EUROGRAPHICS 2026
A. Bönsch and N. Umetani
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 2
STAR – State of The Art Report

Survey on differential estimators for 3d point clouds

Léo Arnal–Anger[1], Thibault Lejemble[2], David Coeurjolly[3], Loïc Barthe[1] and Nicolas Mellado[1]

[1] IRIT, Université de Toulouse, CNRS, Toulouse INP, UT, Toulouse, France
[2] Cerema, Toulouse, France
[3] CNRS, Université Claude Bernard Lyon 1, INSA Lyon, LIRIS, France

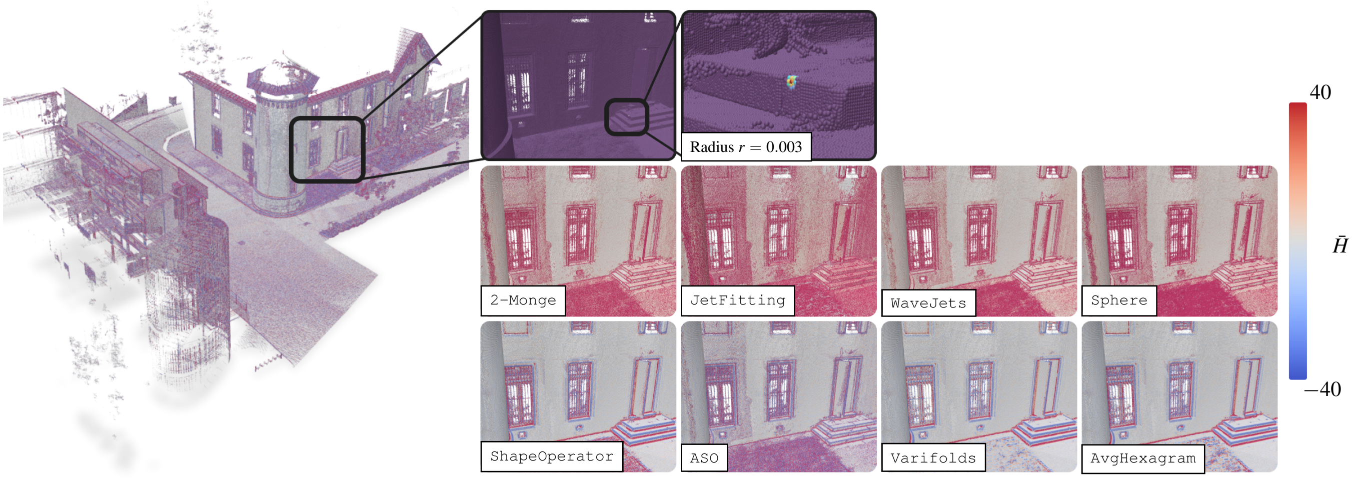


Figure 1: *Computing differential properties on large point clouds (left) is a classical task in many geometry processing pipelines. A lot of related works has been proposed to locally estimate such quantities, as illustrated here with a visual comparison of mean curvature ($\bar{H}$) estimators (values are clamped to the $[-40, 40]$, unsigned/signed curvature values for the top/bottom row respectively).*

Abstract

Recent advancements in 3D scanning technologies, including LiDAR and photogrammetry, have enabled the precise digital replication of real-world objects. These methods are widely used in fields such as GIS, robotics, and cultural heritage. However, the point clouds generated by such scans are often noisy and unstructured, posing challenges for traditional geometry processing tasks. Accurately estimating differential properties like surface curvatures and normals is crucial for tasks such as shape matching and classification, but remains complex due to these inherent challenges. This paper reviews state-of-the-art methods for estimating differential properties from 3D point clouds, with a focus on approaches that offer strong mathematical foundations and theoretical guarantees. We also benchmark these methods using various datasets, evaluating their performance in terms of accuracy, robustness, and efficiency. Our contributions include the release of datasets, tools, and code to promote reproducibility and support future research in this area.

CCS Concepts

• ***Computing methodologies*** $\rightarrow$ ***Shape analysis; Point-based models;***

---

1. Introduction

These last decades have seen a rapid development of 3D scanning technologies allowing nearly anyone to replicate real objects into 3D virtual models. Remote sensing methods like LIDAR provide powerful ways to accurately measure depth in several directions in the 3D space, ranging from small hand-carried devices to airborne systems. Other types of dedicated sensors, often more affordable, also perform depth measurements, such as the pioneering Microsoft

Kinect [Zha12]. In computer vision, techniques like Structure from Motion and Multi-view Stereo [HZ03] infer 3D information solely from several pictures taken from different viewpoints. Photogrammetry [MBM01] encompasses these computational methods, offering tools to automatically reconstruct and analyze 3D scans.

The primary objective of surface scanning techniques is to create 3D digital twins of physical objects, facilitating tasks such as anomaly detection, simulation, interactive visualization, and more. 3D scans have numerous use cases across various domains: GIS, robotics, retro-engineering, and Culture Heritage to name just a few. Modern techniques have become more affordable and can now scan a wide variety of subjects, from small CAD pieces to entire cities and countries, with an unprecedented precision [SM22].

In the 3D scanning pipeline, a point cloud is a fundamental intermediate data structure that is commonly used to represent the surface of a scanned entity, which is either directly obtained from LIDAR measurements, or computed by vision algorithms. Point clouds are typically characterized by an ever-growing number of samples and different kinds of acquisition artifacts. More importantly, point clouds often lack regular structures, especially after the registration of several 3D scans, making them unsuitable for traditional signal processing methods. Therefore, robust and accurate geometry processing algorithms are needed to transform raw 3D data into interpretable structures like meshes [BKP*10], NURBS [PT96], and other higher-level abstractions.

In the upstream part of a standard 3D scanning pipeline, local shape characterization approximates the smooth sampled-surface in the neighborhood of each data sample. The results of this analysis are usually used as input for further geometry processing tasks like shape matching [GMGP05], shape-editing [BDC18], and classification [HLP*21]. In this context, curvature estimation, and more generally differential property estimation, is a fundamental problem where mathematical properties of the unknown smooth surface are estimated from input data points that are scattered in 3D. Intuitively, the goal is to characterize the local geometry by measuring how the surface bends and in which directions. Surface curvatures play an important role in 3D shape analysis as shape descriptors that are invariant to translation and rotation. These geometric features have a strong theoretical foundation, dating back to Gauss work in 1827 [Gau27], providing a framework of analysis with possible theoretical guarantees. In 3D scanning, to address the geometry processing challenges mentioned above, differential estimators working on 3D point clouds must be accurate and precise, while also being robust against noise and other scanning artifacts.

In this work, we provide an overview of the existing methods for estimating differential properties in 3D point clouds. Additionally, we conduct a series of practical experiments to benchmark differential properties estimations using various datasets with different settings. Our contribution is twofold:

- An extensive review of state-of-the-art methods for estimating curvatures, normal vectors, and other differential properties on unstructured 3D point clouds. We specifically focus on methods offering an explicit mathematical formulation (according to fundamental background detailed in Section 3) and providing theoretical guarantees on the computed quantities (see Section 4).
- A benchmark of these methods on different sampled shapes with varying levels of noise, sampling and observation scale (see Sections 5 and 6).

In addition, we released all the materials required to replicate our work and compare upcoming estimators: dataset with ground truth, the tools required to evaluate the benchmark and process the results (scripts and visualization tools)[†], and the source code with our implementation of the presented estimators[‡]. We also provide the raw data used in our experiments, such that practitioners can analyze behaviors related to their specific use-case.

In this work, we found general trends on the estimator behaviors, and we also discuss about general practices in Section 8, *eg*. the impact of the neighborhood queries or the order of the fitted primitive. In section 9, we review high-level potential future work related to the estimation of differential properties on 3d point clouds.

2. Related surveys

3D point cloud processing and local shape characterization methods have been extensively reviewed in the past. The existing surveys mainly focus on different aspects of point cloud processing like shape reconstruction, or on specific differential estimators on 3D data other than point clouds. We briefly describe these prior surveys in the following sections.

2.1. Shape reconstruction

Berger *et al*. [BLN*13] introduced a benchmarking tool to evaluate reconstruction algorithms, while their survey [BTS*17] categorizes these methods based on surface priors such as smoothness and regularity. Sulzer et al. [SMVL25] also summarize surface reconstruction methods based on surface and volumetric representations, that perform interpolation and approximation, including recent deep-learning techniques. Kaiser et al. [KYZB19] study more specific approaches for simple geometric primitive fitting, according to the level of abstraction, the fitted primitive, and the fitting framework. Cheng et al. [CWL*08] outline Moving Least Squares (MLS) surfaces representations. More recently, Tey et al. [TSA*21] review state-of-the-art variants of the MLS method for reconstruction purposes, see Section 5.3 for further discussions.

From a different perspective, several surveys outline 3D point cloud analysis methods depending on the type of sensor used to acquire the data, such as RGB-D camera [ZSG*18]. Other works indirectly include local shape analysis and differential estimations but focus on the application domain like Cultural Heritage [PPY*16, CPPB20] and human body recognition [BDTB18].

All these related works are somewhat crosswise to our problem of curvature estimation in point cloud. For instance, a primitive fitting method [KYZB19] could be used to compute curvatures afterward. However, differential estimators are not precisely investigated in these approaches that are more global and that mainly

---

[†] `https://storm-irit.github.io/pcloud-differential-estimation-benchmark-website/`

[‡] `https://github.com/STORM-IRIT/pcloud-differential-estimation-benchmark`

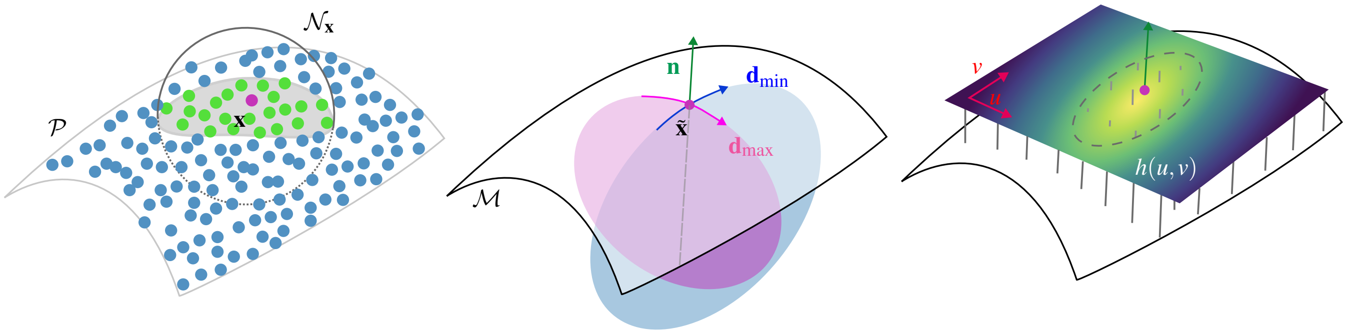


Figure 2: *Given a point cloud $\mathcal{P}$ sampling a smooth manifold $\mathcal{M}$, differential estimators aim at computing quantities at $\mathbf{x} \in \mathcal{P}$, usually from a neighborhood $\mathcal{N}_{\mathbf{x}}$* (left)*, such that they approximate differential measures on $\mathcal{M}$ at some related point $\tilde{\mathbf{x}}$* (center)*, illustrated here with the normal vector and the principal directions. We also illustrate the parameterization of a surface patch as a height function* (right).

focus on 1^{st}-order surface reconstruction. Theoretical guarantees of convergence and robustness are neither provided.

2.2. Differential estimators

Normal vector estimation in point cloud is perhaps the most studied task [KMJ*19]. Many authors propose to benchmark their method on popular datasets such as ABC [KMJ*19] and PCP-Net [GKOM18]. However, only deep-learning techniques are usually tested in these works. Concerning curvature estimation on 3D data, only a few benchmark exists to our knowledge, but they are restricted to triangular meshes [MSR07, VVP*16]. From both a theoretical and applied perspective, discrete curvature is the main topic of a state-of-the-art that discusses ongoing research work on a rich set of discrete spaces [NR17], but 3D point clouds coming from 3D scans are not specifically tackled in this work.

The estimation of differential properties on 3D point clouds has not been reviewed yet despite the number of algorithms proposed through the years. Moreover, there is no practical benchmark that evaluates curvature accuracy, precision, and robustness using a common testing framework in the context of 3D scanning. We propose to overcome this lack of surveys and benchmark with an emphasis on replicability [BCDM20].

3. The fundamentals

In this section, we review the fundamentals of differential geometry [DC16] and introduce the notations used in this survey.

3.1. Smooth setting, fundamental forms and curvature tensor

Let us consider a smooth manifold $\mathcal{M}$ at least twice differentiable. Let us consider a local parameterization of $\mathcal{M}$ in an open set $\mathcal{S}$ at $\mathbf{x}$ as

$$f(u,v) = (u, v, h(u,v)) \,, \tag{1}$$

in a frame $(\mathbf{n}, \mathbf{t}, \mathbf{b})$ centered at $\mathbf{x}$ (respectively the normal, a tangent and a binormal unit vectors at $\mathbf{x}$). In the following, f is called the Monge patch of $\mathcal{M}$ at $\mathbf{x}$. Furthermore, we refer to $h(u,v)$ (also denoted $h(\mathbf{q})$ with $\mathbf{q} = \begin{pmatrix} q_u \\ q_v \end{pmatrix}$, for convenience) as the height function of the patch. According to the Taylor theorem, if h is infinitely differentiable, it can be decomposed as

$$h(u,v) = \sum_{k=0}^{\infty} \sum_{j=0}^{k} \frac{h_{u^{k-j}v^{j}}(0,0)}{(k-j)!\,j!} u^{k-j} v^{j} \,, \tag{2}$$

where $h_{u^{k-j}v^{j}} = \frac{\partial^k h}{\partial u^{k-j} \partial v^{j}}$. If h is only differentiable up to a given order n, then the sum of Equation 2 is truncated and h is approximated up to a residual term $o(u^n + v^n)$.

The first fundamental form of f is given by

$$\mathrm{I} = E du^2 + 2F du dv + G dv^2 \,, \tag{3}$$

where $E = \|f_u\|^2$, $F = f_u \cdot f_v$ and $G = \|f_v\|^2$ (with $f_u = \frac{\partial f}{\partial u}$). Such fundamental form completely captures the local metric properties of $\mathcal{M}$ at $\mathbf{x}$.

In the frame of the Monge patch f, the normal vector $\mathbf{n}_f \in \mathbb{R}^3$ is defined from the surface's derivatives:

$$\mathbf{n}_f = \frac{f_u \times f_v}{\|f_u \times f_v\|} = \frac{(-h_u, -h_v, 1)}{\sqrt{h_u^2 + h_v^2 + 1}} \,, \tag{4}$$

with × the cross product. In the following, we denote $\mathbf{n} \in \mathbb{R}^3$ the normal expressed in the embedding space.

The bending of the surface from its tangent plane is captured by the second fundamental form defined as

$$\mathrm{II} = L du^2 + 2M du dv + N dv^2 \,, \tag{5}$$

where $L = f_{uu} \cdot \mathbf{n}_f$, $M = f_{uv} \cdot \mathbf{n}_f$ and $N = f_{vv} \cdot \mathbf{n}_f$. To simplify the notation, we adopt the classical matrix forms

$$\mathrm{F_I} = \begin{pmatrix} E & F \\ F & G \end{pmatrix} \qquad \mathrm{F_{II}} = \begin{pmatrix} L & M \\ M & N \end{pmatrix} . \tag{6}$$

To observe the variation of the surface normal in a neighborhood, we consider the Gaussian map G that maps the point $\mathbf{x} \in \mathcal{M}$ to $\mathbf{n}_{\mathbf{x}} \in S^2$ the unit sphere. Considering the differential of $\mathrm{D_{\mathbf{x}}G} : \mathrm{T_{\mathbf{x}}}\mathcal{M} \rightarrow$

$\mathrm{T}_{\mathbf{x}}\mathcal{M}$, one can define the Weingarten map (sometimes also called the Shape Operator) as $\mathrm{W}_{\mathbf{x},\mathcal{M}} = -\mathrm{D}_{\mathbf{x}}\mathrm{G}$. In our parametric setting, we simply have a matrix representation $\mathrm{W} = \mathrm{F}_{\mathrm{I}}^{-1}\mathrm{F}_{\mathrm{II}}$, leading to the Gaussian and mean curvature respectively computed as

$$\mathrm{K} = \det(\mathrm{W}), \qquad \mathrm{H} = \frac{1}{2}\,\mathrm{trace}(\mathrm{W})\,. \tag{7}$$

Alternatively, K and H can be expressed w.r.t. the first and second order partial derivatives:

$$\mathrm{K} = \frac{h_{uu}h_{vv} - h_{uv}^2}{(h_u^2 + h_v^2 + 1)^2} \tag{8}$$

$$\mathrm{H} = \frac{(1+h_u^2)h_{uu} + (1+h_v^2)h_{vv} - 2h_u h_v h_{vv}}{2(h_u^2 + h_v^2 + 1)^{\frac{3}{2}}}. \tag{9}$$

The Weingarten map is a 2×2 matrix, but in the literature it is sometimes used to represent the 3×3 matrix W_3 linked to W by the following relation:

$$\mathrm{W} = \mathrm{P}^T\mathrm{W}_3\mathrm{P}, \tag{10}$$

where $\mathrm{P} = [\mathbf{t}\ \mathbf{b}]$. Since the Weingarten map is self-adjoint, a basis exists consisting of the eigenvectors of W. The eigenvectors, combined with their eigenvalues, define the principal curvature directions, $\mathbf{d}_{\min}$ and $\mathbf{d}_{\max}$, and the principal curvatures $\kappa_{\min}, \kappa_{\max}$. Decomposing W_3 provides a third direction corresponding to the normal vector $\mathbf{n}$. Principal curvatures are linked to the Gaussian and mean curvatures such that

$$\mathrm{K} = \kappa_{\min}\kappa_{\max}\,, \qquad \mathrm{H} = \frac{\kappa_{\min} + \kappa_{\max}}{2}\,, \tag{11}$$

$$\kappa_{\min}/\kappa_{\max} = \mathrm{H} \pm \sqrt{\mathrm{H}^2 - \mathrm{K}}. \tag{12}$$

3.2. Manifold sampling and discrete settings

In this survey, we focus on a point cloud, which is a discrete approximation of $\mathcal{M}$ from a finite set $\mathcal{P}$ of samples $\{\mathbf{p}_i\} \in \mathbb{R}^3$, as illustrated in Figure 2. $\mathcal{P}$ may be equipped with unitary normal vectors $\{(\mathbf{p}_i, \mathbf{n}_i)\} \in \mathbb{R}^3 \times \mathcal{S}^2$. As we aim to relate quantities estimated at $\mathbf{p}_i \in \mathcal{P}$ to some expected quantities at some $\mathbf{x} \in \mathcal{M}$, we first need assumptions on the sampling $\mathcal{P}$ of $\mathcal{M}$. The concept of ϵ−sampling by Amenta and Bern [AB98] had been seminal for surface reconstruction results from point clouds. Roughly, $\mathcal{P}$ is an ϵ−sampling of $\mathcal{M}$ if it is sufficiently dense with respect to the distance to the medial axis of $\mathcal{M}$. Under such hypothesis, one can derive properties on the estimated quantities (normal vectors [AB98], area [MT02], curvatures [CSM03], ...). Extensions have been proposed to relax this sampling model while enforcing some stability results [AKF*24]. For instance, some convergence results (*eg*. [MOG11]) can be obtained under the hypothesis that the Hausdorff distance between $\mathcal{P}$ and $\mathcal{M}$ is bounded by some ϵ. In our context, stability or convergence results are stated up to a sampling hypothesis (*eg*. Hausdorff approximation) and on a given class of manifolds $\mathcal{M}$ (*eg*. C^2 with bounded curvature), with a formal proof that the estimation error is bounded by a decreasing function on the sampling parameter ϵ.

As we are focusing on local differential quantities, we denote $\mathcal{N}_{\mathbf{x}} := \{(\mathbf{p}_i, \mathbf{n}_i)\}_{i...n} \subset \mathcal{P}$ the neighborhood of the evaluation point $\mathbf{x} \in \mathbb{R}^3$. For the sake of simplicity, we also use $\mathcal{N}_{\mathbf{x}}$ as the set of indices used to iterate over the neighbors, *eg*. $\sum_{i\in\mathcal{N}_{\mathbf{x}}} \mathbf{p}_i$. The construction of such neighborhood will be discussed later (see Section 5.1), but without loss of generality, one can simply define $\mathcal{N}_{\mathbf{x}}$ as the k-closest points, or the set of points at Euclidean distance r.

Methods	input normals	estimator	sign	Eq.
`Mean` [PWY*07]	oriented	$\bar{\mathbf{n}}$ $\bar{\mathrm{H}}$		(15)
`Cov2D` [BC94, DM14]		$\bar{\mathrm{W}}$		(18)
`NormCov2D` [BC94, DM14]	unoriented	$\bar{\mathrm{W}}$		(21)
`NormCov3D` [LT90, DM14]	unoriented	$\bar{\mathrm{W}}_3$		(17)
`ShapeOperator` [KSNS07]	oriented	$\bar{\mathrm{W}}$	✓	(22)
`PCA` [HJ87]		$\bar{\mathbf{n}}, \bar{\mathbf{t}}, \bar{\mathbf{b}}$		(26)
`2-Monge` [Gra98]		$\bar{\mathrm{W}}$		(33)
`PC-MLS` [RGRG15]		$\bar{\mathrm{W}}$		(36)
`JetFitting` [CP05]		$\bar{\mathrm{W}}$		(38)
`WaveJets` [BDC18]		$\bar{\mathrm{K}}, \bar{\mathrm{H}}$		(40)
`Sphere` [Pra87]		$\bar{\mathbf{n}}, \bar{\mathrm{H}}$		(45)
`APSS` [GG07]	oriented	$\bar{\mathbf{n}}, \bar{\mathrm{H}}$	✓	(45)
`UnorientedSphere` [CGBG13]	unoriented	$\bar{\mathbf{n}}, \bar{\mathrm{H}}$		(45)
`ASO` [LCBM21]	oriented	$\bar{\mathbf{n}}, \bar{\mathrm{W}}$	✓	(49)
`3DQuadric` [DB02]		$\bar{\mathrm{W}}$		
`Varifolds` [BLM22]	oriented	$\bar{\mathrm{W}}_3$	✓	(55)
`AvgHexagram` [LCL*23]	oriented	$\bar{\mathrm{H}}, \bar{\mathrm{K}}, \bar{\mathrm{W}}_3$	✓	(51)

Table 1: *All methods presented in this survey, with the type of input normals: not required, globally oriented, or unoriented (just the direction), the quantities they estimate, if the curvatures are signed or not, and their related equations. For* `Mean`*, the normal estimator is oriented, the curvature is not.*

Finally, when processing geometrical information in $\mathcal{P}$, a weighting kernel is usually considered to scale the importance of each neighbor around $\mathbf{x}$ (see details in Section 5.2). In most cases, such function $\omega_{\mathbf{x}}$ is a monotonic radial decreasing function centered at $\mathbf{x}$. For the sake of simplicity, w_i denotes the weight assigned to the point $(\mathbf{p}_i, \mathbf{n}_i) \in \mathcal{N}_{\mathbf{x}}$, *i.e.* $w_i := \omega_{\mathbf{x}}(\mathbf{p}_i)$.

3.3. Link to integral invariants

Integral Invariants [Con86, HT03, MHYS04, PWY*07] refer to differential quantities of surfaces estimated from local integral computations. In the smooth setting, we analyze the intersection between a sphere or a ball centered at the point of interest, and the smooth object or its boundary defining the surface. The volume (or area), and higher order geometrical moments of the intersection provably capture some curvature information [CRT04, PWY*07, PWHY09]. Convergence and stability proofs are usually obtained when applying the given estimator on a smooth Taylor approximation of the surface such as the one given in Equation 2, as the sphere or ball radius tends to zero. On discrete structures such as meshes [PWY*07], voxel surfaces [LCL17] or point clouds [DM14], the Integral Invariant approach provides a framework to design stability proofs of discrete curvatures estimators.

4. Methods

The estimation of differential properties is a core topic in Geometry Processing. An overview of the existing approaches is given in Table 1. We identify three main families of analytic approaches, based on direct estimation from points and their attributes (Section 4.1), based on local surface models (Section 4.2), and on concepts from the Measure Theory (Section 4.3). The comparative analysis of these estimators is presented in Section 7. We also review Machine Learning methods (Section 4.4). However, due to generalization challenges on unseen data and specific retraining requirements, their experimental evaluation is reported in the supplementary material. In order to compare the reviewed approaches, both theoretically and practically, we define the following elements:

- **Input samples.** For all approaches, the estimators are evaluated at a location $\mathbf{x}$. The set of samples ($\mathbf{p}_i$, and/or in some cases $\mathbf{n}_i$, *eg*. Equation 16) used by an estimator are gathered in the neighborhood $\mathcal{N}_{\mathbf{x}}$.
- **Estimators.** For each method, when available, we present the estimators $\bar{\mathbf{n}}$, $\bar{\mathrm{H}}$ and $\bar{\mathrm{K}}$. When approaches estimate $\bar{\mathrm{W}}$ directly, the curvature estimators $\bar{\mathrm{H}}$ and $\bar{\mathrm{K}}$ are computed using Equation 7, and the principal curvatures (values $\kappa_{\min}, \kappa_{\max}$ and directions $\mathbf{d}_{\min}$ and $\mathbf{d}_{\max}$) using a covariance analysis. In addition, the estimation of $\bar{\mathrm{W}}_3$ provides access to $\bar{\mathrm{W}}$ (see Equation 10), the associated estimators, and the normal vector $\bar{\mathbf{n}}$.
- **Surface orientation and sign.** As detailed in Table 1, some estimators are signed, and some are not. In the context of this survey, an *unsigned* estimator characterizes the surface properties without distinguishing the inner and outer volumes. As a consequence, vectors (*ie*. normal and principal curvature directions) are estimated up to a flip, and curvature estimates are related to a magnitude of curvature, ignoring the difference between concavities and convexities. Since the position of the input samples is not enough to know the surface orientation, all signed estimators take oriented normal vectors as input (except neural networks that rely on other assumptions to estimate the surface orientation).

4.1. Direct point-based estimators

We first review techniques that estimate differential properties directly from the discrete sample distribution. These methods rely on the statistical analysis of point distrubution in a local neighborhood, exploiting weighted averages and covariance matrices constructed from point positions and normal vectors. By analyzing the spatial variations and correlations within the neighborhood (through eigen-analysis or least squares regressions) these approaches directly approximate geometric quantities at the query position (*eg*. the Weingarten map W).

4.1.1. Local averaging

We define the barycenter $\bar{\mathbf{p}}(\mathbf{x})$ as the weighted mean

$$\bar{\mathbf{p}}(\mathbf{x}) = \frac{\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i \mathbf{p}_i}{\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i}\,. \tag{13}$$

In [PWY*07, Theorem 6], the authors show that $\bar{\mathbf{p}}(\mathbf{x})$ is related to the surface mean curvature H such that

$$\bar{\mathbf{p}}(\mathbf{x}) = \begin{pmatrix} O(r^3) \\ O(r^3) \\ \frac{\mathrm{H}r^2}{4} + O(r^3) \end{pmatrix}. \tag{14}$$

With an oriented point cloud, the orientation of the plane defined by a normal vector estimator $\bar{\mathbf{n}}(\mathbf{x})$ and a point $\mathbf{x}$ may be computed as the weighted average of the neighbors' normal vectors of $\mathbf{x}$. Combined with Equation 14, this leads to the following estimator:

$$\bar{\mathrm{H}}(\mathbf{x}) = \frac{4\,\|\bar{\mathbf{p}}(\mathbf{x})-\mathbf{x}\|}{r^2} \tag{15}$$

$$\bar{\mathbf{n}}(\mathbf{x}) = \frac{\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i \mathbf{n}_i}{\|\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i \mathbf{n}_i\|}\,. \tag{16}$$

`Mean`

4.1.2. Normal-based 3D Covariance

As introduced by Liang et al. [LT90], the 3D covariance matrix of the normal vectors estimates the Weingarten map, such that:

$$\bar{\mathrm{W}}_3(\mathbf{x}) = \frac{1}{\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i} \sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i \left(\mathbf{n}_i - \tilde{\mathbf{n}}(\mathbf{x})\right)\left(\mathbf{n}_i - \tilde{\mathbf{n}}(\mathbf{x})\right)^T, \tag{17}$$

`NormCov3D`

with $\tilde{\mathbf{n}} = \sum w_i \mathbf{n}_i / \sum w_i$.

4.1.3. Position-based 2D Covariance

Berkmann et al. [BC94] directly estimate the Weigarten map by computing the 2D covariance matrix of the neighbor points projected onto the tangent plane:

$$\bar{\mathrm{W}}(\mathbf{x}) = \frac{1}{\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i} \sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i \left(\mathbf{r}_i - \bar{\mathbf{r}}(\mathbf{x})\right)\left(\mathbf{r}_i - \bar{\mathbf{r}}(\mathbf{x})\right)^T, \tag{18}$$

`Cov2D`

where

$$s_i = \bar{\mathbf{n}}(\mathbf{x}) \cdot (\mathbf{p}_i - \mathbf{x}), \tag{19}$$

$$\mathbf{r}_i = s_i\, \bar{\mathrm{P}}(\mathbf{x})^T (\mathbf{p}_i - \mathbf{x}), \tag{20}$$

and $\bar{\mathrm{P}}$ projects a point onto an estimated 2D tangent plane as in Equation 10. While such estimation does not properly estimate the curvature values [DM14], it is provably accurate for the estimation of curvature directions.

4.1.4. Normal-based 2D Covariance

Another 2D covariance matrix is formulated by taking the projected normals onto the tangent plane:

$$\bar{\mathrm{W}}(\mathbf{x}) = \frac{1}{\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i} \sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i \left(\mathbf{v}_i - \bar{\mathbf{v}}(\mathbf{x})\right)\left(\mathbf{v}_i - \bar{\mathbf{v}}(\mathbf{x})\right)^T, \tag{21}$$

`NormCov2D`

where $\mathbf{v}_i = \bar{\mathrm{P}}(\mathbf{x})^T \mathbf{n}_i$. It gives similar results as the 3D normal covariance, as demonstrated by Digne et al. [DM14].

4.1.5. Shape operator fitting

Given the normals and positions of the neighbors of $\mathbf{x}$, Kalogerakis et al. [KSNS07] directly fit the Weingarten map using a least-squares regression:

$$\bar{\mathrm{W}}(\mathbf{x}) = \left(\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i \mathbf{v}_i \mathbf{v}_i^T\right)\left(\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i \mathbf{d}_i \mathbf{d}_i^T\right)^{-1}, \tag{22}$$

`ShapeOperator`

where $\mathbf{q}_i$ and $\mathbf{v}_i$ are the projected differences of positions and normals:

$$\mathbf{d}_i = \bar{\mathrm{P}}(\mathbf{x})^T(\mathbf{p}_i - \mathbf{x}) \tag{23}$$

$$\mathbf{v}_i = \bar{\mathrm{P}}(\mathbf{x})^T(\mathbf{n}_i - \mathbf{n}). \tag{24}$$

4.2. Local surface models

In contrast to the direct estimators described in the previous section, the approaches presented here rely on the reconstruction of local surface models. The core idea is to approximate the local geometry of the neighborhood by fitting a (smooth) surface (*eg*. a parametric or an implicit function), and then retrieve the geometric properties from the surface's parameters. In contrast to direct point-based estimators, the curvatures are estimated on the surface (at the projected point) and not at the original location. These methods can be categorized into two main families:

- **2.5D methods** (or Monge patches), which represent the surface as a local heightfield $h(u,v)$. These approaches require a preliminary estimation of a reference frame (typically a tangent plane) to define the parameterization domain.
- **3D methods**, which fit implicit surfaces directly in the 3D space without requiring a prior planar projection.

The following sections review standard models from both categories, ranging from simple plane fitting to higher-order algebraic surfaces. The choice of local surface models in surface reconstruction algorithms is discussed in Section 5.3.

4.2.1. 3D Plane fitting

Plane fitting is a standard procedure in geometry processing. It is used by many algorithms and pipelines [SDC*20, BLL*22] to estimate normal vectors [HJ87, MNG04] and tangent planes [HDD*92]. Let us consider the covariance matrix:

$$\mathrm{C}(\mathbf{x}) = \frac{1}{\sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i} \sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i\,(\mathbf{p}_i - \bar{\mathbf{p}}(\mathbf{x}))\,(\mathbf{p}_i - \bar{\mathbf{p}}(\mathbf{x}))^T, \tag{25}$$

and define the local frame from the eigenvectors of $\mathrm{C}(\mathbf{x})$ (respectively $\mathbf{u}_0^{\mathrm{C}}$, $\mathbf{u}_1^{\mathrm{C}}$, $\mathbf{u}_2^{\mathrm{C}}$ for eigenvalues $\lambda_0^{\mathrm{C}} \leq \lambda_1^{\mathrm{C}} \leq \lambda_2^{\mathrm{C}}$).

For plane fitting, the standard approach is to estimate the best fit plane in the least squares sense, by taking the plane passing by the neighborhood barycenter, and orthogonal to the direction with the smallest variance λ_0^{C} (see covariance matrix construction at Equation 25):

$$\bar{\mathbf{n}}(\mathbf{x}) = \mathbf{u}_0^{\mathrm{C}}, \tag{26}$$

$$\bar{\mathbf{t}}(\mathbf{x}) = \mathbf{u}_1^{\mathrm{C}}, \tag{27}$$

$$\bar{\mathbf{b}}(\mathbf{x}) = \mathbf{u}_2^{\mathrm{C}}. \tag{28}$$

`PCA`

With this approach, the projection of a point onto the locally fitted plane generates a mean curvature flow [DMSL11, Theorem 2]:

$$\mathrm{proj}(\mathbf{x}) - \mathbf{x} = -\frac{Hr^2}{4}\bar{\mathbf{n}}(\mathbf{x}) + O(r^3), \tag{29}$$

where $\mathrm{proj}(\mathbf{x})$ is the orthogonal projection of $\mathbf{x}$ onto the fitted plane.

When normal vectors are available, an alternative approach is to take the plane passing by the mean position and orthogonal to the mean direction of normal vectors $\mathbf{n}_i$ (see normal estimator at Equation 16).

4.2.2. 2.5D Monge-Patch fitting

As detailed in Equation 1, a Monge-Patch is defined by a bivariate polynomial $h(u,v)$ representing the surface as a local heightfield. In this section, we detail several propositions for h and the properties of the associated estimators. In a local frame defined by a 3×3 matrix $\bar{\mathrm{B}}(\mathbf{x}) = [\bar{\mathbf{n}}(\mathbf{x}), \bar{\mathbf{t}}(\mathbf{x}), \bar{\mathbf{b}}(\mathbf{x})]$, each point $\mathbf{p}_i$ is represented by its height h_i and tangential components $\mathbf{q}_i \in \mathbb{R}^2$

$$\begin{pmatrix} h_i \\ \mathbf{q}_i \end{pmatrix} = \bar{\mathrm{B}}(\mathbf{x})^T\,(\mathbf{p}_i - \bar{\mathbf{p}}(\mathbf{x})). \tag{30}$$

Given a model $h(u,v)$, a fitted plane and its associated frame, (*eg*. computed using Equations 26, 27 and 28), Monge Patch fitting can be expressed as the following least-squares problem:

$$\underset{h}{\arg\min} \sum_{i\in\mathcal{N}_{\mathbf{x}}} w_i\,(h(\mathbf{q}_i) - h_i)^2. \tag{31}$$

In the following sections, we review the differential estimators induced by several models.

4.2.2.1. 2.5D quadrics This model defines the local heightfield as quadratic polynomial [Gra98]:

$$h(\mathbf{q}) = u_c + \mathbf{u}_\ell^T\mathbf{q} + \mathbf{q}^T\mathrm{U}_q\mathbf{q}, \tag{32}$$

where $u_c \in \mathbb{R}$, $\mathbf{u}_\ell \in \mathbb{R}^2$ and $\mathrm{U}_q = \begin{pmatrix} u_{q_a} & u_{q_b} \\ u_{q_b} & u_{q_c} \end{pmatrix} \in \mathbb{R}^{2\times2}$ are respectively the constant, linear and quadratic parameters. The Weingarten map can be directly computed from this model, as detailed in Section 1.1 of the supplementary. For the sake of readability, we only detail the estimators $\bar{\mathrm{H}}(\mathbf{x})$ and $\bar{\mathrm{K}}(\mathbf{x})$ (derived using Equations 8 and 9, but one can retrieve the other estimators from $\bar{\mathrm{W}}$), which are expressed as:

$$\bar{\mathrm{K}}(\mathbf{x}) = \frac{4u_{q_a}u_{q_c} - 4u_{q_b}^2}{([\mathbf{u}_\ell]_0^2 + [\mathbf{u}_\ell]_1^2 + 1)^2}, \tag{33}$$

$$\bar{\mathrm{H}}(\mathbf{x}) = \frac{u_{q_a}(1 + [\mathbf{u}_\ell]_1^2) + u_{q_c}(1 + [\mathbf{u}_\ell]_0^2) - 2u_{q_b}\,[\mathbf{u}_\ell]_0\,[\mathbf{u}_\ell]_1}{([\mathbf{u}_\ell]_0^2 + [\mathbf{u}_\ell]_1^2 + 1)^{\frac{3}{2}}}. \tag{34}$$

`2-Monge`

4.2.2.2. Parabolic cylinder From the expression of such quadrics, Ridel et al. [RGRG15] fit a parabolic cylinder to ensure the developpability of the surface. First, they compute the eigenvectors of $\mathcal{H}(\mathbf{x}) \in \mathbb{R}^{2\times 2}$ the Hessian matrix of $h(\mathbf{q})$ (respectively $\mathbf{u}_0^{\mathcal{H}}$, $\mathbf{u}_1^{\mathcal{H}}$, for eigenvalues $|\lambda_0^{\mathcal{H}}| \leq |\lambda_1^{\mathcal{H}}|$), with h defined as a quadratic polynomial, see Equation 32, and $\mathcal{H}$ being equal to the Weingarten map of h under the assumption $\mathrm{P}(\mathbf{x}) = \mathrm{T}(\mathbf{x})$, the tangent plane (see Section 1.2 of the supplementary).

The key idea of their work is to cancel the minimal curvature by ignoring surface variations in direction $\mathbf{u}_0$, leading to the new expression of the parametric surface:

$$h(\mathbf{q}) = u_c + \mathbf{u}_\ell^T \mathbf{q} + a\mathbf{q}^T \left(\mathbf{u}_1^{\mathcal{H}} (\mathbf{u}_1^{\mathcal{H}})^T\right) \mathbf{q}, \qquad (35)$$

with $a \in \mathbb{R}$ being a parameter optimized by the method. This formulation also provides a direct estimator of the Weingarten map (see Section 1.1 of the supplementary). The mean and Gaussian curvature estimators are expressed as:

$$\bar{\mathrm{K}}(\mathbf{x}) = 0, \qquad (36)$$

$$\begin{aligned}\bar{\mathrm{H}}(\mathbf{x}) = &\frac{a\left(\left[\mathbf{u}_1^{\mathcal{H}}\right]_0^2 (1 + [\mathbf{u}_\ell]_1^2) + \left[\mathbf{u}_1^{\mathcal{H}}\right]_1^2 (1 + [\mathbf{u}_\ell]_0^2)\right)}{([\mathbf{u}_\ell]_0^2 + [\mathbf{u}_\ell]_1^2 + 1)^{\frac{3}{2}}} \\ &- \frac{2a\left(\left[\mathbf{u}_1^{\mathcal{H}}\right]_0 \left[\mathbf{u}_1^{\mathcal{H}}\right]_1 [\mathbf{u}_\ell]_0 [\mathbf{u}_\ell]_1\right)}{([\mathbf{u}_\ell]_0^2 + [\mathbf{u}_\ell]_1^2 + 1)^{\frac{3}{2}}}.\end{aligned} \qquad (37)$$

`PC-MLS`

4.2.2.3. Jets The polynomial fitting of osculating Jets [CP05] is a method that generalizes the 2.5D bivariate polynomial to higher degrees:

$$h(u,v) = \sum_{k=0}^{d} \sum_{j=0}^{k} b_{k-j,j}\, u^{k-j} v^{j}, \qquad (38)$$

with $b_{i,j} \in \mathbb{R}$ the polynomial coefficients, and d the polynomial degree. In practice, the authors first fit a local frame using covariance analysis (see Section 4.1), refine it by fitting the model introduced in Equation 38 with a degree $d \geq 2$ (denoted as d-jet). The Monge patch coefficients $b_{i,j}$ are finally obtained by fitting the same model of (potentially different) degree d, denoted as d-Monge. The coefficients directly estimate the Taylor decomposition of the patch (see Equation 2), giving access to the fundamental forms, and thus the Weingarten map estimator.

4.2.2.4. Wavejets Bearzi et al. [BDC18] propose to decompose the bivariate Taylor approximation (Equation 2) into a radial polynomial and angular oscillations, and call their new set of basis functions WaveJets . Given a plane fitted at $\mathbf{x}$ using a radius r, the local heightfield is expressed in polar coordinate such that $h(u,v) = h(\rho\cos\theta, \rho\sin\theta)$, denoted $h(\rho,\theta)$, and such that $\rho \in [0,r]$. The associated Taylor expansion is defined as

$$h(\rho,\theta) = \sum_{n=-\infty}^{\infty} \sum_{k=|n|}^{\infty} \rho^k \phi_{k,n} e^{\mathbf{i}n\theta}, \qquad (39)$$

with $\phi_{k,n} = \sum_{j=0}^{k} \frac{1}{j!(k-j)!} b(k,j,n) h_{u^{k-j}v^{j}}(0,0)$, and

- $b(k,j,n) = 0$ if k and n do not have the same parity,
- $b(k,j,n) = \frac{1}{2^k \mathbf{i}^j} \sum_{\ell=0}^{(n-k)/2} \binom{k-j}{\ell} \binom{j}{\frac{n-k}{2} - \ell} (-1)^\ell$ otherwise.

Each $\phi_{k,n}$ is related to an order of radial derivation k and to a number of oscillations n. By explicitly writing the link between $\phi_{k,n}$ and the derivatives of f, the mean and Gaussian curvatures are given as follows:

$$\bar{\mathrm{K}}(\mathbf{x}) = \frac{4\phi_{2,0}^2 - 16\phi_{2,-2}\phi_{2,2}}{(1 + 4\phi_{1,-1}\phi_{1,1})^2}, \qquad (40)$$

$$\bar{\mathrm{H}}(\mathbf{x}) = \frac{2\phi_{2,0}(1 + 4\phi_{1,-1}\phi_{1,1}) + 4\phi_{2,-2}\phi_{1,1}^2 + 4\phi_{2,2}\phi_{1,-1}^2}{(1 + 4\phi_{1,-1}\phi_{1,1})^{\frac{3}{2}}}. \qquad (41)$$

`WaveJets`

The authors also introduce the following estimators:

$$\kappa_{\mathrm{min}} = 2\left(\phi_{2,0} + \phi_{2,2} + \phi_{2,-2}\right), \qquad (42)$$

$$\kappa_{\mathrm{max}} = 2\left(\phi_{2,0} - \phi_{2,2} - \phi_{2,-2}\right), \qquad (43)$$

which are based on the assumption that the fitted tangent plane is aligned with the surface tangent, *ie.* h_u and h_v vanish to 0. In practice, this simplification does not always hold (see Figure 3) and might introduce instabilities in the principal curvature estimation (see Section 7).

4.2.2.5. Impact of the tangent plane estimation In practice, all the methods based on Monge patch fitting raise ambiguities regarding the Weingarten map estimation. As they rely on a preliminary frame estimation, the Weingarten map is expressed in the fitting frame ($\bar{\mathrm{P}}(\mathbf{x})$ in Figure 3), which is not aligned with the surface normal ($\bar{\mathbf{n}}(\mathbf{x})$ in Figure 3) Hence, the normal $\bar{\mathbf{n}}(\mathbf{x})$ estimated by the surface variations is not orthogonal to the principal curvature directions estimated by W, which is expressed relatively to $\bar{\mathrm{P}}(\mathbf{x})$. In order to reduce the impact of these differences, one can iterate several Monge patch fits (as in [CP05]) to refine the tangent plane and align it to the normal vector.

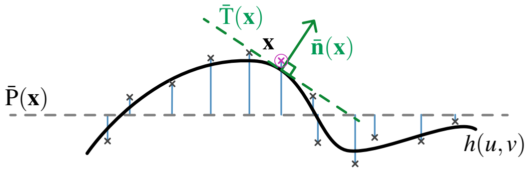


Figure 3: *Monge-Patch estimators are expressed relatively to a fitted plane $\bar{\mathrm{P}}(\mathbf{x})$ which is not aligned with the reconstructed surface normal $\bar{\mathbf{n}}(\mathbf{x})$ of $\bar{\mathrm{T}}(\mathbf{x})$, leading to potential inconsistencies in the estimation.*

4.2.3. 3D Algebraic Sphere

In a seminal work, Pratt [Pra87] introduces an approach to directly fit a sphere to scattered points. In 3D, the sphere is defined *algebraically*, *ie.* as the 0-set of the polynomial $f : \mathbb{R}^3 \to \mathbb{R}$:

$$f(\mathbf{x}) = u_c + \mathbf{u}_\ell^T \mathbf{x} + u_q \mathbf{x}^T \mathbf{x}, \qquad (44)$$

with $u_c \in \mathbb{R}, \mathbf{u}_\ell \in \mathbb{R}^3$ and $u_q \in \mathbb{R}$ the constant, linear and quadratic parameters of the sphere. In contrast to the standard center/radius representation, algebraic spheres smoothly degenerate to planes when $u_q \to 0$ (while center and radius go to infinity).

By definition, $f(\mathbf{x})$ can be scaled by any multiplicative factor without affecting the 0-set of the function, and thus the sphere. In order to get a unique relationship between $f(\mathbf{x})$ and the sphere, the author constrains the gradient of $f(\mathbf{x})$ to be unitary when $f(\mathbf{x}) = 0$, by normalizing $f(\mathbf{x})$ by the Pratt's norm [Pra87] that is equal to $\sqrt{\|\mathbf{u}_\ell\|^2 - 4u_c u_q}$.

From these normalized parameters, two differential estimators can be derived, the mean curvature (also being the inverse radius of the sphere), and the normal vector:

$$\bar{\mathbf{n}}(\mathbf{x}) = \frac{\mathbf{u}_\ell(\mathbf{x})}{\|\mathbf{u}_\ell(\mathbf{x})\|}\,, \tag{45}$$

$$\bar{\mathrm{H}}(\mathbf{x}) = 2u_q(\mathbf{x}). \tag{46}$$

`Sphere`, `APSS` [GG07], `UnorientedSphere` [CGBG13]

Several variants have been proposed to fit algebraic spheres by optimizing w.r.t. 3D coordinates only [Pra87, GG07], 3D points with oriented normal vectors [GG07, GGG08], and 3D points with non-oriented normal vectors [CGBG13].

More recently, Lejemble et al. [LCBM21] propose a method to compute the Algebraic Shape Operator (ASO), by using the smooth kernel used in the Weighted Least Squares algebraic sphere regression to differentiate the fitted scalar field of the sphere. Since the parameters of the sphere depend on $\mathbf{x}$ through the weighting kernel ω (*cf*. Section 5.2), and the kernel itself is twice differentiable, the gradient is computed as follows:

$$\nabla f(\mathbf{x}) = \nabla u_c + \mathbf{u}_\ell + \nabla \mathbf{u}_\ell^T \mathbf{x} + 2u_q \mathbf{x} + \nabla u_q \mathbf{x}^T \mathbf{x}, \tag{47}$$

and the Hessian matrix:

$$\begin{aligned}\nabla^2 f(\mathbf{x}) = {} & \nabla^2 u_c + \nabla \mathbf{u}_\ell + \nabla \mathbf{u}_\ell^T + \nabla^2 \mathbf{u}_\ell \mathbf{x} + 2\nabla u_q \mathbf{x}^T \\ & + \mathbf{x}^T \mathbf{x} \nabla^2 u_q + 2u_q I_3 + 2\mathbf{x}\nabla u_q^T,\end{aligned} \tag{48}$$

where $\nabla^2 \mathbf{u}_\ell \mathbf{x}$ is the product between the rank-3 tensor $\nabla^2 \mathbf{u}_\ell$ and the 3D point $\mathbf{x} = [x\ y\ z]^T$, giving $\nabla^2 \mathbf{u}_\ell \mathbf{x} = x\nabla^2 \mathbf{u}_{lx} + y\nabla^2 \mathbf{u}_{ly} + z\nabla^2 \mathbf{u}_{lz}$.

The curvatures of ASO are estimated directly from the Weingarten Map W (see Equation 7) of the implicit surface, which is computed as follows:

$$\bar{\mathbf{n}}(\mathbf{x}) = \frac{\nabla f(\mathbf{x})}{\|\nabla f(\mathbf{x})\|}\,, \tag{49}$$

$$\bar{\mathrm{W}}(\mathbf{x}) = \mathrm{P}^T(\mathbf{x}) \frac{\nabla^2 f(\mathbf{x})}{\|\nabla f(\mathbf{x})\|} \mathrm{P}(\mathbf{x})\,, \tag{50}$$

`ASO`

with P being the transfer matrix between the 2D tangent plane and the 3D space, as in Equation 10. The authors [LCBM21] also provide proofs of convergence and stability for $\bar{\mathrm{H}}$ and $\bar{\mathbf{n}}$ in the oriented case [GGG08], as well as for other geometric features computed from the sphere regression [NGB*12].

4.2.4. 3D algebraic surfaces

Some approaches have also been proposed to fit algebraic surfaces of higher degrees [Tau93, DB02, KG18], which provides analytic access to its derivatives and the Weingarten map. Alternatively, specific methods focus on fitting quadrics to oriented points [Š04, BBN*20], although they typically require manual parameter setting. In this survey, we evaluate the estimator of W proposed by [DB02], denoted as `3DQuadric`. The definition of this estimator w.r.t. the surface formulation is given in Section 2 of the supplementary document.

4.3. Measure theory approaches

A third category of estimators draws upon concepts from Measure Theory. Unlike local surface fitting or direct differential estimation, these approaches define geometric quantities through integration over local spatial domains rather than pointwise differentiation. By modeling the point cloud as a discrete measure or a generalized surface (such as currents or varifolds), these methods allow for rigorous stability proofs, ensuring consistent curvature estimation even in the presence of noise or when the underlying geometry lacks smoothness (*eg*. near sharp features). The following sections review three prominent families of such estimators: those based on Voronoi diagrams, Normal Cycles and Varifolds. Finally, we briefly mention alternative approaches based on implicit representations.

4.3.1. Voronoi covariance measure

In their seminal paper, Amenta and Bern [AB98] demonstrated that in the absence of noise, if $\mathcal{P}$ samples a smooth surface densely enough, its Voronoi cells will be elongated in the direction of the normal to the surface. This led to normal vector estimators based on either extremal vertices of each Voronoi cell (poles) [AB98], or its principal component analysis [ACSTD07] for better robustness to noise. Mérigot et al. [MOG11] further extend this approach considering convolved covariance matrices of Voronoi cells from which curvature information are obtained. A key contribution of such Voronoi Covariance Measure (VCM) method is that stability results can be obtained on the curvature estimation even with mild assumptions on $\mathcal{P}$ (Hausdorff approximation of the smooth manifold). However, as evaluated in [LCBM21], the error of the VCM estimation remains too high on real data in comparison with other approaches. By replacing the Euclidean metric involved in the Voronoi diagram construction (*eg*. with a distance to a measure), Le Cuel et al. [CLMT14] demonstrate that additional robustness to outliers can be obtained for normal vector estimation.

4.3.2. Corrected normal current

As an extension the normal cycle approaches [Win82, Fu94, CSM03] designed for triangulated surfaces, corrected curvature measures have led to a new family of curvature estimators on polygonal surfaces [LRTC20, LRT22]. The core idea is to uncouple geometry and tangent bundle information using a Grassmannian embedding on $\mathbb{R}^3 \times \mathcal{S}^2$ of the surface elements, leading to close form formulas for the interpolated area and corrected curvature measures (mean, Gaussian and anisotropic measures) for a triangle τ with vertices $(\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k)$ and normals $(\mathbf{n}_i, \mathbf{n}_j, \mathbf{n}_k)$:

$$\begin{aligned}
\mu_\tau^{(0)} &= \frac{1}{2}\,\bar{\mathbf{n}} \cdot \left((\mathbf{p}_j - \mathbf{p}_i) \times (\mathbf{p}_k - \mathbf{p}_i) \right), \\
\mu_\tau^{(1)} &= \frac{1}{2}\,\bar{\mathbf{n}} \cdot \left((\mathbf{n}_k - \mathbf{n}_j) \times \mathbf{p}_i \right. \\
&\qquad \left. + (\mathbf{n}_i - \mathbf{n}_k) \times \mathbf{p}_j + (\mathbf{n}_j - \mathbf{n}_i) \times \mathbf{p}_k \right), \\
\mu_\tau^{(2)} &= \frac{1}{2}\,\mathbf{n}_i \cdot (\mathbf{n}_j \times \mathbf{n}_k), \\
\mu_\tau^{(\mathbf{XY})} &= \frac{1}{2}\,\bar{\mathbf{n}} \cdot \left((\mathbf{Y} \cdot (\mathbf{n}_k - \mathbf{n}_i))\,\mathbf{X} \times (\mathbf{p}_j - \mathbf{p}_i) \right. \\
&\qquad \left. - (\mathbf{Y} \cdot (\mathbf{n}_j - \mathbf{n}_i))\,\mathbf{X} \times (\mathbf{p}_k - \mathbf{p}_i) \right),
\end{aligned}$$

where $\bar{\mathbf{n}} = \frac{1}{3}(\mathbf{n}_i + \mathbf{n}_j + \mathbf{n}_k)$, and $\mathbf{XY}$ being pairs of the basis vectors $\{\mathbf{e}_i\}_{1..3}$. Curvatures are obtained by summing these quantities over sampled triangles $\tau \in \mathrm{T}$ around a point:

$$\bar{\mathrm{H}} = \sum_{\tau\in\mathrm{T}} \mu_\tau^{(1)} / \sum_{\tau\in\mathrm{T}} \mu_\tau^{(0)}, \tag{51}$$

$$\bar{\mathrm{K}} = \sum_{\tau\in\mathrm{T}} \mu_\tau^{(2)} / \sum_{\tau\in\mathrm{T}} \mu_\tau^{(0)}, \tag{52}$$

`AvgHexagram` (for T as in Figure 4)

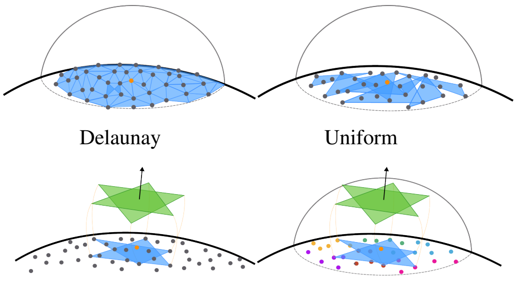
From the per triangle anisotropic measure $\mu_\tau^{(\mathbf{XY})}$, one can define the 3×3 tensor $\tilde{\mathrm{W}}_3$ such that

$$\left(\tilde{\mathrm{W}}_3\right)_{ij} = \sum_{\tau\in\mathrm{T}} \mu_\tau^{(\mathbf{e}_i, \mathbf{e}_j)}, \tag{53}$$

which converges to the second fundamental form. Once symmetrized (see [LRTC20]), for some big constant C, principal curvature values and directions can be estimated from eigenvalues and eigenvectors of the following operator

$$\bar{\mathrm{W}}_3(\mathbf{x}) = \frac{1}{\sum_{\tau\in T} \mu_\tau^{(0)}} \left(\frac{1}{2}(\tilde{\mathrm{W}}_3 + \tilde{\mathrm{W}}_3^T) + C\,\bar{\mathbf{n}}\bar{\mathbf{n}}^T \right). \tag{54}$$

`AvgHexagram` (for T as in Figure 4)


Delaunay Uniform

Hexagram AvgHexagram

Figure 4: *Triangle sampling strategy for the corrected normal current approach: using a local Delaunay reconstruction in the neighborhood (`Delaunay`), a random sampling of triangles (`Uniform`), using only two triangles and 6 probes of $\mathcal{P}$, without (`Hexagram`) or with averaging closest normal vectors (`AvgHexagram`).*

As described in the original paper, the estimated $\bar{\mathrm{W}}_3(\mathbf{x})$ is forced to be aligned with the normal of the input point, thus, `AvgHexagram` is not a normal estimator.

Lachaud et al. [LCL*23] further extend this approach to point clouds using various deterministic or stochastic strategies to define the set of triangle T to consider (see Figure 4). Theoretical stability results provide pointwise accuracy of the curvature estimates even in the presence of noise in point position and normal vector.

4.3.3. Varifolds

A varifold [Alm65] is a generalization of the notion of differentiable submanifolds and 3D surfaces from the viewpoint of geometric measure theory [S*84]. Concepts like length, area, and curvature usually defined for smooth spaces are generalized to irregular sets that lack smoothness and have singularities like junctions. Formally, a d-dimensional varifold in $\mathbb{R}^n$ is defined as a Radon measure on the product space $\mathbb{R}^n \times G_{d,n}$, where $G_{d,n}$ is the Grassmanian (*ie*. the space of all d-dimensional linear subspace in $\mathbb{R}^n$). A 2-varifold in 3D defines a measure on both point coordinates and tangent planes.

Point cloud varifolds have been introduced in the context of surface approximation [BLM17], and definitions for second order properties have been recently provided [BLM22] for oriented point clouds. The proposed method needs an estimate of the oriented normal vector $\mathbf{n}$ at $\mathbf{x}$ to first compute W_3 as

$$\bar{\mathrm{W}}_3 = W^{\perp} * \mathbf{n}, \tag{55}$$

`Varifolds`

with $*$ the tensor-vector product such that $(\bar{\mathrm{W}}_3)_{ij} = \sum_k W^{\perp}_{ijk}\mathbf{n}_k$. Similarly to `AvgHexagram`, this method does not estimate normal vectors. Then, they estimate curvatures using Equations 7 and 11. The $3 \times 3 \times 3$ tensor $W^{\perp}$ is called the *approximate orthogonal weak second fundamental form* [BLM22, Equation 7.2], defined as:

$$\begin{aligned}
W^{\perp}_{jkl} = &\frac{r}{\sum_{i\in\mathcal{N}_\mathbf{x}} w_i \|\mathbf{x} - \mathbf{p}_i\|} \sum_{i\in\mathcal{N}_\mathbf{x}} w_i \left(\mathrm{P}_{\mathbf{n}_i} \frac{\mathbf{x} - \mathbf{p}_i}{\|\mathbf{x} - \mathbf{p}_i\|} \right) \\
&\cdot \left((\mathrm{P}_{\mathbf{n}_i} - \mathrm{P}_\mathbf{n})_{kl} e_j + (\mathrm{P}_{\mathbf{n}_i} - \mathrm{P}_\mathbf{n})_{jl} e_k - (\mathrm{P}_{\mathbf{n}_i} - \mathrm{P}_\mathbf{n})_{jk} e_l \right)
\end{aligned} \tag{56}$$

where $\mathrm{P}_\mathbf{n} = \mathrm{I}_3 - \mathbf{n}\mathbf{n}^T$ is the orthogonal projection matrix on the plane defined by $\mathbf{n}$, e_i is the i^{th} canonical basis vector of $\mathbb{R}^3$, and w_i is computed using the derivative of the bump kernel $\Omega(x) = \exp(-\frac{x^2}{1-x^2})$ (see Section 5.2 for the definition and usage of kernels).

4.3.4. Alternative implicit representations

Differential quantities can be estimated from an implicit representation of the point cloud (see [Gol05] for generic formulas). We do not include in this survey the rich literature on implicit function reconstruction, *eg*. distance fields or signed distance fields [BDS*18, FC24]. Note that, as previously mentioned and experimented (see Section 6), many signed estimators degrade significantly with inconsistent normal orientation. Strategies to sign distance fields ([BDS*18]) could also be considered to achieve a globally consistent orientation [XDW*23].

4.4. Deep learning

While the methods discussed in the previous sections rely on explicit geometric priors, deep learning approaches infer geometric properties from extensive training sets. Instead of hand-crafting features to define local geometry, these methods leverage neural networks to learn complex features directly from examples.

This offers several advantages: deep learning models often demonstrate superior robustness to noise and outliers, effectively learning to "denoise" geometry in ways that are difficult to formalize analytically. However, this often comes at the cost of precision and generalization: models trained on synthetic datasets often fail to preserve their accuracy on real-world acquisitions. Furthermore, unlike measure-theory approaches (Section 4.3) which provide proofs of convergence, deep learning models often lack theoretical guarantees regarding their stability on unseen data. Given these differences, and to ensure fair comparison over the methods' capabilities without extensive retraining, we report the experimental evaluation of learning-based approaches in the supplementary material.

Interestingly, recent works show a convergence between these two worlds [BSG20, ZLD*21, LZW*22, ZJW*23]: rather than purely regressing geometric values, many modern architectures are designed to enhance classical pipelines, *eg*. by learning optimal weights for a polynomial fit (Section 4.2.2.3), rather than replacing the fit entirely. In this survey, we categorize these methods into regression-based approaches, weight estimation and local surface estimation models.

4.4.1. Regression-based model

Boulch et al. [BM16] utilize a Hough transform-based representation of local shape patches combined with a convolutional neural network to learn normal directions. This method, however, is constrained by its reliance on a 2D-based learning representation and a loss of orientation information. The PCPNet approach [GKOM18] uses the PointNet architecture [QSMG17] in a local patch-based multi-scale form to estimate consistently oriented normals from point clouds, leveraging both local and global information. A similar approach has been proposed by Zhou et al. [ZHLL20], using a feature constraint mechanism called Local Plane Features Constraint. More recently, Yi *et al.* [YdSEZ*24] propose to estimate normals by distinguishing between sharp and non-sharp feature points. PointNorm-Net [ZNZ*25] introduces a multi-modal normal distribution estimation paradigm in a self-supervised manner. This approach addresses the limitation of supervised models trained on synthetic datasets that perform poorly on real-world point clouds.

4.4.2. Weight estimation model

Some deep learning methods are dedicated to estimate the weights $\{w_i\}$ of the point set instead of using the commonly used kernels presented in Section 5.2. The learned weights are later used to fit planes [LOM20] or n-Jets [BSG20, ZLD*21, LZW*22, ZJW*23] as described in Sections 4.2.1 and 4.2.2.3. Some of them also estimate offset points [ZLD*21,LZW*22,ZJW*23] to increase the consistency of the polynomial order by adjusting the point distribution. Zhang et al. [ZCZ*22] propose to directly estimate normals according to a regression model that takes as input an optimized point set provided by a weight learning model. Zhou *et al*. [ZLW*24] identify critical points through a hybrid local–global framework that geometrically represents the latent tangent plane, thereby enhancing surface fitting and normal estimation.

4.4.3. Local surface estimation model

Cao et al. [CZB*22] propose to estimate planes using a differentiable RANSAC model called DSAC. Other methods use models to fit hypersurfaces in the feature space, resulting in unoriented estimated normals [LLC*22] or oriented normals [LFS*23]. The NeAF model [LZM*23] predicts a neural angle field to estimate more accurate and robust normal vectors using a learned implicit angle function. Recently, IGF-Fit model [LSY24] introduces a learning-based framework that estimates unoriented normals by leveraging the derivatives of an implicit surface fitted to local patch neighbors.

4.4.4. Curvature estimation

Two of the methods discussed earlier, PCPNet [GKOM18] and DeepFit [BSG20], are also dedicated to estimating curvatures. PCPNet uses the PointNet architecture to directly estimate both normals and curvatures. DeepFit, on the other hand, employs an n-order Jet of weighted points to fit local surfaces, allowing for the estimation of various geometric properties, including curvatures, without the need for predefined scales and weights.

5. From theory to practice

In this section, we review the practical concepts and tools that need to be considered to implement the differential estimators reviewed in this survey. Specifically, we focus on neighborhood collection (Section 5.1), on the neighborhood weighting functions (Section 5.2), and on the use of iterative schemes to regularize the estimations (Section 5.3).

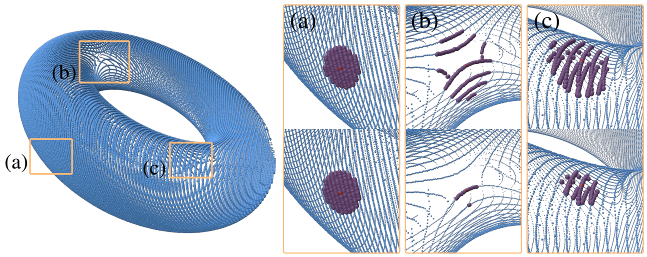


Figure 5: *Impact of the local sampling pattern on the neighborhood collection: for three query points (a), (b) and (c), we construct $\mathcal{N}_{\mathbf{x}}$ either from a kNN search with $k = 200$ neighbors (first row), or from a radius search with $r = 0.05$ (second row). The point cloud is normalized to fit in $[-1,1]^3$.*

5.1. Neighborhood queries

We identify two important aspects related to the definition of $\mathcal{N}_x$ on large point cloud $\mathcal{P}$: the query type (by number of neighbors k or by radius r, see Figure 5), and the spatial data structure used to speed up computations.

5.1.1. Types of query and data structure

A first approach consists in collecting the k-nearest neighbors to $\mathbf{x}$. A strong benefit of this approach is to generate neighborhoods of constant size k, a good property for parallel processing and deep learning architectures. On the other hand, as illustrated in Figure 5, variation of density affects the size of the coverage of the k-neighborhood. Anisotropic sampling patterns, such as those generated by LIDAR devices, require to increase the number of neighbors to avoid degenerated cases. Outlier points can also be connected to other points while being very far away (a property used by some outlier detection algorithms [Sot07]).

Alternatively, one can define $\mathcal{N}_x$ by collecting the points within a ball of radius r centered at $\mathbf{x}$. This type of query is independent of the sampling and naturally implements the scale r used in convergence proofs (*eg*. Equation 14). Also, as r is defined as a length, it can be set according to the scene properties, especially when the coordinates scale is known, *eg*. when acquired with LIDAR sensors. On the downside, collecting neighborhoods with large radius on highly sampled point clouds might lead to very large collections with varying sizes, affecting further processing. This can be solved by using multi-resolution representations [LMBM20].

Each query type can be efficiently implemented using dedicated spatial data structures such as partitioning trees (*eg*. kd-trees [Ben75] or octrees [Mea82]). Once constructed, the expected complexity of a nearest neighbor search becomes $O(\log|\mathcal{P}|)$ for fixed dimension d. For large sets of neighborhood points, implementations of approximated nearest neighbors techniques could also be considered [ML09].

5.1.2. Neighborhood graphs

So far, the data structure and neighborhood queries have been considered in ambient space $\mathbb{R}^3$, which could be problematic when processing thin-sheet objects. Ideally, when the point set samples an unknown 2-d surface embedded in $\mathbb{R}^3$, we may be interested in a definition of neighborhood that would match the geodesic metric of the surface. One way to approximate this structure is to consider a graph structure: from a kNN or radius search (with small k or radius values) in $\mathbb{R}^3$, we connect each point to its k closest neighbors. Then, larger neighborhood sets are given from a breadth first propagation from the query point of the graph, at the cost of more expensive neighbor queries and higher memory requirements than spatial partitioning trees. Alternatively, other types of graph exist such as the Riemannian graph [HDD*92], the Gabriel graph [GS69, PCBS16], or the Delaunay triangulation [MOG11, PRLH*22].

In the following experiments, only the radius search strategy is considered for simplicity in comparing estimators, except for the comparison with deep learning techniques (see Table 1 in the supplementary document).

5.2. Weighting function

As mentioned in Section 3.2, the importance of each neighbor of $\mathbf{x}$ is usually measured by a weighting function $\omega_{\mathbf{x}}(\mathbf{p}_i)$. A standard approach used in surface reconstruction [BTS*17] is to decrease the weight of the neighbors $\mathbf{p}_i$ according to their distance from the evaluation point $\mathbf{x}$, by defining $\omega_{\mathbf{x}}$ as a monotonic radial decreasing function with compact support:

$$\omega_{\mathbf{x}}(\mathbf{p}_i) = \begin{cases} \Omega\left(\frac{\|\mathbf{p}_i - \mathbf{x}\|}{r}\right) & \text{if } \|\mathbf{p}_i - \mathbf{x}\| \leq r, \\ 0 & \text{otherwise} \end{cases} \tag{57}$$

with r the evaluation radius, and $\Omega(t)$: $[0:1] \rightarrow [0:1]$ a weighting kernel. The use of ω_x has several objectives. First, the compact support of $\omega_{\mathbf{x}}$ enforces locality and allows the use of accelerating data structures for efficiency purpose (*cf*. Section 5.1). Secondly, it increases the spatial coherence of the estimator and its robustness to noise and outliers. Finally, it allows for estimating second-order differential properties thanks to kernel differentiation, as done by `ASO` and `Varifold`. These estimators indeed require a non-null twice continuously differentiable weighting function.

In the literature, many different kernels Ω have been used, as illustrated in Figure 6 and Table 2. According to our experiments, this choice is actually not critical for the accuracy of the estimation. Several other aspects can be taken into account when choosing a kernel. With Moving Least Squares (MLS) approaches like `APSS` and `ASO`, the continuity of the reconstructed surface directly depends on the continuity of $\omega_{\mathbf{x}}$ as discussed in Section 5.3). In this context, other kinds of functions exist that ensure interpolation [AA09] or that control approximation smoothness [Wen95]. The weight function is also the central element of robust iterative methods such as those based on M-estimation [KSNS07], and Iterative Reweighting Least Squares [ÖGG09]. These methods usually adjust the weights during several iterations over the neighborhood. Recently, Mercier *et al*. [MLR*22] introduced an approach to control the amount of approximation using a level-of-details approach incorporated in the MLS framework. Note that the weighting function can also include user-defined weights per points based on density estimation [CPJG09], and other points attributes such as colors and measurements uncertainties.

5.3. Iterative schemes

The Moving Least Squares (MLS) [CWL*08] are an approximation framework popularized in the graphics community by Levin [Lev98, Lev04]. In Geometry Processing, Point Set Surfaces [ABCO*03, AK04] approximate the underlying surface of a set of scattered points using the MLS. Tey et al. [TSA*21] overview several variants of the MLS, focusing on improvements in weighting functions [AD01, MB08, LWK12, ZFH20], discrete norms [KL06, LLY14, Lev15], iterative schemes [AR00, FCOS05, FZ07, LSY17, ZFH20], and hybrid strategies [GNNB08, DSP16, LL20] to overcome its limitations while processing discontinuous fields.

Given an evaluation point $\mathbf{x}$, the point cloud $\mathcal{P} = \{\mathbf{p}_i\}_{i\ldots n}$, oriented or not, and $\mathcal{N}_{\mathbf{x}} \subset \mathcal{P}$ the weighted neighborhood of the input $\mathbf{x}$, a geometric primitive is locally fitted through a weighted *least squares* minimization. The evaluation point is then projected onto this newly fitted primitive, and a new weighted neighborhood is collected from this new position. From this, it is possible to repeat this operation either until convergence [Lev04], or until an arbitrary number of times. It defines the projection operator onto the

Methods	Og. Kernel	Exp. Kernel
`Mean` [PWY*07]	constant	quartic
`Cov2D` [BC94, DM14]	constant	quartic
`NormCov2D` [BC94, DM14]	constant	quartic
`NormCov3D` [LT90, DM14]	constant	quartic
`ShapeOperator` [KSNS07]	constant	quartic
`PCA` [HJ87]	constant	quartic
`2-Monge` [Gra98]	quartic	quartic
`PC-MLS` [RGRG15]	quartic	quartic
`JetFitting` [CP05]	constant	constant
`WaveJets` [BDC18]	exponential	quartic
`Sphere` [Pra87]	constant	quartic
`APSS` [GG07]	quartic	quartic
`UnorientedSphere` [CGBG13]	quartic	quartic
`ASO` [LCBM21]	quartic $\star$	quartic $\star$
`3DQuadric` [DB02]	constant	quartic
`Varifolds` [BLM22]	bump $\star$	bump $\star$
`AvgHexagram` [LCL*23]	constant	constant

Table 2: *Weighting kernels (or their derivatives $\star$) used in the original publication (Og.) and used in our experiments (Exp.). Note that a 'constant' kernel refers to uniform weighting ($w_i = 1$) for all neighbors.*

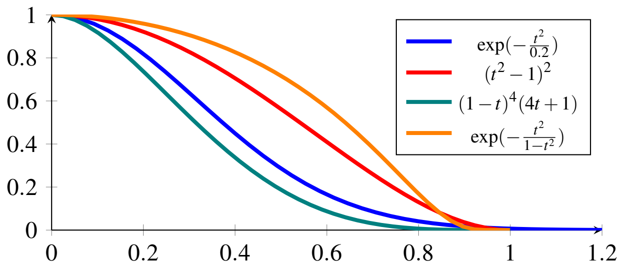


Figure 6: *Standard weighting kernels Ω: Gaussian, quartic, Wendland [Wen95], and bump [BLM22].*

MLS surface [AA04]. The radius r of the neighborhood controls the approximation, producing smooth surfaces in any case (noisy point clouds), and the level of continuity of the resulting MLS surface is determined by the level of differentiability of the weighting kernel $\Omega(t)$ [Lev98] at the neighborhood border (*eg*. when $t \to 1$, see Equation 57).

Alternative approaches based on robust statistics [SDC*20, WZG*23] have also been proposed to optimize the estimators parameter setting (*eg*. neighbors filtering).

6. Experimental settings

The goal of this work is to provide a comprehensive comparison of differential estimators taking as input 3D point clouds. In addition to the theoretical comparisons, we also developed a practical benchmarking tool allowing to quantitatively evaluate and compare the estimators.

Replicability. In order to be able to replicate (according to the recommendations of [BCDM20]) and pursue this work with upcoming research outcomes, the following open-source material are available online[§]:

- our implementation of the estimators (Table 1)) as a C++ library,
- the collection of 3D point clouds with ground truth values, as detailed in Section 6.2,
- the benchmark *per se*: a set of tools to run the estimators and compute the metrics according to the ground truth values as detailed in Section 6.5,
- an online website providing access (visualization and download links) to the experimental results detailed in section 7 (see online website [¶]): values estimated for each 3D point cloud (per scale, method, ...), raw scores for all the experiments, and tools to aggregate and compare the estimator behaviors. The website will also provide high-resolution versions of the images, as well as interactive 3D viewers and plots visualizers.

As a long-term objective, we would like to extend the website to allow for submissions from the community, such that upcoming estimators could be compared and publicized against those evaluated in this study.

Performance reporting. All the experiments have been ran using an Intel Core i9-12900 2.40Ghz with 64GB RAM. All the estimators are implemented in C++ and run on the CPU, and neighborhood queries were sped up using KdTree [MLG*20] (see Section 6.4). For some experiments we report performances of learning-based methods by using metrics computed in previous studies, as we failed at reproducing the expected performances of these approaches on our dataset.

6.1. Estimators

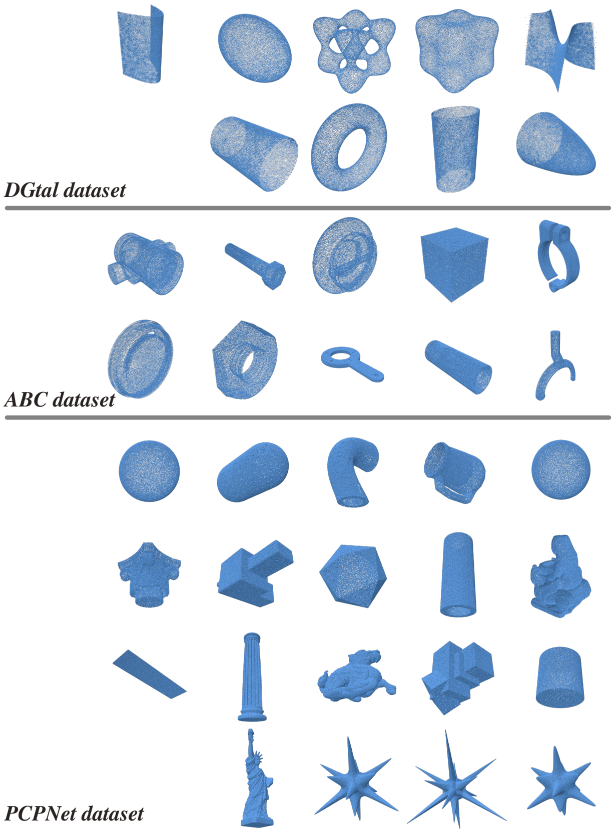
Our benchmark implements the approaches introduced in Table 1. We standardized the weighting strategy by using a `quartic` kernel for all applicable methods (see Table 2). Exceptions include `JetFitting`, where the reference implementation [PC24] enforces a constant weight, as well as `Varifolds` and `AvgHexagram`, where the weighting scheme is intrinsic to the method's formulation. To ensure fair comparisons, we compute the estimator without considering iterative or robust fitting techniques (as in [KSNS07] for `ShapeOperator` or in APSS [GG07]): for each point, we collect the neighborhood, run the estimator once, and report the estimated values.

For the approaches that require an input tangent plane (`2-Monge`, `PC-MLS`, `JetFitting`, `WaveJets`), we use the estimation produced by `PCA`, as it is usually done in previous work.

The `NormCov3D` and `Varifolds` approaches estimate W_3. In

[§] https://github.com/STORM-IRIT/pcloud-differential-estimation-benchmark

[¶] https://storm-irit.github.io/pcloud-differential-estimation-benchmark-website/



Dataset	#Shapes	Type of perturbation	#variants
DGtal dataset	9	P, N, F, O, H	486
ABC dataset	10	N, H	120
PCPNet dataset	19	P, G/S	108
Total			714

Figure 7: ***Illustration of the datasets.*** *The DGtal dataset is synthesized using the DGtal library [DGt], with each shape sampled with 50,000 points. The ABC dataset shapes are given by Poisson disk sampling of some ABC meshes. Several perturbations are applied to the datasets for experimentation purposes, including positional noise (P), normal noise (N), normal flipping (F), outliers (O),* LIDAR*-like density variations using Helios++ (H), and density variation included in PCPNet dataset (G/S, gradient or stripped).*

order to compute W using Equation 10, we estimate the local frame using Equations 26, 27 and 28, and flip the estimated normal to match the input normal orientation. From our experiments, `Norm-Cov2D` (which is also expressed relatively to the covariance plane) gives the same estimation results than `NormCov3D` in this setting. Hence, to improve readability, we report values for `NormCov3D` only in the website.

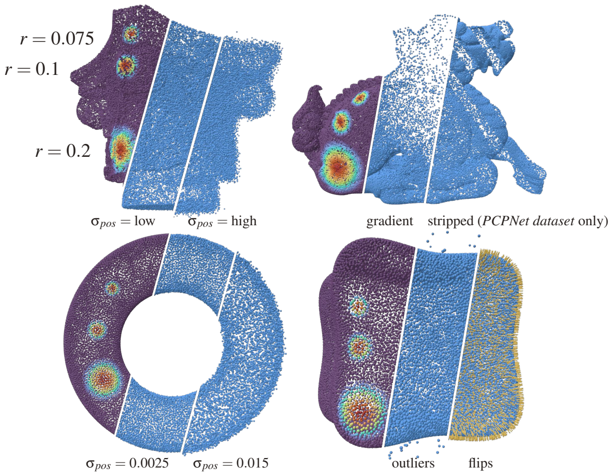


Figure 8: *Examples point clouds with various noise models, and various radius search parameters r (shapes from the DGtal dataset and PCPNet dataset).*

6.2. Datasets

An illustration of the datasets is provided in Figure 7. To ensure consistency, all shapes presented below are scaled to fit in a $[-1,1]^3$ domain. For datasets with some noise perturbation or resampling strategies, the scaling ratio is determined based on the noise-free version and lowest sampled version. Our evaluation database is composed of analytic shapes on which ground-truth differential properties are known:

- **Implicit shapes.** The DGtal [DGt] dataset is the base of our benchmarking tool as it provides sampling features of implicit polynomial surfaces while providing all the differential properties from the implicit formulation. The initial samples are obtained using a Lloyd relaxation algorithm restricted to the zero level set of the implicit functions. Each point set $\mathcal{P}$ is equipped with unit normal vectors, unit curvature directions, and curvatures computed from the polynomial implicit representation.
- **PCPNet dataset.** Given its widespread use in deep learning, our goal is to evaluate the performance of methods against the test set of PCPNet shapes [GKOM18], which comprises 19 different shapes, with 100,000 points each. For each shape, it provides three levels of noise and two different density types (striped and gradient). Originating from meshed objects, this dataset offers ground-truth normal vectors and mesh-estimated curvatures.
- **CAD objects.** For the estimation of properties on objects, like Computer-Aided Design (CAD) objects, we use the *ABC dataset* [KMJ*19] as point clouds, sampled from the meshes by using a Poisson disk sampling.

To evaluate the robustness to noise, an additive isotropic Gaussian noise is added to the point positions $\{\mathbf{p}_i\}$, with 5 level of standard deviation: $\sigma_{pos} \in \{0.0, 0.0025, 0.005, 0.0075, 0.01, 0.015\}$. An example of such kind of noise is provided in Figure 8. Similarly, noise is added to the input normal vectors $\{\mathbf{n}_i\}$ following an isotropic Gaussian noise with a standard deviation $\sigma_{norm} \in \{0.0, 0.025, 0.05, 0.075, 0.1, 0.15\}$. On the overall dataset, these values experimentally correspond to an average angular deviation of $\{0°, 1.79°, 3.59°, 5.38°, 7.18°, 10.77°\}$ respectively. For stability with respect to normal vector orientation, the data is randomly perturbed by a flipping of the normal vectors, with probabilities $\{0.0, 0.01, 0.05, 0.1\}$. Finally, outliers are generated with a probability in $\{0.0, 0.01, 0.05, 0.1\}$ and Gaussian noise is added to them with a standard deviation in $\{0.1, 0.15, 0.2\}$.

6.3. Sampling and density variations

Poisson disk sampling is a method for generating point distributions on 3D meshes, ensuring that each point is separated by a minimum distance from its neighbors (*eg.* [CJW*09, Yuk15, GYJZ15]). The *ABC dataset* is created using this method, providing a uniform and isotropic sampling of the surfaces, and avoiding clustering in densely triangulated areas.

LIDAR-like density variation. We have considered the Helios++ library [WEW*22] to simulate LIDAR sampling (using high-end device Riegl vz400) on input meshes (see Figure 9). The normal of each point of the output is reconstructed taking the nearest triangle orientation. We use this to assess the robustness of differential properties estimations below the LIDAR anisotropic sampling, while providing the ground truth. For this purpose, synthetic LIDAR sampling has been added to the 9 shapes from the *DGtal dataset* where Poisson surface reconstruction [KBH06] have been applied (using CloudCompare [Clo] implementation). Simulated LIDAR point clouds may thus have artifacts such as varying density and holes. The ten CAD objects of the *ABC dataset* [KMJ*19] have been sent to Helios++.

6.4. Neighborhood queries

In our experiments, we have considered range neighbor queries, *ie.* all the neighbors included in an Euclidean ball of radius r centered at the evaluation point $\mathbf{x}$, such that $\mathcal{N}_\mathbf{x} \in \mathcal{B}_\mathbf{x}(r)$. This allows us to compare the convergence of the estimators when the radius tends to zero, to be as close as theoretical guarantees [LCBM21, DM14, PWHY09]. We have choosen to use different scales to compare the estimators, $\mathcal{N}_\mathbf{x} \in \mathcal{B}_\mathbf{x}(r)$ with $r \in \{0.05, 0.075, 0.1, 0.25, 0.15, 0.175, 0.2\}$

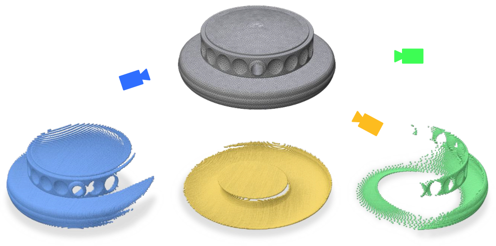


Figure 9: *Illustration of the process using Helios++. The input mesh (top) is placed within a simulated scene, where LIDAR sensors (blue, yellow, and green for the example) scan the environment from different viewpoints, each producing a distinct point cloud. The final point cloud is obtained by merging these outputs.*

6.5. Evaluation metrics

To maintain consistency, the metrics primarily use the root mean square error (RMSE) for each point cloud. In order to ensure fair comparisons between oriented and non-oriented methods, the RMSE is computed between absolute values of the estimation and ground truth values:

$$\mathrm{RMSE} = \sqrt{\sum_n \frac{(|\hat{v}_n| - |v_n|)^2}{n}}, \tag{58}$$

with $|v_n|$ the absolute value obtained using the estimator and $\hat{v}_n$ the associated ground truth. For directional data such as normal vectors and curvature directions, the Root Mean Square Angular Error (RMSAE) is computed in the interval $[0°, 90°]$. In order to avoid errors due to numerical instabilities, we do not evaluate the error on curvature directions for planar and spherical local patches, *ie.* when $|k_1 - k_2| < 1e^{-6}$.

Finally, several methods [RGRG15, Gra98, LCBM21, GG07, CGBG13, ABCO*03] use a projection operator to estimate differential properties at the projected point, even without using an iterative scheme (see discussion in Section 5.3). Since the ground truth is generated before the estimations, the evaluations of a point are compared to the ground truth at the initial point position, even in the presence of noise.

7. Results

In this section, we detail a comparative evaluation of the estimators presented above. A complete analysis is available in the companion website (see supplementary materials). In Figure 10, we present some partial results that will be used for the discussion. More precisely, we illustrate the averaged error measurements for the *DGtal dataset* and *DGtal+Helios dataset*, of the mean curvature estimation $\bar{\mathrm{H}}$, minimum curvature direction $\bar{\mathbf{d}}_{\min}$ and normal vector estimation $\bar{\mathbf{n}}$. We insist on the fact that only partial results are shown hereafter. Performances are highly context dependent. Results on a specific dataset can be obtained on the companion website or reproduced following the provided benchmarking source code. We illustrate qualitative results in Figure 11. In Section 8, we provide a higher-level discussion. A comparison with learning-based approaches on the *PCPNet dataset* is presented in supplementary.

7.1. Performances

All estimators rely on a local patch of points $\mathcal{N}_\mathbf{x}$. Once constructed (see Section 5.1), most estimated quantities are computed in linear time with respect to the size of $\mathcal{N}_\mathbf{x}$ (with an additional eigenvalue decomposition of 3×3 covariance matrices or $\bar{\mathrm{W}}_3(\mathbf{x})$). Only `JetFitting` and `WaveJets` require solving a linear system to compute their fitting. In Figure 12, we detail the average timing for each scale and method.

In the same figure, we also present the results of a numerical precision test for high-performance design on specific devices or GPUs. On our platform, switching the computation scalar type from `double` to `float` has a negligible impact on the computation time (we report timings for the estimators while excluding the neighborhood collection). On the other hand, we observed that

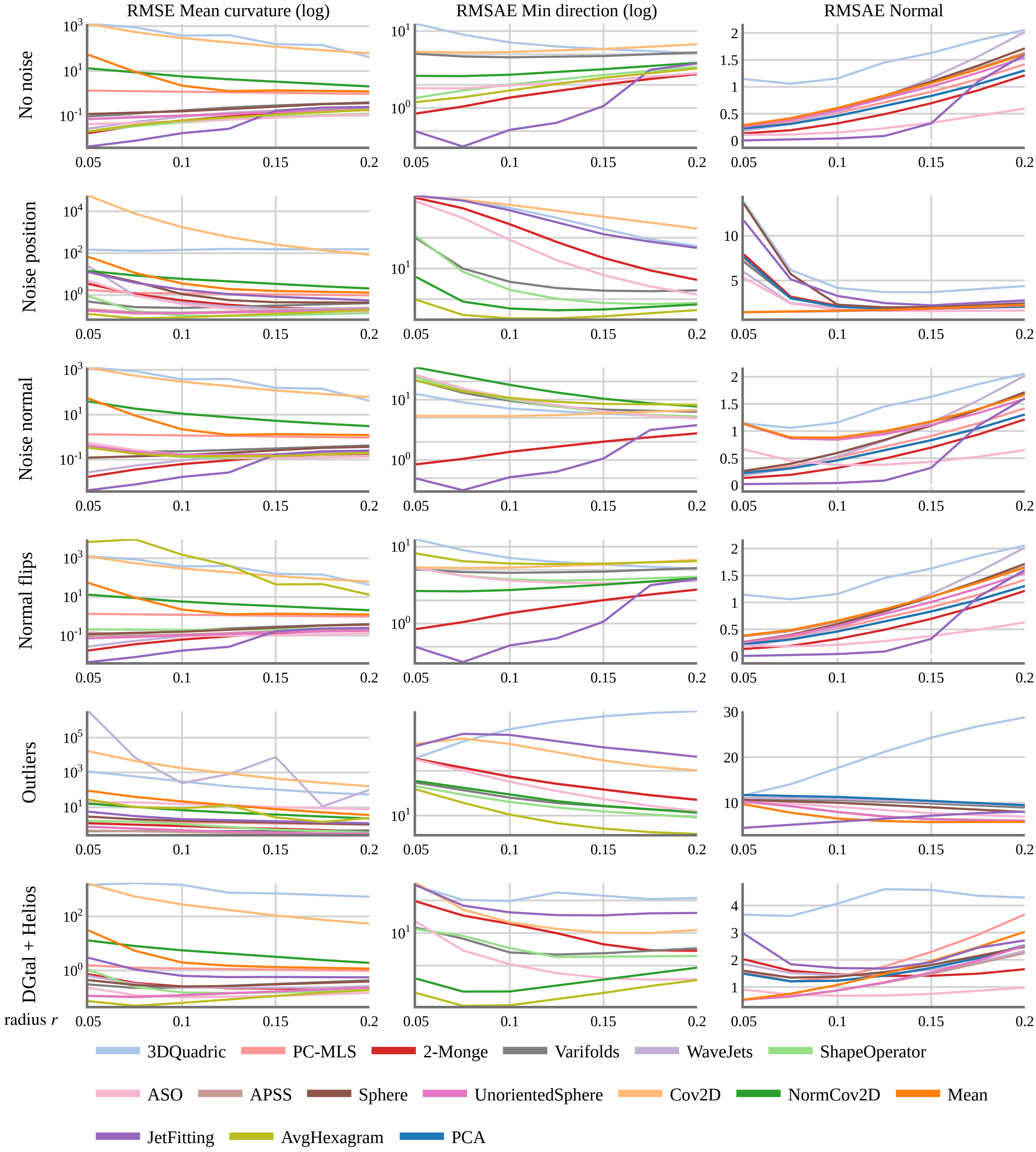


Figure 10: *This figure presents the average RMS errors in (absolute) mean curvature values, minimal curvature direction, and normal vectors, using DGtal dataset. The graphs compare error measurements under various noises at different scales (radius on the abscissa), averaged over all levels of noise. From top to bottom: no noise, noise on position, noise on normal vectors, normals flip, outliers and non-uniform sampled geometry (DGtal + Helios).*

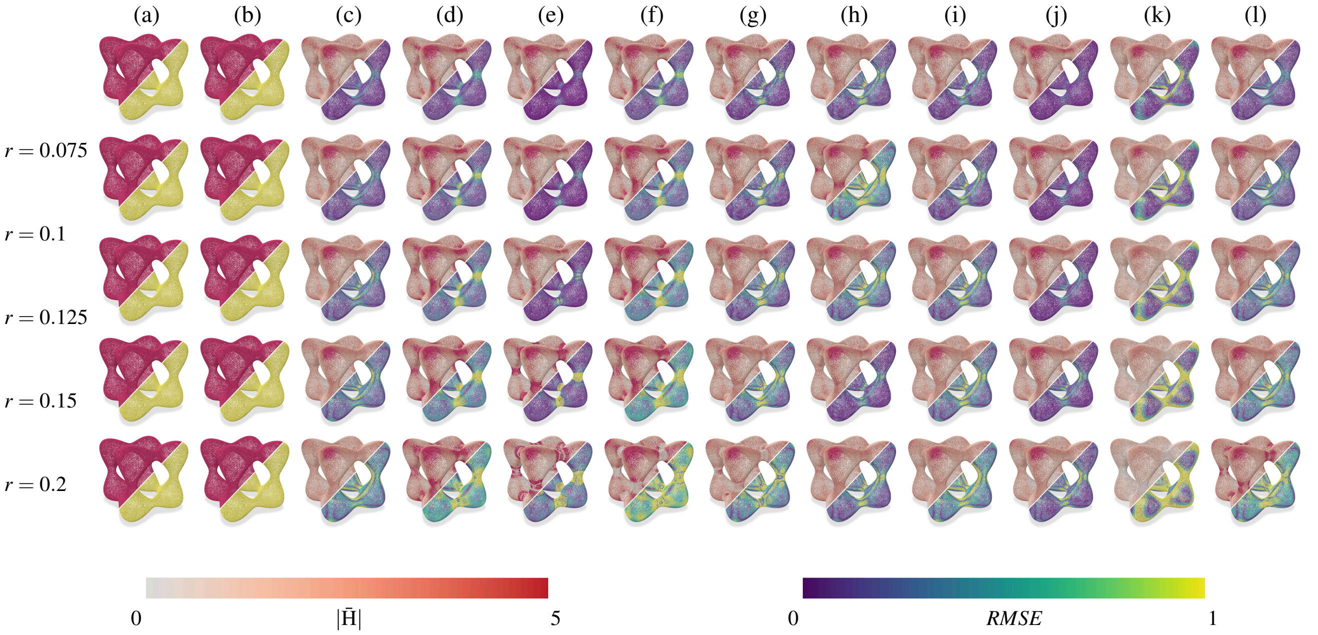


Figure 11: *Visual comparison of mean curvature estimation (top left parts), and errors (bottom right parts) on the* `Goursat-Hole` *shape sampled with 50,000 points (noise-free), for various radii and methods (values are clamped to the* $[0,5]$ *or* $[0,1]$ *intervals respectively): (a)* `Cov2D` *[BC94, DM14], (b)* `NormCov2D` *[BC94, DM14], (c)* `ShapeOperator` *[KSNS07], (d)* `2-Monge` *[Gra98], (e)* `JetFitting` *[CP05], (f)* `WaveJets` *[BDC18], (g)* `Sphere` *[Pra87], (h)* `APSS` *[GG07], (i)* `UnorientedSphere` *[CGBG13], (j)* `ASO` *[LCBM21], (k)* `Varifolds` *[BLM22], (l)* `AvgHexagram` *[LCL*23]*

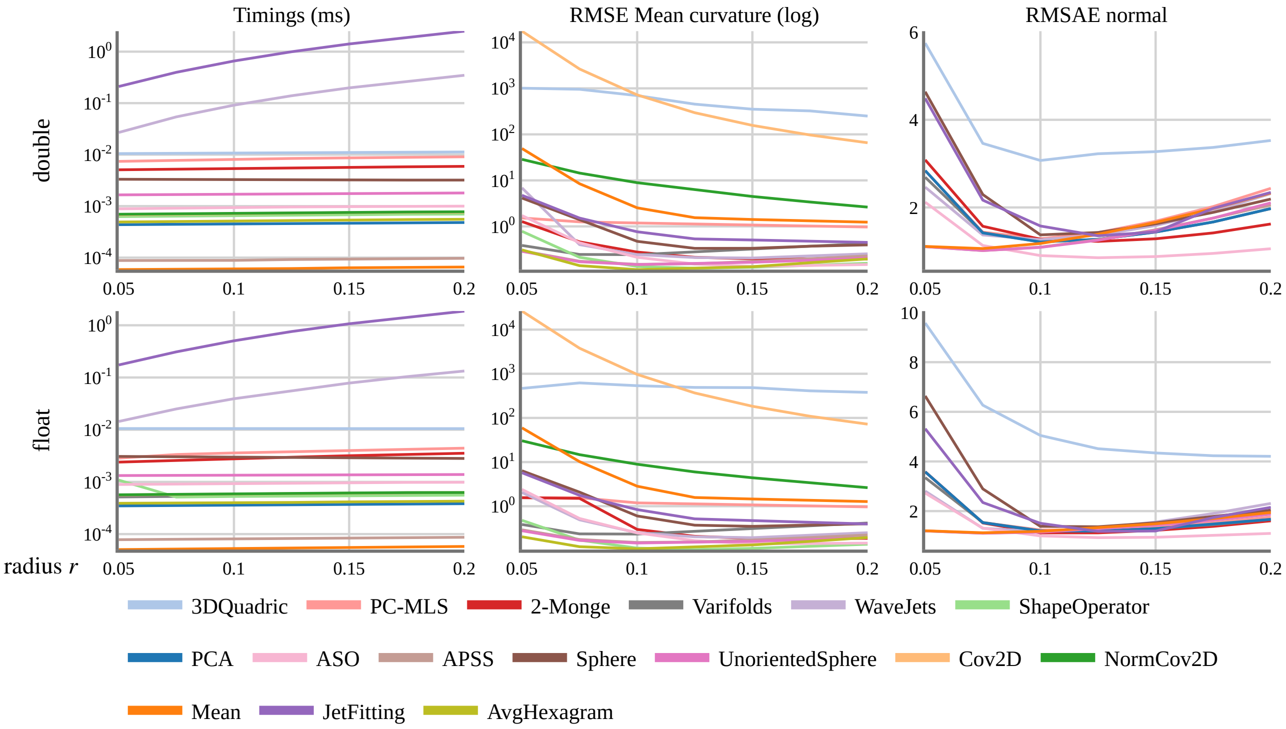


Figure 12: *Left: This figure presents the timings (estimator computation only, excluding* $\mathcal{N}_{\mathbf{x}}$ *construction) using* `double` *(top) and* `float` *(bottom) internal scalar types, averaging at each scale given in abscissa (DGtal dataset with position and normal noise variations). Center and right: Impact of the computation precision on the error of the mean curvature and normal estimators, respectively.*

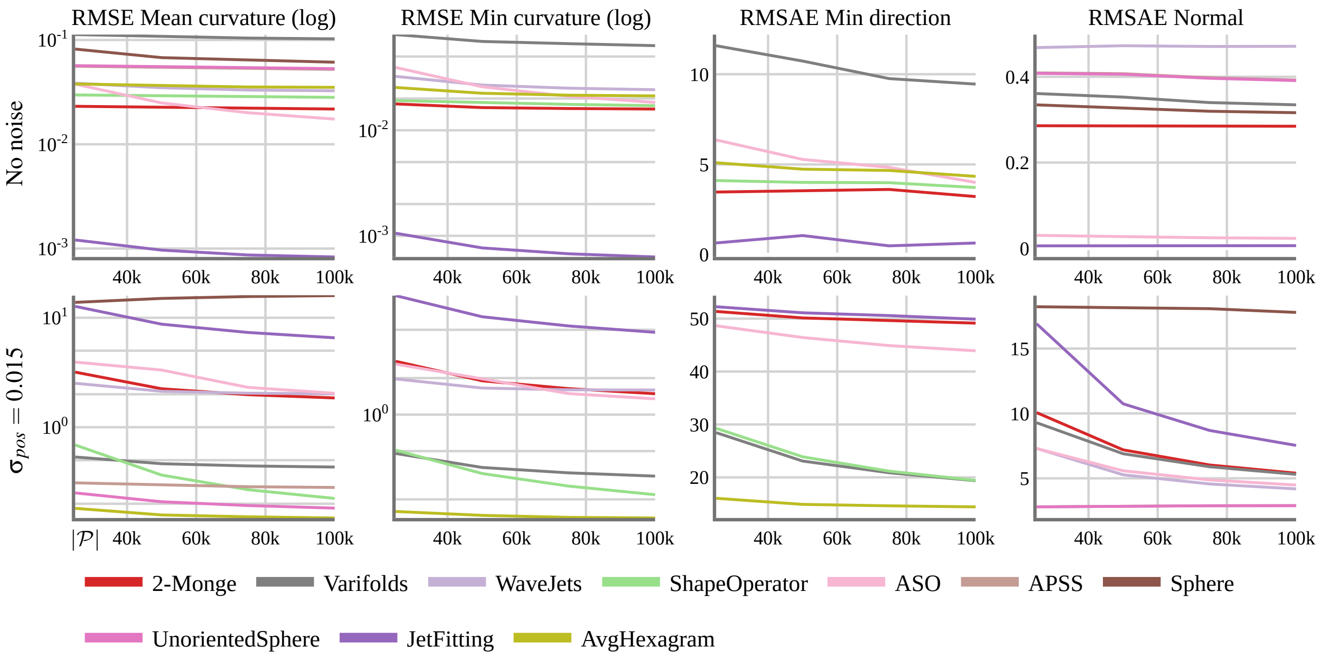


Figure 13: *Convergence experiment: on a smooth implicit surface (*`Goursat` *shape of DGtal dataset, see Figure. 8, bottom right), with a fixed radius* $r = 0.075$ *(without –first row– or with some noise on the positions –second row–). As we increase the number of samples in abscissa, we evaluate the estimation errors (RMSE).*

the estimation of vectors (*eg*. normal or principal directions of curvatures) are more impacted by floating point computations.

7.2. Robustness to noise and sampling

The errors reported in Figure 10 show that methods tend to be affected differently by noise on position and normal vectors. As a general comment, local surface models (Section 4.2) lead to more stable estimations. `AvgHexagram` also has good performances but is more sensitive to normal vector perturbations (normal flips). As expected, perturbing positions strongly impacts the estimators at small scales (when r is small). Some methods (*eg*. `AvgHexagram` for min curvature direction or `ASO` for normal estimation) produce results comparable to the noise-free situation at larger scales. In contrast, LIDAR sampling affects the estimators both at small and large scales.

7.3. Stability evaluation

In Figure 13, we evaluate the convergence property of some estimators providing stability results. In most mathematical frameworks (*eg*. the integral invariant one, see Section 3.3), the stability results are stated for a radius r tending to zero with increasing density of the local patch. To mimic this behavior, we fix the radius r and increase the number of samples on a smooth manifold. While most methods show some numerical convergence properties, `JetFitting` clearly outperforms the other approaches on noise-free surfaces. When adding some noise (as illustrated in Figure 8), performances of `JetFitting` collapse. In this specific setting, `AvgHexagram` provides a more robust estimation.

7.4. Acquired data

To validate the estimators on real-world raw data, we report qualitative results on two indoor scanned models from the Rohbau3D [RB25] dataset. Figure 14 illustrates the estimations using a fixed neighborhood radius of $r = 0.05m$.

To ensure a fair visual comparison in the absence of ground-truth curvatures, we primarily report results using methods that do not require input normal vectors. We observe that `JetFitting` tends to produce noisy estimations for the mean curvature, likely due to overfitting on the raw sensor noise, contrarily to `PC-MLS` that tends to smooth out the noise on planar regions. The `2-Monge`, `WaveJets`, and `Sphere` methods offer similar visual consistency, balancing noise reduction and feature preservation. For completeness, we also provide results on these indoor datasets using oriented methods in the supplementary material (see Figure 1).

We extend our qualitative evaluation in the supplementary material (see Figure 2 and 3) to four outdoor models sourced from two different datasets (Rohbau3D [RB25] and OpenTrench3D [HJP*24]). For each of them, we report the mean curvature estima-

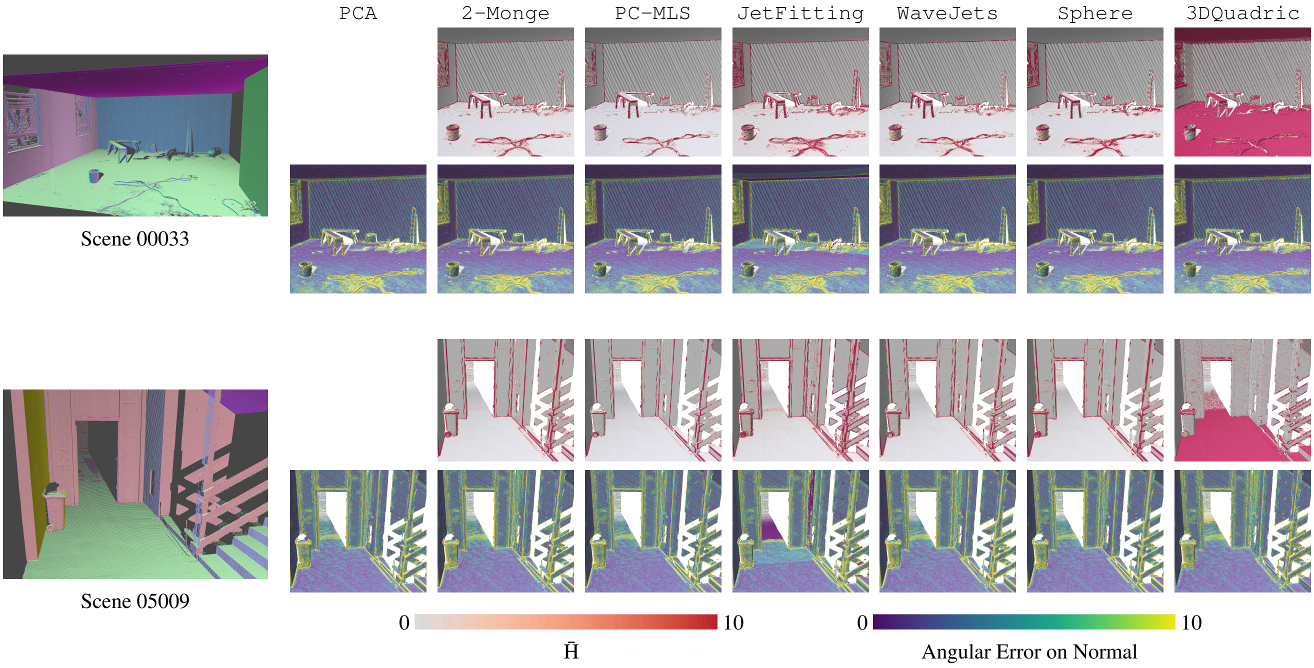


Figure 14: *Mean curvature estimation H̄ and Angular Error (in degrees) on normals on two 3D indoor scanned models from the* `Rohbaud3D` *dataset [RB25] using a radius of* $r = 0.05m$.

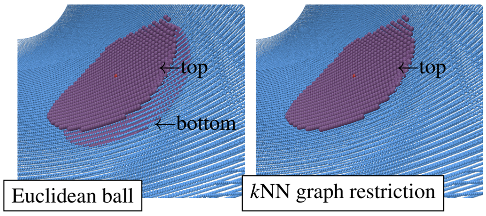


Figure 15: *Influence of the neighborhood selection for close-sheets*

tion on single scale, demonstrating the behavior of the estimators on larger and raw acquisitions.

8. Discussion

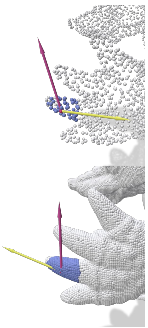
In this section, we discuss a few areas of concern when using a differential estimator in a user-specific setting.

Neighborhood queries and radii. When evaluating differential quantities on the *ABC dataset* (see Figure 15) , poor results can occur due to thin sheet artifacts when using ambient space query strategies (see section 5.1). In such situations, even though there is an associated computational overhead, neighborhood graph strategies should be preferred (see companion website). Concerning the size or radius of the local patch definition, we are facing a classical noise/ratio issue that should be addressed depending on the targeted point clouds. Note that in some previous works, this radius parameter had been used to define a *scale-space* representation of the point cloud differentials [PKG03, NGB*12, LCL*23].

`JetFitting` as a *gold standard*. Since the seminal work of Cazals and Pouget [CP05], and due to its wide adoption in computational geometry libraries such as CGAL [PC24], `JetFitting` has been considered a baseline for many geometry processing pipelines on point clouds. While excellent results can be obtained on noise-free surfaces, our experiments show that even small perturbations in position can lead to significant estimation errors. In that situation, alternatives such as `ShapeOperator` or `AvgHexagram` should be preferred.

Frame computation. As detailed in Section 6.1, Monge-based estimators require an input frame to express the point cloud coordinates as a height-field, which is usually computed using `PCA`. As illustrated in Figure 16, the frame computed using `PCA` may rotate inconsistently depending on the neighborhood size. As a result, the Monge patch is fitted from a frame where the point cloud cannot be expressed as a function of the height. This problem disappears when using estimators working in 3D, such as `APSS`.

Order of the fitted primitive. Cheng *et al*. [CZ09] discussed that a higher order polynomial fitting for $h(u,v)$ will not reduce the errors. In fact, it tends to overfit the data, and induce higher errors in case of noisy data, which is often the case. From our experiments, the sampling density affects `3DQuadric`, which tends to vary w.r.t. the point cloud density, as visible in the right inset.



Errors in *PCPNet dataset* ground truth values. In order to compare the different approaches, we have proposed new datasets with analytical ground truth values. In addition, we also used the

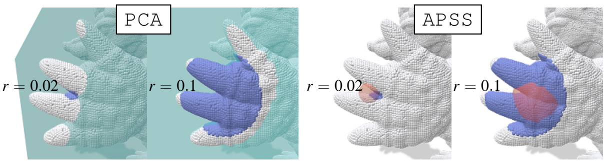


Figure 16: *Local frame computation using `PCA` and `APSS` at different scales.*

PCPNet dataset, which is commonly used in several papers. However, as visible in the accompanying website, we observed that several point clouds have inconsistent ground truth values of curvature for this specific dataset (as provided by the authors). We believe that these artifacts might have significantly affected both the optimization and the evaluation of previous work.

9. Future works

As summarized in the previous sections, the literature on differential estimators for point clouds is extremely rich, both in terms of mathematical approaches and technical details, catering to a wide range of user needs. In this section, we review high-level potential future work.

First, when normal vectors are required, most estimators require a consistent normal bundle. Missing orientation information, or noisy ones (*cf*. the normal orientation flipping experiment of Section 6.2 and Figure 10) may considerably affect the estimator performances. Reorienting consistently a normal vector field is still an active research field (*eg*. [XDW*23]). Following the construction of `UnorientedSphere` [CGBG13], there is room for differential estimators on unoriented normal vectors only, skipping the need of a reorientation preprocessing step.

In this survey, we have only considered explicit and radial smoothing kernels (see Section 5.2). Some deep learning methods [LOM20, BSG20, ZLD*21, ZJW*23, LZW*22] aim at estimating the weights to fit a `PCA` or `JetFitting` for instance. An interesting future work would be to also consider these data-driven weighting techniques in a broader class of local differential estimators.

Such data-driven approaches could also be an interesting way to handle extreme non-uniform and anisotropic density in point clouds, as seen in those captured by LIDAR devices. Stable quantities could be obtained though the weighting function as discussed earlier, or by resampling the point cloud. This would imply an iterative process where the local surface patch is reconstructed while being used for resampling during the estimation.

Considering that point clouds are generated from acquisition devices, another interesting avenue is to incorporate knowledge from the acquisition process in the estimator computation. For instance, Comino *et al*. [CACB18] studied the link between the distance to the sensor location, the neighborhood size, and the estimator error bounds. In the case of mono-photon LIDAR devices, Tachella *et al*. [TAM*19] have demonstrated the local surface priors (*eg*. [GG07]) can be evaluated directly on the acquired photon distribution during the point cloud reconstruction step.

As demonstrated by Himeur *et al*. [HLP*21], using efficient estimators helps to reduce the energy consumption of learning-based classifiers. As point clouds get bigger (modern devices may produce several billions of points during a single acquisition session) and are used in more and more contexts and industries [Dha24] (including mobile devices), we expect that the optimization of the energy consumption of point cloud processing algorithms will gain more attention in the upcoming years.

Finally, this survey mainly focuses on the first and second order differential quantities. In geometry processing, higher order quantities could be of interest for ridge-valley line extraction on surfaces (*eg*. [MBF92, OBS04, GI04, KK06]). While some techniques, such as `JetFitting` [CP05] or `WaveJets` [BDC18] could directly estimate such higher order quantities, a dedicated analysis would be interesting.

10. Conclusion

In this work, we provide a review of the literature on differential estimators for 3D point cloud. In addition to describing the different approaches and their mathematical framework, we provide a practical tool helping practitioners to quantitatively compare existing approaches and future proposals. As detailed in Section 9, the estimation of differential properties is a core topic offering a wide range of perspectives, with a potentially strong theoretical and practical impact.

Acknowledgments

This work is partially supported by the French National Research Agency within the StableProxies (ANR-22-CE46-0006) and SSLAM (ANR-22-CE23-0004) projects.

References

[AA04] ALEXA M., ADAMSON A.: On Normals and Projection Operators for Surfaces Defined by Point Sets. In *SPBG'04 Symposium on Point - Based Graphics 2004* (2004), Gross M., Pfister H., Alexa M., Rusinkiewicz S., (Eds.), The Eurographics Association. doi:10.2312/SPBG/SPBG04/149-155. 12

[AA09] ALEXA M., ADAMSON A.: Interpolatory point set surfaces—convexity and hermite data. *ACM Trans. Graph. 28*, 2 (may 2009). doi:10.1145/1516522.1516531. 11

[AB98] AMENTA N., BERN M.: Surface reconstruction by voronoi filtering. In *Proceedings of the Fourteenth Annual Symposium on Computational Geometry* (New York, NY, USA, 1998), SCG '98, Association for Computing Machinery, p. 39–48. doi:10.1145/276884.276889. 4, 8

[ABCO*03] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics 9* (2003). doi:10.1109/TVCG.2003.1175093. 11, 14

[ACSTD07] ALLIEZ P., COHEN-STEINER D., TONG Y., DESBRUN M.: Voronoi-based variational reconstruction of unoriented point sets. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing* (Goslar, DEU, 2007), SGP '07, Eurographics Association, p. 39–48. 8

[AD01] Armentano M. G., Durán R. G.: Error estimates for moving least square approximations. *Applied Numerical Mathematics 37*, 3 (2001), 397–416. `doi:10.1016/S0168-9274(00)00054-4`. 11

[AK04] Amenta N., Kil Y. J.: Defining point-set surfaces. In *ACM SIGGRAPH 2004 Papers, SIGGRAPH 2004* (2004). `doi:10.1145/1186562.1015713`. 11

[AKF*24] Attali D., Kouřimská H. D. P., Fillmore C., Ghosh I., Lieutier A., Stephenson E., Wintraecken M.: Tight bounds for the learning of homotopy à la niyogi, smale, and weinberger for subsets of euclidean spaces and of riemannian manifolds, 2024. `arXiv:2206.10485`. 4

[Alm65] Almgren F.: The theory of varifolds. *Mimeographed notes* (1965). 9

[AR00] Atkinson A., Riani M.: *Regression and the Forward Search*. Springer New York, New York, NY, 2000, pp. 16–42. URL: `https://doi.org/10.1007/978-1-4612-1160-0_2`, `doi:10.1007/978-1-4612-1160-0_2`. 11

[BBN*20] Birdal T., Busam B., Navab N., Ilic S., Sturm P.: Generic primitive detection in point clouds using novel minimal quadric fits. *IEEE Transactions on Pattern Analysis and Machine Intelligence 42*, 6 (2020), 1333–1347. `doi:10.1109/TPAMI.2019.2900309`. 8

[BC94] Berkmann J., Caelli T.: Computation of surface geometry and segmentation using covariance techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence 16*, 11 (1994), 1114–1116. `doi:10.1109/34.334391`. 4, 5, 12, 16

[BCDM20] Bonneel N., Coeurjolly D., Digne J., Mellado N.: Code replicability in computer graphics. *ACM Trans. Graph. 39*, 4 (aug 2020). `doi:10.1145/3386569.3392413`. 3, 12

[BDC18] Bearzi Y., Digne J., Chaine R.: Wavejets: A local frequency framework for shape details amplification. *Computer Graphics Forum 37* (2018), 13–24. `doi:10.1111/CGF.13338`. 2, 4, 7, 12, 16, 19

[BDS*18] Barill G., Dickson N. G., Schmidt R., Levin D. I. W., Jacobson A.: Fast winding numbers for soups and clouds. *ACM Trans. Graph. 37*, 4 (July 2018). `doi:10.1145/3197517.3201337`. 9

[BDTB18] Berretti S., Daoudi M., Turaga P., Basu A.: Representation, analysis, and recognition of 3d humans: A survey. *ACM Trans. Multimedia Comput. Commun. Appl. 14*, 1s (mar 2018). `doi:10.1145/3182179`. 2

[Ben75] Bentley J. L.: Multidimensional binary search trees used for associative searching. *Communications of the ACM 18* (9 1975), 509–517. `doi:10.1145/361002.361007`. 11

[BKP*10] Botsch M., Kobbelt L., Pauly M., Alliez P., Levy B.: *Polygon Mesh Processing*. A K Peters, 2010. `doi:10.1201/b10688`. 2

[BLL*22] Bai C., Liu G., Li X., Yang Y., Li Z.: Robust and accurate normal estimation in 3d point clouds via entropy-based local plane voting. *Measurement Science and Technology 33* (6 2022), 095202. `doi:10.1088/1361-6501/AC7035`. 6

[BLM17] Buet B., Leonardi G. P., Masnou S.: A varifold approach to surface approximation. *Archive for Rational Mechanics and Analysis 226* (2017). `doi:10.1007/s00205-017-1141-0`. 9

[BLM22] Buet B., Leonardi G. P., Masnou S.: Weak and approximate curvatures of a measure: A varifold perspective. *Nonlinear Analysis, Theory, Methods and Applications 222* (2022). `doi:10.1016/j.na.2022.112983`. 4, 9, 12, 16

[BLN*13] Berger M., Levine J. A., Nonato L. G., Taubin G., Silva C. T.: A benchmark for surface reconstruction. *ACM Trans. Graph. 32*, 2 (Apr. 2013). `doi:10.1145/2451236.2451246`. 2

[BM16] Boulch A., Marlet R.: Deep learning for robust normal estimation in unstructured point clouds. In *Eurographics Symposium on Geometry Processing* (2016), vol. 35. `doi:10.1111/cgf.12983`. 10

[BSG20] Ben-Shabat Y., Gould S.: Deepfit: 3d surface fitting via neural network weighted least squares, 2020. `doi:10.1007/978-3-030-58452-8_2`. 10, 19

[BTS*17] Berger M., Tagliasacchi A., Seversky L. M., Alliez P., Guennebaud G., Levine J. A., Sharf A., Silva C. T.: A survey of surface reconstruction from point clouds. *Computer Graphics Forum 36* (2017). `doi:10.1111/cgf.12802`. 2, 11

[CACB18] Comino M., Andujar C., Chica A., Brunet P.: Sensor-aware normal estimation for point clouds from 3d range scans. *Computer Graphics Forum 37*, 5 (2018), 233–243. `doi:https://doi.org/10.1111/cgf.13505`. 19

[CGBG13] Chen J., Guennebaud G., Barla P., Granier X.: Non-oriented mls gradient fields. *Computer Graphics Forum 32* (2013). `doi:10.1111/cgf.12164`. 4, 8, 12, 14, 16, 19

[CJW*09] Cline D., Jeschke S., White K., Razdan A., Wonka P.: Dart throwing on surfaces. In *Proceedings of the Twentieth Eurographics Conference on Rendering* (Goslar, DEU, 2009), EGSR'09, Eurographics Association, p. 1217–1226. `doi:10.1111/j.1467-8659.2009.01499.x`. 14

[CLMT14] Cuel L., Lachaud J.-O., Mérigot Q., Thibert B.: Robust geometry estimation using the generalized voronoi covariance measure. *SIAM Journal on Imaging Sciences 8* (8 2014), 1293–1314. `doi:10.1137/140977552`. 8

[Clo] CloudCompare: 3d point cloud and mesh processing software open source project. `https://www.cloudcompare.org/`. 14

[Con86] Connolly M. L.: Measurement of protein surface shape by solid angles. *J. Mol. Graph. 4*, 1 (Mar. 1986), 3–6. `doi:10.1016/0263-7855(86)80086-8`. 4

[CP05] Cazals F., Pouget M.: Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design 22* (2 2005), 121–146. `doi:10.1016/J.CAGD.2004.09.004`. 4, 7, 12, 16, 18, 19

[CPJG09] Cipriano G., Phillips Jr. G. N., Gleicher M.: Multi-scale surface descriptors. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1201–1208. `doi:10.1109/TVCG.2009.168`. 11

[CPPB20] Costantino C., Prati D., Predari G., Bartolomei C.: 3d laser scanning survey for cultural heritage. a flexible methodology to optimize data collection. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 43-B2-2020* (2020), 821–828. `doi:10.5194/isprs-archives-XLIII-B2-2020-821-2020`. 2

[CRT04] Clarenz U., Rumpf M., Telea A.: Robust feature detection and local classification for surfaces based on moment analysis. In *IEEE Transactions on Visualization and Computer Graphics* (2004), vol. 10. `doi:10.1109/TVCG.2004.34`. 4

[CSM03] Cohen-Steiner D., Morvan J. M.: Restricted delaunay triangulations and normal cycle. In *Proceedings of the Annual Symposium on Computational Geometry* (2003). `doi:10.1145/777792.777839`. 4, 8

[CWL*08] Cheng Z.-Q., Wang Y.-Z., Li B., Xu K., Dang G., Jin S.-Y.: A survey of methods for moving least squares surfaces. In *Proceedings of the Fifth Eurographics / IEEE VGTC Conference on Point-Based Graphics* (Goslar, DEU, 2008), SPBG'08, Eurographics Association, p. 9–23. `doi:10.2312/VG/VG-PBG08/009-023`. 2, 11

[CZ09] Cheng Z., Zhang X.: Estimating differential quantities from point cloud based on a linear fitting of normal vectors. *Science in China, Series F: Information Sciences 52* (2009). `doi:10.1007/s11432-009-0061-5`. 18

[CZB*22] Cao J., Zhu H., Bai Y., Zhou J., Pan J., Su Z.: Latent tangent space representation for normal estimation. *IEEE Transactions on Industrial Electronics 69* (2022). `doi:10.1109/TIE.2021.3053904`. 10

[DB02] Douros I., Buxton B.: Three-dimensional surface curvature estimation using quadric surface patches. *Scanning 2002 Proceedings 44* (2002). 4, 8, 12

[DC16] Do Carmo M. P.: *Differential geometry of curves and surfaces: revised and updated second edition*. Courier Dover Publications, 2016. 3

[DGt] DGtal: Digital geometry tools and algorithms library. `http://dgtal.org`. 13

[Dha24] Dharmadhikari S.: *3D Point Cloud Processing Software Market Report 2024 (Global Edition)*. Cognitive Market Research, 2024. 19

[DM14] Digne J., Morel J. M.: Numerical analysis of differential operators on raw point clouds. *Numerische Mathematik 127* (6 2014), 255–289. `doi:10.1007/S00211-013-0584-Y`. 4, 5, 12, 14, 16

[DMSL11] Digne J., Morel J.-M., Souzani C.-M., Lartigue C.: Scale space meshing of raw data point sets. *Computer Graphics Forum 30*, 6 (2011), 1630–1642. `doi:10.1111/j.1467-8659.2011.01848.x`. 6

[DSP16] Dabboura E., Sadat H., Prax C.: A moving least squares meshless method for solving the generalized kuramoto-sivashinsky equation. *Alexandria Engineering Journal 55* (9 2016), 2783–2787. `doi:10.1016/J.AEJ.2016.07.024`. 11

[FC24] Feng N., Crane K.: A heat method for generalized signed distance. *ACM Trans. Graph. 43*, 4 (2024). 9

[FCOS05] Fleishman S., Cohen-Or D., Silva C. T.: Robust moving least-squares fitting with sharp features. *ACM Trans. Graph. 24*, 3 (July 2005), 544–552. `doi:10.1145/1073204.1073227`. 11

[Fu94] Fu J. H. G.: Curvature measures of subanalytic sets. *American Journal of Mathematics 116* (8 1994), 819. `doi:10.2307/2375003`. 8

[FZ07] Fasshauer G. E., Zhang J. G.: Iterated approximate moving least squares approximation. *Computational Methods in Applied Sciences 5* (2007), 221–239. `doi:10.1007/978-1-4020-6095-3_12`. 11

[Gau27] Gauss K. F.: *General Investigations of Curved Surfaces*. Independently published, 1827. 2

[GG07] Guennebaud G., Gross M.: Algebraic point set surfaces. *ACM Transactions on Graphics 26* (7 2007). `doi:10.1145/1276377.1276406`. 4, 8, 12, 14, 16, 19

[GGG08] Guennebaud G., Germann M., Gross M.: Dynamic Sampling and Rendering of Algebraic Point Set Surfaces. *Computer Graphics Forum* (2008). `doi:10.1111/j.1467-8659.2008.01163.x`. 8

[GI04] Goldfeather J., Interrante V.: A novel cubic-order algorithm for approximating principal direction vectors. *ACM Trans. Graph. 23*, 1 (jan 2004), 45–63. `doi:10.1145/966131.966134`. 19

[GKOM18] Guerrero P., Kleiman Y., Ovsjanikov M., Mitra N. J.: Pcpnet learning local shape properties from raw point clouds. *Computer Graphics Forum 37* (2018), 75–85. `doi:10.1111/CGF.13343`. 3, 10, 13

[GMGP05] Gelfand N., Mitra N. J., Guibas L. J., Pottmann H.: Robust global registration. In *Proceedings of the Third Eurographics Symposium on Geometry Processing* (Goslar, DEU, 2005), SGP ’05, Eurographics Association, p. 197–es. `doi:10.2312/SGP/SGP05/197-206`. 2

[GNNB08] Gois J. P., Nakano A., Nonato L. G., Buscaglia G. C.: Front tracking with moving-least-squares surfaces. *Journal of Computational Physics 227* (11 2008), 9643–9669. `doi:10.1016/J.JCP.2008.07.013`. 11

[Gol05] Goldman R.: Curvature formulas for implicit curves and surfaces. *Computer Aided Geometric Design 22*, 7 (2005), 632–658. 9

[Gra98] Gray A.: Modern differential geometry of curves and surfaces with mathematica (second edition). *Computers & Mathematics with Applications 36* (1998). `doi:10.1016/s0898-1221(98)91133-6`. 4, 6, 12, 14, 16

[GS69] Gabriel K. R., Sokal R. R.: A new statistical approach to geographic variation analysis. *Systematic Biology 18*, 3 (09 1969), 259–278. `doi:10.2307/2412323`. 11

[GYJZ15] Guo J., Yan D.-M., Jia X., Zhang X.: Efficient maximal poisson-disk sampling and remeshing on surfaces. *Computers & Graphics 46* (2015), 72–79. 14

[HDD*92] Hoppe H., DeRose T., Duchamp T., McDonald J., Stuetzle W.: Surface reconstruction from unorganized points. *Computer Graphics (ACM) 26* (1992). `doi:10.1145/142920.134011`. 6, 11

[HJ87] Hoffman R., Jain A. K.: Segmentation and classification of range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-9*, 5 (1987), 608–620. `doi:10.1109/TPAMI.1987.4767955`. 4, 6, 12

[HJP*24] Hansen L. H., Jensen S. B., Philipsen M. P., Møgelmose A., Bodum L., Moeslund T. B.: OpenTrench3D: A photogrammetric 3d point cloud dataset for semantic segmentation of underground utilities. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2024), pp. 7646–7655. `doi:10.1109/CVPRW63382.2024.00760`. 17

[HLP*21] Himeur C.-E., Lejemble T., Pellegrini T., Paulin M., Barthe L., Mellado N.: Pcednet: A lightweight neural network for fast and interactive edge detection in 3d point clouds. *ACM Trans. Graph. 41*, 1 (Nov. 2021). `doi:10.1145/3481804`. 2, 19

[HT03] Hulin D., Troyanov M.: Mean curvature and asymptotic volume of small balls. *The American mathematical monthly 110*, 10 (2003), 947–950. `doi:10.1080/00029890.2003.11920037`. 4

[HZ03] Hartley R., Zisserman A.: *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2

[KBH06] Kazhdan M., Bolitho M., Hoppe H.: Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing* (Goslar, DEU, 2006), SGP ’06, Eurographics Association, p. 61–70. `doi:10.2312/SGP/SGP06/061-070`. 14

[KG18] Khameneifar F., Ghorbani H.: On the curvature estimation for noisy point cloud data via local quadric surface fitting. *Computer-Aided Design and Applications 16* (2018). `doi:10.14733/cadaps.2019.140-149`. 8

[KK06] Kim S.-K., Kim C.-H.: Finding ridges and valleys in a discrete surface using a modified mls approximation. *Computer-Aided Design 38*, 2 (2006), 173–180. `doi:10.1016/j.cad.2005.10.004`. 19

[KL06] Komargodski Z., Levin D.: Hermite type moving-least-squares approximations. *Computers & Mathematics with Applications 51* (4 2006), 1223–1232. `doi:10.1016/J.CAMWA.2006.04.005`. 11

[KMJ*19] Koch S., Matveev A., Jiang Z., Williams F., Artemov A., Burnaev E., Alexa M., Zorin D., Panozzo D.: Abc: A big cad model dataset for geometric deep learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019). 3, 13, 14

[KSNS07] Kalogerakis E., Simari P., Nowrouzezahrai D., Singh K.: Robust Statistical Estimation of Curvature on Discretized Surfaces. In *Geometry Processing* (2007), Belyaev A., Garland M., (Eds.), The Eurographics Association. `doi:/10.2312/SGP/SGP07/013-022`. 4, 6, 11, 12, 16

[KYZB19] Kaiser A., Ybanez Zepeda J. A., Boubekeur T.: A Survey of Simple Geometric Primitives Detection Methods for Captured 3D Data. *Computer Graphics Forum* (2019). `doi:10.1111/cgf.13451`. 2

[LCBM21] Lejemble T., Coeurjolly D., Barthe L., Mellado N.: Stable and efficient differential estimators on oriented point clouds.

Computer Graphics Forum 40 (8 2021), 205–216. doi:10.1111/CGF.14368. 4, 8, 12, 14, 16

[LCL17] LACHAUD J.-O., COEURJOLLY D., LEVALLOIS J.: Robust and convergent curvature and normal estimators with digital integral invariants. *Modern Approaches to Discrete Curvature* (2017), 293–348. 4

[LCL*23] LACHAUD J. O., COEURJOLLY D., LABART C., ROMON P., THIBERT B.: Lightweight curvature estimation on point clouds with randomized corrected curvature measures. *Computer Graphics Forum 42* (8 2023), e14910. doi:10.1111/CGF.14910. 4, 9, 12, 16, 18

[Lev98] LEVIN D.: The approximation power of moving least-squares. *Mathematics of Computation 67* (1998). doi:10.1090/s0025-5718-98-00974-0. 11, 12

[Lev04] LEVIN D.: Mesh-independent surface interpolation. In *Geometric Modeling for Scientific Visualization* (Berlin, Heidelberg, 2004), Brunnett G., Hamann B., Müller H., Linsen L., (Eds.), Springer Berlin Heidelberg, pp. 37–49. doi:10.1007/978-3-662-07443-5_3. 11

[Lev15] LEVIN D.: Between moving least-squares and moving least-ℓ1. *BIT Numerical Mathematics 55* (9 2015), 781–796. doi:10.1007/S10543-014-0522-0/METRICS. 11

[LFS*23] LI Q., FENG H., SHI K., GAO Y., FANG Y., LIU Y.-S., HAN Z.: SHS-Net: Learning signed hyper surfaces for oriented normal estimation of point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Los Alamitos, CA, USA, June 2023), IEEE Computer Society, pp. 13591–13600. doi:10.1109/CVPR52729.2023.01306. 10

[LL20] LI X., LI S.: On the augmented moving least squares approximation and the localized method of fundamental solutions for anisotropic heat conduction problems. *Engineering Analysis with Boundary Elements 119* (10 2020), 74–82. doi:10.1016/J.ENGANABOUND.2020.07.007. 11

[LLC*22] LI Q., LIU Y.-S., CHENG J.-S., WANG C., FANG Y., HAN Z.: HSurf-Net: Normal estimation for 3D point clouds by learning hyper surfaces. In *Advances in Neural Information Processing Systems (NeurIPS)* (2022), vol. 35, Curran Associates, Inc., pp. 4218–4230. 10

[LLY14] LEE Y. J., LEE S., YOON J.: A framework for moving least squares method with total variation minimizing regularization. *Journal of Mathematical Imaging and Vision 48* (3 2014), 566–582. doi:10.1007/S10851-013-0428-5/METRICS. 11

[LMBM20] LEJEMBLE T., MURA C., BARTHE L., MELLADO N.: Persistence analysis of multi-scale planar structure graph in point clouds. *Computer Graphics Forum 39*, 2 (2020), 35–50. doi:https://doi.org/10.1111/cgf.13910. 11

[LOM20] LENSSEN J. E., OSENDORFER C., MASCI J.: Deep iterative surface normal estimation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2020). doi:10.1109/CVPR42600.2020.01126. 10, 19

[LRT22] LACHAUD J. O., ROMON P., THIBERT B.: Corrected curvature measures. *Discrete and Computational Geometry 68* (9 2022), 477–524. doi:10.1007/S00454-022-00399-4. 8

[LRTC20] LACHAUD J. O., ROMON P., THIBERT B., COEURJOLLY D.: Interpolated corrected curvature measures for polygonal surfaces. *Computer Graphics Forum 39* (8 2020), 41–54. doi:10.1111/CGF.14067. 8, 9

[LSY17] LI W., SONG G., YAO G.: Piece-wise moving least squares approximation. *Applied Numerical Mathematics 115* (5 2017), 68–81. doi:10.1016/J.APNUM.2017.01.001. 11

[LSY24] LYU B., SHEN L.-Y., YUAN C.-M.: Igf-fit: Implicit gradient field fitting for point cloud normal estimation. *Graphical Models 133* (2024), 101214. doi:10.1016/j.gmod.2024.101214. 10

[LT90] LIANG P., TODHUNTER J. S.: Representation and recognition of surface shapes in range images: a differential geometry approach. *Comput. Vision Graph. Image Process. 52*, 1 (Aug. 1990), 78–109. doi:10.1016/0734-189X(90)90124-E. 4, 5, 12

[LWK12] LI J., WANG H., KIM N. H.: Doubly weighted moving least squares and its application to structural reliability analysis. *Structural and Multidisciplinary Optimization 46* (7 2012), 69–82. doi:10.1007/S00158-011-0748-2/METRICS. 11

[LZM*23] LI S., ZHOU J., MA B., LIU Y.-S., HAN Z.: Neaf: learning neural angle fields for point normal estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2023), AAAI Press. doi:10.1609/aaai.v37i1.25224. 10

[LZW*22] LI K., ZHAO M., WU H., YAN D.-M., SHEN Z., WANG F.-Y., XIONG G.: GraphFit: Learning multi-scale graph-convolutional representation for point cloud normal estimation. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII* (Berlin, Heidelberg, 2022), Springer-Verlag, p. 651–667. doi:10.1007/978-3-031-19824-3_38. 10, 19

[MB08] MOST T., BUCHER C.: New concepts for moving least squares: An interpolating non-singular weighting function and weighted nodal least squares. *Engineering Analysis with Boundary Elements 32* (6 2008), 461–470. doi:10.1016/j.enganabound.2007.10.013. 11

[MBF92] MONGA O., BENAYOUN S., FAUGERAS O.: From partial derivatives of 3-d density images to ridge lines. In *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (1992), pp. 354–359. doi:10.1109/CVPR.1992.223165. 19

[MBM01] MIKHAIL E. M., BETHEL J. S., MCGLONE J. C.: *Introduction to modern photogrammetry*. John Wiley & Sons, 2001. 2

[Mea82] MEAGHER D.: Geometric modeling using octree encoding. *Computer Graphics and Image Processing 19*, 2 (1982), 129–147. doi:10.1016/0146-664X(82)90104-6. 11

[MHYS04] MANAY S., HONG B.-W., YEZZI A. J., SOATTO S.: Integral invariant signatures. In *European Conference on Computer Vision* (2004), Springer, pp. 87–99. 4

[ML09] MUJA M., LOWE D. G.: Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1) 2*, 331-340 (2009), 2. 11

[MLG*20] MELLADO N., LEJEMBLE T., GUENNEBAUD G., BARLA P., ET AL.: Ponca: a point cloud analysis library. https://github.com/poncateam/ponca/, 2020. 12

[MLR*22] MERCIER C., LESCOAT T., ROUSSILLON P., BOUBEKEUR T., THIERY J.-M.: Moving level-of-detail surfaces. *ACM Transactions on Graphics (Proc. SIGGRAPH 2022) 41*, 4 (2022). doi:10.1145/3528223.3530151. 11

[MNG04] MITRA N. J., NGUYEN A., GUIBAS L.: Estimating surface normals in noisy point cloud data. In *International Journal of Computational Geometry and Applications* (2004), vol. 14. doi:10.1142/s0218195904001470. 6

[MOG11] MERIGOT Q., OVSJANIKOV M., GUIBAS L. J.: Voronoi-based curvature and feature estimation from point clouds. *IEEE Transactions on Visualization and Computer Graphics 17* (2011). doi:10.1109/TVCG.2010.261. 4, 8, 11

[MSR07] MAGID E., SOLDEA O., RIVLIN E.: A comparison of gaussian and mean curvature estimation methods on triangular meshes of range image data. *Computer Vision and Image Understanding 107*, 3 (2007), 139–159. doi:10.1016/j.cviu.2006.09.007. 3

[MT02] MORVAN J.-M., THIBERT B.: On the approximation of a smooth surface with a triangulated mesh. *Computational Geometry 23*, 3 (2002), 337–352. doi:10.1016/S0925-7721(02)00097-4. 4

[NGB*12] NICOLAS M., GUENNEBAUD G., BARLA P., REUTER P., SCHLICK C.: Growing least squares for the analysis of manifolds in scale-space. *Eurographics Symposium on Geometry Processing 31* (2012), 1691–1701. doi:10.1111/J.1467-8659.2012.03174.X. 8, 18

[NR17] Najman L., Romon P.: *Modern approaches to discrete curvature*, vol. 2184. Springer, 2017. 3

[OBS04] Ohtake Y., Belyaev A., Seidel H.-P.: Ridge-valley lines on meshes via implicit surface fitting. In *ACM SIGGRAPH 2004 Papers* (New York, NY, USA, 2004), SIGGRAPH '04, Association for Computing Machinery, p. 609–612. doi:10.1145/1186562.1015768. 19

[ÖGG09] Öztireli A. C., Guennebaud G., Gross M.: Feature Preserving Point Set Surfaces based on Non-Linear Kernel Regression. *Computer Graphics Forum* (2009). doi:10.1111/j.1467-8659.2009.01388.x. 11

[PC24] Pouget M., Cazals F.: Estimation of local differential properties of point-sampled surfaces. In *CGAL User and Reference Manual*, 5.6.1 ed. CGAL Editorial Board, 2024. URL: https://doc.cgal.org/5.6.1/Manual/packages.html#PkgJetFitting3. 12, 18

[PCBS16] Palma G., Cignoni P., Boubekeur T., Scopigno R.: Detection of Geometric Temporal Changes in Point Clouds. *Computer Graphics Forum* (2016). doi:10.1111/cgf.12730. 11

[PKG03] Pauly M., Keiser R., Gross M.: Multi-scale Feature Extraction on Point-Sampled Surfaces. *Computer Graphics Forum* (2003). doi:10.1111/1467-8659.00675. 18

[PPY*16] Pintus R., Pal K., Yang Y., Weyrich T., Gobbetti E., Rushmeier H.: A survey of geometric analysis in cultural heritage. *Computer Graphics Forum 35*, 1 (2016), 4–31. doi:https://doi.org/10.1111/cgf.12668. 2

[Pra87] Pratt V.: Direct least-squares fitting of algebraic surface. *Computer Graphics (ACM) 21* (1987). doi:10.1145/37402.37420. 4, 7, 8, 12, 16

[PRLH*22] Portaneri C., Rouxel-Labbé M., Hemmer M., Cohen-Steiner D., Alliez P.: Alpha wrapping with an offset. *ACM Trans. Graph. 41*, 4 (July 2022). doi:10.1145/3528223.3530152. 11

[PT96] Piegl L., Tiller W.: *The NURBS book*. Springer, 1996. 2

[PWHY09] Pottmann H., Wallner J., Huang Q. X., Yang Y. L.: Integral invariants for robust geometry processing. *Computer Aided Geometric Design 26* (1 2009), 37–60. doi:10.1016/J.CAGD.2008.01.002. 4, 14

[PWY*07] Pottmann H., Wallner J., Yang Y. L., Lai Y. K., Hu S. M.: Principal curvatures from the integral invariant viewpoint. *Computer Aided Geometric Design 24* (11 2007), 428–442. doi:10.1016/J.CAGD.2007.07.004. 4, 5, 12

[QSMG17] Qi C. R., Su H., Mo K., Guibas L. J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017* (2017), vol. 2017-January. doi:10.1109/CVPR.2017.16. 10

[RB25] Rauch L., Braml T.: Rohbau3D: A shell construction site 3d point cloud dataset. *Scientific Data 2025 12:1 12* (8 2025), 1478–. doi:10.1038/s41597-025-05827-7. 17, 18

[RGRG15] Ridel B., Guennebaud G., Reuter P., Granier X.: Parabolic-cylindrical moving least squares surfaces. *Computers and Graphics (Pergamon) 51* (2015). doi:10.1016/j.cag.2015.05.006. 4, 7, 12, 14

[S*84] Simon L., et al.: Lectures on geometric measure theory, 1984. 9

[SDC*20] Sanchez J., Denis F., Coeurjolly D., Dupont F., Trassoudaine L., Checchin P.: Robust normal vector estimation in 3d point clouds through iterative principal component analysis. *ISPRS Journal of Photogrammetry and Remote Sensing 163* (2020), 18–35. doi:10.1016/j.isprsjprs.2020.02.018. 6, 12

[SM22] Stoker J., Miller B.: The accuracy and consistency of 3d elevation program data: A systematic analysis. *Remote Sensing 14*, 4 (2022), 940. 2

[SMVL25] Sulzer R., Marlet R., Vallet B., Landrieu L.: A survey and benchmark of automatic surface reconstruction from point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence 47*, 3 (2025), 2000–2019. doi:10.1109/TPAMI.2024.3510932. 2

[Sot07] Sotoodeh S.: Hierarchical clustered outlier detection in laser scanner point clouds. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 36*, 3/W52 (2007), 383–388. 11

[TAM*19] Tachella J., Altmann Y., Mellado N., Mccarthy A., Tobin R., Stuart Buller G., Tourneret J.-Y., Mclaughlin S.: Real-time 3D reconstruction from single-photon lidar data using plug-and-play point cloud denoisers. *Nature Communications 10* (2019), 4984. doi:10.1038/s41467-019-12943-7. 19

[Tau93] Taubin G.: An improved algorithm for algebraic curve and surface fitting. In *1993 (4th) International Conference on Computer Vision* (1993), pp. 658–665. doi:10.1109/ICCV.1993.378149. 8

[TSA*21] Tey W. Y., Sidik N. A. C., Asako Y., Muhieldeen M. W., Afshar O.: Moving least squares method and its improvement: A concise review. *Journal of Applied and Computational Mechanics 7* (4 2021), 883–889. doi:10.22055/JACM.2021.35435.2652. 2, 11

[Š04] Šír Z.: Fitting of piecewise polynomial implicit surfaces. In *Geometry and Computer Graphics* (Ostrava, 2004), pp. 202–206. 8

[VVP*16] Váša L., Vaněček P., Prantl M., Skorkovská V., Martínek P., Kolingerová I.: Mesh Statistics for Robust Curvature Estimation. *Computer Graphics Forum* (2016). doi:10.1111/cgf.12982. 3

[Wen95] Wendland H.: Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in computational Mathematics 4* (1995), 389–396. 11, 12

[WEW*22] Winiwarter L., Esmoris Pena A. M., Weiser H., Anders K., Martinez Sanchez J., Searle M., Hofle B.: Virtual laser scanning with helios++: A novel take on ray tracing-based simulation of topographic full-waveform 3d laser scanning. *Remote Sensing of Environment 269* (2022). doi:https://doi.org/10.1016/j.rse.2021.112772. 14

[Win82] Wintgen P.: Normal cycle and integral curvature for polyhedra in riemannian manifolds. *Differential geometry 21* (1982). 8

[WZG*23] Wang W., Zhang Y., Ge G., Jiang Q., Wang Y., Hu L.: Indoor point cloud segmentation using a modified region growing algorithm and accurate normal estimation. *IEEE Access 11* (2023). doi:10.1109/ACCESS.2023.3270709. 12

[XDW*23] Xu R., Dou Z., Wang N., Xin S., Chen S., Jiang M., Guo X., Wang W., Tu C.: Globally consistent normal orientation for point clouds by regularizing the winding-number field. *ACM Trans. Graph. 42*, 4 (July 2023). doi:10.1145/3592129. 9, 19

[YdSEZ*24] Yi Y., de Silva Edirimuni D., Zhu Y., Gao S., Wang Z., Robles-Kelly A., Lu X.: Point cloud normal estimation via representation learning on height maps. In *Proceedings of the 6th ACM International Conference on Multimedia in Asia* (New York, NY, USA, 2024), MMAsia '24, Association for Computing Machinery. doi:10.1145/3696409.3700185. 10

[Yuk15] Yuksel C.: Sample Elimination for Generating Poisson Disk Sample Sets. *Computer Graphics Forum* (2015). doi:10.1111/cgf.12538. 14

[ZCZ*22] Zhang J., Cao J. J., Zhu H. R., Yan D. M., Liu X. P.: Geometry guided deep surface normal estimation. *CAD Computer Aided Design 142* (2022). doi:10.1016/j.cad.2021.103119. 10

[ZFH20] Zheng S., Feng R., Huang A.: A modified moving least-squares suitable for scattered data fitting with outliers. *Journal of Computational and Applied Mathematics 370* (5 2020). doi:10.1016/j.cam.2019.112655. 11

[Zha12] Zhang Z.: Microsoft kinect sensor and its effect. *IEEE MultiMedia 19*, 2 (2012), 4–10. doi:10.1109/MMUL.2012.24. 2

[ZHLL20] Zhou J., Huang H., Liu B., Liu X.: Normal estimation for 3d point clouds via local plane constraint and multi-scale selection. *CAD Computer Aided Design 129* (2020). doi:10.1016/j.cad.2020.102916. 10

[ZJW*23] Zhou J., Jin W., Wang M., Liu X., Li Z., Liu Z.: Improvement of normal estimation for point clouds via simplifying surface fitting. *CAD Computer Aided Design 161* (2023). doi:10.1016/j.cad.2023.103533. 10, 19

[ZLD*21] Zhu R., Liu Y., Dong Z., Wang Y., Jiang T., Wang W., Yang B.: AdaFit: Rethinking learning-based normal estimation on point clouds. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 6098–6107. doi:10.1109/ICCV48922.2021.00606. 10, 19

[ZLW*24] Zhou J., Li Y., Wang M., Li N., Li Z., Wang W.: Robust point cloud normal estimation via multi-level critical point aggregation. *Visual Computer 40* (10 2024), 7369–7384. doi:10.1007/S00371-024-03532-X/FIGURES/7. 10

[ZNZ*25] Zhang J., Nie M., Zou C., Liu J., Liu L., Cao J.: PointNorm-Net: Self-Supervised Normal Prediction of 3D Point Clouds via Multi-Modal Distribution Estimation . *IEEE Transactions on Pattern Analysis & Machine Intelligence 47*, 08 (Aug. 2025), 6515–6530. doi:10.1109/TPAMI.2025.3562051. 10

[ZSG*18] Zollhöfer M., Stotko P., Görlitz A., Theobalt C., Niessner M., Klein R., Kolb A.: State of the Art on 3D Reconstruction with RGB-D Cameras. *Computer Graphics Forum* (2018). doi:10.1111/cgf.13386. 2