Anisotropic Denoising of 3D Point Clouds by Aggregation of Multiple Surface-Adaptive Estimates

Zhongwei Xu, Alessandro Foi, *Senior Member, IEEE*

Abstract—3D point clouds commonly contain positional errors which can be regarded as noise. We propose a point cloud denoising algorithm based on aggregation of multiple anisotropic estimates computed on local coordinate systems. These local estimates are adaptive to the shape of the surface underlying the point cloud, leveraging an extension of the Local Polynomial Approximation (LPA) - Intersection of Confidence Intervals (ICI) technique to 3D point clouds. The adaptivity due to LPA-ICI is further strengthened by the dense aggregation with data-driven weights. Experimental results demonstrate state-of-the-art restoration quality of both sharp features and smooth areas.

Index Terms—3D Point Cloud, denoising, anisotropic, surface-adaptive filtering.

1 Introduction

POINT-CLOUD data, which consist of collections of 3D point locations along with some other affiliated properties (colors, point orientations, etc.) discretely representing 3D scenes or objects, are widely used in many applications, including cultural heritage preservation [1], autonomous vehicles [2] and virtual reality [3]. These data can be the direct output of various 3D scanning devices, or be computed from photographs taken at multiple viewpoints using photogrammetry techniques [4], [5]. Such adaptive procedures are invariably prone to errors, e.g., due to vibrations or scattering during the scanning process, or because the multi-view photographs are not ideal (e.g., subject to reflections, over or under exposed, blurred, grainy). These measurement errors result in noise corrupting the positions of the points in the cloud. To permit the most effective use of the acquired point clouds, it is essential to first remove the noise.

In this paper we develop an anisotropic denoising algorithm for point clouds based on a dense aggregation of moving least squares (MLS) estimates defined on directional neighborhoods that are locally adaptive to the shape of the surface underlying the point cloud. Specifically, the Local Polynomial Approximation (LPA) - Intersection of Confidence Intervals (ICI) technique [6], [7] is employed to automatically determine adaptive directional neighborhoods for each point. This procedure identifies the largest neighborhoods whose content fits the desired polynomial smoothness; hence they avoid edges and singularities and constitute a robust adaptive support for a local polynomial MLS estimation. The final estimate of the point cloud is obtained by combining all overlapping local estimates through a novel regularized aggregation procedure specific for point clouds.

The LPA-ICI technique was originally developed for the pointwise adaptive nonparametric estimation of 1-D signals [6], and further developed into an adaptive anistropic filter for images and videos. Thanks to its design flexibility and accurate adaptation to the geometry of the image content, it has found successful application to image denoising and other image and video restoration tasks [8]. These properties make it a natural candidate also for dealing with point cloud data.

The specific technical contributions in this work include:

- the design of the adaptive anisotropic neighborhood structure with respect to the local coordinate system (LCS);
- a novel aggregation strategy where the final estimate simultaneously fits all overlapping local estimates;
- a robust method for estimating the noise variance of noisy point clouds;
- a robust method for estimating the surface sample density of point clouds;
- a fully automatic algorithm for point cloud denoising that embeds the above four items and can adapt to data with unknown sampling density and noise variance;
- a scheme for the efficient recursive application of this point cloud denoising algorithm.

2 Related work

Many algorithms for point cloud denoising have been proposed in the last two decades. A classical approach based on the MLS approximation is introduced in [9]. However, due to fixed circular symmetric (isotropic) weights in the MLS, this method tends to over-smooth sharp features in the point cloud. This issue has been addressed by various anisotropic approaches: [10] extended the popular bilateral filter from image [11] and mesh denoising [12] to point cloud denoising; by first detecting the location of sharp features,

- *Z. Xu is with Noiseless Imaging Oy (Ltd), 33720, Tampere, Finland. E-mail: xu@noiselessimaging.com*
- *A. Foi is with Tampere University, 33720, Tampere Finland, and with Noiseless Imaging Oy.*

[13] classifies piecewise smooth regions that are separately filtered, while [14] applies piecewise polynomials (splines) for better local approximation of smoothness and singularities. The anisotropic mean curvature flow technique is employed by [15] and [16] to recognize and recover the sharp edges during the filtering; [17] takes the bilaterally filtered point normals as the indicator of surface discontinuities (sharp features), then treats the points from the other side of the discontinuity as outliers during their robust approximation procedure, thus preserving sharpness. Sparse recovery methods using global ℓ_1 and ℓ_0 minimization are presented in [18] and [19], respectively, demonstrating the potential to reconstruct sharp edges of basic geometrical objects (e.g., dodecahedron, pyramid) even when the noise level is high. The locally optimal projection (LOP) operator introduced in [20] incorporates multivariate ℓ_1 median fitting and a repulsion function to project a number of evenly distributed points onto the underlying surface of the noisy point cloud. Due to desired features such as not relying on local orientation of the input noisy points and robustness to outliers, several algorithms were developed based on LOP: [21] accelerates the time-consuming LOP operator by describing local point distribution with a Gaussian mixture model (GMM); since both [20] and [21] still run in an isotropic manner, [22] and [23] adopted a strategy similar to that introduced in [17] by incorporating the filtered normal information of the input points in their LOP-like projection operators, making them anisotropic. The idea of implementing signal processing on graphs [24] for point cloud denoising appears in [25], where convex optimization is applied on the graph structure built on the noisy point cloud. More recently, [26] proposed a rolling filter for removing small-scale geometric features in a point cloud while retaining the large-scale structures.

Due to the crucial role orientation played in many point cloud processing applications, such as surface reconstruction and point-based rendering, considerable efforts were also invested in developing accurate normal estimation algorithms. The main challenge is obtaining correct normals at points close to sharp features, where the estimate of normal based on local neighborhood is often corrupted by points from different sides of the surface discontinuities. Techniques proposed to tackle this issue, include the iterative bilateral normal filtering [17], the anisotropic neighborhood based estimation [27], low-rank approximation using non-local similarities [28], and deep-learning based methods [29], [30]. We note that our proposed algorithm does not crucially rely on an accurate normal estimation of each point, as we explain further in this paper, thus we adopt just the standard approach of local principal component analysis (PCA) [27] due to its simplicity and computational efficiency.

The proposed method possesses several advantages over previous approaches to point cloud denoising: compared to methods based on bilateral weights [10], [17], the LPA-ICI is intrinsically more robust, because it is based on *testing the statistical consistency* of a *multiscale* sequence of local estimators, rather than the mere position of individual noisy samples, this makes our method more effective under heavy noise; by utilizing *asymmetric directional* neighborhoods for the LPA-ICI, we can adapt to edges and discontinuities using much larger supports (hence stronger noise attenuation) than classical MLS based on symmetric weights, as illustrated in Fig. 1; as known from image denoising [31], the *aggregation* of the overlapping local estimates improves the stability and quality of the estimation; with the proper modeling of the residual noise in the denoised point cloud, further iterations of the proposed denoising algorithm allow to attenuate more noise without distorting fine features; thanks to its adaptivity and rigorous statistical modeling, our method is able to preserve the structure and fine features faithfully and with metrological accuracy, without incurring in oversimplification of the fine geometries or shape bias such as can be observed in [18], [19]; unlike LOP-like anisotropic algorithms [22], [23], the proposed method does not introduces gaps near sharp edges, thus the post-upsampling process for filling the gaps, which is not cheap and error-prone, is avoided.

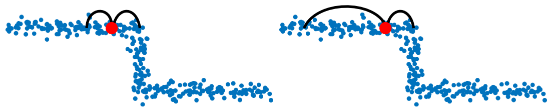


Fig. 1. Adaptive isotropic neighborhoods (left) and adaptive directional neighborhoods (right) indicated by black curved lines for a point (in red) located close to a sharp edge in a 2-D noisy point cloud. Using adaptive directional neighborhoods provides larger supports for a local estimate even when the points are close to sharp features.

At the time of writing of the present paper, methods based on deep neural networks are making their debut for denoising point clouds [32], after impressive results were already demonstrated for the restoration of images and videos (e.g., [33], [34], [35]). These methods, designed for powerful parallel GPUs, crucially depend on expensive training over massive datasets and are often impaired when the data to be filtered deviates significantly from the training material. As a model-based method, the proposed algorithm does not rely on training and thus generalizes and adapts to multiple sampling and measurements scenarios, where it attains consistent quality without need of additional training.

3 Observation model and notation

We consider a noisy point cloud $P = \{p_i, i = 1, \ldots, I\}$ of the form

$$p_i = q_i + \eta_i, \qquad p_i, q_i, \eta_i \in \mathbb{R}^3,$$

where q_i is a point from the unknown noise-free point cloud Q, $\eta_i \sim \mathcal{N}(\mu, \Sigma)$ is i.i.d. zero-mean Gaussian white noise, with $\mu = [0, 0, 0]^T$ and $\Sigma = \sigma^2\mathbf{1}$, $\mathbf{1}$ being the 3×3 identity matrix.

Each point is a 3-component column vector representing a spatial position in $\mathbb{R}^3$. We assume that point clouds are discrete noisy samples of an underlying surface (2-dimensional manifold) which we denote by $\mathcal{S}$. As the noise is modeled as isotropic, the point-to-surface error measured along the normal to the surface has also variance σ^2.

The aim of denoising is to obtain from P a point cloud $\hat{Q} = \{\hat{q}_i, i = 1, \ldots, I\}$ that is close to $\mathcal{S}$, while not being necessarily close to Q.

Since we focus on positional noise filtering, in this work we expect the input noisy point clouds are without normal information, as well as the denoised ones.

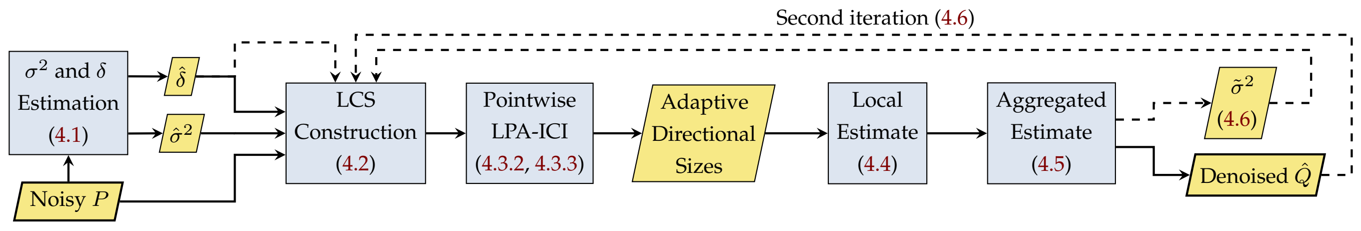


Fig. 2. Flowchart of the proposed point cloud denoising algorithm (indices of the corresponding subsections are in parentheses). Taking an un-oriented and un-organized point cloud P as the input, our algorithm first estimates its noise variance σ^2 and surface sample density δ (Section 4.1), which are utilized throughout our algorithm enabling adaptivity to point clouds of various scales and noise levels; the local coordinate system (LCS) for each point $p_i \in P$ is then computed (Section 4.2) using local PCA and employed for building the directional neighborhoods of p_i, which have shape of right rectangular prisms (Section 4.3.1 and Fig. 4); thereafter, the LPA-ICI technique is applied within the LCS of p_i to select the adaptive size for each directional neighborhood of p_i, such that the directional neighborhoods are large in smooth areas and small when approaching sharp edges (Sections 4.3.2, 4.3.3, and Fig. 6); with its own directional neighborhoods, we can already compute the local estimate of p_i (Section 4.4), however it is a rather weak estimate; a novel strategy to aggregate the overlapping adaptive local estimates (Section 4.5) of p_i is therefore introduced to gain more stable and finer estimation; a second iteration of this denoising process can be applied on the denoised $\hat{Q}$, by modeling the residual noise variance $\tilde{\sigma}^2$ in $\hat{Q}$ (Section 4.6). The dashed arrows indicate the start of the second iteration.

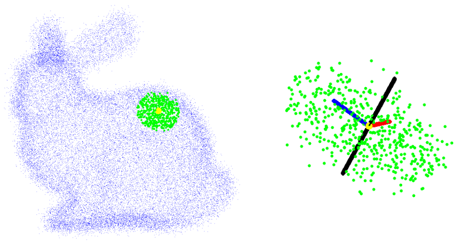


Fig. 3. Computation of the LCS for a point in the noisy Bunny point cloud. Left: a point (in yellow) and its KNN (in green); right: zoomed-in view of the KNN and the three principal axes of the LCS (red and blue being the first and second principal axes, and black being the third principal axis).

4 Proposed Denoising Algorithm

Fig. 2 provides a flowchart of the proposed algorithm, with the pipeline summarized in the caption, which we describe in detail in the following subsections.

4.1 Surface sample density and noise variance estimation

In order to develop a robust and practical denoising algorithm suitable for real world use cases, where often the essential information of P, i.e. its noise variance σ^2 and surface sample density (denoted by δ), are unknown, we introduce a novel strategy to estimate them, as we detail in Appendix I, and utilize these two parameters throughout the proposed algorithm.

4.2 LCS construction

For each point $p_i \in P$, we find its K nearest neighbors (KNN) and compute their three principal components, denoted by c_i, d_i, and e_i (1st, 2nd, and 3rd components, respectively). These are three column vectors of unit norm (i.e. versors). The LCS of p_i, denoted by $\mathcal{L}_i$, is then built having p_i as its origin and using c_i, d_i, and e_i as the versors of its three axes, as illustrated in Fig. 3. This is an efficient, yet rudimentary, approach for LCS construction. The versors c_i and d_i span a plane that, roughly, is locally parallel to the underlying surface at p_i, while e_i acts as the local surface normal [27].

The local coordinates of any point $p_m \in P$ with respect to $\mathcal{L}_i$ are denoted as $p_m^{\mathcal{L}_i}[x_m^{\mathcal{L}_i}, y_m^{\mathcal{L}_i}, z_m^{\mathcal{L}_i}]^T$ and they can be easily computed by $p_m^{\mathcal{L}_i} = [c_i, d_i, e_i]^T(p_m - p_i)$. Trivially, $p_i^{\mathcal{L}_i} = [0,0,0]^T$ and $c_i^{\mathcal{L}_i} = [1,0,0]^T$, $d_i^{\mathcal{L}_i} = [0,1,0]^T$, $e_i^{\mathcal{L}_i} = [0,0,1]^T$.

Although this simple LCS computation provides only a rough estimate of the local surface orientation and normal, this does not reflect into the quality of the final point-cloud estimate by our algorithm. As explained in the following sections, the LCS is used to define an adaptive directional neighborhood and, due to adaptivity, the resulting neighborhood is only marginally impacted by moderate errors in the surface normal estimation.

4.3 Adaptive directional sizes

The idea of anisotropic LPA-ICI is to model the local smoothness of the underlying surface $\mathcal{S}$ with respect to different directional neighborhoods with varying sizes. Points within each directional neighborhood are treated as part of a locally smooth portion of $\mathcal{S}$, where a low-order local polynomial model fits the data. Anisotropy follows from allowing neighborhoods along different directions to have different sizes. Compared to adaptive isotropic neighborhoods, the adaptive directional ones can attain large sizes even in areas that are close to sharp features, thus providing better supports for local estimates, as illustrated in Fig. 1.

In the following subsections, we concentrate on the peculiar and novel aspects involved when considering the LPA-ICI for point clouds. We refer our readers to [36], [37], [38] for details about this versatile established technique.

4.3.1 Structure of the directional neighborhoods

In case of point clouds, directional neighborhoods are subvolumes of $\mathbb{R}^3$ which are intersected with P in order to select points. Concretely, for each p_i, we consider four directional neighborhoods each within the four quadrants defined by the first two principal axes of $\mathcal{L}_i$. Each directional neighborhood is a right rectangular prism, as illustrated by the example in Fig. 4. The base of the prism is a $h \times h$ square, where h defines the size of the directional neighborhood; the height of the prism is, by construction of $\mathcal{L}_i$, along the normal and it is set equal to $\max\{6\sigma, 2h\}$, so that the prism is tall enough to include noisy samples from the neighborhood of p_i (i.e. $\pm 3\sigma$ about the floor of the LCS) without intersecting other portions of the manifold.

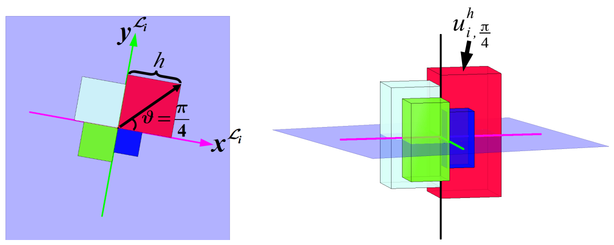


Fig. 4. The directional neighborhoods of p_i, as seen from atop (along the $z^{\mathcal{L}_i}$ axis of the local coordinate system) (left) and sideways (right). The arrow with angle $\vartheta = \frac{\pi}{4}$ indicates the direction of the neighborhood in red, denoted by $u^h_{i,\frac{\pi}{4}}$; the blue plane represents the $(x^{\mathcal{L}_i}, y^{\mathcal{L}_i})$-plane; the magenta, green, and black lines represent the three axes of the local coordinate system $\mathcal{L}_i$.

We denote each directional neighborhood of p_i as $u^h_{i,\vartheta}$, where $h \in \mathbb{R}^+$ is the size and $\vartheta \in \Theta = \{\frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}\}$ gives direction within the $(x^{\mathcal{L}_i}, y^{\mathcal{L}_i})$-plane, as also shown in Fig. 4. Further, let $M^h_{i,\vartheta}$ be the indices of the points inside $u^h_{i,\vartheta}$, i.e. $P \cap u^h_{i,\vartheta} = \{p_m, m \in M^h_{i,\vartheta}\}$. For $h=0$ we set $u^0_{i,\vartheta} = \{p_i\}$ and $M^0_{i,\vartheta} = \{i\}$, because the probability that $u^0_{i,\vartheta}$ contains any point other than p_i is equal to zero.

4.3.2 *Pointwise polynomial estimate*

The fitting of the local polynomial model over each directional neighborhood is operated through the LPA kernels $g^h_{i,\vartheta}$, which are computed with respect to the coordinates $x^{\mathcal{L}_i}_m$, $y^{\mathcal{L}_i}_m$, $m \in M^h_{i,\vartheta}$, as

$$g^h_{i,\vartheta}(k) = \boldsymbol{\phi}(k,:)\left(\boldsymbol{\phi}^T\boldsymbol{\phi}\right)^{\dagger}\boldsymbol{\phi}(1,:)^T, \quad k=1,\ldots,\left|M^h_{i,\vartheta}\right|, \qquad (1)$$

where $\dagger$ is the matrix pseudoinverse, $\boldsymbol{\phi}(k,:)$ denotes the k-th row of $\boldsymbol{\phi}$, and the matrix $\boldsymbol{\phi}$ is formed by horizontal concatenation of column vectors $\phi_{l,r}$ of length $\left|M^h_{i,\vartheta}\right|$, each representing a bivariate monomial with non-negative integer exponents l, r for the polynomial approximation:

$$\boldsymbol{\phi} = [\phi_{0,0}\,, \cdots, \phi_{l,r}\,, \cdots]\,,$$
$$\phi_{l,r} = \begin{bmatrix} \left(x^{\mathcal{L}_i}_{m_1}\right)^l\left(y^{\mathcal{L}_i}_{m_1}\right)^r \\ \vdots \\ \left(x^{\mathcal{L}_i}_{m_k}\right)^l\left(y^{\mathcal{L}_i}_{m_k}\right)^r \\ \vdots \end{bmatrix}, \qquad \left\{m_k\right\}_{k=1}^{\left|M^h_{i,\vartheta}\right|} = M^h_{i,\vartheta}\,.$$

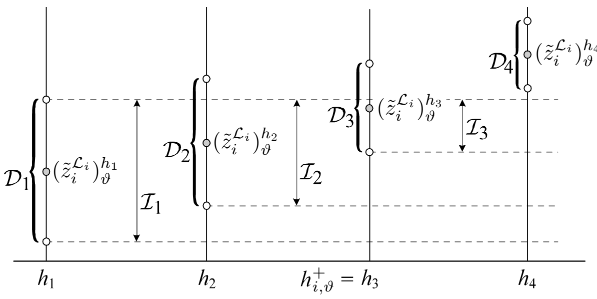
For an order-O LPA, the degree of each monomial $\phi_{l,r}$ entering $\boldsymbol{\phi}$ is restricted to $l+r \leq O$; thus the matrix $\boldsymbol{\phi}$ has size $\left|M^h_{i,\vartheta}\right| \times 3$ and $\left|M^h_{i,\vartheta}\right| \times 6$ for order-1 and order-2 LPA, respectively. In our algorithm we set $m_1 = i$ (i.e. the index of the current point p_i), thus $x^{\mathcal{L}_i}_{m_1} = y^{\mathcal{L}_i}_{m_1} = 0$ and for order-1 LPA, $\boldsymbol{\phi}(1,:) = [1,\,0,\,0]$.

These kernels yield a polynomial approximation of the normal elevation of $\mathcal{S}$ at p_i with respect to $\mathcal{L}_i$ as

$$\left(\tilde{z}^{\mathcal{L}_i}_i\right)^h_\vartheta = \sum_{k=1}^{\left|M^h_{i,\vartheta}\right|} z^{\mathcal{L}_i}_{m_k}\, g^h_{i,\vartheta}(k)\,. \qquad (2)$$

Due to the specific construction (1), the LPA approximation (2) is equivalent to first solving a least-squares fitting problem over the polynomials of order O,

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \sum_{k=1}^{\left|M^h_{i,\vartheta}\right|}\left(z^{\mathcal{L}_i}_{m_k} - \sum_{l+r<O}\alpha_{l,r}\left(x^{\mathcal{L}_i}_{m_k}\right)^l\left(y^{\mathcal{L}_i}_{m_k}\right)^r\right)^2, \qquad (3)$$



$\mathcal{D}_1$
$(\tilde{z}^{\mathcal{L}_i}_i)^{h_1}_\vartheta$
$\mathcal{I}_1$
$\mathcal{D}_2$
$(\tilde{z}^{\mathcal{L}_i}_i)^{h_2}_\vartheta$
$\mathcal{I}_2$
$\mathcal{D}_3$
$(\tilde{z}^{\mathcal{L}_i}_i)^{h_3}_\vartheta$
$\mathcal{I}_3$
$\mathcal{D}_4$
$(\tilde{z}^{\mathcal{L}_i}_i)^{h_4}_\vartheta$
h_1
h_2
$h^+_{i,\vartheta} = h_3$
h_4

Fig. 5. Illustration of the Intersection of Confidence Interval (ICI) rule.

and then sampling the fitted polynomial at $\left(x^{\mathcal{L}_i}_{m_1}\right)^l\left(y^{\mathcal{L}_i}_{m_1}\right)^r$:

$$\left(\tilde{z}^{\mathcal{L}_i}_i\right)^h_\vartheta = \sum_{l+r<O}\alpha^*_{l,r}\left(x^{\mathcal{L}_i}_{m_1}\right)^l\left(y^{\mathcal{L}_i}_{m_1}\right)^r.$$

4.3.3 *Adaptive size selection*

Anisotropy of the estimate entails the adaptive selection of the size of the directional neighborhood of each point and for each direction. We employ the ICI technique for this task.

Concretely, let $H = \{h_1, \ldots, h_J\} \subset \mathbb{R}^+$ be a predefined set of increasing sizes and compute (2) for each $h \in H$, using the same fixed polynomial order for all sizes and yielding a set of estimates $\{(\tilde{z}^{\mathcal{L}_i}_i)^{h_1}_\vartheta, \ldots, (\tilde{z}^{\mathcal{L}_i}_i)^{h_J}_\vartheta\}$. Since $M^h_{i,\vartheta}$ grows with the size h, these estimates are characterized by decreasing variance and increasing bias with h. Although the bias is unknown, the pointwise variance is easily computed as

$$\left(\sigma^h_{i,\vartheta}\right)^2 = \sigma^2\|g^h_{i,\vartheta}\|^2_2\,. \qquad (4)$$

The ICI rule selects from $\{(\tilde{z}^{\mathcal{L}_i}_i)^{h_1}_\vartheta, \ldots, (\tilde{z}^{\mathcal{L}_i}_i)^{h_J}_\vartheta\}$ an adaptive estimate $(\tilde{z}^{\mathcal{L}_i}_i)^{h^+_{i,\vartheta}}_\vartheta$ that aims at optimizing the bias-variance trade-off. Specifically, the ICI progressively intersects the confidence intervals associated to the estimates, starting from the smallest size h_1 and increasing the size as long as the intervals have a common non-empty intersection. The confidence interval $\mathcal{D}_j$ is computed as,

$$\mathcal{D}_j = \left[(\tilde{z}^{\mathcal{L}_i}_i)^{h_j}_\vartheta - \Gamma\sigma^{h_j}_{i,\vartheta},\ (\tilde{z}^{\mathcal{L}_i}_i)^{h_j}_\vartheta + \Gamma\sigma^{h_j}_{i,\vartheta}\right] \qquad (5)$$

where $\Gamma > 0$ is a threshold parameter.

The adaptive size $h^+_{i,\vartheta}$ is the largest one before the intersection is empty. The corresponding adaptive directional neighborhood is denoted as $u^+_{i,\vartheta}$. Ideally, if all estimates are unbiased, the corresponding confidence intervals have (with overwhelming probability), a common intersection which contains the true value of the normal elevation, and the ICI selects the largest neighborhood. An illustration of ICI rule is drawn in Fig. 5. The idea of the LPA-ICI is that, by monitoring the intersection of the confidence intervals, it is able to detect the increase of bias that results from the neighborhoods expanding over an edge or sharp features incompatible with the assumed polynomial smoothness, and it adaptively balances this with the decreasing variance, towards minimization of the mean squared error [6].

In Fig. 6, we show examples of the adaptive directional neighborhoods computed by the LPA-ICI for three points located within a smooth area, in the vicinity of an edge, or in the vicinity of a corner in a noisy Cube point cloud. Parameters of the ground-truth Cube point cloud, as well as

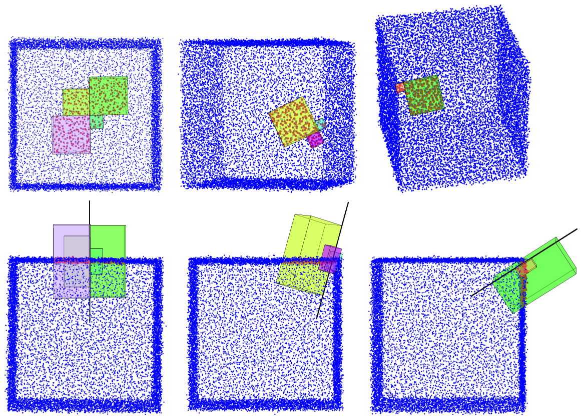


Fig. 6. Examples of the adaptive directional neighborhoods computed by LPA-ICI for three points located within a smooth area (left), in the vicinity of an edge (center), and in the vicinity of a corner (right) in the noisy Cube point cloud (noise $\sigma = 0.4$), as visualized from atop (top) and sideways (bottom). The points located inside the neighborhood are drawn in red. Observe that the directional neighborhoods are consistently small when approaching edges/corners, as also illustrated in Fig. 8.

TABLE 1
Parameters of the synthetic ground-truth and raw scanned point clouds used in the experiments. Please refer to the footnote[1] for more details.

	Point cloud	I (number of points)	δ (surface sample density)	diameter
Synthetic	Fandisk	6475	1.00	74.80
	Bunny	35947	1.00	161.63
	Armadillo	172974	1.00	395.03
	Cube	13826	1.00	82.02
	Dodecahedron	44228	1.00	124.61
	Sphere	10201	1.00	54.25
	Happy	100000	1.00	298.43
	Box Push	100000	1.00	208.66
	Dragon	100000	1.00	259.01
	Galera	100000	1.00	228.85
	Netsuke	100000	1.00	279.01
	Column Head	100000	1.00	189.32
Raw	Iron	161004	1.00	562.98
	Shutter Blind	291220	1.00	752.56

of the other point clouds used in the paper, are summarized in Table 1. Observe that the directional neighborhoods are consistently small when approaching sharp edges/corners. It can also be observed that, in the vicinity of edges and corners, the estimated normal e_i (illustrated by the black line in Fig. 6) can be inaccurate. However, such a moderately inaccurate normal does not compromise the final estimate of our algorithm, as it only marginally affects the adaptive directional size selection of LPA-ICI (see Fig. 7) and it is not used for defining the aggregated estimate in the following step (Section 4.5) of our algorithm.

The visual evaluation of the selection of the adaptive directional sizes is given in Fig. 8. The directional size of p_i along a specific direction θ (in the case of the figure, a direction parallel to an edge of the cube), is defined as

$$h^+_{i,\theta} = \frac{\sum_{\vartheta\in\Theta} h^+_{i,\vartheta} \max\{0, \langle\vartheta^{\mathcal{L}_i}, \theta\rangle\}}{\sum_{\vartheta\in\Theta} \max\{0, \langle\vartheta^{\mathcal{L}_i}, \theta\rangle\}}\,, \tag{6}$$

where $\langle\vartheta^{\mathcal{L}_i}, \theta\rangle$ is the inner product between the direction ϑ in the LCS of p_i and the direction θ, treated as two versors in $\mathbb{R}^3$. In the figure we see that points within the smooth faces have mostly larger directional sizes (indicated by bright colors), which decrease when moving toward edges in the direction indicated by the arrow. Points close to these edges are all with very dark colors, indicating small directional sizes, which reduces the risks of over-smoothing during our following denoising process.

4.4 Local estimates with adaptive directional neighborhood size

The local estimate with the adaptive directional size $h^+_{i,\vartheta}$ can be thus written with respect to the LCS of p_i as

$$\tilde{p}^{\mathcal{L}_i}_{i,\vartheta} = \left[0, 0, (\tilde{z}^{\mathcal{L}_i}_i)^{h^+_{i,\vartheta}}_{\vartheta}\right]^T, \tag{7}$$

or with respect to the canonical coordinates as

$$\tilde{p}_{i,\vartheta} = p_i + (\tilde{z}^{\mathcal{L}_i}_i)^{h^+_{i,\vartheta}}_{\vartheta} e_i\,, \tag{8}$$

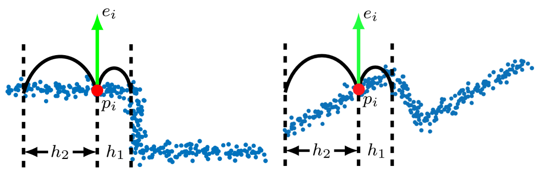


Fig. 7. Directional size selection on a 2-D noisy point cloud, based on an accurate normal estimation (left) and moderately inaccurate normal estimation (right), where LPA-ICI can still function correctly.

where e_i is the third principal component of the KNN of p_i, defining the LCS $\mathcal{L}_i$ as in Section 4.2.

In Fig. 8, a number of erroneous small sizes (i.e. dark color) in the middle of smooth faces can also be observed; these errors are due to tails of the noise distribution affecting the precision of LPA-ICI. Due to these errors, the individual local estimates $\tilde{p}_{i,\vartheta}$ (8) are in fact rather weak estimates of p_i; it is thus standard practice to strengthen the estimation by fusing or aggregating multiple LPA-ICI estimates (see, e.g., [31], [39]). A special aggregated estimate for point cloud denoising is introduced in the following subsection.

4.5 Aggregated estimate

For each point p_i, we have now four directional neighborhoods with adaptive sizes $h^+_{i,\vartheta}$, $\vartheta \in \Theta$ and corresponding estimates (7)-(8). There are in principle three alternative approaches to estimate $\hat{q}_i$: 1) combine the four adaptive estimates (8) $\tilde{p}_{i,\vartheta}$, $\vartheta \in \Theta$; 2) define a new local estimate based on all points in the union of the four directional

1. Armadillo and Bunny are from the Stanford 3D Scanning Repository [40]. Fandisk is downloaded from [41]. Cube, Dodecahedron and Sphere are made by the authors. Each face of the Cube is a regular grid 49×49 points. Happy, Box Push, Dragon, Galera, Netsuke and Column Head are from [42]. The raw scanned Iron and Shutter Blind point clouds are provided by [22]. We rescaled all point clouds to a common $\delta = 1$ to simplify the comparisons of results. The original δ of Armadillo, Bunny, Fandisk, Shutter Blind and Iron are 4.71, 664102.44, 883.99, 160253.81 and 1.65, respectively. The original δ of Happy, Box Push, Dragon, Galera, Netsuke and Column Head are 2047833, 228.03, 1509735.90, 1898.38, 11.53, 57.38, respectively. We compute the diameter of a point cloud as the largest distance among all points in that cloud.

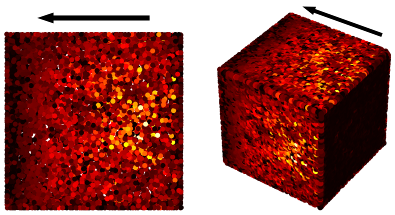


Fig. 8. Front and side views of the noisy Cube point cloud: each point's directional neighborhood size along the direction indicated by the arrow (which is parallel to an edge of the cube) is computed using (6); points with bright colors indicate large directional sizes and vice versa. Note that, in these plots the noisy points have been projected to the original noise-free Cube surfaces for better visualization purpose. Observe that the directional sizes decrease when approaching the edges pointed by the arrow, and they can be still large when close to the edges in the opposite direction, which is an advantage of using adaptive *directional* neighborhoods instead of adaptive *isotropic* ones, as illustrated in Fig. 1.

neighborhoods, i.e. $\{p_m, m \in M_{i,\vartheta}^{h_{i,\vartheta}^+}, \vartheta \in \Theta\}$; 3) define an aggregated estimate that simultaneously fits the many local estimates from overlapping adaptive directional neighborhoods of different points.

The first approach is the simplest but is known to be relatively weak and not particularly competitive already in the basic case of image denoising. The second approach assumes that the four neighborhoods share a unique underlying smooth surface, something which does not happen when p_i is on an edge.

To take advantage of the multiplicity of overlapping estimates for each point (see, e.g., Fig. 9), our method develops along the third approach. In particular, we define our estimate as

$$\hat{q}_i = \underset{q}{\operatorname{argmin}} \Bigg(\sum_{(n,\vartheta)\in\mathcal{K}_i} w_{n,\vartheta}^2 \, \mathrm{d}^2(q, \tilde{\mathcal{S}}_{n,\vartheta}^+) + \lambda^2 \sigma^{-2} \|q - p_i\|_2^2 \sum_{(n,\vartheta)\in\mathcal{K}_i} w_{n,\vartheta}^2 \Bigg), \tag{9}$$

where $\mathcal{K}_i$ provides the point and direction indices (n,ϑ) of any adaptive directional neighborhood $u_{n,\vartheta}^+$ containing p_i, $\tilde{\mathcal{S}}_{n,\vartheta}^+$ is the polynomial surface fitted by the LPA on the adaptive directional neighborhood $u_{n,\vartheta}^+$, d is the point-to-surface distance, $\lambda > 0$ is a regularization parameter, and $w_{n,\vartheta}^2 = \varepsilon_{n,\vartheta}^{-2}$ are weights inversely proportional to the average quadratic fit of the surface to the data

$$\varepsilon_{n,\vartheta}^2 = \left| M_{n,\vartheta}^{h_{n,\vartheta}^+} \right|^{-1} \sum_{m\in M_{n,\vartheta}^{h_{n,\vartheta}^+}} \mathrm{d}^2(p_m, \tilde{\mathcal{S}}_{n,\vartheta}^+), \tag{10}$$

$\left| M_{n,\vartheta}^{h_{n,\vartheta}^+} \right|$ being the number of points included by $u_{n,\vartheta}^+$.

The first addend in (9) promotes estimates $\hat{q}_i$ that are close to the polynomial surfaces fitted within adaptive neighborhoods that contain the noisy point p_i. As illustrated in Fig. 10, since the point-to-surface distance is measured along the surface normals, minimization of the first addend alone (i.e. $\lambda = 0$) may elicit large drifts of the estimates along directions tangential to the surfaces; the second addend in (9) is a quadratic regularization term that prevents such large drifts. The weights $w_{n,\vartheta}$ give more importance to surfaces $\tilde{\mathcal{S}}_{n,\vartheta}^+$ that enjoy a better fit to the data in their own adaptive neighborhood. Ideally, the fit should be tested

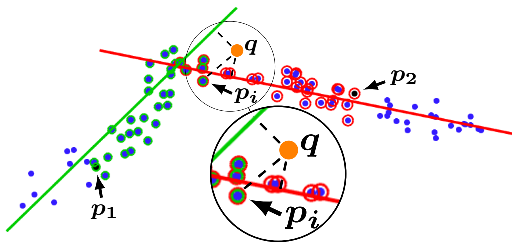


Fig. 9. 2-D illustration of the quantities being summed in (9). The green and red lines represent two order-1 LPA surfaces $\tilde{\mathcal{S}}_{n,\vartheta}^+$ computed using noisy points that belong to the directional neighborhoods of p_1 and p_2, respectively. Points in the neighborhood of p_1 are circled in green, and points in the neighborhood of p_2 are circled in red. Some points (including p_i) belong simultaneously to both neighborhoods, and are thus circled in both green and red. The dashed black lines visualize the distances $\mathrm{d}^2(q, \tilde{\mathcal{S}}_{n,\vartheta}^+)$ of a candidate point q to the two lines, as well as the distance of q to p_i. In this example, we see that the green line fits the points circled in green not as well as the red line fits the points circled in red; hence, in the weighted sum, the distance between q and the green line gets a smaller weight $w_{n,\vartheta}$ than the distance between q and the red line. The minimization of (9) is illustrated in Fig. 10.

against the noise-free data, i.e. replacing in (10) p_m by q_m; however the latter is not available.

Order-1 case

For order-1 LPA, the surfaces $\tilde{\mathcal{S}}_{n,\vartheta}^+$ are flat planes. Each of these planes is characterized by the point $\tilde{p}_{n,\vartheta}$ which lies within the plane, and by the normal vector to the plane, which we denote by $\nu_{n,\vartheta}$. Note that $\nu_{n,\vartheta}$ is not the vector e_n used for LCS (Section 4.2). In fact, $\nu_{n,\vartheta} = [c_n, d_n, e_n]\nu_{n,\vartheta}^{\mathcal{L}_n}$, where $\nu_{n,\vartheta}^{\mathcal{L}_n}$ is the equivalent LCS representation of $\nu_{n,\vartheta}$. This $\nu_{n,\vartheta}^{\mathcal{L}_n}$ can be conveniently computed via (2) using LPA kernels corresponding to the $x^{\mathcal{L}_n}$ and $y^{\mathcal{L}_n}$ components of the fitted order-1 polynomial (3) as

$$\nu_{n,\vartheta}^{\mathcal{L}_n} = \begin{bmatrix} -\sum_{k=1}^{\left|M_{n,\vartheta}^{h_{n,\vartheta}^+}\right|} z_{m_k}^{\mathcal{L}_n} \, \boldsymbol{\phi}(k,:) \left(\boldsymbol{\phi}^T \boldsymbol{\phi}\right)^{\dagger} [0 \;\, 1 \;\, 0]^T \\ -\sum_{k=1}^{\left|M_{n,\vartheta}^{h_{n,\vartheta}^+}\right|} z_{m_k}^{\mathcal{L}_n} \, \boldsymbol{\phi}(k,:) \left(\boldsymbol{\phi}^T \boldsymbol{\phi}\right)^{\dagger} [0 \;\, 0 \;\, 1]^T \\ 1 \end{bmatrix}.$$

Then (9) becomes

$$\hat{q}_i = \underset{q}{\operatorname{argmin}} \Bigg(\sum_{(n,\vartheta)\in\mathcal{K}_i} w_{n,\vartheta}^2 \left(\langle q - \tilde{p}_{n,\vartheta}, \nu_{n,\vartheta} \rangle\right)^2 + \lambda^2 \sigma^{-2} \|q - p_i\|_2^2 \sum_{(n,\vartheta)\in\mathcal{K}_i} w_{n,\vartheta}^2 \Bigg), \tag{11}$$

which is quadratic on q and can be solved by zeroing the gradient as

$$\sum_{(n,\vartheta)\in\mathcal{K}_i} w_{n,\vartheta}^2 \nu_{n,\vartheta} \langle \hat{q}_i - \tilde{p}_{n,\vartheta}, \nu_{n,\vartheta} \rangle + \frac{\lambda^2}{\sigma^2} (\hat{q}_i - p_i) \sum_{(n,\vartheta)\in\mathcal{K}_i} w_{n,\vartheta}^2 = 0, \tag{12}$$

i.e.

$$\sum_{(n,\vartheta)\in\mathcal{K}_i} w_{n,\vartheta}^2 \left(\nu_{n,\vartheta} \langle \hat{q}_i, \nu_{n,\vartheta} \rangle + \lambda^2 \sigma^{-2} \hat{q}_i \right) = \sum_{(n,\vartheta)\in\mathcal{K}_i} w_{n,\vartheta}^2 \left(\nu_{n,\vartheta} \langle \tilde{p}_{n,\vartheta}, \nu_{n,\vartheta} \rangle + \lambda^2 \sigma^{-2} p_i \right), \tag{13}$$

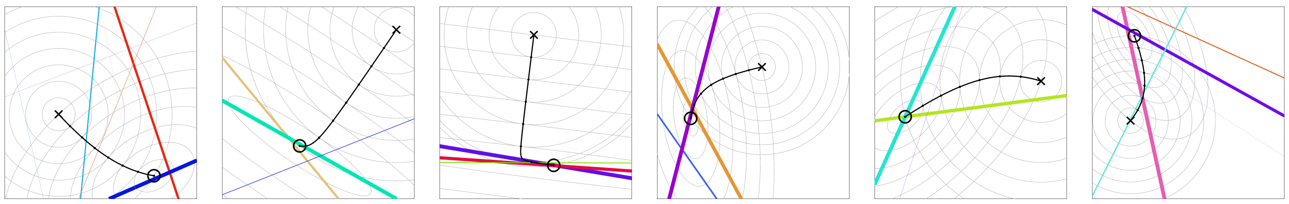


Fig. 10. 2-D illustrations of the minimization (9). The colored straight lines represent a number of fitted order-1 LPA surfaces $\tilde{\mathcal{S}}^{+}_{n,\vartheta}$; the lines are drawn with thickness proportional to their respective weights $w_{n,\vartheta}$. The solid black line shows the trajectory of the solution $\hat{q}_i$ for λ ranging from 0 (marked by circle, coinciding with the minimizer of the first addend in (9)) to $+\infty$ (marked by cross, trivially coinciding with the noisy point p_i). In the order-1 case, both addends in (9) are quadratic, resulting in elliptical contours for the first addend (where the ellipticity depend on weights and orientation of the fitted lines) and in circular contours for the second addend. For any given value of λ, the solution $\hat{q}_i$ is located where elliptical and circular contours are tangent to each other. All denoising experiments reported in the paper use a fixed value of $\lambda = 0.06/\sqrt{\hat{\delta}}$ (see Section 5.2).

which using matrix-vector notation becomes $A\hat{q}_i = b$, where

$$A = \sum_{(n,\vartheta)\in\mathcal{K}_i} w^2_{n,\vartheta}\left(\nu_{n,\vartheta}\,\nu^T_{n,\vartheta} + \lambda^2\sigma^{-2}\mathbf{1}\right), \qquad (14)$$

$$b = \sum_{(n,\vartheta)\in\mathcal{K}_i} w^2_{n,\vartheta}\left(\nu_{n,\vartheta}\,\nu^T_{n,\vartheta}\,\tilde{p}_{n,\vartheta} + \lambda^2\sigma^{-2}p_i\right). \qquad (15)$$

Thus, $\hat{q}_i$ is obtained by left matrix division of the 3×3 matrix A into the vector b. This solution is particularly efficient since it does not require storing in memory the multiple estimates for $(n,\vartheta)\in\mathcal{K}_i$, as these are progressively aggregated into the A and b buffers.

If $\left|M^{h^+_{n,\vartheta}}_{n,\vartheta}\right| < 3$, the local estimate is discarded, as the plane $\tilde{\mathcal{S}}^{+}_{n,\vartheta}$ and its normal $\nu_{n,\vartheta}$ could not be defined. For the order-1 case, we also found empirically that the weights

$$w_{n,\vartheta} = \min\left\{\left(\varepsilon^2_{n,\vartheta} - \frac{3\sigma^2}{4}\right)^{-1},\ \frac{1}{\sqrt{2\sigma^2}}\left|M^{h^+_{n,\vartheta}}_{n,\vartheta}\right|^{\frac{1}{2}}\right\} \qquad (16)$$

yield better results, as they mimic ideal weights computed from q_m balancing the fit error with the variance error due overfit given few noisy samples in the neighborhood.

All experiments presented in the remainder are based on the order-1 case.

4.6 Further denoising iterations and residual noise variance

Further iterations of the denoising can improve the quality of the estimate. Analogous to the first iteration, three inputs are required (see Fig. 2): the denoised point cloud $\hat{Q}$ from the previous iteration, its residual noise variance (denoted by $\tilde{\sigma}^2_i$ for $\hat{q}_i \in \hat{Q}$), and the surface sample density, which remains approximately equal to δ.

We model the variance of the residual noise after the first iteration as

$$\tilde{\sigma}^2_i = (1.0806\varrho_i - 0.2424\sigma)^2, \qquad (17)$$

$$\varrho^2_i = \frac{\sum_{(n,\vartheta)\in\mathcal{K}_i}\left(\sigma^{h^+}_{n,\vartheta}\right)^2}{\sum_{(n,\vartheta)\in\mathcal{K}_i}1} = \frac{\sum_{(n,\vartheta)\in\mathcal{K}_i}\sigma^2\|g^{h^+}_{n,\vartheta}\|^2_2}{\sum_{(n,\vartheta)\in\mathcal{K}_i}1}, \qquad (18)$$

where ϱ^2_i is a compound variance that averages all pointwise variances $(\sigma^{h^+}_{n,\vartheta})^2$ (4) of the directional estimate (7) with the adaptive size $h^+_{n,\vartheta}$ that concur to the estimation of $\hat{q}_i$.

The coefficients in (17) were fitted by linear regression between $\{\varrho_i, i = 1,\ldots,I\}$ and the standard deviations of $\{\hat{q}_i, i = 1,\ldots,I\}$ measured from Monte-Carlo simulations (using all the synthetic point clouds in Table 1).

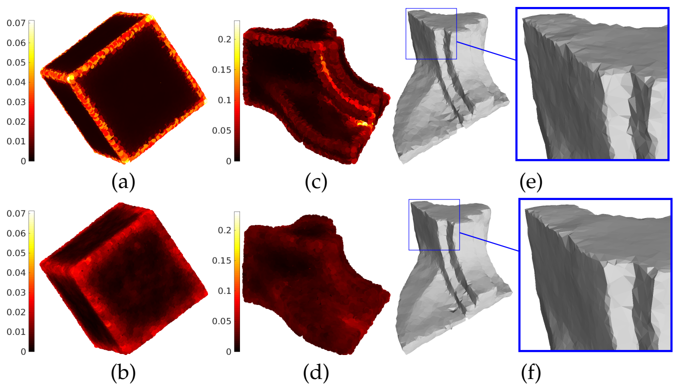


Fig. 11. (a) & (c) Residual noise variance distribution after the first denoising iteration, computed from Monte-Carlo experiments on the Cube and Fandisk point clouds by adding noise with σ = 0.4; (b) & (d) the distribution of our modeled residual variances (17); points with bright colors have large variances and vice versa; (e) & (f) reconstructed 3D surfaces of the denoised Fandisk point clouds after the first and second denoising iteration.

Fig. 11 (a)-(d) illustrates the qualitative and quantitative match of the distribution of $\tilde{\sigma}^2_i$ with ground-truth computed via Monte-Carlo experiments (shown here only for reference, and not available in the operation of the proposed algorithm).

Similar to earlier work [43] on iterative LPA-ICI denoising, the variances $\tilde{\sigma}^2_i$ are further scaled by a fixed constant $0 < \alpha \leq 1$, in order to compensate for the fact that residual noise after the first iteration is no longer spatially independent but features correlation due to the operated filtering.

Whereas in the first iteration σ is a constant value for all the points, in the second iteration different points are subject to different noise variances. Thus, formulas in Sections 4.3.1-4.5 hold upon replacing σ with $\alpha\tilde{\sigma}_i$ for the corresponding $\hat{q}_i$. Specifically, these straightforward replacements apply to the height of the prism in Section 4.3.1, to equations (4), (9), (12), (13), (14), and (15), while in (16), for each $(n,\vartheta)\in\mathcal{K}_i$, we replace σ^2 with $\left(\sum_{m\in M^{h^+_{n,\vartheta}}_{n,\vartheta}}\alpha^2\tilde{\sigma}^2_m\right)/\left|M^{h^+_{n,\vartheta}}_{n,\vartheta}\right|$, i.e. the average variance over points $\{p_m, m\in M^{h^+_{n,\vartheta}}_{n,\vartheta}\}$.

Fig. 11 (e-f) provides a visual comparison of the denoised Fandisk point cloud after the first and second iteration, illustrating the improvement in the filtering along edges and in smooth regions.

In principle, a third and more iterations could be added also. However, we did not investigate this possibility noting that the benefit after the second iteration is typically only

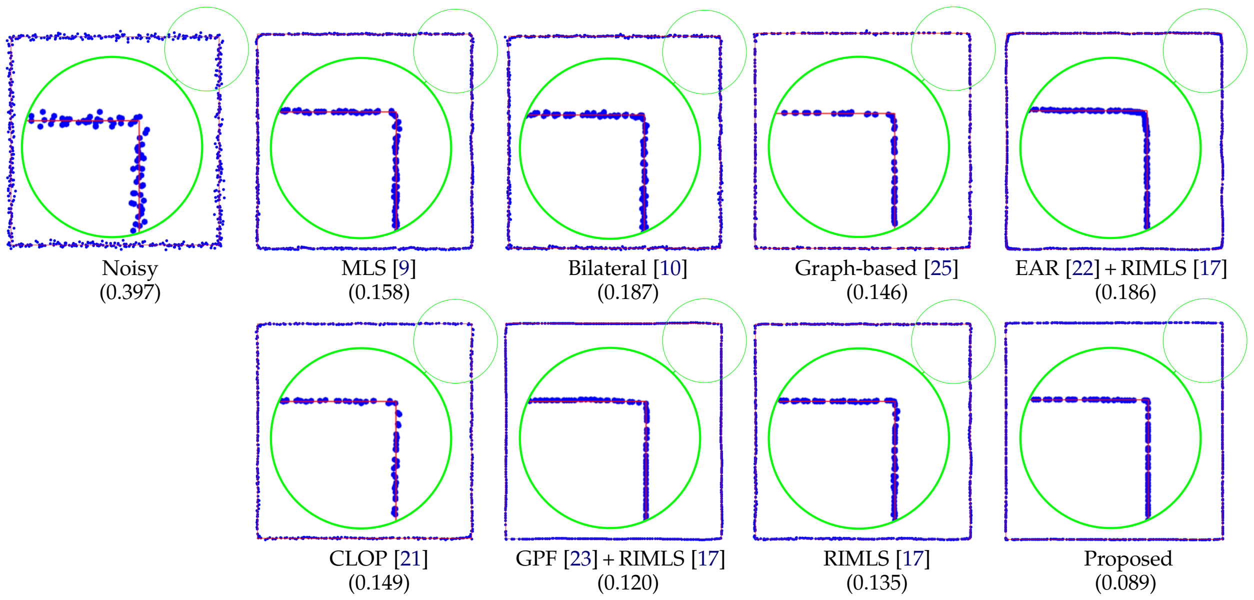


Fig. 12. The cross-section view of the noisy Cube point cloud (σ = 0.4) and the denoised point clouds from different algorithms, where the cross-section view of ground-truth surface is indicated by red lines. The root mean squared point-to-surface distance (RMSD$_\perp$) of each result is reported within parentheses. Observe that our denoised result is tightly aligned with the ground-truth surface, including sharp edges, as also suggested by its RMSD$_\perp$, which is the smallest of all. Some reconstructed surfaces of the denoised Cube are presented in Fig. 13.

marginal [43] and the modeling of the residual noise gets challenging.

5 Experimental Results and Analysis

We performed extensive experiments to validate the accuracy and advantages of the proposed denoising algorithm, as detailed in the following subsections.

All the reported results by the proposed method can be reproduced by the software included in the supplementary materials.

5.1 Comparison methods

The compared popular and state-of-the-art point cloud denoising algorithms include the classical MLS [9] method, the bilateral filter [10], the graph-based regularization algorithm [25], PointCleanNet (PCN) [32], edge-aware resampling (EAR) [22], continuous LOP (CLOP) [21], GMM-inspired filter (GPF) [23], and RIMLS [17]. The MLS method is implemented using the Point Cloud Library (PCL) [44]. The bilateral filter, graph-based regularization algorithm, PCN, EAR, CLOP and GPF are implemented from the source codes provided by their respective authors. For RIMLS we used the corresponding function integrated in the Meshlab software [45]. For PCN, we applied the denoising script with 11 iterations, which is the same as in the experiments in [32]. For GPF, we do not execute the gap filling but only the edge-aware upsampling, as we noticed the former would worsen the results. For EAR and GPF, we applied RIMLS post-filtering, following the authors' suggested pipeline. Both MLS and RIMLS are based on a second-order polynomial surface regression.

5.2 Parameter settings

Unless otherwise noted, the reported results by the proposed algorithm are obtained using two iterations and the following parameters: KNN $K = 50$; LPA order is 1 and LPA sizes

$$H = \{0, 3(\sqrt{2})^0/\sqrt{\hat{\delta}}, 3(\sqrt{2})^1/\sqrt{\hat{\delta}}, \ldots, 3(\sqrt{2})^4/\sqrt{\hat{\delta}}\}, \quad (19)$$

where $\hat{\delta}$ is the estimated surface sample density of the considered noisy point cloud; ICI threshold $\Gamma = 0.55$ (first iteration) and $\Gamma = 0.85$ (second iteration); $\alpha = 0.533$; the regularization parameter λ in (9) is set to $0.06/\sqrt{\hat{\delta}}$. Our algorithm is applied using values of $\hat{\sigma}$ and $\hat{\delta}$ estimated by the methods described in the Appendix I, thus making the procedure wholly unsupervised.

It is worth noting that the value of the ICI threshold Γ directly affects the quality of the output: too large or too small Γ result in either oversmoothing or underfiltering. In particular, we illustrate the effect of using very large Γ in Section 5.7. The chosen values $\Gamma = 0.55, 0.85$ are quantitatively consistent with those used in earlier works using directional LPA-ICI for denoising (e.g., $\Gamma = 0.8$ for [43]). Likewise, the exponential rule adopted for H (19), where the base surface of the directional neighborhood doubles at each next scale, is standard in multiscale and multiresolution signal processing. The factor δ provides normalization with respect to the surface density of the point cloud. The maximum value of H determines the strongest smoothing possible with one iteration of the algorithm. The minimum non-zero value $h_1 \in H$ determines a neighborhood base surface of $9/\hat{\delta}$ so that $M_{i,\vartheta}^{h_1} = 9$ on average, enabling a meaningful definition of the LPA kernel (1).

5.3 Experiments on synthetic data

The ground-truth synthetic point clouds used in the experiments are specified in Table 1. The noisy ones are obtained by adding Gaussian white noise with different σ values to the ground-truth point clouds. We provide in supplementary materials all the noisy and ground-truth point clouds.

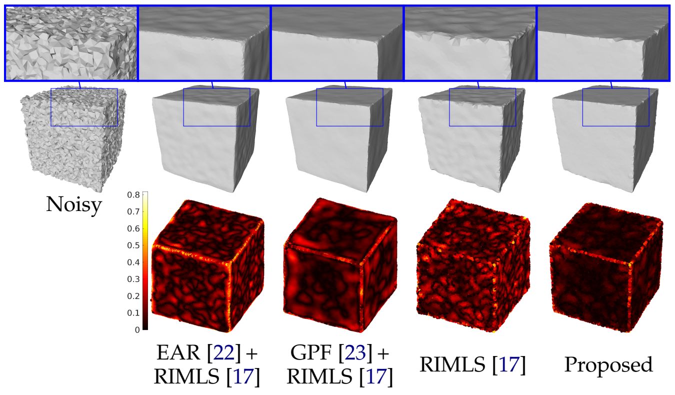


Fig. 13. The same noisy ($\sigma = 0.4$) and denoised Cube point clouds as in Fig. 12, with reconstructed 3D surfaces (top) and with the color of each point indicating its point-to-surface distance (bottom), for some competitive methods and the proposed algorithm. The breaks on the edges of the reconstructed surface for RIMLS and our results are due to lower number of points (number of points = 13826), while results from EAR and GPF are both upsampled following their pipelines (number of points = 43828 for both).

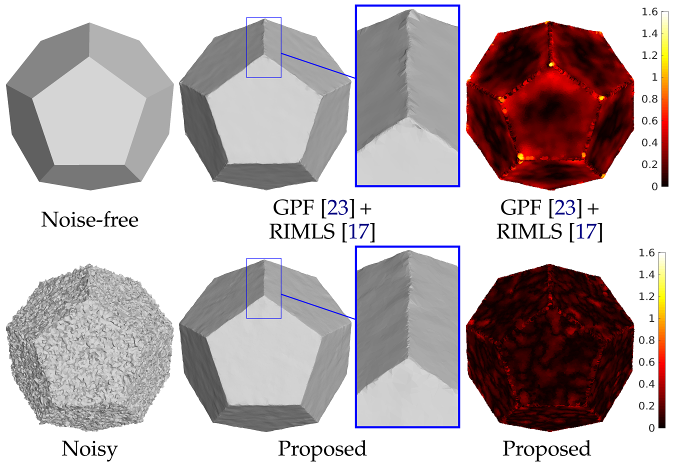


Fig. 14. Left and center: reconstructed 3D surfaces of noise-free, noisy (σ = 0.638) and denoised Dodecahedron point cloud results from GPF and the proposed algorithm; right: corresponding denoised point clouds, where the color indicates the point-to-surface distance. The root mean squared point-to-surface distances of the noisy and denoised point clouds from GPF and our method are 0.630, 0.251 and 0.109, respectively.

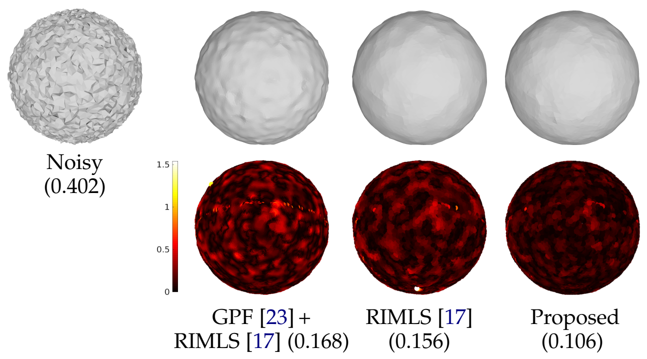


Fig. 15. Top: reconstructed 3D surfaces of the noisy Sphere point cloud (σ = 0.4) and the denoised results from different algorithms; bottom: the corresponding denoised point clouds with the color of each point indicating its point-to-surface distance against the ground truth. The root mean squared point-to-surface distances of the noisy and denoised point clouds are reported within parentheses.

With the availability of ground truth, we provide quantitative denoising results of the experiments on synthetic point clouds, where both point-to-surface distance and normal angular error are evaluated. The point-to-surface distance is also represented by colors in the plots, where brighter colors indicates larger distances and vice versa. Description of the error metrics is in Appendix II.

Visual results are presented with reconstructed 3D surfaces of the denoised point clouds. All the surfaces in the paper are reconstructed by the software [46].

The parameters of the comparison algorithms were manually tuned in order to minimize the root mean squared point-to-surface distance separately for each of the considered noise levels on the synthetic point clouds. The proposed algorithm uses instead the above-mentioned single set of parameters for all noise levels, i.e. it is fully unsupervised.

The cross-sections in Fig. 12 show that the denoised points by the proposed method are closely aligned with the underlying ground-truth surface (indicated by red lines), within smooth areas as well as at sharp edges, whereas the other methods either leave some noise (e.g., CLOP and RIMLS) or yield positional distortion around sharp edges (e.g., EAR and GPF). This behaviour can be observed also in Fig. 13, where the proposed method reconstructs smoother and flatter cube faces and sharper and straighter edges, as confirmed by the darker colors in the point-to-surface distances plot. A similar performance advantage of our algorithm is demonstrated in Fig. 14, which reproduces an experiment on the Dodecahedron point cloud from [23].

For denoising those point clouds with curved shapes, our algorithm, even using order-1 LPA approximation, still outperforms the others as shown in Fig. 15 for the Sphere point cloud denoising, thanks to its adaptivity and the strength of aggregated estimation.

Denoising results on point clouds of more complex shape, including Fandisk, Bunny and Armadillo, further attest the effectiveness of our algorithm. Quantitative results in Table 2 show that our proposed algorithm attains the smallest point-to-surface distance in the near totality of cases. Although not specifically designed for the normal estimation task (a problem which is inherently different from denoising), our method does achieve highly competitive results, with both low Root Mean Squared Angular Error with threshold (RMSAE$_\tau$) and large Proportion-of-Good-Points (PGP) values, as reported in Table 3.

Visual assessment of some of these denoised point clouds are presented in Fig. 16, 18, and 19. It can be observed that, for all three noise levels, our algorithm not only well preserves the sharp features (e.g., sharp edges in Fandisk and Bunny, teeth and claw tips of Armadillo), but also filters the noise better in smooth areas (e.g., back of Bunny and chest of Armadillo). Other methods seem to encounter more difficulty at balancing this trade-off, e.g., the over-smoothed edges from MLS in Fig. 16, or noise-remaining smooth areas such as the back of Bunny from RIMLS in Fig. 18 and the chest of Armadillo in MLS and RIMLS results in Fig. 19, although they attain similar quantitative performance in these two cases in Table 2 and 3.

Similar to Fig. 13, 14, and 15, in Fig. 17, the uniformly dark colors on our denoised point clouds indicate the consis-

TABLE 2
Root mean squared point-to-surface distance (RMSD$_{\perp}$) results for noisy and denoised point clouds. The green and red background colors indicate the best and second best results, respectively.

	σ	Root mean squared point-to-surface distance									
		Noisy	MLS [9]	Bilateral [10]	Graph-based [25]	PCN [32]	EAR [22] + RIMLS [17]	CLOP [21]	GPF [23] + RIMLS [17]	RIMLS [17]	Proposed
Fandisk	0.2	0.1978	0.1532	0.1372	0.1300	0.1976	0.1776	0.1374	0.1189	0.1011	0.0979
	0.4	0.3910	0.2230	0.2261	0.2240	0.3896	0.2200	0.2054	0.2154	0.1707	0.1694
	0.8	0.7565	0.3770	0.4208	0.3645	0.7535	0.3272	0.3574	0.4385	0.3202	0.2963
Bunny	0.2	0.2002	0.0960	0.1053	0.1350	0.2352	0.1643	0.1007	0.1229	0.0958	0.0908
	0.4	0.3980	0.1648	0.1846	0.2137	0.3243	0.2308	0.1614	0.2107	0.1572	0.1423
	0.8	0.7854	0.2931	0.3341	0.3558	0.4129	0.3057	0.2789	0.3679	0.2525	0.2311
Arma-dillo	0.2	0.1999	0.0963	0.1148	0.1376	0.3403	0.1557	0.1016	0.1363	0.0982	0.0981
	0.4	0.3979	0.1627	0.1880	0.2173	0.4377	0.2269	0.1672	0.2381	0.1596	0.1562
	0.8	0.7922	0.2833	0.3381	0.3616	0.5866	0.3250	0.2783	0.4151	0.2669	0.2544

TABLE 3
Root mean squared angular error with threshold (RMSAE$_{\tau}$) of normals and the proportion of good points (PGP) in parentheses for noisy and denoised point clouds (definitions of RMSAE$_{\tau}$ and PGP are given in Appendix II). The green and red background colors indicate the best and second best results, respectively.

	σ	RMSAE$_{\tau}$ (PGP in parentheses)									
		Noisy	MLS [9]	Bilateral [10]	Graph-based [25]	PCN [32]	EAR [22] + RIMLS [17]	CLOP [21]	GPF [23] + RIMLS [17]	RIMLS [17]	Proposed
Fandisk	0.2	1.1890 (42.93%)	0.7722 (76.06%)	0.7287 (78.69%)	0.8004 (74.22%)	1.1892 (42.92%)	0.8438 (71.33%)	0.8133 (73.35%)	0.6708 (81.92%)	0.6482 (83.15%)	0.6843 (81.11%)
	0.4	1.4859 (10.58%)	1.0152 (58.49%)	1.1450 (47.09%)	1.0693 (53.85%)	1.4863 (10.53%)	0.9448 (64.08%)	0.9588 (62.97%)	0.8891 (68.14%)	0.7953 (74.59%)	0.8127 (73.39%)
	0.8	1.5443 (3.37%)	1.3099 (30.63%)	1.4027 (20.37%)	1.3967 (21.03%)	1.5438 (3.43%)	1.1089 (50.37%)	1.2547 (36.38%)	1.1000 (51.17%)	1.0418 (56.23%)	1.0689 (53.88%)
Bunny	0.2	1.2121 (40.68%)	0.5368 (88.61%)	0.5210 (89.27%)	0.9474 (63.91%)	1.1720 (44.57%)	0.8286 (72.47%)	0.5195 (89.31%)	0.6642 (82.43%)	0.5274 (88.99%)	0.5301 (88.86%)
	0.4	1.4895 (10.14%)	0.8334 (72.17%)	1.0315 (57.16%)	1.2935 (32.35%)	1.4143 (19.05%)	1.0383 (56.59%)	0.7154 (79.58%)	0.9817 (61.25%)	0.7160 (79.54%)	0.7058 (80.10%)
	0.8	1.5470 (3.02%)	1.1456 (47.07%)	1.3498 (26.30%)	1.4803 (11.25%)	1.4721 (12.25%)	1.0589 (54.83%)	1.0223 (57.93%)	0.9605 (62.90%)	0.8654 (69.95%)	0.8748 (69.28%)
Armadillo	0.2	1.2311 (38.79%)	0.5660 (87.32%)	0.5885 (86.27%)	0.9802 (61.34%)	1.2769 (34.11%)	0.7925 (74.86%)	0.6037 (85.52%)	0.6744 (81.86%)	0.5726 (87.01%)	0.6036 (85.52%)
	0.4	1.4954 (9.42%)	0.8778 (69.10%)	1.0743 (53.49%)	1.3228 (29.24%)	1.4627 (13.37%)	1.0989 (51.32%)	0.8195 (73.10%)	0.9888 (60.67%)	0.8415 (71.62%)	0.8379 (71.85%)
	0.8	1.5471 (3.01%)	1.1867 (43.16%)	1.3723 (23.81%)	1.4906 (10.00%)	1.5083 (7.85%)	1.1022 (51.02%)	1.0912 (52.02%)	1.1428 (47.31%)	1.0233 (57.84%)	1.0145 (58.56%)

tently low point-to-surface distances on both smooth areas and sharp edge regions.

We observe in Table 2 and Table 3 that the PCN denoising method is rather ineffective at filtering these noisy point clouds, which have relatively lower noise and lower sampling density than those on which this method had been trained. Therefore, to enable a fair comparison against this recent method, we provide a separate set of experiments on the noisy point clouds from the PCN's very own testing dataset, as well as on noisier versions of some of the other point clouds. These are challenging experiments on extremely noisy data, as illustrated by Fig. 20. The quantitative results in Table 4 show that PCN attains a denoising quality that is competitive with that of RIMLS, while the proposed method is able to outperform them both, and often by a very significant margin.

5.4 Experiments on raw scanned data

We ran experiments also on two raw scanned point clouds, namely Iron and Shutter Blind. These point clouds are included in supplementary materials and their parameters are listed in Table 1. Ground-truth data is not available, preventing a quantitative evaluation, we therefore present only visual results in Fig. 21 and 22. As always, our algorithm is applied with the parameters specified in Section 5.2 and using the noise variances estimated on each noisy point cloud (see Appendix I). Parameters of the compared algorithms are tuned by minimizing the root mean squared point-to-surface distance on the synthetic Fandisk, Bunny and Armadillo point clouds corrupted by noise with the variance estimated on the raw scanned point clouds. In this way we ensure that the other algorithms are applied using ideal parameters for this noise level. Similar to the results on synthetic data, our algorithm demonstrates superior recovery of smooth areas (e.g., the smooth region enclosed in the blue box in Fig. 21 and in the red box in Fig. 22) and of fine features and sharp edges (e.g., the region in blue boxes in Fig. 21 and 22).

5.5 Processing times

The average processing times for denoising the Bunny point clouds with different noise levels (as shown in Table 2 and Fig. 18) is reported in Table 5. For all algorithms, the processing time changes roughly proportional to the number of points in a point cloud.

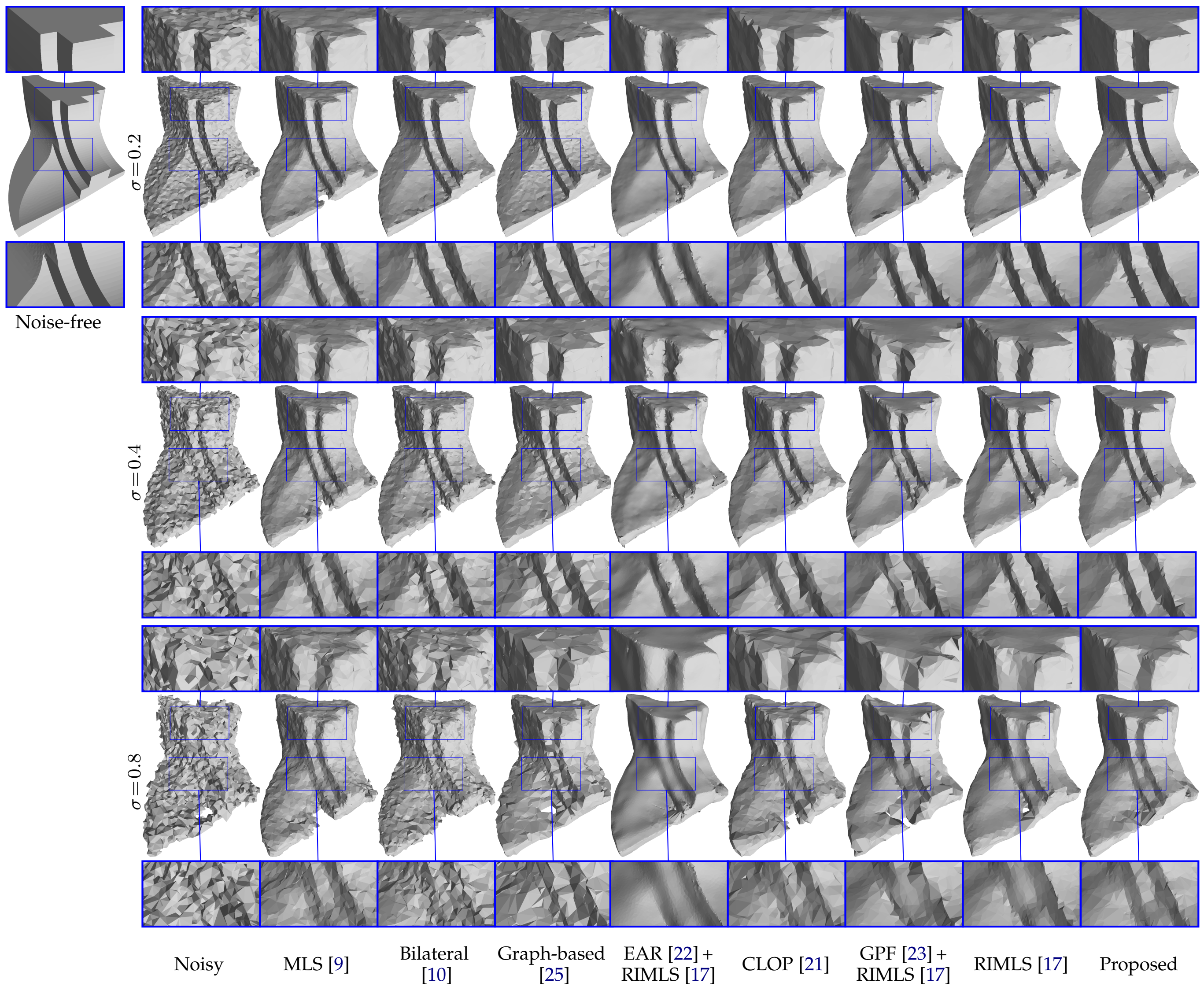


Fig. 16. Reconstructed 3D surfaces of noise-free, noisy, and denoised Fandisk point clouds. Fig. 17 illustrates these results with color-coding based on the point-to-surface distance to the ground truth. Zoomed-in view is shown in blue box for each result.

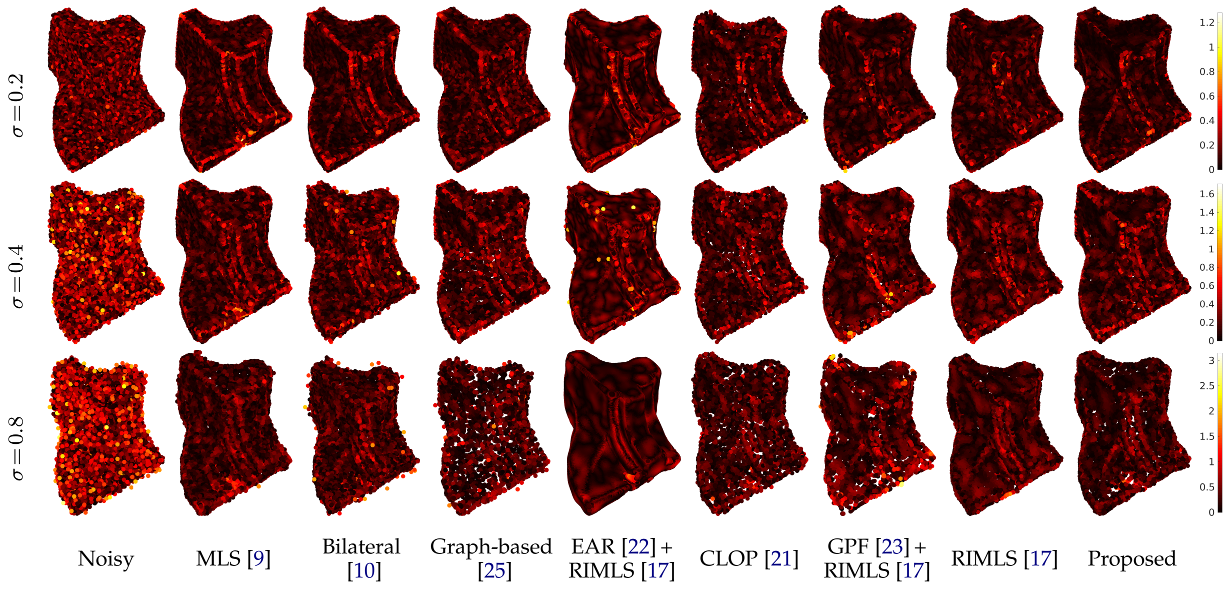


Fig. 17. The same noisy and denoised Fandisk point clouds as in Fig. 16, with the color of each point indicating its point-to-surface distance. The figures in each row share a common color scaling.

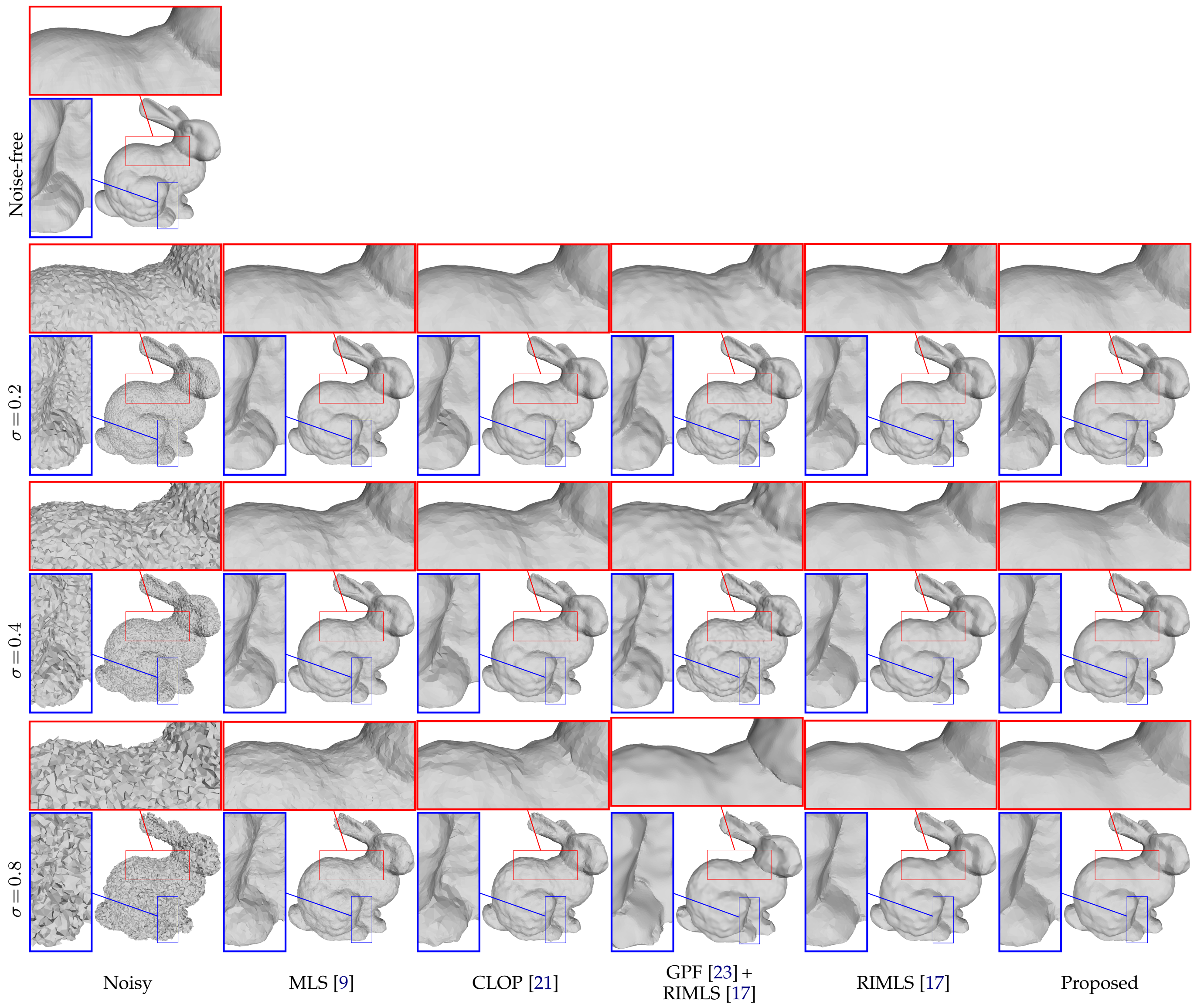


Fig. 18. Reconstructed 3D surfaces of noise-free, noisy, and denoised Bunny point clouds.

The proposed algorithm enjoys a graceful quality/complexity trade-off, since the aggregated estimate (Section 4.5) can be computed from a reduced number of local estimates. Specifically, and referring to (9), in order to define $\hat{q}_i$ it is enough that the set $\mathcal{K}_i$ contains at least one pair of indices, i.e. that p_i has been included in at least one directional neighborhood for some point in the cloud. Thus, it is not necessary to compute local estimates for every $i \in \{1, \ldots, I\}$. Furthermore, the algorithm can be speeded up also by reducing the largest neighborhood size in H. To evaluate these options quantitatively, we experimented using different proportions of local estimates (where 100% means computing local estimates for every $i \in \{1, \ldots, I\}$) and by restricting the LPA sizes to two subsets of the default full set (19), namely $H = \{0, 3(\sqrt{2})^0/\sqrt{\hat{\delta}}, \ldots, 3(\sqrt{2})^3/\sqrt{\hat{\delta}}\}$ and $H = \{0, 3(\sqrt{2})^0/\sqrt{\hat{\delta}}, \ldots, 3(\sqrt{2})^2/\sqrt{\hat{\delta}}\}$, for which $\max\{H\}\sqrt{\hat{\delta}}$ is respectively equal to $3(\sqrt{2})^3$ and 6, as opposed to $\max\{H\}\sqrt{\hat{\delta}} = 12$ (19). Fig. 23 summarizes the results obtained by denoising several noisy point clouds (σ = 0.2, 0.4, 0.8) in terms of relative root mean squared point-to-surface distance (relative RMSD$_\perp$) versus relative processing time, i.e. the ratios between the RMSD$_\perp$ and processing time obtained with reduced settings and those using (19) and 100% of the local estimates. An effective graceful trade-off corresponds to being able to significantly reduce the relative processing time while maintaining the relative RMSD$_\perp$ close to 1. The results show that one can halve the processing times simply by computing only a smaller portion of local estimates, and with negligible impact on the RMSD$_\perp$. The processing time can be reduced even further, by using smaller H: on relatively complex point clouds like Bunny, Fandisk, or Armadillo, we can get down to one fourth of the original processing time without affecting the RMSD$_\perp$, as the local estimates only seldom benefit from $h > 6/\sqrt{\hat{\delta}}$ due to the intricate surface; however, on a basic point cloud like Cube, featuring large, perfectly flat faces, the availability of very large sizes is very beneficial and reducing H does bear a negative impact. Likewise, one can expect that increasing $\max\{H\}\sqrt{\hat{\delta}}$ beyond 12 can help at

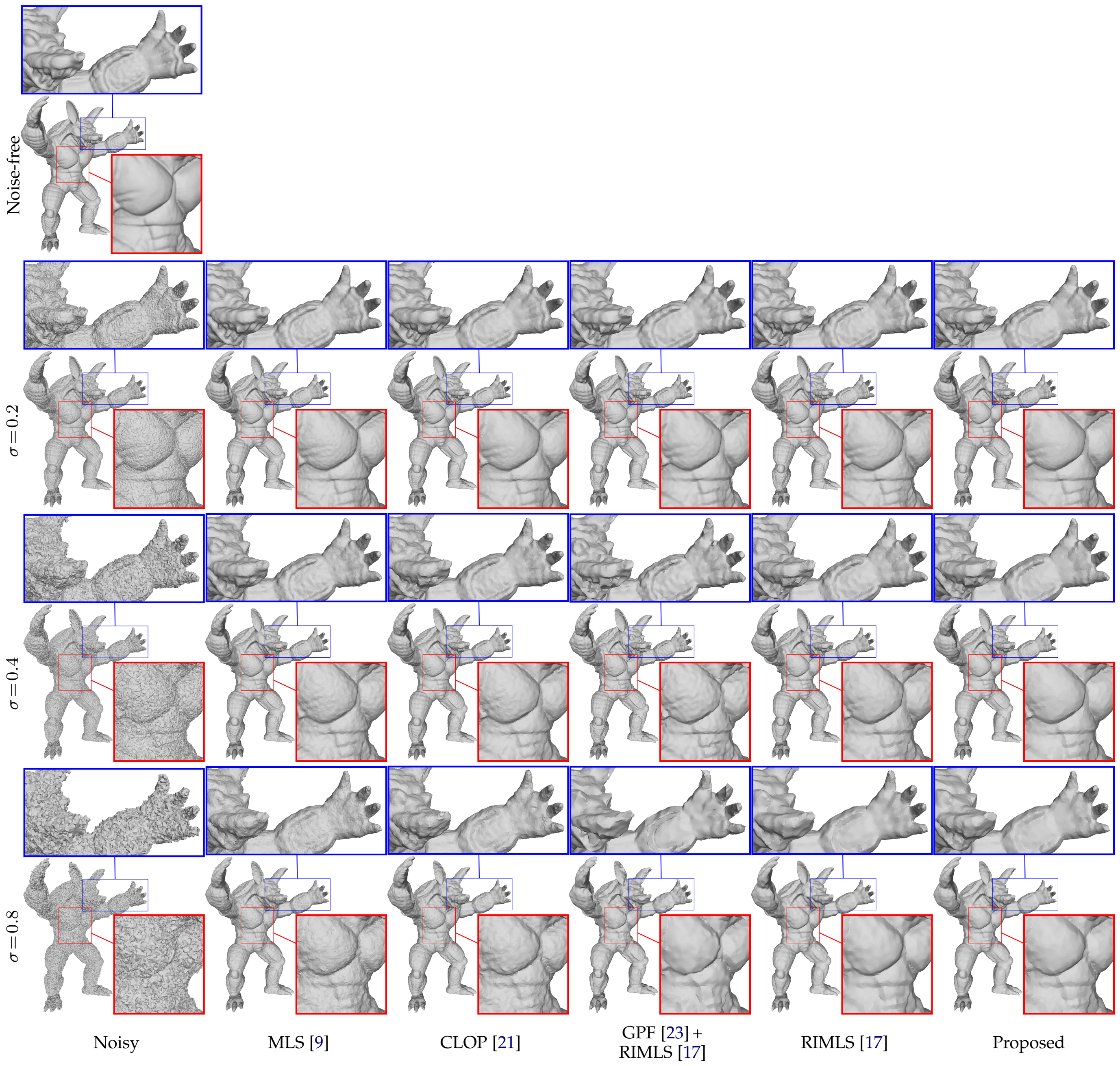


Fig. 19. Reconstructed 3D surfaces of noise-free, noisy, and denoised Armadillo point clouds. A zoomed-in smooth region is shown in the red box, and a fine feature region in the blue box.

particularly high noise levels and on very large flat areas.

5.6 Importance of directional adaptivity

To give an indication of the importance of using adaptive directional neighborhoods in our algorithm, we performed the following separate experiment on the noisy Fandisk, Bunny, and Armadillo point clouds. Using for simplicity only a single iteration, we apply the proposed algorithm adopting either the usual adaptive directional neighborhoods, or a basic adaptive isotropic neighborhood, namely, a sphere with adaptive radius. In either case, the length or radius of the neighborhood is selected from the same set of sizes H, while Γ has been tuned to maximize quality of results. The averaged root mean squared point-to-surface distance over the three point clouds is, for each noise level $\sigma = 0.2, 0.4, 0.8$ respectively, 0.095, 0.159, and 0.270 when using adaptive directional neighborhoods, and 0.109, 0.170, and 0.272 when using adaptive isotropic neighborhoods. These numerical differences are concentrated in the vicinity of edges surrounded by large smooth areas, where directional neighborhoods are able to extend asymmetrically with respect to the edge, whereas isotropic neighborhoods are either small or extend over the edge (resulting in smearing of the edge), which in turn leads to a less favourable trade-off between bias and variance.

5.7 Large Γ for simplification of small structures

By using significantly larger Γ values than those indicated in Section 5.2, the sensitivity of the ICI criterion is reduced and our algorithm can remove the minor structures while

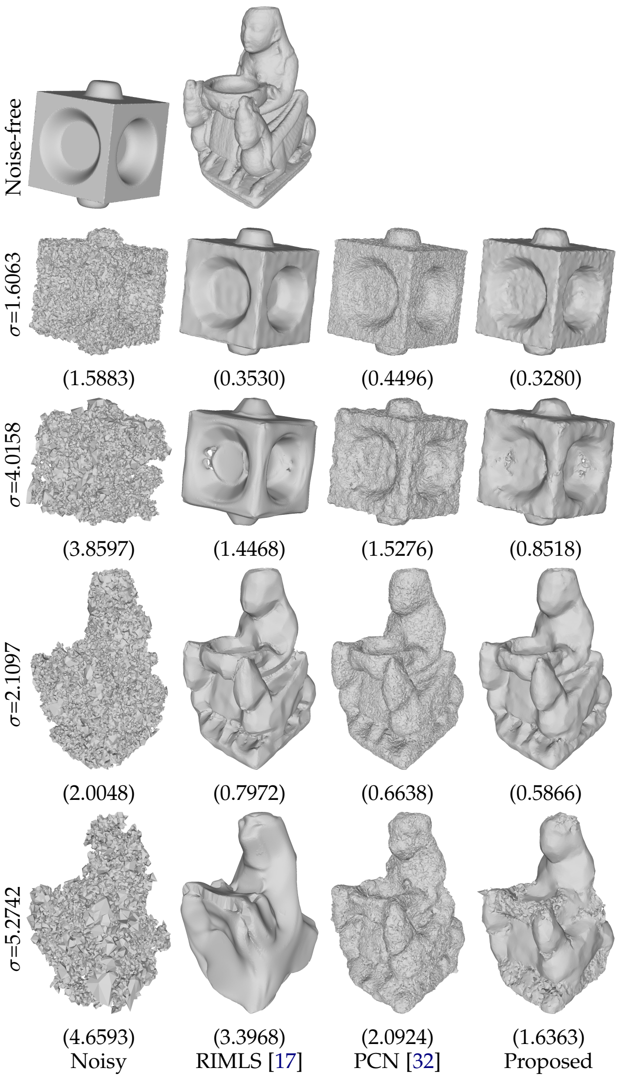


Fig. 20. Reconstructed surfaces of noise-free, heavily noisy Box Push (top) and Galera (bottom), and their denoised results by different algorithms. The root mean squared point-to-surface distances of the noisy and denoised point clouds are reported within parentheses.

TABLE 4
Root mean squared point-to-surface distance results for denoising very strong noise. The σ values for the PCN testing data [32] correspond to ratios between the noise standard deviation and the diagonal of the bounding box of the noise free point cloud equal to 0.0025, 0.005, 0.01, 0.025, as per the authors' original data. The green and red background colors indicate the best and second best results, respectively.

Point clouds	σ	RMSD$_\perp$		
		RIMLS [17]	PCN [32]	Proposed
Happy	0.7080	0.3318	0.6050	0.2965
	1.4160	0.7191	0.7512	0.5179
	2.8321	1.6702	1.0032	0.9737
	7.0802	3.4888	3.5914	2.3947
Box Push	0.4016	0.1321	0.3080	0.0909
	0.8032	0.2082	0.3603	0.1593
	1.6063	0.3530	0.4496	0.3280
	4.0158	1.4468	1.5276	0.8518
Dragon	0.6285	0.2457	0.4842	0.2209
	1.2571	0.5737	0.6434	0.3953
	2.5142	1.3156	0.9109	0.8265
	6.2855	4.2242	2.7852	2.1238
Galera	0.5274	0.2062	0.3964	0.1865
	1.0548	0.3615	0.4961	0.3193
	2.1097	0.7972	0.6638	0.5866
	5.2742	3.3968	2.0924	1.6363
Netsuke	0.5445	0.2225	0.4546	0.2076
	1.0889	0.3897	0.5724	0.3514
	2.1779	0.8251	0.7358	0.6093
	5.4447	3.0742	1.7705	1.5359
Column Head	0.3731	0.2613	0.5764	0.2365
	0.7462	0.4186	0.6727	0.3759
	1.4925	0.7688	0.7970	0.5951
	3.7312	1.5579	1.4372	1.1161
Armadillo	3.00	1.1925	0.9463	0.7304
	6.40	3.0477	1.8531	1.6680
Bunny	3.00	1.3276	1.4028	0.8365
	6.40	3.0758	5.7496	1.8038
Dodecahedron	3.00	0.5038	0.6991	0.4348
	6.40	1.7029	3.5237	1.0001
Cube	3.00	0.9311	2.8314	0.7307
	6.40	3.1323	6.0160	1.9020

preserving only the major features of a point cloud. This can further improve the denoising performance when point clouds feature only simple basic shapes, such as Dodecahedron and Sphere shown in Fig. 24 (top), and can be desirable in certain applications that require additional shape simplification beyond noise reduction, as illustrated in Fig. 24 (bottom) where the small details on the face and legs of Armadillo are cleaned while the edges of the main shape are untouched.

6 Discussion and conclusion

We presented a novel point cloud denoising algorithm based on aggregation of multiple polynomial surfaces computed on directional neighborhoods that are locally adaptive to the shape of the underlying surface of the point cloud. The LPA-ICI technique is performed with respect to the LCS of each point, providing its adaptive directional neighborhood sizes; a dense aggregation of one point's overlapping local polynomial estimates from different adaptive directional neighborhoods delivers the stable and accurate final estimate of the point. Increased effectiveness is achieved through a second iteration of filtering, which is based on modeling the residual noise variance distribution in the denoised point cloud. The proposed algorithm, due to its adaptive design, reduces the noise while preserving fine features and sharp edges of the object surface underlying a point cloud. Results obtained with a baseline implementation of the proposed algorithm with simple first-order polynomials demonstrate superior restoration quality than of competitive methods based on higher-order models. The method has been validated over a set of synthetic and real raw scanned point clouds, under various levels of noise, being applied in fully unsupervised and automatic way, without need of any separate parameter tuning, and providing consistent quality over all these cases.

The proposed algorithm is developed under the assumption of an additive Gaussian noise with uniform isotropic variance σ^2 and roughly uniform surface sampling density δ. In some applications these hypotheses do not hold over the whole point cloud; e.g., depending on the point acquisi-

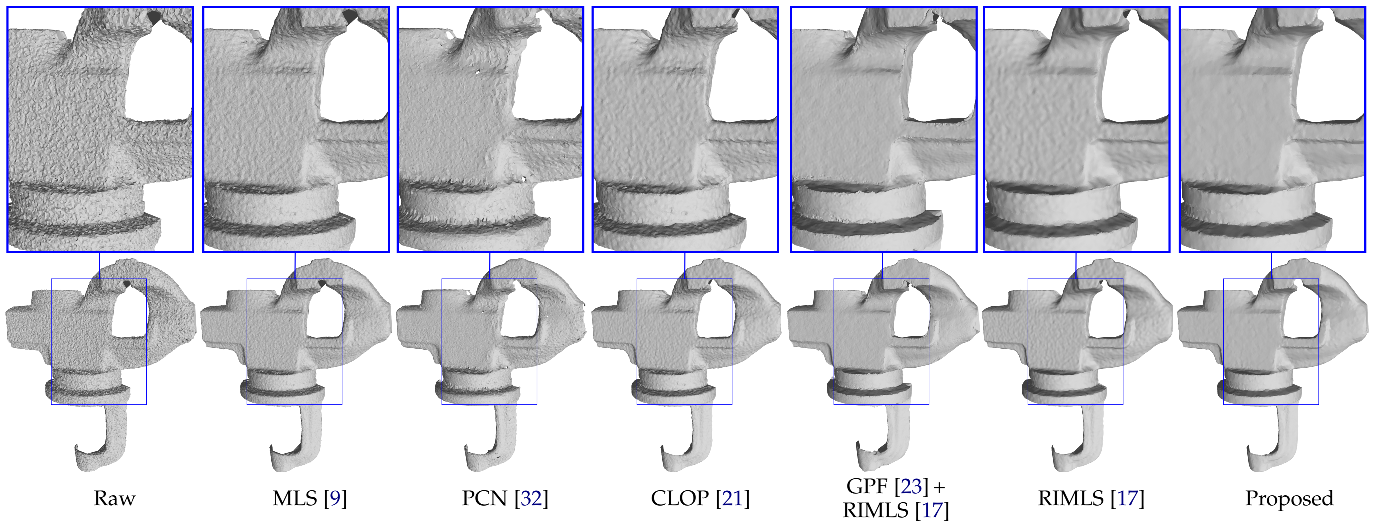


Fig. 21. Reconstructed surfaces of the raw scanned Iron point cloud (see Table 1 for its parameters) and of the denoised point clouds from different algorithms. A zoomed-in area with smooth region and fine features are shown in the blue boxes.

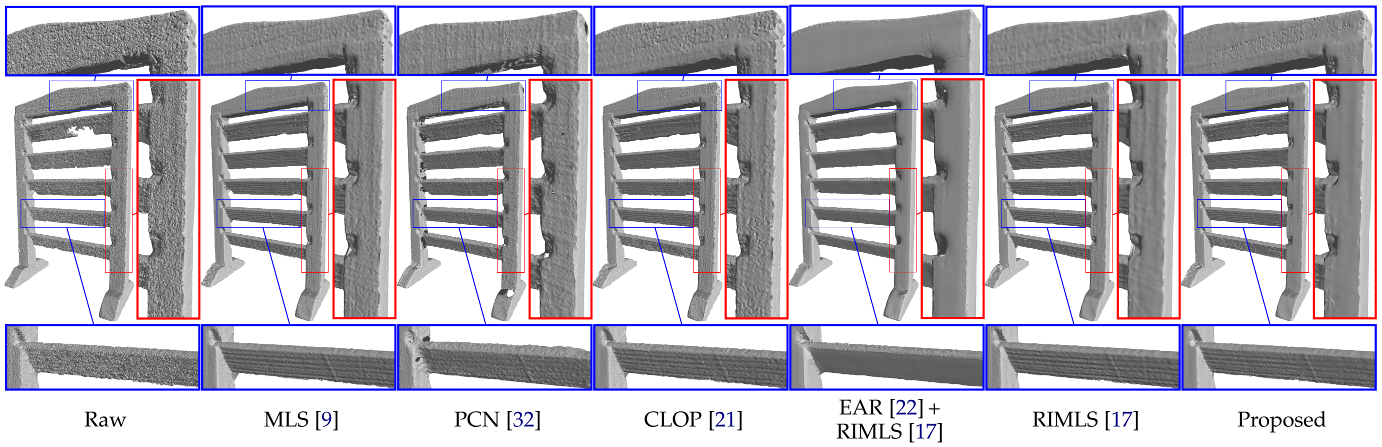


Fig. 22. Reconstructed surfaces of the raw scanned Shutter Blind point cloud (see Table 1 for its parameters) and of the denoised point clouds from different algorithms. A zoomed-in smooth area is shown in the red box, and fine feature regions in the blue boxes.

TABLE 5
Average processing time (in seconds) of different algorithms on Bunny point cloud: MLS, Bilateral filtering, and CLOP are compiled C++ programs; Graph-based regularization is in MATLAB ".m" code format; PCN codes are written in Python language and use the Nvidia CUDA Deep Neural Network library; RIMLS was implemented by the corresponding function integrated in the Meshlab software; EAR and GPF are binary ".exe" programs with user interfaces. Our proposed algorithm is in binary MEX format compiled from the C code generated from MATLAB ".m" sources using MATLAB Coder. All algorithms, except PCN, were run on Intel Core i7-3520M CPU (2.90GHz×4) and 16GB memory; PCN was run on AMD Ryzen 2700X CPU (3.70GHz×8), 32GB memory and Nvidia GeForce GTX 1080 Ti GPU. Quality/complexity trade-off for the proposed algorithm is further reported in Fig. 23.

	MLS [9]	Bilateral [10]	Graph-based [25]	EAR [22] + RIMLS [17]	CLOP [21]	GPF [23] + RIMLS [17]	PCN [32]	RIMLS [17]	Proposed
Bunny (σ = 0.2, 0.4, 0.8)	0.77	1.27	0.75	118.54	8.43	107.98	625.18	5.94	21.26

tion or scanning process, errors and sampling density may be larger or smaller depending on the surface orientation or object distance from the capture system. On this regard, we can however observe that the actual filtering method is intrinsically local and it can be modified easily so to adopt local or semilocal estimates of σ^2 and δ. As a matter of fact, the second iteration of the method already uses a local estimate of σ^2. Likewise, even though the adopted estimators of σ^2 and δ (see Appendix I) are based on global robust statistics, they can be modified in order to output the relevant local or semilocal estimates (e.g., estimates conditioned upon either the location or local normal orientation), as appropriate for the target application. Furthermore, a preprocessing outlier-removal stage (e.g., [25]) becomes necessary when the noisy point cloud includes outlier points.

In this work we adopted an adaptive neighborhood structure with four quadrants; more complex designs with a higher number of narrower sectors are possible (see, e.g., [37], [39]) but make the adaptive size selection less stable. Although one may speculate that adopting higher-order models into the proposed algorithm could bring some improvement on complex, highly textured point clouds, we must note that this would require a different and more complicated aggregation procedure than (11), arguably lim-

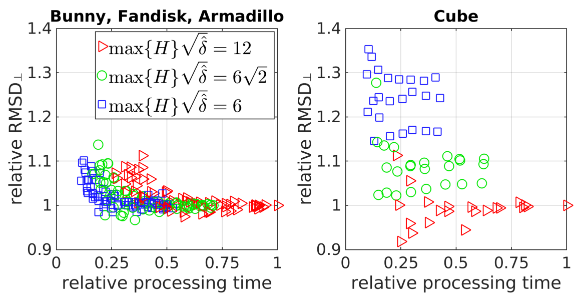


Fig. 23. Relative RMSD$_\perp$ vs. relative processing time plots for various point clouds of different noise levels (σ = 0.2, 0.4, 0.8), where a relative value is the proportion of that value against that from using $\max\{H\}\sqrt{\hat{\delta}}$=12 (19) and 100% of the local estimates.

iting the practicality of the approach. Therefore, we did not pursue this opportunity.

A partitioning operator can be integrated into the proposed algorithm for dealing with big point cloud data. In particular, octree partitioning (see Fig. 25) can be used to process independently the sub point cloud in each octant (i.e. each leaf) of the octree: to enable full LPA-ICI filtering and aggregation, adjacent octants must share an overlap slab with thickness at least equal to $\max\{H\}$.

Finally, the proposed method can benefit from various forms of parellelization, which are currently not leveraged by the current implementation. Specifically, the LCS construction and the pointwise LPA-ICI can be processed independently for each point in the cloud in embarrassingly parallel manner and different octants of the octree partition processed in parallel also.

Appendix I: Estimation of Noise Standard Deviation and of Surface Sample Density

We developed the following method for estimating the noise standard deviation. The method extends to point clouds the approach based on the median of absolute deviation (MAD) [47] of high-frequency subbands which is established in image processing, prominently in the wavelet literature.

Analogous to a detail subband of the Haar wavelet (i.e. first-order finite differences), we first construct a vector $d = [d_1, \ldots, d_I]$ formed by the difference between adjacent points along the third dimension of the LCS, where

$$d_i = \frac{z_i^{\mathcal{L}_i} - z_t^{\mathcal{L}_i}}{\sqrt{2}},$$

and p_t is the closest point to p_i along the $(x^{\mathcal{L}_i}, y^{\mathcal{L}_i})$-plane:

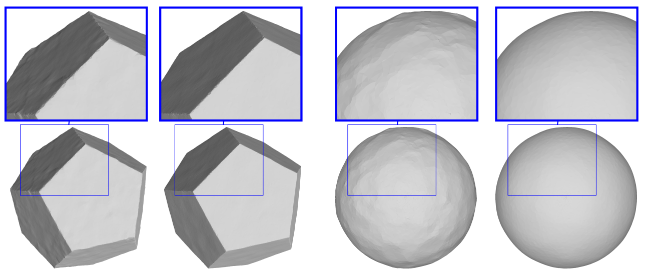
$$p_t = \underset{p_j \in P, p_j \neq p_i}{\operatorname{argmin}} \left\| (x_i^{\mathcal{L}_i}, y_i^{\mathcal{L}_i}) - (x_j^{\mathcal{L}_i}, y_j^{\mathcal{L}_i}) \right\|_2 .$$

Then, we estimate the standard deviation of the noise σ as

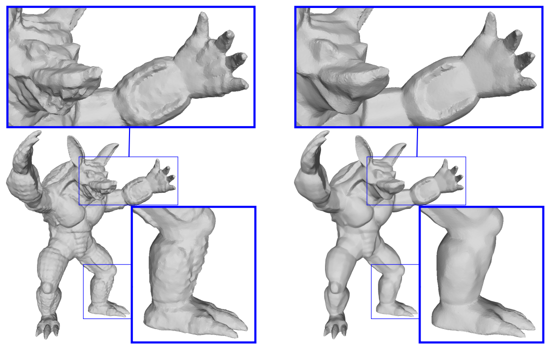
$$\hat{\sigma} = \frac{\operatorname{median}(|d|)}{0.6745} . \tag{20}$$

For estimating the surface sample density, similar to the above procedure we construct a vector $v = [v_1, \ldots, v_I]$ formed by the normalized planar variance within the LCS,

$$v_i = K^{-1} \underset{p_j \in U_i^K}{\operatorname{var}} \left\{ (x_j^{\mathcal{L}_i}, y_j^{\mathcal{L}_i}) \right\} ,$$



Γ = 0.55, 0.85 (0.1086) Γ = 5.00, 5.00 (0.0983) Γ = 0.55, 0.85 (0.1061) Γ = 5.00, 5.00 (0.0783)



Γ = 0.55, 0.85 (0.1562) Γ = 5.00, 5.00 (0.2629)

Fig. 24. Reconstructed 3D surfaces of the denoised Dodecahedron (noise σ=0.638), Sphere (noise σ=0.4), and Armadillo (noise σ=0.4), using the proposed algorithm with different Γ settings. Within each pair, the result on the left-hand side is with for the default Γ setting as described in Section 5.2, whereas the one in the right-hand side uses significantly larger values, leading to more aggressive smoothing. The root mean squared point-to-surface distances against the ground truth are reported in parentheses.

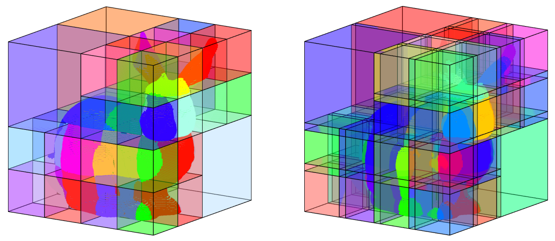


Fig. 25. Example of the octree partitioning on the Bunny point cloud without (left) and with overlap of octants (right).

where U_i^K is the KNN of p_i, from which the LCS is also defined, as described in Section 4.2. Then, the estimate of the surface sample density is

$$\hat{\delta} = \frac{1}{2\pi \operatorname{median}(v)} . \tag{21}$$

The scaling constants in (20) and (21) correspond to assuming, respectively, that the noise is Gaussian distributed and that the point cloud samples within the LCS are uniformly distributed with respect to the planar components of the LCS.

The accuracy of the $\hat{\sigma}$ and $\hat{\delta}$ estimates is of course affected by the actual level of noise relative to sampling density, which in turn affects the construction of the LCS via PCA of the KNN. To obtain accurate estimates under stronger noise and/or at higher density, it is necessary to use larger values of K in the construction of the LCS. Thus, we progressively increase K and repeat the estimation of σ and δ whenever the current value of the product $\hat{\sigma}\sqrt{\hat{\delta}}$

exceeds a given threshold. Specifically, starting from $K=50$, if $\hat{\sigma}\sqrt{\hat{\delta}} \geq 1.5$ we repeat the estimation with $K=200$; then, if $\hat{\sigma}\sqrt{\hat{\delta}} \geq 3.5$ we repeat the estimation with $K=300$; finally, if $\hat{\sigma}\sqrt{\hat{\delta}} \geq 4.5$ we repeat the estimation with $K=500$. If, at any stage, the threshold is not exceeded, the algorithm returns the current estimates $\hat{\sigma}$ and $\hat{\delta}$. This progressive increase of K concerns exclusively the estimation of σ and δ: inside the denoising algorithm, the LCS for the LPA-ICI denoising is always built with $K=50$, irrespective of what value of K had been used for the preliminary estimation of σ and δ.

Appendix II: Error metrics

A. Point-to-surface distance

The point-to-surface distance (also called point-to-plane distance) has been employed as the quality measurement of point cloud denoising [25], [48] and point cloud compression [49] algorthms. It is also applied in Iterative Closest Point (ICP) algorithms [50], [51] for point cloud registration.

We compute the point-to-surface distance as follows,

1) for each $\hat{q}_i$ in the denoised point cloud $\hat{Q}$, we first find its nearest point (in Euclidean distance) in the ground-truth Q, which can differ from q_i and which we denote by q_{i^*};
2) the surface normal at q_{i^*}, represented by n_{i^*}, is then estimated using PCA on q_{i^*}'s KNN in Q (we choose $K=5$).
3) the point-to-surface distance $D_{i\perp}$ of $\hat{q}_i$ is then computed as $D_{i\perp} = \|\langle \hat{q}_i - q_{i^*}, n_{i^*} \rangle\|_2$;
4) the root mean squared point-to-surface distance (RMSD$_\perp$) is thus $\sqrt{\frac{1}{I}\sum_{i=1}^{I} D_{i\perp}^2}$.

B. Normal angular error and proportion of good points

Measuring the angular differences between each surface normal in the denoised point cloud and its corresponding one in the ground truth gives another perspective of the denoising performance of an algorithm.

After obtaining n_{i^*} as in the above computation of the point-to-surface distance, we compute the surface normal at $\hat{q}_i$ in $\hat{Q}$ through the PCA of its KNN in $\hat{Q}$ (again $K=5$), which we denote by $\hat{n}_i$; n_{i^*} is treated as the ground-truth value for $\hat{n}_i$ and by $\sphericalangle\{\hat{n}_i, n_{i^*}\} \in [0, \pi/2]$ we denote their absolute angular difference.

The Root Mean Square Angular Error with threshold τ (RMSAE$_\tau$) [52], [53] is computed as

$$\text{RMSAE}_\tau = \sqrt{\frac{1}{I}\sum_{i=1}^{I} f_\tau^2(\hat{n}_i, n_{i^*})}\ , \tag{22}$$

where

$$f_\tau(\hat{n}_i, n_{i^*}) = \begin{cases} \sphericalangle\{\hat{n}_i, n_{i^*}\} & \text{if } \sphericalangle\{\hat{n}_i, n_{i^*}\} < \tau\,, \\ \pi/2 & \text{otherwise}\,. \end{cases}$$

The RMSAE$_\tau$ treats all those angular errors larger than τ as they were as bad as $\pi/2$, and in this way it avoids the drawback of the common RMSAE (without threshold) which favors smoothness everywhere at the expense of sharp reconstruction of edges and corners [29], [52].

The proportion of good points (PGP) [29], [53], i.e. the points in $\hat{Q}$ whose angular error is smaller than τ, provides an additional measure of the surface normal estimation quality; it is computed as

$$\text{PGP} = \frac{1}{I}\sum_{i=1}^{I} N(\hat{n}_i, n_{i^*}), \tag{23}$$

where

$$N(\hat{n}_i, n_{i^*}) = \begin{cases} 1 & \text{if } \sphericalangle\{\hat{n}_i, n_{i^*}\} < \tau\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

When computing RMSAE$_\tau$ and PGP, we set $\tau = \pi/18$ (i.e. 10°), as recommended by [29], [52], [53].

Acknowledgments

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7-PEOPLE-2013-ITN) under grant agreement n° 607290 SpaRTaN, and from the Academy of Finland under project n° 310779).

References

[1] F. Lozes, A. Elmoataz, and O. Lézoray, "PDE-based graph signal processing for 3-D color point clouds: opportunities for cultural heritage," *IEEE Signal Process. Mag.*, vol. 32, no. 4, pp. 103–111, July 2015.

[2] A. Holgado-Barco, B. Riveiro, D. González-Aguilera, and P. Arias, "Automatic inventory of road cross-sections from mobile laser scanning system," *Comput. Aided Civil Infrastr. Eng.*, vol. 32, no. 1, pp. 3–17, 2017. [Online]. Available: http://dx.doi.org/10.1111/mice.12213

[3] E. Shellshear, R. Berlin, and J. S. Carlson, "Maximizing smart factory systems by incrementally updating point clouds," *IEEE Comput. Graphics App.*, vol. 35, no. 2, pp. 62–69, Mar. 2015.

[4] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[5] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 8, pp. 1362–1376, Aug. 2010.

[6] A. Goldenshluger and A. Nemirovski, "On spatially adaptive estimation of nonparametric regression," *Math. Meth. Stat.*, vol. 6, no. 2, pp. 135–170, 1997.

[7] V. Katkovnik, "A new method for varying adaptive bandwidth selection," *IEEE Trans. Signal Process.*, vol. 47, no. 9, pp. 2567–2571, 1999.

[8] A. Foi, V. Katkovnik, and K. Egiazarian, "Pointwise shape-adaptive DCT for high-quality denoising and deblocking of grayscale and color images," *IEEE Trans. Image Process.*, vol. 16, no. 5, pp. 1395–1411, 2007.

[9] D. Levin, "Mesh-independent surface interpolation," *Geometric Modeling for Scientific Visualization*, vol. 3, pp. 37–49, 2003.

[10] J. Digne, "The Bilateral Filter for Point Clouds," *Image Process. On Line*, 2015.

[11] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Sixth Int. Conf. Comput. Vis.*, 1998, pp. 839–846.

[12] S. Fleishman, I. Drori, and D. Cohen-Or, "Bilateral mesh denoising," *ACM Trans. Graphics*, vol. 22, no. 3, pp. 950–953, July 2003. [Online]. Available: http://doi.acm.org/10.1145/882262.882368

[13] S. Fleishman, D. Cohen-Or, and C. T. Silva, "Robust moving least-squares fitting with sharp features," in *ACM Trans. Graphics*, vol. 24, no. 3. ACM, 2005, pp. 544–552.

[14] Y. Lipman, D. Cohen-Or, and D. Levin, "Data-dependent MLS for faithful surface approximation," in *Proc. 5th Eurographics Sym. Geom. Process.* Eurographics Association, 2007, pp. 59–67.

[15] C. Lange and K. Polthier, "Anisotropic smoothing of point sets," *Comput. Aided Geom. Des.*, vol. 22, no. 7, pp. 680–692, 2005.

[16] A. Wetzler, G. Rosman, and R. Kimmel, "Patch-space Beltrami denoising of 3D point clouds," in *IEEE 27th Conv. Elec. Electron. Eng. Israel*, 2012, pp. 1–5.

[17] A. C. Öztireli, G. Guennebaud, and M. Gross, "Feature preserving point set surfaces based on non-linear kernel regression," in *Comput. Graphics Forum*, vol. 28, no. 2, 2009, pp. 493–501.

[18] H. Avron, A. Sharf, C. Greif, and D. Cohen-Or, "ℓ1-sparse reconstruction of sharp point set surfaces," *ACM Trans. Graphics*, vol. 29, no. 5, p. 135, 2010.

[19] Y. Sun, S. Schaefer, and W. Wang, "Denoising point sets via L0 minimization," *Comput. Aided Geom. Des.*, vol. 35, pp. 2–15, 2015.

[20] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer, "Parameterization-free projection for geometry reconstruction," in *ACM Trans. Graphics*, vol. 26, no. 3. ACM, 2007, p. 22.

[21] R. Preiner, O. Mattausch, M. Arikan, R. Pajarola, and M. Wimmer, "Continuous projection for fast L1 reconstruction." *ACM Trans. Graph.*, vol. 33, no. 4, pp. 47–1, 2014.

[22] H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, and H. R. Zhang, "Edge-aware point set resampling," *ACM Trans. Graphics*, vol. 32, no. 1, p. 9, 2013.

[23] X. Lu, S. Wu, H. Chen, S.-K. Yeung, W. Chen, and M. Zwicker, "GPF: GMM-inspired feature-preserving point set filtering," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 8, pp. 2315–2326, 2018.

[24] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, 2013.

[25] Y. Schoenenberger, J. Paratte, and P. Vandergheynst, "Graph-based denoising for time-varying point clouds," in *3DTV-Conference (3DTV-CON)*. IEEE, 2015, pp. 1–4.

[26] Y. Zheng, G. Li, X. Xu, S. Wu, and Y. Nie, "Rolling normal filtering for point clouds," *Comput. Aided Geom. Des.*, vol. 62, pp. 16–28, 2018.

[27] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," *Proc. 19th Annual Conf. Comput. Graphics Interactive Tech.*, vol. 26, no. 2, pp. 71–78, July 1992. [Online]. Available: http://doi.acm.org/10.1145/142920.134011

[28] X. Lu, S. Schaefer, J. Luo, L. Ma, and Y. He, "Low rank matrix approximation for geometry filtering," *arXiv preprint arXiv:1803.06783*, 2018.

[29] A. Boulch and R. Marlet, "Deep learning for robust normal estimation in unstructured point clouds," in *Comput. Graphics Forum*, vol. 35, no. 5, 2016, pp. 281–290.

[30] P. Guerrero, Y. Kleiman, M. Ovsjanikov, and N. J. Mitra, "PCPNet: Learning local shape properties from raw point clouds," *Comput. Graphics Forum*, vol. 37, no. 2, pp. 75–85, 2018.

[31] A. Foi and V. Katkovnik, "From local polynomial approximation to pointwise shape-adaptive transforms: An evolutionary nonparametric regression perspective," in *Proc. SMMSP*, 2006.

[32] M.-J. Rakotosaona, V. La Barbera, P. Guerrero, N. J. Mitra, and M. Ovsjanikov, "PointCleanNet: learning to denoise and remove outliers from dense point clouds," *Comput. Graphics Forum*, 2019.

[33] W. Bae, J. Yoo, and J. Chul Ye, "Beyond deep residual learning for image restoration: Persistent homology-guided manifold simplification," in *Proc. 2017 IEEE Conf. Comput. Vis. Pattern Recogn. Workshops*, 2017, pp. 145–153.

[34] C. Cruz, A. Foi, V. Katkovnik, and K. Egiazarian, "Nonlocality-reinforced convolutional neural networks for image denoising," *IEEE Signal Process. Lett.*, vol. 25, no. 8, pp. 1216–1220, 2018.

[35] X. Li, W. Dong, and G. Shi, *Sparsity-Based Denoising of Photographic Images: From Model-Based to Data-Driven*. Cham: Springer, 2018, pp. 37–62. [Online]. Available: https://doi.org/10.1007/978-3-319-96029-6_2

[36] V. Katkovnik, A. Foi, K. Egiazarian, and J. Astola, "Directional varying scale approximations for anisotropic signal processing," in *12th European Signal Process. Conf.* IEEE, 2004, pp. 101–104.

[37] A. Foi, "Anisotropic nonparametric image processing: theory, algorithms and applications," Ph.D. dissertation, Dip. di Matematica, Politecnico di Milano, ERLTDD-D01290, 2005.

[38] V. Katkovnik, K. Egiazarian, and J. Astola, *Local approximation techniques in signal and image processing*. SPIE Bellingham, 2006.

[39] V. Katkovnik, A. Foi, K. Egiazarian, and J. Astola, "From local kernel to nonlocal multiple-model image denoising," *Int. J. Comput. Vision*, vol. 86, no. 1, p. 1, 2010.

[40] "The Stanford 3D Scanning Repository." [Online]. Available: http://graphics.stanford.edu/data/3Dscanrep/

[41] [Online]. Available: https://github.com/areslp/normal/blob/master/fandisk.ply

[42] M.-J. Rakotosaona, "PointCleanNet." [Online]. Available: https://github.com/mrakotosaon/pointcleannet

[43] A. Foi, V. Katkovnik, K. Egiazarian, and J. Astola, "A novel anisotropic local polynomial estimator based on directional multiscale optimizations," in *Proc. 6th IMA Int. Conf. Math. Signal Process.*, 2004, pp. 79–82. [Online]. Available: http://www.cs.tut.fi/~lasip/2D/

[44] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE Int. Conf. Robotics and automation (ICRA)*. IEEE, 2011, pp. 1–4. [Online]. Available: http://pointclouds.org/

[45] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, "Meshlab: an open-source mesh processing tool," in *Eurographics Italian Chap. Conf.*, vol. 2008, 2008, pp. 129–136.

[46] L. Giaccari, "Surface Reconstruction Toolbox," Sept. 2017. [Online]. Available: https://github.com/LuigiGiaccari/Surface-Reconstruction-Toolbox/releases

[47] F. R. Hampel, "The influence curve and its role in robust estimation," *J. Am. Stat. Assoc.*, vol. 69, no. 346, pp. 383–393, 1974.

[48] A. Javaheri, C. Brites, F. Pereira, and J. Ascenso, "Subjective and objective quality evaluation of 3D point cloud denoising algorithms," in *Proc. IEEE Int. Conf. Multim. Expo Workshops*, 2017, pp. 1–6.

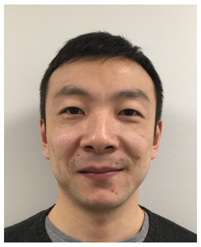
[49] D. Tian, H. Ochimizu, C. Feng, R. Cohen, and A. Vetro, "Geometric distortion metrics for point cloud compression," in *IEEE Int. Conf. Image Process.*, 2017, pp. 3460–3464.

[50] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image Vision Comput.*, vol. 10, no. 3, pp. 145–155, 1992.

[51] K.-L. Low, "Linear least-squares optimization for point-to-plane ICP surface registration," *Chapel Hill, University of North Carolina*, vol. 4, no. 10, 2004.

[52] A. Boulch and R. Marlet, "Fast and robust normal estimation for point clouds with sharp features," in *Comput. Graphics Forum*, vol. 31, no. 5, 2012, pp. 1765–1774.

[53] J. Zhang, J. Cao, X. Liu, J. Wang, J. Liu, and X. Shi, "Point cloud normal estimation via low-rank subspace clustering," *Computers & Graphics*, vol. 37, no. 6, pp. 697–706, 2013.



Zhongwei Xu received the B.S. degree in Electrical Engineering and the M.Sc. degree in Computer Science from the Xidian University, China, in 2008 and 2011, respectively. He received the Ph.D. degree in Computer Science from the Universitat Autónoma de Barcelona, Spain, in 2015. His research interests are the restoration and enhancement of 3D point cloud data and digital images, and image and video coding.

Alessandro Foi received the M.Sc. degree in Mathematics from the Università degli Studi di Milano, Italy, in 2001, the Ph.D. degree in Mathematics from the Politecnico di Milano in 2005, and the D.Sc.Tech. degree in Signal Processing from Tampere University of Technology, Finland, in 2007. He is Professor of Signal Processing at Tampere University, Finland. His research interests include mathematical and statistical methods for signal processing, functional and harmonic analysis, and computational modeling of the human visual system. His work focuses on spatially adaptive (anisotropic, nonlocal) algorithms for the restoration and enhancement of digital images, on noise modeling for imaging devices, and on the optimal design of statistical transformations for the stabilization, normalization, and analysis of random data. He is a Senior Member of the IEEE, Member of the Image, Video, and Multidimensional Signal Processing Technical Committee of the IEEE Signal Processing Society, an Associate Editor for the SIAM Journal on Imaging Sciences, and a Senior Area Editor for the IEEE Transactions on Computational Imaging.