A review of algorithms for filtering the 3D point cloud

Article *in* Signal Processing Image Communication · May 2017

DOI: 10.1016/j.image.2017.05.009

CITATIONS
23

READS
4,587

6 authors, including:

Xian-Feng Han
Tianjin University
8 PUBLICATIONS **39** CITATIONS
SEE PROFILE

Jesse Sheng Jin
Tianjin University
62 PUBLICATIONS **320** CITATIONS
SEE PROFILE

Signal Processing: Image Communication 57 (2017) 103–112

Contents lists available at ScienceDirect

Signal Processing: Image Communication

journal homepage: www.elsevier.com/locate/image

A review of algorithms for filtering the 3D point cloud

Xian-Feng Han *, Jesse S. Jin, Ming-Jie Wang, Wei Jiang, Lei Gao, Liping Xiao

School of Computer Software, Tianjin University, 30072, Tianjin, China
National Key Laboratory of Science and Technology on Aerospace Intelligent Control, Beijing 100085, China

ARTICLE INFO

Keywords:
3D point cloud
Filtering methods
Feature-preserving
Noise reduction

ABSTRACT

In recent years, 3D point cloud has gained increasing attention as a new representation for objects. However, the raw point cloud is often noisy and contains outliers. Therefore, it is crucial to remove the noise and outliers from the point cloud while preserving the features, in particular, its fine details. This paper makes an attempt to present a comprehensive analysis of the state-of-the-art methods for filtering point cloud. The existing methods are categorized into seven classes, which concentrate on their common and obvious traits. An experimental evaluation is also performed to demonstrate robustness, effectiveness and computational efficiency of several methods used widely in practice.

1. Introduction

The 3D point cloud [1–3], a new primitive representation for objects, has became increasingly prevalent in many research fields [2], such as object recognition [4] and reconstruction [5,6], due to its simplicity, flexibility and powerful representation capability. In contrast to triangle meshes, the point cloud does not require to store or maintain the polygonal-mesh connectivity [7] or topological consistency [8]. Processing and manipulating point cloud therefore can demonstrate better performance and lower overhead. These prominent advantages make the research on processing point cloud a hot topic.

The rapid development of low-cost sensors, such as Kinect [9–11] and time of flight cameras [5,12], makes it easy to obtain point cloud for growing communities. The point cloud acquired with these sensors, however, inevitably suffers from noise contamination and contains outliers [13,14] due to the limitations of sensors [5], the inherent noise of the acquisition device [15], the lighting or reflective nature of the surface or artifact in the scene [16]. Therefore, it is necessary to perform filtering operations on raw point clouds to obtain accurate point clouds that are suitable for further processing.

In recent years, although a large number of methods contributing to 3D filtering have been proposed, most of these are devised for meshes and only a few approaches directly operate on point cloud. In addition, there is no survey paper giving an insightful analysis of these filtering methods for point cloud.

Compared with the existing literature, the main contributions of this work are as follows: (i) To the best of our knowledge, this is the first review paper in the literature that focuses on algorithms for filtering 3D point cloud at present. (ii) This paper provides readers with a comprehensive review of the state-of-the-art methods covered in early work. (iii) A comparative summary of traits of these methods is demonstrated in table form. (iv) This paper carries out an experiment concerning on performance comparison of several widely used methods.

The remainder of this paper is organized as follows. Section 2 presents an overview of filtering approaches for 3D point cloud. And then experimental results and discussion are illustrated in Section 3. Conclusions are drawn in Section 4.

2. Methods for filtering point cloud

Filtering is an area of intensive research and the crucial step of the processing pipeline for a wide range of applications. The main filtering approaches for 3D point cloud can be categorized into the following seven groups, where four classifications (statistical-based, neighborhood-based, projection-based and PDEs-based filtering) are from [17].

2.1. Statistical-based filtering techniques

In the context of filtering point cloud, many techniques utilize the adaptation of the statistical conceptions, which are suitable for the nature of the point cloud.

* Corresponding author at: School of Computer Software, Tianjin University, 30072, Tianjin, China.
E-mail addresses: hanxianf@163.com (X.-F. Han), jinsheng@tju.edu.cn (J.S. Jin), 18768126670@163.com (M.-J. Wang), jiangweitju@163.com (W. Jiang), thrstone@sina.cn (L. Gao), xlp027@sina.com (L. Xiao).

http://dx.doi.org/10.1016/j.image.2017.05.009
Received 22 November 2016; Received in revised form 28 February 2017; Accepted 15 May 2017
Available online 22 May 2017
0923-5965/

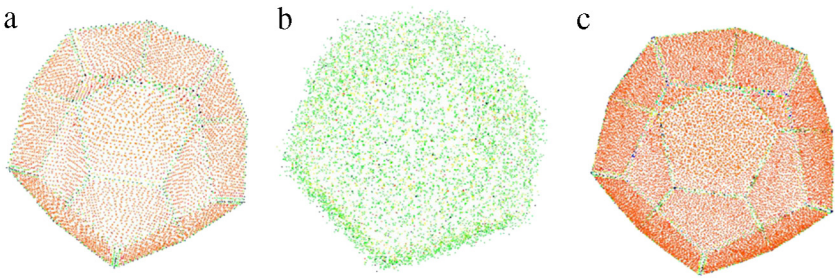

Fig. 1. Denoise point sets via L0 minimization. (a) Original model; (b) input; (c) L0 minimization, primitively shown in [23].

Schall et al. [18] filtered point cloud using a kernel based clustering approach. They first accumulated the local likelihoods L_i computed on every point p_i to define the likelihood function L modeling the probability for the noisy point cloud. Next, they moved the points to positions of high probability utilizing an iterative scheme motivated by mean shift technique to smooth the point cloud. This method achieves effectiveness in filtering and robustness in outlier detection. However, sharp features are not treated with emphasis in their method.

Narváez et al. [15] proposed a new weighted variant of the principal component analysis method for denoising point cloud, which used weighting factors assignment by inversely proportional repartition of the sum of distance to the mean. Then, the factors and the weighted mean are used to estimate a weighted covariance matrix. By realizing an eigen-analysis of the matrix, a fitting plane, expanded by the eigen-vectors corresponding to the largest eigen-values, and a normal vector to this plane oriented in the direction of eigen-vector corresponding to the smallest eigen-value are obtained at each point. The variations make the algorithm robust to the noise and outliers. In addition, the operator $p' = p + tn_p$ [19] is applied to shift the mean along the normal direction to preserve shape features.

Jenke et al. [20] first employed Bayesian statistics for denoising point cloud. They found a measurement model $P(D|S)$, which specified the probability distribution of estimated point cloud S agreeing with measured data D. Then, they defined three prior probabilities, such as density priors, smoothness priors and priors for sharp features, to form $P(S) = \frac{1}{Z} P_{density}(S) P_{smooth}(S) P_{discrete}(S) \cdot w(S)$, Z was a normalization constant. Finally, they maximized a posteriori $P(S|D)$ to remove noise while preserving features.

$$S_{MAP} = argmax_S P(S|D) = argmax_S P(D|S) P(S) \tag{1}$$

Kalogerakis et al. [21] provided a robust statistical framework to filter point clouds. In their framework, an Iteratively Least Squares (IRLS) approach estimates curvature tensor and assigns weights to samples at each iteration to refine each neighborhood around every point. The computed curvatures and the final statistical weights are utilized to correct normal. The robustly estimated curvatures and normal can drive the outlier rejection and denoise point cloud in a feature-preserving manner based on a global energy minimization process.

Avron et al. [22] introduced L_1-*sparsity* paradigm to denoise the point cloud. Firstly, a re-weighted L_1 minimization process is used to restore point orientations. Then, point position is reconstructed by assuming a local planarity criterion so as to preserve shape features. Although, this method can achieve reasonable results, points on an edge are sometimes not faithfully recovered [23]. Meanwhile, since L_0 is a sparser solution than L_1, Sun et al. provided an L_0 minimization method, which is directly used to denoise point cloud by applying a similar L_0 optimization procedure to estimate normals followed by repositioning points along the normal directions in order to better maintain the sharp features (see Fig. 1).

Orts-Escolano et al. [24] first used a 3D filtering and downsampling technique based on Growing Neural Gas (GNG) [25–27] network. This is a growth process to produce a GNG network to represent a raw point cloud using a set of 3D neurons and interconnection among them. This method can preserve the topology of point cloud and deal with outliers in point cloud. Therefore, filtered point cloud using GNG can improve keypoints detection performance [28] and yield better recognition results [4,29]. This method produced a GNG network mapping to the point cloud (shown in Fig. 2).

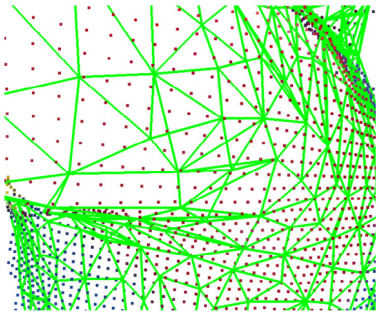

Fig. 2. GNN representation of point cloud, primitively shown in [4].

2.2. Neighborhood-based filtering techniques

Neighborhood-based filtering techniques determine the filtered position of a point using similarity measures between a point and its neighborhood which has a strong influence on the efficiency and effectiveness of the filtering approach [17]. As described in the following methods, the similarity can be defined by positions of points, normals or regions.

The bilateral filter, originally introduced by Tomasi and Manduchi [30], is an edge-preserving [31] smoothing filter, which is extended to 3D meshed denoising [32–34]. However, these methods require a mesh generation process, which itself suffers from noise [35]. In order to tackle this problem, the bilateral filter is applied directly on the point cloud [6,36–38] based on point positions and intensity. The w_s and w_r are the spatial and range weight, respectively,

$$w_s = \exp\left(-\frac{(i-x)^2 + (j-y)^2}{2\sigma_s^2}\right) \tag{2}$$

$$w_r = \exp\left(-\frac{(I(i,j) - I(x,y))^2}{2\sigma_r^2}\right) \tag{3}$$

where (i, j) is the neighborhood of (x, y), $I(i, j)$ presents the intensity at (x, y), σ_s and σ_r are the standards of Gaussian functions.

In order to reduce time complexity, Xu et al. [39] replaced the weight of gray domain in the bilateral filter with a binary function (4) to achieve a better performance. However, this kind of filters deals with the point cloud containing intensity components. As a consequence, normal [35,40–43], being as one of the important attributes of point cloud, is considered in the process of bilateral filter of which the weight

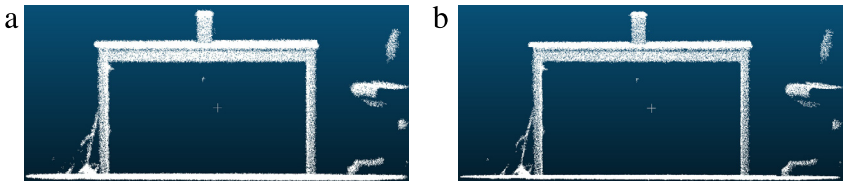

Fig. 3. Denoise point cloud via a normal based bilateral filter. (a) Noisy model; (b) filtered result.

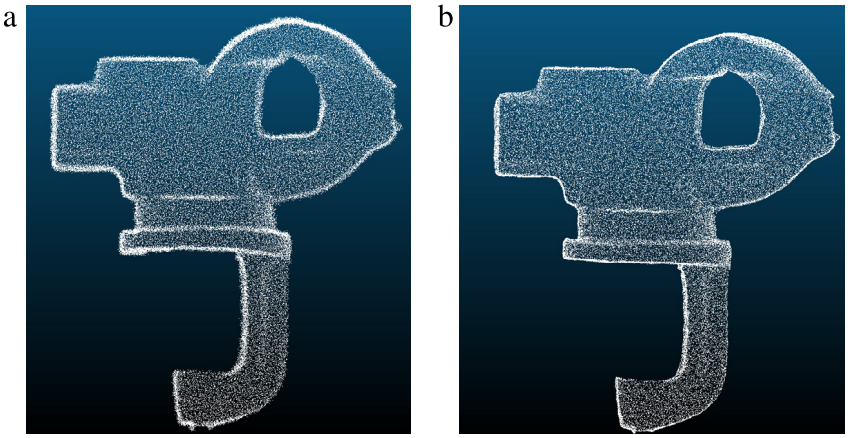

Fig. 4. Denoise point sets via WLOP. (a) Input noisy model; (b) filtered result.

is defined as a function of spatial location and normal information of points, shown in Eq. (5).

$$w_r = \begin{cases} 1 & |I(x,y) - I(i,j)| \le 3, I(x,y) \neq 0 \\ 0 & |I(x,y) - I(i,j)| > 3, I(x,y) \neq 0 \end{cases} \tag{4}$$

$$w = f\left[d\left(p,q\right)\right] \times g\left[c\left(\mathbf{n}_p, \mathbf{n}_q\right)\right] \tag{5}$$

where f and g are the Gaussian function with σ_f and σ_g. $d(p,q)$ defines certain distance (e.g. Euclidean distance) between point p and its neighbor q. $c(\mathbf{n}_p, \mathbf{n}_q)$ indicates the normal relation at points p and q, such as inner product of normal vector $\langle \mathbf{n}_p, \mathbf{n}_q \rangle$ [35], $\left\|(p-q)\cdot \mathbf{n}_p \mathbf{n}_q\right\|$ [40], $\left(\mathbf{n}_p - \mathbf{n}_q\right)^2$ [41] and $\mathbf{n}_p \cdot \left(\mathbf{n}_p - \mathbf{n}_q\right)$ [42] (see Fig. 3).

Motivated by the mean shift filtering for images, Hu et al. [44] formulated a 3D mean shift based anisotropic filter algorithm by taking the vertex normal, curvature as well as position into account. According to the similar modes of points produced via mean shift procedure, they designed a cluster scheme to provide an adaptive neighbor searching method to determine the neighborhoods of each point. Finally, they applied a trilateral filtering to the adaptive neighborhoods to denoise the point cloud whilst preserving features.

Based on the similar idea as bilateral filtering, Schall et al. [17] first extended non-local means method proposed by Buades et al. [45] for image filtering to 3D point cloud. They introduced a new non-local similarity measure Φ_s which took the local square neighborhoods around points into account. $\Phi_s\left(p_i, p_j\right) = e^{-Sim(p_i,p_j)^2/s^2}$, $Sim\left(p_i, p_j\right) = \sum_{o\in O}\left|(p_{i+o} - p_{j+o})\cdot d_i\right|^2 / \|O\|$, where p_j is a point of neighborhood $N(p_i)$, o denotes the offset between the center point and an arbitrary neighborhood point. d_i is the displacement direction of point p_i, which can either be an estimated normal or the line-of-sight of camera lying on the types of noise. This method yields a more accurate filtering result and possesses a better feature preservation characteristic.

Huhle et al. [12] took color information and intra-patch similarity into account to propose a robust non-local means filter. This method first detects outliers in the input data and then performs a feature-preserving smoothing processing. Experimental results show that this denoising algorithm exhibited a superior performance.

Wang et al. [46] presented an effective algorithms consisting of two steps: outliers filtering and noise smoothing. They designed a connectivity-based approach based on the properties of the relative deviation of the local neighborhood and the average local neighborhood to remove sparse outliers. After the process, they used a clustering-based scheme to further eliminate the small cluster outliers. Furthermore, they performed normal filtering by iteratively updating weighted normal of point. Points are updated to match the filtered normal. Experimental results show that this method obtains satisfactory results.

2.3. Projection-based filtering approaches

Projection-based filtering approaches adjust the position of each point in a point cloud via different projection strategies to filter point cloud.

Inspired by the concept of L_1 median [47–50] from statistics, Lipman et al. [51] introduced a parameterization-free projection operator, that was, Locally Optimal Projection (LOP) operator [52]. The rationale of this method is to iteratively project a subset of the input point cloud onto this point cloud to reduce noise and outliers. However, if the input point cloud is highly non-uniform, projection by LOP tends to become non-uniformity, which is undesirable in shape features preservation and normal estimation.

Huang et al. [49] incorporated locally adaptive density weights of each point into LOP, that was WLOP (see Fig. 4), to produce an evenly distributed points set. An important parameter which controls the amount of smoothness in LOP is h, the support size of the weight function Θ. Too large value of h causes the shrinkage of the point cloud while too small value results in little effect of denoising. To solve this problem, Ye et al. [53] projected only z coordinates of the point-set, then x and y could be calculated through re-projection. This method not only obtains smoothness and the preservation of geometric structure, but also removes isolated outlier points.

To address problems mentioned above, Liao et al. [54] integrated a feature preservation weight $\theta_r(x) = e^{-\left(\mathbf{n}_i^T\left(p_i - q_j\right)\right)^2/\sigma_p^2}$ into the formula, which penalized large variation in geometry similarity, into the term E_1 of L_1 median, while retained the repulsion term E_2 unchanged to form the feature-preserving locally optimal projection (FLOP).

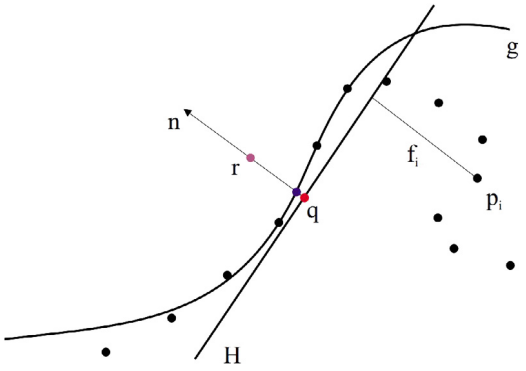

Fig. 5. The projection of MLS, originally shown in [19].

Since LOP is an isotropic operator, Huang et al. [55] replaced $\theta\left(\left\|p_i - q_j\right\|\right)$ in the term E_1 with following equation

$$\Theta\left(\mathbf{n}_i, p_i - q_j\right) = e^{-\left(\mathbf{n}_i^T\left(p_i - q_j\right)\right)^2/\sigma_p^2} \tag{6}$$

where $\mathbf{n}_i$ denoted normal at point p_i, which was estimated based on an anisotropic neighborhood followed by being smoothed using bilateral normal smoothing. This anisotropic LOP could obtain better sharp feature preservation.

Since the above methods require high computational effort, Preiner et al. [56] designed an efficient variant of the locally optimal projection operator. They employed the Gaussian Mixture Model formed by regularizing the hierarchical EM algorithm to cluster points to represent the point cloud's density. Then, they redefined and evaluated the attractive force to obtain the continuous LOP (CLOP). Finally, CLOP was applied to the Gaussians instead of points to achieve speedups.

There are many approaches built upon the work over Moving Least Squares by Levin [57–59]. Alexa handled noise problem by iteratively projecting the points onto the MLS surface [19,60] defined by two steps (see Fig. 5). The first step finds a local reference plane $H = \left\{x \in R^3 | < \mathbf{n}, x > - D = 0\right\}$ by locally minimizing $\sum_{p\in P}(< \mathbf{n}, p > - D)^2 e^{-\|r-q\|}/h^2$, where q is the projection of r onto H and h is a global scale factor controlling the degree of smoothing. A local polynomial approximation with respect to the reference domain is then computed (see Fig. 6). However, in terms of irregularly sampled point cloud, it is difficult to find a suitable h. Pauly et al. [61] collected the k-nearest neighbors and dynamically chose $h = r/3$ to ensure that points in the k-neighborhood contributed to the least-squares optimization.

The drawback of MLS is that the process involves a non-linear optimization for finding the reference plane [8]. It is computational expensive. Alexa and Adamson [62] proposed a simpler projection procedure, in which local reference planes were defined by a weighted average position at x. $a(x) = \sum_i \theta\left(\left\|x - p_i\right\|\right) p_i / \sum_i \theta\left(\left\|x - p_i\right\|\right)$ and a normal was computed using weighted averages of input normal $\mathbf{n}_i$. $\mathbf{n}(x) = \sum_i \theta\left(\left\|x - p_i\right\|\right) \mathbf{n}_i / \left\|\sum_i \theta\left(x - p_i\right) \mathbf{n}_i\right\|$. Amenta and Kil [63] presented an explicit version of MLS surface definition in terms of the critical points of an energy function e_{MLS} on lines determined by a vector field $\boldsymbol{n}(x) = \operatorname{argmin} e_{MLS}$ and gave a simple procedure that produced a point of the MLS surface, which was not considered in Levin's paper.

In fact, MLS is a low-pass filter, which could smooth shape features of point cloud. Thoughtfully, based on M-estimator procedure and moving least squares method, Mederos et al. [64] gave a new smoothing operator $Q(r) = r + t_r \boldsymbol{n}_r$ computed by using an effective numerical optimization procedure to preserve salient features, where n_r and t_r were determined by the fitting of a plane H in the neighborhood of point r. Fleishman et al. [65] introduced a robust moving least squares technique based on forward-search paradigm to deal with noise, outliers and sharp features. Their work classifies the neighborhood of the point into subsets of outlier-free smooth regions of the surface, and then the moving least squares projection mechanism is operated on points. But this method requires very dense point clouds and is time-consuming. Adamson and Alexa [66] adopted cell complexes to preserve the shape features by decomposing the object into cells of different dimensions, while this decomposition demanded effort by the users.

From the above approaches, it can be seen that the plane fitting operation is unstable in regions of high curvature if the sampling rate is below a threshold. Guennebaud and Gross [67] thus performed algebraic sphere fitting to improve stability where planar MLS fails.

Fua et al. [68] fitted a local quadric patch to a neighborhood of every point. They then iteratively smoothed the raw point clouds by using these estimated surfaces. In order to deal with outliers, a metric was defined to measure whether or not two points belong to the same local surface. This process was curvature-preserving and could not smooth out relevant features.

Wang et al. [69] used the mean shift clustering and adaptive scale sampling consensus to compute the best tangent planes for all points so as to detect and remove outliers. Subsequently, they dichotomized the feature and non-feature points and denoised them, respectively, by projecting them onto the corresponding quadratic surfaces fitted using their neighborhoods, which were determined based on the normal-based region growing technique.

2.4. Signal processing based method

Signal processing methodology can also be extended to point cloud filtering. Based on Taubin's method [70] that applied the Laplacian operators to filter the mesh, Linsen [71] developed a filtering operator

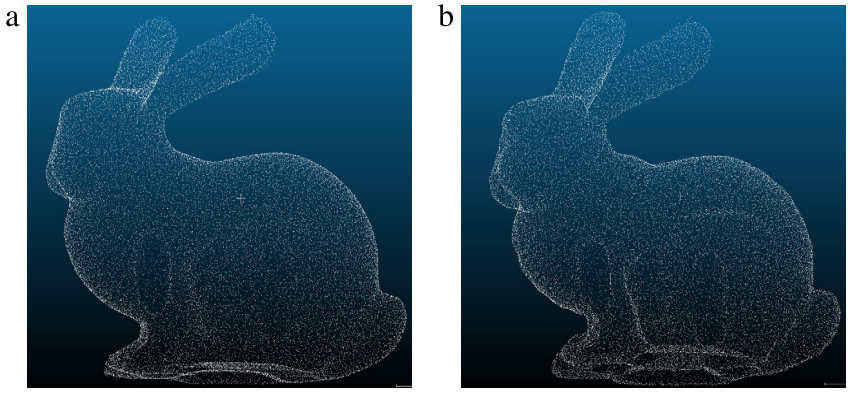

Fig. 6. Denoise point sets via MLS. (a) Input noisy model; (b) filtered result.

containing three features: locality, non-shrinkage and including geometrical aspects. Pauly et al. [72] introduced the discrete approximation of Laplacian to point cloud. In order to handle shrinkage, their method approximates the local volume change and compensates the shrinkage by displacing the neighbors of point $\mathbf{p}$ with $(\mathbf{p}-\mathbf{p}')/k, \mathbf{p}' = \mathbf{p} + \lambda \triangle \mathbf{p}$, where $\triangle\mathbf{p}$ is discrete approximation of the Laplacian at point $\mathbf{p}$. This method, however, could smooth features and lead to the vertex drift.

Motivated by Fourier transformation, Pauly and Gross [73] used spectral processing to filter point cloud. They applied a discrete Fourier transform to obtain a spectral decomposition of point cloud. The elaborate filtering operation was then carried out by manipulating the frequency spectrum via the Wiener filter.

Rosman et al. [74] first constructed the collaborative patch P_i using a set of similar patches for each patch $\widehat{P}_i$ based on certain distance measurement, and then their algorithm included two phases. Phase I prevents shrinkage using the eigen-functions of the Laplace–Beltrami operator of the collaborative patch, followed by Wiener filtering based on the denoised estimation from phase I.

2.5. PDEs-based filtering technique

PDE (Partial Differential Equations), one of the most important tools widely used in computer vision and computer graphic, has been successfully applied to many applications tasks such as image or mesh denoising. PDEs-based techniques for filtering point cloud can be considered as an extension of that to the triangular meshes.

By assembling the local finite matrices constructed over small point neighborhoods into a single matrix, Clarenz et al. [75] presented a framework for processing point cloud via PDEs. They discretized and solved an anisotropic geometric diffusion equation for filtering point cloud. This method could smooth out the noise whereas features were preserved or enhanced.

Extending Taubin's work [76], Lange and Polthier [77] took the directional curvatures, principal curvatures and the Weingarten map into account to obtain an anisotropic geometric mean curvature flow method to filter point cloud. First, they used directional curvatures to produce a Weingarten map by discretizing an integral formula. Next, they computed the eigen-values and eigen-vectors corresponding to principal curvatures and directions respectively. Finally, they defined an anisotropic Laplacian for mean curvature flow technique by applying curvature information to modify Laplacian. This method can detect features such as edges but need to choose edge quotients manually.

Based on covariance analysis and constructed directional curvature, Xiao et al. [78] proposed an anisotropic curvature flow approach. The corresponding equation they used is $\partial u/\partial t = g\left(-\bar{k}\mathbf{n}\right) + (1-g)(I-u)$, where G_u is a Gaussian kernel, $\bar{k}$ denotes the directional curvature. The terms $g\left(\left|k_H\left(G_u \times u\right)\right|\right)$ and *1-g* work as a moderated selector of the diffusion process. *I–u* is a forcing term to eliminate shrinkage. Experimental results show that their method achieves a better performance in noise removal, feature preservation, anti-shrinking and anti-point drifting.

To extend many processing methods on images into 3D point cloud, Lozes et al. [79] constructed weighted arbitrary graphs for representing point cloud, which needed to consider local neighborhood information. The transposition of framework of PDEs, such as *p*-Laplacian operator [80] and PDEs morphological operators, was adapted to these arbitrary graphs (that is PDEs on weighted graphs [81]) to filter point cloud.

Because PDEs-based techniques need to compute the partial differential properties, it is time consuming to use these methods to filter point cloud.

2.6. Hybrid filtering technique

Hybrid strategies usually use two or more filtering techniques together to deal with the raw point clouds.

Liu et al. [82] developed an iterative framework to process point cloud. They first applied a Weighted Locally Optimal Projection (WLOP) operator to efficiently filter out noise. Then, they used a mean-shift-based outlier removal operator to detect and eliminate outliers. However, the limitations of their algorithm are its difficulty in recovering sharp features and its computational costs.

Zaman et al. [16] proposed a density-based approach for denoising point cloud. First, they used particle-swam optimization technique to select the bandwidth for approximation of kernel density estimation (KDE). Then, mean-shift clustering algorithm was utilized to the KDE to remove outliers. Finally, the noise in the remaining points was reduced by applying bilateral filter. The result showed that this method is robust with highly noisy point cloud.

2.7. Other methods

There are many other methods used for filtering point cloud. Szeliski and Tonnesen et al. [83] developed oriented particles (point clouds). Each point, having a position and its own local coordinate frame, defined both a normal and a local tangent plane of the represented surface. They devised a set of potential functions to group these points into surface-like arrangements. Their method could shape the surface by moving points around.

Voxel Grid (VG) filtering method first defines a 3D voxel grid (3D boxes in 3D space) on a point cloud. Then, in each voxel, a point is chosen to approximate all the points that lie on that voxel. Normally, the centroid of these points or the center of this voxel is used as the approximation. The former is slower than the later, while its representative of underlying surface is more accurate. The VG method usually leads to geometric information loss.

Quadtree-based filtering method creates a quadtree as a data structure to organize a point cloud. This representation can improve the speed of neighborhood search, which is beneficial to the efficiency of the filtering algorithms. And then, the filtering strategies or methods (like neighborhood-based approaches or projection-based methods) are used to filter the point clouds.

Digne et al. [84] introduced a similarity based filtering to denoise point cloud. They decomposed the input point clouds into a smooth part S_{smooth} yielded by using the mean curvature motion and a high frequency term $\underset{V}{\rightarrow}$. And then a non-local strategy, filtering high frequency based on L2-distance similarity between local descriptors computed for each point, was used to denoise $\underset{V}{\rightarrow}$ to achieve shape denoising. Nevertheless, this algorithm relied on point densities and cannot differentiate texture and structured noise.

Table 1 presents a summary of the filtering methods in terms of noise removal, feature preservation, outlier removal and other performance. These methods are listed chronologically by year of publication.

3. Experimental results and discussion

After a comprehensive analysis of the major 3D point cloud filtering algorithms, we now proceed to conduct experiments aiming at comparing and evaluating the performance of selected point cloud filtering methods.

3.1. Selected methods

In our experiments, we select 7 different filtering methods which are widely cited and used in comparison. These algorithms have been already briefly introduced in Section 2 and include: Voxel Grid filter (VG), Normal-based Bilateral Filter (NBF), Moving Least Square (MLS), Weighted Locally Optimal Projection (WLOP), Edge Aware Resample (EAR) and L0 minimization (L0). These filtering algorithms are tested on different point cloud models corrupted with Gaussian noise to evaluate corresponding performance.

Table 1
Methods for 3D point cloud filtering.

No.	Reference	Category(section)	Noise removal	Feature preserving	Outliers removal	Performance
1	P. Fua et al. [68] 1996	Projection(1.3)	✓	✓	✓	Be inherently local and parallel
2	Szeliski et al. [83] 1998	Other(1.7)	✓	✓	×	Be helpful for rendering
3	Alexa et al. [19] 2001	Statistical(1.1)	✓	×	×	Controls the fidelity of point cloud
4	Linsen et al. [71] 2001	SP(1.4)	✓	×	×	handles a number of measuring errors
5	Pauly and Gross [73] 2001	SP(1.4)	✓	✓	×	Be efficient, robust and amenable to hardware acceleration
6	Pauly et al. [72] 2002	SP(1.4)	✓	×	✓	Guarantees bounded curvature and stability
7	Amenta et al. [63] 2004	Projection(1.3)	✓	×	✓	Be better than MLS [19]
8	Mederos et al. [64] 2003	Projection(1.3)	✓	✓	✓	Outperforms MLS [19]
9	Clarenz et al. [75] 2004	PDEs(1.5)	✓	✓	×	Be quite suitable for surface fairing application
10	Jones et al. [40] 2004	Neighborhood(1.2)	✓	✓	×	Preserves much more features contributing to rendering
11	Miropolsky A. et al. [35] 2004	Neighborhood(1.2)	✓	✓	×	This method is simple, fast and accurate
12	Fleishman et al. [65] 2005	Projection(1.3)	✓	✓	✓	Handles complex shapes and suppresses the shrinkage
13	Lange and Polthier [77] 2005	PDEs(1.5)	✓	✓	×	Well enhances geometric features (edges and corners)
14	Schall et al. [18] 2005	Statistical(1.1)	✓	✓	×	Works well in combination with surface reconstruction method
15	Adamson and Alexa [66] 2006	Projection(1.3)	✓	✓	✓	Be new and more flexibility
16	Esmeide A. et al. [15] 2006	Statistical(1.1)	✓	✓	✓	Shrinkage prevention and Bias correction
17	Hu et al. [44] 2006	Neighborhood(1.2)	✓	✓	×	Avoids local shape corruption and Outperforms bilateral fitlering
18	Jenke et al. [20] 2006	Statistical(1.1)	✓	✓	×	Works robustlyin practice
19	Xiao et al. [78] 2006	PDEs(1.5)	✓	✓	×	Better than bilateral filter and Laplacian method [71]
20	Lipman et al. [51] 2007	Projection(1.3)	✓	×	✓	Outperforms MLS [19]
21	Huhle et al. [12] 2008	Neighborhood(1.2)	✓	✓	✓	Yields unbiased results
22	Kalogerakis et al. [21] 2008	Statistical(1.1)	✓	✓	✓	Works effective point clouds with varying amounts of noise and outliers
23	Schall et al. [17] 2008	Neighborhood(1.2)	✓	✓	×	Outperforms bilateral filter and Be flexibly applicable to noisy models
24	Huang et al. [49] 2009	Projection(1.3)	✓	✓	✓	Does not require normal estimation, local plane fitting and other parametric representation
25	Avron et al. [22] 2010	Statistical(1.1)	✓	✓	×	Outperforms LOP [51] and RMLS [65]
26	Ye et al. [53] 2011	Projection(1.3)	✓	✓	✓	Avoids shrinkage and improves LOP [51]
27	Digne et al. [84] 2012	Other(1.7)	✓	✓	×	Outperforms NL means [17] and Bilateral filter
28	Liu et al. [82] 2012	Combination(1.6)	✓	×	✓	Outperforms MLS [19] and WLOP [49]
29	Huang et al. [55] 2013	Projection(1.3)	✓	✓	✓	Outperforms MLS [19] and RMLS [65]
30	Liao et al. [54] 2013	Projection(1.3)	✓	✓	✓	Accelerates LOP [51]
31	Orts-Escolano et al. [24] 2013	Statistical(1.1)	✓	✓	✓	Outperforms voxel grid filter
32	Rosman et al. [74] 2013	SP(1.4)	✓	✓	×	Outperforms NL means [17], Bilateral filter and MLS [19]
33	Wang et al. [46] 2013	Neighborhood(1.2)	✓	✓	✓	Outperforms WLOP [49], RMLS [65] and [84]
34	Wang et al. [69] 2013	Projection(1.3)	✓	✓	✓	Outperforms RMLS [65] in terms of sharp edges and corners preservation
35	Lozes et al. [79] 2014	PDEs(1.5)	✓	✓	×	Enables a better visualization of the point cloud
36	Ma, S. et al. [36] 2014	Neighborhood(1.2)	✓	✓	×	Gets more accurate results
37	Preiner et al. [56] 2014	Projection(1.3)	✓	✓	×	Be more accuracy than WLOP [49] and Speedups using GPU
38	Sun et al. [23] 2015	Statistical(1.1)	✓	✓	×	Outperforms WLOP [49], EAR [55] and RMLS [65]
39	Zaman et al. [16] 2016	Combination(1.6)	✓	✓	✓	Deals with highly noisy data robustly

VG, NBF and MLS are available in the Point Cloud Library(PCL) Version 1.7.2. WLOP and EAR were implemented in C++, while the others (namely RMLS and L0) were programmed by MATLAB R2016a. The experiments are carried out on a PC with Intel(R) Core(TM) i7-4790 CPU @ 3.60 GHz and 16 GB memory.

3.2. Efficiency

A comprehensive evaluation of the considered filtering algorithms is given with respect to computational efficiency. We calculate the running time required by each method to filter a model of a point cloud. Since the number of points is one of the main factors that would affect the computational time, models with various numbers of points are chosen to test the performance of these methods.

3.3. Measures

In order to evaluate the quality of results, two error metrics δ and D_{mean} are used. δ represents the averaged angle over all angles between the ground truth point normals and the resulting point normals. D_{mean}

Table 2
Timings for our method and other filtering methods on different models (in ms).

Model	Points	VG	NBF	MLS	RMLS	WLOP	EAR	L0
Office chair	29,148	86	8,269	7,776	13,556	14,962	71,082	309,610
i-H Bunny	34,834	95	5,428	8,015	17,239	17,735	87,920	297,046
Julius	36,201	165	11,516	9,875	19,741	21,125	73,998	318,776
Sofa	48,668	222	14,836	15,863	30.636	24,923	98,276	420,314
Coffee-mug	49,017	313	12,750	11,161	23,824	25,096	122,659	341,659
Bowl	53,166	283	13,853	13,662	25,440	27,285	101,910	365,420
Iron	85,574	324	24,144	22,558	44,632	46,702	175,409	559,228
Armadillo	172,974	767	20,225	23,803	50,069	99,650	365,790	1,211,477
Dragon	437,645	865	514,836	80,013	164,247	227,273	940,147	975,000
Table	460,400	2,050	590,851	89,597	185,530	261,235	995,090	1,087,200

Table 3
Error metrics for different methods.

Model	Error	VG	NBF	MLS	RMLS	WLOP	EAR	L0
Chair	δ	32.125	20.336	8.962	8.932	8.904	8.785	8.752
	D_{mean}	0.427	0.307	0.133	0.132	0.134	0.132	0.130
i-H bunny	δ	10.223	6.021	5.862	6.105	5.710	5.553	5.559
	D_{mean}	0.185	0.092	0.089	0.099	0.089	0.085	0.086
Sofa	δ	12.850	10.126	10.388	7.211	5.334	5.186	5.032
	D_{mean}	0.174	0.169	0.167	0.150	0.087	0.087	0.080
Julius	δ	13.139	10.647	3.409	3.372	2.166	3.378	3.364
	D_{mean}	0.181	0.177	0.058	0.058	0.038	0.058	0.049
Coffee-mug	δ	12.962	12.729	7.655	7.375	6.447	6.498	6.487
	D_{mean}	0.210	0.201	0.120	0.116	0.100	0.101	0.101
Bowl	δ	25.494	21.676	6.149	6.003	7.667	7.311	7.386
	D_{mean}	0.501	0.312	0.086	0.074	0.107	0.102	0.104
Iron	δ	11.504	7.513	5.844	5.915	6.182	6.180	5.837
	D_{mean}	0.193	0.116	0.091	0.093	0.097	0.096	0.090
Armadillo	δ	9.071	3.616	3.599	4.406	5.124	5.139	5.005
	D_{mean}	0.117	0.052	0.052	0.061	0.097	0.075	0.066
Dragon	δ	26.375	23.036	20.816	21.090	20.816	17.646	16.494
	D_{mean}	0.421	0.359	0.299	0.303	0.310	0.262	0.202
Table	δ	32.481	29.313	17.010	17.449	28.540	21.918	15.998
	D_{mean}	0.553	0.502	0.277	0.285	0.463	0.358	0.254

is the average distant from the resulting points to the corresponding ground truth points.

3.4. Results and discussion

Table 2 gives the running time of different methods over different point cloud models with different numbers of points. Some observations can be found among these algorithms. Firstly, voxel grid approach is the most efficient filter, which is nearly two or three orders of magnitude faster compared with others. Secondly, it is worth pointing out that when the number of points in models is less than 200,000, NBF can achieve almost the same performance in terms of efficiency as MLS, while with the increase of points, NBF becomes very time-consuming. RMLS and WLOP are two or three times slower than MLS and NBF on the whole. Finally, due to different strategies for normal estimation, normal filtering and point updating, L0 and EAR, therefore, are the most computationally expensive algorithms.

The error metrics for different methods are presented in Table 3. According to the error metrics, it is evident that there is an obvious difference between ground truth point clouds and filtered ones by VG, indicating that VG cannot obtain the results as expected. The filtering effectiveness of NBF is similar to VG in some extend apart from the Armadillo model. In addition, for most point cloud models, EAR and L0 achieves a better filtering performance with relatively lower δ and D_{mean}.

Fig. 7 shows the filtered results of these four methods on bowl models with Gaussian noise ($\sigma = 0.005$). It can be seen that the bowl is still noisy after filtering by NBF and VG, while MLS, WLOP and EAR can produce good visual results. Experimental results tested on sofa models with Gaussian noise ($\sigma = 0.01$) are illustrated in Fig. 8. WLOP, EAR and L0 can preserve the shape feature better than other methods. And it can be seen from Fig. 9 that EAR and L0 yield a better results compared to the others in terms of noise removal and feature preserving.

Overall, as for real-time systems on point clouds with a large number of points, VG can be recommended as a better choice regarding to computational efficiency. However, its filtering effect is unsatisfactory. In contrast, EAR and L0 demonstrate a sufficient performance in terms of noise removal, together with feature preserving, yielding a relatively accurate models for further processing (e.g. object recognition). It has to be noted that they are both computationally expensive. MLS can be considered as a relatively good trade-off which provides a balance between running efficiency as well as the filtering effectiveness.

4. Conclusion

This paper has placed a strong emphasis on the comprehensive review of the state-of-the-art algorithms for filtering 3D point cloud. Although there are a few existing research on point cloud filtering, it is believed that filtering on the raw point cloud, being as a crucial step of point cloud processing pipeline, remains a challenging task. A brief discussion of future research directions are presented as follows.

(1) Combination of color and geometric information: For point clouds, especially these containing color information, a pure color or pure geometric attributes based method cannot work well. Hence, it is expected to combine the color and geometric information in the filtering process to further increase the performance of a filtering scheme.

(2) Time complexity reduction: Because point clouds contain a large number of points, some of which can be up to hundreds of thousands

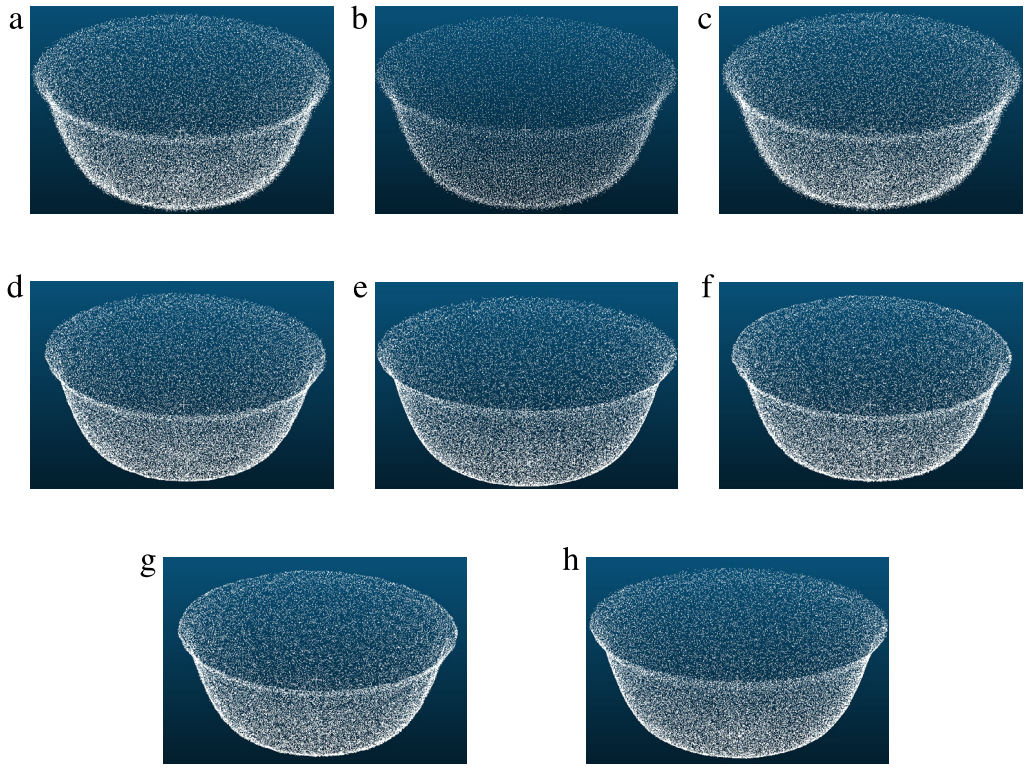

Fig. 7. (a) Noisy bowl model; (b) filtering result with VG; (c) NBF; (d) MLS; (e) RMLS; (f) WLOP; (g) EAR; (h) L0.

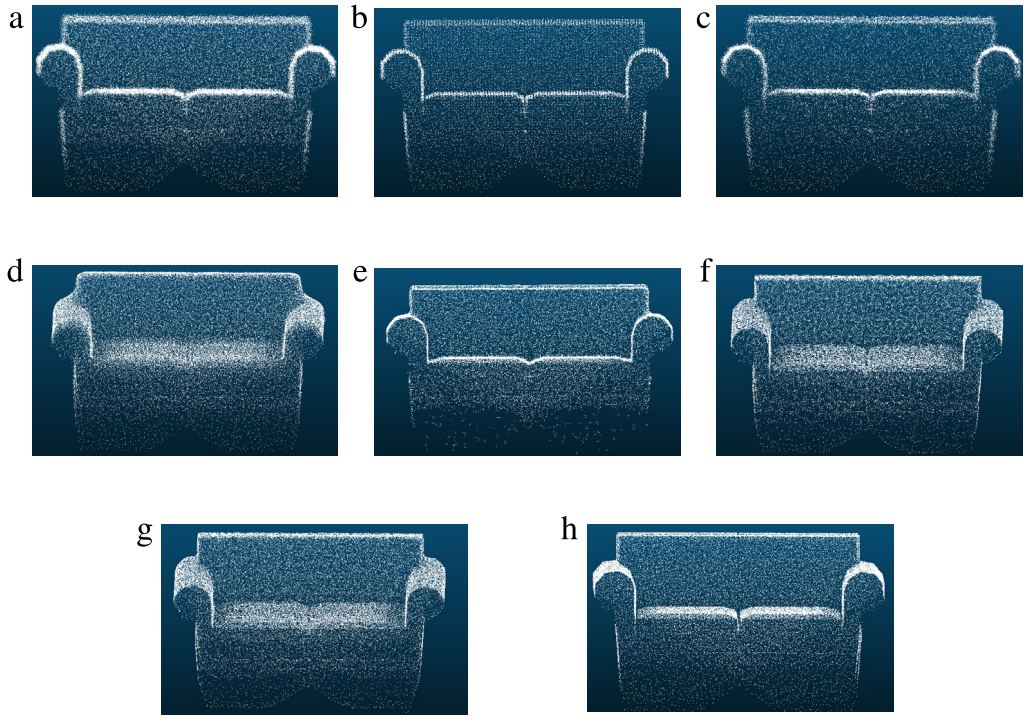

Fig. 8. (a) Noisy sofa model; (b) filtering result with VG; (c) NBF; (d) MLS; (e) RMLS; (f) WLOP; (g) EAR; (h) L0.

or even millions of points, computation on these point clouds is time consuming. It is necessary to develop filtering technologies to filter point cloud effectively to reduce time complexity.

(3) Filtering on point cloud sequence: Since object recognition from a point cloud sequence will become the future research direction. And filtering the point cloud sequence will help to improve the performance and accuracy of object recognition.

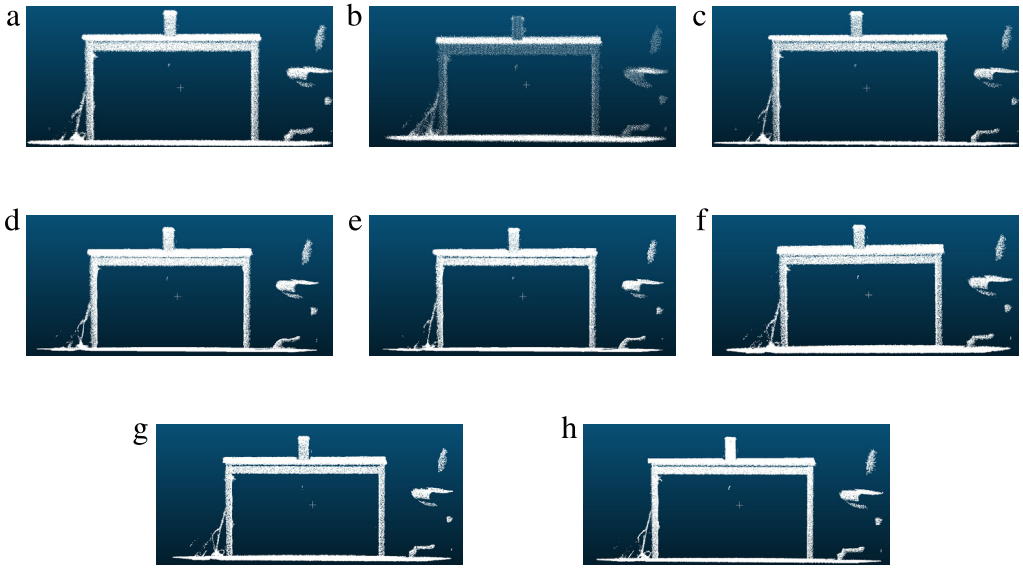

Fig. 9. (a) Noisy table model; (b) filtering result with VG; (c) NBF; (d) MLS; (e) RMLS; (f) WLOP; (g) EAR; (h) L0.

References

[1] R.B. Rusu, S. Cousins, 3d is here: Point cloud library (pcl), in: IEEE International Conference on Robotics and Automation, 2011, pp. 1–4.

[2] A. Aldoma, Z.C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, et al., Tutorial: Point cloud library: Three-dimensional object recognition and 6 dof pose estimation. IEEE Robot. Autom. Mag. 19 (3) (2012) 80–91.

[3] M. Saval-Calvo, S. Orts-Escolano, J. Azorin-Lopez, J. Garcia-Rodriguez, A. Fuster-Guillo, V. Morell-Gimenez, et al., A comparative study of downsampling techniques for non-rigid point set registration using color, in: Bioinspired Computation in Artificial Systems, Springer, 2015, pp. 281–290.

[4] J.C. Rangel, V. Morell, M. Cazorla, S. Orts-Escolano, J. García-Rodríguez, Object recognition in noisy rgb-d data, in: Bioinspired Computation in Artificial Systems, Springer, 2015, pp. 261–270.

[5] J. Park, H. Kim, Y.W. Tai, M.S. Brown, I. Kweon, High quality depth map up-sampling for 3d-tof cameras, in: International Conference on Computer Vision, Barcelona, Nov., 2011, pp. 1623–1630.

[6] N.A.I.M. Rosli, A. Ramli, Mapping bootstrap error for bilateral smoothing on point set, in: AIP Conference Proceedings, Penang, Malaysia, 2014, pp. 149–154.

[7] H. Pfister, M. Gross, Point-based computer graphics, IEEE Comput. Graph. Appl. 24 (4) (2004) 22–23.

[8] L. Kobbelt, M. Botsch, A survey of point-based techniques in computer graphics, Comput. Graph. 28 (6) (2004) 801–814.

[9] D. Holz, S. Behnke, Fast range image segmentation and smoothing using approximate surface reconstruction and region growing, in: International Conference on Intelligent Autonomous Systems, 2012, pp. 61–73.

[10] J. Han, L. Shao, D. Xu, J. Shotton, Enhanced computer vision with microsoft kinect sensor: A review, IEEE Trans. Cybernet. 43 (5) (2013) 1318–1334.

[11] W. Xu, I.S. Lee, S.K. Lee, B. Lu, E.J. Lee, Multiview-based hand posture recognition method based on point cloud, Ksii Trans. Internet Inf. Syst. 9 (7) (2015) 2585–2598.

[12] B. Huhle, T. Schairer, P. Jenke, W. Strasser, Robust non-local denoising of colored depth data, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Anchorge, AK, June, 2008, pp. 1–7.

[13] J. Landa, D. Procházka, J. Šťastný, Point cloud processing for smart systems, Acta Univ. Agricult. Silvicult. Mendelianae Brunensis 61 (7) (2013) 2415–2421.

[14] H. Xie, K.T. McDonnell, H. Qin, Surface Reconstruction of Noisy and Defective Data Sets, Visualization, IEEE, Austin, TX, USA, Oct., 2004, pp. 259–266.

[15] E.A.L. Narváez, N.E.L. Narváez, Point cloud denoising using robust principal component analysis, in: Proceedings of the First International Conference on Computer Graphics Theory and Applications, Setúbal, Portugal, February, 2006, pp. 51–58.

[16] F. Zaman, Y.P. Wong, B.Y. Ng, Density-based denoising of point cloud, 2016. ArXiv preprint arXiv:160205312.

[17] O. Schall, A. Belyaev, H.P. Seidel, Adaptive feature-preserving non-local denoising of static and time-varying range data, Comput. Aided Des. 40 (6) (2008) 701–707.

[18] O. Schall, A. Belyaev, H.P. Seidel, Robust filtering of noisy scattered point data, in: Proceedings Eurographics/IEEE VGTC Symposium Point-Based Graphics, Stony Brook, NY, USA, June, 2005, pp. 71–144.

[19] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C.T. Silva, Point set surfaces, in: Proceedings of the Conference on Visualization, San Diego, CA, USA, Oct., 2001, pp. 21–28.

[20] P.M.W. Jenke, M. Bokeloh, A. Schilling, W. Straßer, Bayesian point cloud reconstruction, Comput. Graph. Forum 25 (3) (2006) 379–388.

[21] E. Kalogerakis, D. Nowrouzezahrai, P. Simari, K. Singh, Extracting lines of curvature from noisy point clouds, Comput. Aided Des. 41 (4) (2009) 282–292.

[22] H. Avron, A. Sharf, C. Greif, D. Cohen-Or, L1-sparse reconstruction of sharp point set surfaces, ACM Trans. Graph. 29 (5) (2010) 1–12.

[23] Y. Sun, S. Schaefer, W. Wang, Denoising point sets via l0 minimization, Comput. Aided Geom. Design 35 (2015) 2–15.

[24] S. Orts-Escolano, V. Morell, J. Garcia-Rodriguez, M. Cazorla, Point cloud data filtering and downsampling using growing neural gas, in: The 2013 International Joint Conference on Neural Networks, IJCNN, Dallas, TX, Aug. 2013, pp. 1–8.

[25] S. Orts-Escolano, J. Garcia-Rodriguez, V. Morell, M. Cazorla, J.A.S. Perez, A. Garcia-Garcia, 3d surface reconstruction of noisy point clouds using growing neural gas: 3d object/scene reconstruction, Neural Process. Lett. 43 (2) (2015) 1C23.

[26] M. Saval-Calvo, S. Orts-Escolano, J. Azorin-Lopez, J. Garcia-Rodriguez, A. Fuster-Guillo, V. Morell-Gimenez, et al., Non-rigid point set registration using color and data downsampling, in: International Joint Conference on Neural Networks, IJCNN, Killarney, July, 2015, pp. 1–8.

[27] B. Fritzke, A growing neural gas network learns topologies, Adv. Neural Inf. Process. Syst. 7 (1995) 625–632.

[28] J. Garcia-Rodriguez, M. Cazorla, S. Orts-Escolano, V. Morell, Improving 3d keypoint detection from noisy data using growing neural gas, in: International Conference on Artificial Neural Networks: Advences in Computational Intelligence, 2013, pp. 480–487.

[29] J.C. Rangel, V. Morell, M. Cazorla, S. Orts-Escolano, J. García-Rodríguez, Object recognition in noisy rgb-d data using gng, PAA Pattern Anal. Appl. (2016) 1–16.

[30] C. Tomasi, R. Manduchi, Bilateral filtering for gray and color images, in: International Conference on Computer Vision, Bombay, Jan., 1998, pp. 839–846.

[31] S. Paris, A gentle introduction to bilateral filtering and its applications in: ACM SIGGRAPH, San Diego, Californila, 2007, pp. 853–879.

[32] S. Fleishman, I. Drori, D. Cohen-Or, Bilateral mesh denoising, ACM Trans. Graph. 22 (3) (2003) 950–953.

[33] T.R. Jones, F. Dur, M. Desbrun, Non-iterative, feature-preserving mesh smoothing, ACM Trans. Graph. 22 (3) (2003) 943–949.

[34] K.W. Lee, W.P. Wang, Feature-preserving mesh denoising via bilateral normal filtering, in: International Conference on Computer Aided Design and Computer Graphics, Hong Kong, China, Dec., 2005, pp. 275–280.

[35] A. Miropolsky, A. Fischer, Reconstruction with 3d geometric bilateral filter, in: Proceedings of the Ninth ACM Symposium on Solid Modeling and Applications, 2004, pp. 225–229.

[36] S. Ma, C. Zhou, L. Zhang, W. Hong, Depth image denoising and key points extraction for manipulation plane detection, in: Intelligent Control and Automation, Shenyang, June, 2014, pp. 3315–3320.

[37] M. Hernandez, J. Choi, G. Medioni, Near laser-scan quality 3-d face reconstruction from a low-quality depth stream, Image Vis. Comput. 36 (2015) 61–69.

[38] B.Q. Shi, J. Liang, Q. Liu, B.Q. Shi, J. Liang, Q. Liu, Adaptive simplification of point cloud using -means clustering, Comput. Aided Des. 43 (8) (2011) 910–922.

[39] W. Xu, I.S. Lee, S.K. Lee, B. Lu, E.J. Lee, Multiview-based hand posture recognition method based on point cloud, Ksii Trans. Internet Inf. Syst. 9 (7) (2015) 2585–2598.

[40] T.R. Jones, F. Durand, M. Zwicker, Normal improvement for point rendering, IEEE Comput. Graph. Appl. 24 (4) (2004) 53–56.

[41] M. Wand, A. Berner, M. Bokeloh, P. Jenke, A. Fleck, M. Hoffmann, et al., Processing and interactive editing of huge point clouds from 3d scanners, Comput. Graph. 32 (2) (2008) 204–220.
[42] B. Moorfield, R. Haeusler, R. Klette, Bilateral filtering of 3d point clouds for refined 3d roadside reconstructions, in: Computer Analysis of Images and Patterns, 2015, pp. 394–402.
[43] S.E. Nasab, S.F. Ghaleh, S. Ramezanpour, S. Kasaei, Permutohedral lattice in 3d point cloud processing, in: International Symposium on Telecommunications, 2014, pp. 289–294.
[44] G. Hu, Q. Peng, A.R. Forrest, Mean shift denoising of point-sampled surfaces, Vis. Comput. 22 (3) (2006) 147–157.
[45] A. Buades, B. Coll, J.M. Morel, A non-local algorithm for image denoising, in: IEEE Computer Society Conference on Computer Vision & Pattern Recognition, San Diego, CA, USA, June, 2005, pp. 60–65.
[46] J. Wang, K. Xu, L. Liu, J. Cao, S. Liu, Z. Yu, et al., Consolidation of low-quality point clouds from outdoor scenes, Comput. Graph. Forum 32 (5) (2013) 207–216.
[47] B. Brown, Statistical uses of the spatial median, J. R. Stat. Soc. Ser. B Stat. Methodol. 45 (1) (1983) 25–30.
[48] C.G. Small, A survey of multidimensional medians, Int. Stat. Rev. 58 (3) (1990) 263–277.
[49] H. Huang, D. Li, H. Zhang, U. Ascher, D. Cohen-Or, Consolidation of unorganized point clouds for surface reconstruction, ACM Trans. Graph. 28 (5) (2009) 89–97.
[50] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, et al., L1-medial skeleton of point cloud, ACM Trans. Graph. 32 (4) (2013) 96.
[51] Y. Lipman, D. Cohen-Or, D. Levin, H. Tal-Ezer, Parameterization-free projection for geometry reconstruction, ACM Trans. Graph. 26 (3) (2007) 221–225.
[52] A. Tagliasacchi, H. Zhang, D. Cohen-Or, Curve skeleton extraction from incomplete point cloud, ACM Trans. Graph. 28 (3) (2009) 341–352.
[53] M. Ye, X. Wang, R. Yang, L. Ren, M. Pollefeys, Accurate 3d pose estimation from a single depth image, in: International Conference on Computer Vision, Barcelona, Nov., 2011, pp. 731–738.
[54] B. Liao, C. Xiao, L. Jin, H. Fu, Efficient feature-preserving local projection operator for geometry reconstruction, Comput. Aided Des. 45 (5) (2013) 861–874.
[55] H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, H. Zhang, Edge-aware point set resampling, ACM Trans. Graph. 32 (1) (2013) 60–61.
[56] R. Preiner, O. Mattausch, M. Arikan, R. Pajarola, M. Wimmer, Continuous projection for fast l1 reconstruction, ACM Trans. Graph. 33 (4) (2014) 70–79.
[57] D. Levin, The approximation power of moving least-squares, Math. Comp. 67 (224) (2000) 1517–1531.
[58] D. Levin, Mesh-independent surface interpolation, in: Geometric Modeling for Scientific Visualization, Springer, Berlin Heidelberg, 2004, pp. 37–49.
[59] T. Dey, S. Goswami, J. Sun, Smoothing Noisy Point Clouds with Delaunay Preprocessing and MLS, The Ohio State University Technical Report No OSU-CISRC-3/04-TR17, 2004.
[60] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C.T. Silva, Computing and rendering point set surfaces, IEEE Trans. Vis. Comput. Graph. 9 (1) (2003) 3–15.
[61] M. Pauly, M. Gross, L.P. Kobbelt, Efficient simplification of point-sampled surfaces, in: Proceedings of the Conference on Visualization, Boston, MA, USA, Oct., 2002, pp. 163–170.
[62] M. Alexa, A. Adamson, On normals and projection operators for surfaces defined by point sets, in: Proceedings of the First Eurographics Conference on Point-Based Graphics, Aire-la-Ville, Switzerland, 2004, pp. 149–155.
[63] N. Amenta, Y.J. Kil, Defining point-set surfaces, ACM Trans. Graph. 23 (3) (2004) 264–270.
[64] B. Mederos, L. Velho, L.H. De Figueiredo, Robust smoothing of noisy point clouds, in: Proc SIAM Conference on Geometric Design & Computing, 2003, pp. 405–416.
[65] S. Fleishman, D. Cohen-Or, C.T. Silva, Robust moving least-squares fitting with sharp features, ACM Trans. Graph. 24 (3) (2005) 544–552.
[66] A. Adamson, M. Alexa, Point-sampled cell complexes, ACM Trans. Graph. 25 (3) (2006) 671–680.
[67] G. Guennebaud, M. Gross, Algebraic point set surfaces, ACM Trans. Graph. 26 (3) (2007) 23.
[68] P. Fua, Reconstructing surfaces from unstructured 3d points, in: Proc Image Understanding Workshop, 1996, pp. 615–625.
[69] J. Wang, Z. Yu, W. Zhu, J. Cao, Feature-preserving surface reconstruction from unoriented noisy point data, Comput. Graph. Forum 32 (1) (2013) 164–176.
[70] G. Taubin, A signal processing approach to fair surface design, in: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, New York, USA, 1995, pp. 351–358.
[71] L. Linsen, Point Cloud Representation, 2001.
[72] M. Pauly, L. Kobbelt, M. Gross, Multiresolution Modeling of Point-Sampled Geometry, 2002.
[73] M. Pauly, M. Gross, Spectral processing of point-sampled geometry, in: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, New York, USA, 2001, pp. 379–386.
[74] G. Rosman, A. Dubrovina, R. Kimmel, Patch-collaborative spectral point cloud denoising, Comput. Graph. Forum 32 (8) (2013) 1–12.
[75] U. Clarenz, M. Rumpf, A. Telea, Fairing of point based surfaces, in: Proceedings of Computer Graphics International, Crete, June, 2004, pp. 600–603.
[76] G. Taubin, Estimating the tensor of curvature of a surface from a polyhedral approximation, in: International Conference on Computer Vision, Washington, DC, USA, 1995, pp. 902–907.
[77] C. Lange, K. Polthier, Anisotropic smoothing of point sets, Comput. Aided Geom. Design 22 (7) (2005) 680–692.
[78] C. Xiao, Y. Miao, S. Liu, Q. Peng, A dynamic balanced flow for filtering point-sampled geometry, Vis. Comput. 22 (3) (2006) 210–219.
[79] F. Lozes, A. Elmoataz, O. Lézoray, Partial difference operators on weighted graphs for image processing on surfaces and point clouds, IEEE Trans. Image Process. 23 (9) (2014) 3896–3909.
[80] V.T. Ta, A. Elmoataz, O. Lézoray, Nonlocal pdes-based morphology on weighted graphs for image and data processing, IEEE Trans. Image Process. 20 (6) (2011) 1504–1516.
[81] A.E. Chakik, X. Desquesnes, A. Elmoataz, Fast 3d surface reconstruction from point clouds using graph-based fronts propagation, ACCV, Berlin, Heidelberg, 2012, pp. 309–320.
[82] S. Liu, K.C. Chan, C.C. Wang, Iterative consolidation of unorganized point clouds, IEEE Comput. Graph. Appl. 32 (3) (2012) 70–83.
[83] R. Szeliski, Surface modeling with oriented particle systems, ACM Siggraph Comput. Graph. 26 (2) (1998) 185–194.
[84] J. Digne, Similarity based filtering of point clouds, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPRW, Providence, RI, 2012, pp. 73–79.