CONTOUR-ENHANCED RESAMPLING OF 3D POINT CLOUDS VIA GRAPHS

Siheng Chen[1,2], *Dong Tian*[1], *Chen Feng*[1], *Anthony Vetro*[1] *and Jelena Kovačević*[2,3]

[1] Mitsubishi Electric Research Laboratories, Cambridge, MA, USA
[2]Dept. of ECE, [3]Dept. of BME, Carnegie Mellon University, Pittsburgh, PA, USA

ABSTRACT

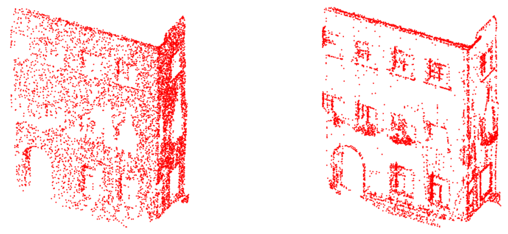
To reduce storage and computational cost for processing and visualizing large-scale 3D point clouds, an efficient resampling strategy is needed to select a representative subset of 3D points that can preserve contours in the original 3D point cloud. We tackle this problem by using graph-based techniques as graphs can represent underlying surfaces and lend themselves well to efficient computation. We first construct a general graph for a 3D point cloud and then propose a graph-based metric to quantify the contour information via high-pass graph filtering. Finally, we obtain an optimal resampling distribution that preserves the contour information by solving an optimization problem. When browsing, the proposed graph-based resampling performs better than uniform resampling both for toy point clouds as well as real large-scale point clouds. Furthermore, as neither mesh construction nor surface normal calculation is involved, the proposed graph-based method is computationally more efficient than the mesh-based methods.

Index Terms— 3D point cloud, graph signal processing, high-pass filtering, resampling strategy

1. INTRODUCTION

With the growth of 3D sensing technologies, 3D point clouds have become an important and practical representation of 3D objects and surrounding environments in many applications, such as virtual reality, mobile mapping, scanning of historical artifacts, 3D printing and digital elevation models [1]. A challenge in 3D point cloud processing is how to handle a large number of incoming 3D points [2, 3]; in applications such as digital documentation of historical buildings and terrain visualization, we need to store billions of incoming 3D points. A large-scale point cloud makes storage and subsequent processing inefficient.

To solve this problem, an approach is to consider efficient data structures to represent 3D point clouds, such as voxel or octree representations [4, 5, 6, 7]; a drawback of this approach comes from discretization error. Another approach is to consider mesh simplification [8, 9, 10]; a drawback of this



(a) Uniform resampling. (b) Proposed resampling.

Fig. 1: Proposed resampling strategy enhances the contour information. The original point cloud is obtained from `http://semantic3d.net/view_dbase.php?chl=1`. Plots (a) and (b) resample 2% points from a 3D point cloud of a building containing $381,903$ points. The method in (b) preserves more contours than the one in (a).

approach is that mesh simplification may change the positions of original points and causes distortion. Compression of dynamic 3D point clouds is considered in [11, 12]; it cannot, however, be used for static 3D point clouds.

In this paper, we consider point cloud resampling; that is, we aim to preserve a representative subset of points from a dense point cloud. Our specific goal is to design an efficient resampling strategy to preserve contour information, which is critical for visualization tasks. The benefits of working with such resampled point cloud are: (1) we significantly reduce the number of points to achieve low-cost processing; and (2) we efficiently enhance contours to avoid saturation problem during visualization; see Fig. 1 for an example.

Conventional contour detection in point clouds typically requires intensive computation to construct meshes and calculate surface normals [13, 14]; we avoid those issues by using a general graph, which captures local dependencies between 3D points and discretizes the original surface of an object. The proposed method is rooted in graph signal processing, which provides a framework to explore the interaction between the graph structure and the signal residing on it [15, 16]. Each coordinate associated with a point in 3D space is treated as a graph signal indexed by the nodes of the underlying graph, allowing us to capture both local and

The work has been done while Siheng Chen was an intern at MERL during the summer of 2016.

global structure in point clouds. To further reduce the computational cost, we propose a randomized resampling strategy on graphs to choose a subset of points, which is more efficient than deterministic sampling strategies [17, 18]. In particular, we first design high-pass graph filters to detect contours in a 3D point cloud and then use a graph resampling strategy to choose samples. The main idea is to select samples according to a nonuniform resampling distribution, which provably preserves contour information in the original point cloud. The main advantage of a randomized resampling strategy is that once the resampling distribution is obtained, it is efficient to take arbitrary number of samples [19].

2. GRAPHS FOR POINT CLOUDS

In this section, we cover the background material. We first formulate a task of resampling a 3D point cloud and then introduce graph signal processing, which lays the foundation for our proposed methods.

Problem Formulation. We consider a matrix representation of a point cloud with N points,

$$\mathrm{X} = \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \mathbf{s}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_N \end{bmatrix}^T \in \mathbb{R}^{N \times 3},$$

where $\mathbf{s}_i \in \mathbb{R}^N$ represents the ith coordinate and $\mathbf{x}_i \in \mathbb{R}^3$ represents the 3D coordinates of the ith point. The number of points N is usually huge for large-scale point clouds, making it a challenge to store, process, and visualize. To reduce the storage and computational cost, we consider choosing a subset of representative points from the original point cloud. Since the original point cloud is sampled from an object, we call this task *resampling*. We resample M $(M < N)$ points from a point cloud leading to the resampled point cloud, $\mathrm{X}_{\mathcal{M}} = \Psi \mathrm{X} \in \mathbb{R}^{M \times 3}$, where $\mathcal{M} = (\mathcal{M}_1, \dots, \mathcal{M}_M)$ denotes the sequence of resampled indices, $\mathcal{M}_i \in \{1, \dots, N\}$ and $|\mathcal{M}| = M$, and the resampling operator Ψ is a linear mapping from $\mathbb{R}^N$ to $\mathbb{R}^M$, defined as $\Psi_{i,j}$ is 1 when $j = \mathcal{M}_i$ and 0, otherwise.

The complexity of computing resampling is a critical issue, because we work with a large-scale point cloud. To solve this problem, we consider a randomized resampling strategy, where the resampled indices are chosen according to a resampling distribution. Let $\{\pi_i\}_{i=1}^N$ be a series of resampling probabilities, where π_i denotes the probability of selecting the ith sample in each random trial. Once the resampling distribution is chosen, it efficiently generates samples. The goal here is to find a resampling distribution that preserves contour information in the original point cloud.

Graph construction for point clouds. A graph is a natural and efficient way to represent a point cloud because it represents a discretized version of an original surface. In computer graphics, polygon meshes, as a class of graphs with particular connectivity restrictions, are extensively used to represent the shape of an object [20, 12]; however, mesh construction usually requires sophisticated geometry analysis and the mesh representation may not be suitable for analyzing point clouds because of connectivity restrictions. Here we extend polygon meshes to general graphs by relaxing the connectivity restrictions. Such graphs are easier to construct and flexibly capture geometry information.

We construct a general graph of a point cloud by encoding the local geometry information through an adjacency matrix $\mathrm{W} \in \mathbb{R}^{N \times N}$. The edge weight between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$\mathrm{W}_{i,j} = \begin{cases} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{\sigma^2}}, & \|\mathbf{x}_i - \mathbf{x}_j\|_2 \le \tau; \\ 0, & \text{otherwise}, \end{cases} \qquad (1)$$

where variance σ and threshold τ are parameters. Equation (1) shows that when the Euclidean distance of two points is smaller than a threshold τ, we connect these two points by an edge and the edge weight depends on the similarity of two points in the 3D space. The weighted degree matrix D is a diagonal matrix with diagonal element $\mathrm{D}_{i,i} = \sum_j \mathrm{W}_{i,j}$ reflecting the density around the ith point. This graph can be efficiently constructed via a tree data structure, such as octree [6, 7]. Here we only use the 3D coordinates to construct a graph, but it is also feasible to take other attributes into account in (1).

3. GRAPH-BASED RESAMPLING

The proposed method includes a high-pass graph filter, which finds contours, and a randomized resampling strategy, which reduces the number of 3D points.

Local variation via high-pass graph filtering. A graph filter is a system that takes a graph signal as an input and produces another graph signal as an output. Let $\mathrm{A} \in \mathbb{R}^{N \times N}$ be a *graph shift operator*, which is the most elementary graph filter. Here we use the transition matrix as a graph shift operator; that is $\mathrm{A} = \mathrm{D}^{-1}\,\mathrm{W}$, where D is a degree matrix and W is an adjacency matrix. The graph shift replaces the signal value at a node with a weighted linear combination of values at its neighbors; that is, $\mathbf{y} = \mathrm{A}\,\mathbf{s} \in \mathbb{R}^N$, where $\mathbf{s} \in \mathbb{R}^N$ is an input graph signal (attribute of a point cloud). A linear, shift-invariant graph filter is a polynomial in the graph shift [15] $h(\mathrm{A}) = \sum_{\ell=0}^{L-1} h_\ell \, \mathrm{A}^\ell$ where $h_i, i = 0, 1, \dots, L-1$ are filter coefficients and L is the length of this graph filter. Its output is given by the matrix-vector product $\mathbf{y} = h(\mathrm{A})\mathbf{s} \in \mathbb{R}^N$.

In image processing, a high-pass filter is used to extract edges and contours. Similarly, we use a high-pass graph filter to extract contours in a 3D point cloud, that is, those points that break the trend formed by their neighboring points and carry information. In some prior work, sophisticated geometry-related computation is required, such as surface normals, to detect contours [13]; we instead identify contour points by the response of high-pass graph filtering. The response of the ith point is

$$f_i(\mathrm{X}) = \| \, (h(\mathrm{A})\,\mathrm{X})_i \|_2^2 \,, \qquad (2)$$

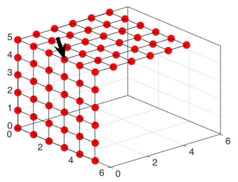


Fig. 2: The pairwise difference based local variation (4) cannot capture the contour points connecting two faces.

where $h(\mathrm{A})$ is a high-pass graph filter and $(\cdot)_i$ extracts the ith row. We call $f(\mathrm{X}) \in \mathbb{R}^N$ local variation, because it quantifies the energy of response after high-pass graph filtering.

A basic high-pass graph filter design is a Haar-like [21, 22] high-pass graph filter $h_{\mathrm{HH}}(\mathrm{A}) = \mathrm{I} - \mathrm{A}$. In the graph vertex domain, the response of the ith point is

$$(h_{\mathrm{HH}}(\mathrm{A})\,\mathrm{X})_i \;=\; \mathbf{x}_i - \sum_{j \in \mathcal{N}_i} \mathrm{A}_{i,j}\,\mathbf{x}_j, \tag{3}$$

where $\mathcal{N}_i$ denotes the neighborhood of the ith point. Because A is a transition matrix, $\sum_{j\in\mathcal{N}_i} \mathrm{A}_{i,j} = 1$ and $h_{\mathrm{HH}}(\mathrm{A})$ compares the difference between a point and the weighted linear combination of its neighbors. The geometric interpretation is that (3) calculates the distance between the original point and the convex combination of its neighbors, reflecting how much information we know about a point from its neighbors. When the local variation of a point is large, the distance between this point and the convex combination of its neighbors is long and this point provides a large amount of information.

Note that while the ordinary graph Laplacian matrix is often used to measure variation, it may not capture geometry change in some cases [16]. Let $\mathrm{L} = \mathrm{D} - \mathrm{W} \in \mathbb{R}^{N\times N}$ be a ordinary graph Laplacian matrix, the graph Laplacian based total variation $\mathrm{Tr}\left(\mathrm{X}^T\,\mathrm{L}\,\mathrm{X}\right) \;=\; \sum_i \sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2$, and a pairwise difference based variation of the ith point

$$f_i^{(P)}(\mathrm{X}) \;=\; \sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}\left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2. \tag{4}$$

Consider Fig. 2; the points are uniformly spread along the faces of a cube. Each point connects to its adjacent four points with the same edge weight. The pairwise difference based local variations of all the points are the same, indicating no contour in this point cloud. However, some points (such as the one indicated by a black arrow) should be contour points. On the other hand, a Haar-like high-pass graph filter can detect the contour points connecting two faces; see Fig. 3.

Randomized resampling strategy. We resample a point cloud M times. In the jth step, we independently choose a point $\mathcal{M}_j = i$ with probability π_i. Let $\Psi \in \mathbb{R}^{M\times N}$ be the resampling operator in Section 2 and $\mathrm{S} \in \mathbb{R}^{N\times N}$ be a diagonal rescaling matrix with $\mathrm{S}_{i,i} = 1/\sqrt{M\pi_i}$. We evaluate the performance of a resampling operator by a reconstruction error,

$$D_{f(\mathrm{X})}(\Psi) \;=\; \left\|\mathrm{S}\,\Psi^T\Psi f(\mathrm{X}) - f(\mathrm{X})\right\|_2^2, \tag{5}$$

where $\|\cdot\|_2$ is the spectral norm and feature $f(\cdot)$ is defined in (2) with $h(\cdot)$ defined in (3). $\Psi^T\Psi \in \mathbb{R}^{N\times N}$ is a zero-padding operator, which a diagonal matrix with diagonal elements $(\Psi^T\Psi)_{i,i} > 0$ when the ith point is sampled, and 0, otherwise. The zero-padding operator $\Psi^T\Psi$ ensures that the resampled points and the original point cloud have the same size. S is used to compensate non-uniform weights during resampling. $\mathrm{S}\,\Psi^T$ is the most naive interpolation operator that reconstructs the original feature $f(\mathrm{X})$ from its resampled version $\Psi f(\mathrm{X})$ and $\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})$ represents the preserved features after resampling in a zero-padded form. Lemma 1 below shows that $\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})$ is an unbiased estimator to the original feature.

Lemma 1. Let $\mathrm{X} \in \mathbb{R}^{N\times 3}$ be a point cloud. Then,

$$\mathbb{E}_{\Psi\sim\pi}\left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right) \;=\; f(\mathrm{X}).$$

The evaluation metric $D_{f(\mathrm{X})}(\Psi)$ measures the reconstruction error; that is, how much feature information is lost after resampling without using sophisticated interpolation operator. When $D_{f(\mathrm{X})}(\Psi)$ is small, preserved features after resampling are close to the original features, meaning that little information is lost.

The expectation $\mathbb{E}_{\Psi\sim\pi}\left(D_{f(\mathrm{X})}(\Psi)\right)$ is the expected error caused by resampling and quantifies the performance of a resampling distribution π. Our goal is to minimize $\mathbb{E}_{\Psi\sim\pi}\left(D_{f(\mathrm{X})}(\Psi)\right)$ over π to obtain an optimal resampling distribution in terms of preserving features $f(\mathrm{X})$. We now derive the mean square error of the objective function (5).

Theorem 1. The mean square error of (5) between the preserved feature and the original feature is

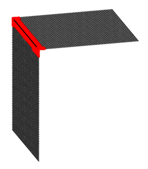
$$\mathbb{E}_{\Psi\sim\pi} D_{f(\mathrm{X})}(\Psi) \;=\; \mathrm{Tr}\left(f(\mathrm{X})\,\mathrm{Q}\,f(\mathrm{X})^T\right), \tag{6}$$

where $\mathrm{Q} \in \mathbb{R}^{N\times N}$ is a diagonal matrix with $\mathrm{Q}_{i,i} = 1/\pi_i - 1$.

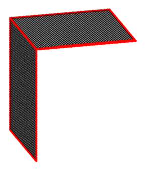
The optimal resampling distributions can then be obtained by minimizing the mean square error (6).

Theorem 2. The optimal resampling strategy according to (5) is $\pi_i^* \;\propto\; \|f_i(\mathrm{X})\|_2 \;\propto\; \left\|\left(h_{\mathrm{HH}}(\mathrm{A})\,\mathrm{X}\right)_i\right\|_2^2$, with $f(\cdot)$ from (2) and $h_{\mathrm{HH}}(\cdot)$ from (3).

We omit the proofs due to limited space. The main idea is to minimize (6) with respect to π being a valid probability distribution. The optimal resampling distribution in Theorem 2 is proportional to the magnitude of response after high-pass graph filtering; that is, points associated with high magnitudes have high probability to be selected. In practice, we directly resample points according to the optimal resampling distribution in Theorem 2. The computation only involves a matrix multiplication, which takes $O(\|\mathrm{vec}(\mathrm{A})\|_0)$, where $\|\mathrm{vec}(\mathrm{A})\|_0$ is the number of nonzero elements in the graph shift A.



(a) Difference of normals [23].



(b) Local variation (2).

Fig. 3: Haar-like high-pass graph filtering based local variation (2) outperforms the difference of normals.

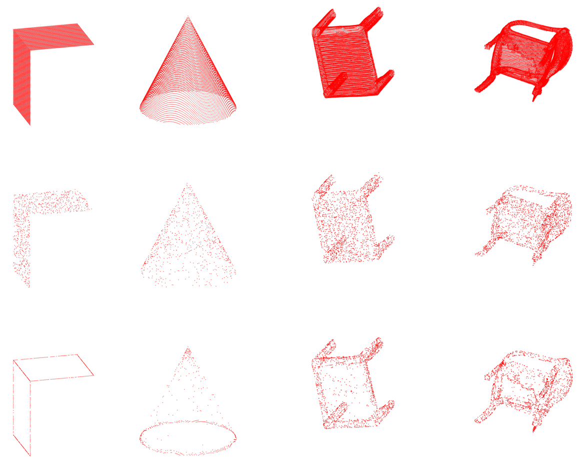


Fig. 4: Haar-like high-pass graph filtering based local variation (2) outperforms pairwise difference based local variation (4). The first row shows the original point clouds; the second and third rows show the resampled versions with respect to two local variations: pairwise difference based local variation (4) and Haar-like high-pass graph filtering based local variation (2), respectively. The two resampled versions have the same number of points—10% of points in the original point cloud.

4. EXPERIMENTS

As it is hard to quantitatively evaluate the quality of resampled 3D point cloud due to the irregular distribution of 3D points and the lack of a proper perception model, we validate our proposed method by visual comparison. The quantitative comparison is left for future work.

First, we compare the proposed method with a typical benchmark algorithm, difference of normals (DoN) [23], which is used for segmentation and contour detection. As a contour detection technique, DoN computes the difference between surface normals calculated at two scales. We used a 3D point cloud with two faces containing $10,000$ points as shown in Fig. 3; the difference of normals (DoN) method (first column) is illustrated on the left and the proposed Haar-like high-pass graph filtering based local variation (2) on the right. In each plot, we highlight the points with the top 10% largest DoN scores/local variations. In Fig. 3 (a), we see that DoN fails to find the boundary in the plane because the surface normal does not change. The performance of DoN is also sensitive to pre-designed radius. For example, the difference of normals cannot capture precise contours in the hinge. On the other hand, local variation captures all the contours precisely in Fig. 3 (b). Further, DoN needs to compute the first principle component of the neighboring points for each 3D point, which is computationally inefficient. The local variation only involves a sparse matrix and vector multiplication, which is computationally efficient.

We further compare the performance between Haar-like high-pass graph filtering based local variation (2) and pairwise difference based local variation (4). Examples of point cloud, including hinge, cone, table, chair, sofa and trash container were used as in Fig. 4. The second and third rows show the resampled versions with respect to different local variations: pairwise difference based local variation (4) and Haar-like high-pass graph filtering based local variation (2), respectively. Both resampled versions collects 10% points in the original point cloud. For two simulated objects, the hinge and the cone (first two columns), the pairwise difference based local variation (4) fails to detect contour and the Haar-like high-pass graph filtering based local variation (2) detects all the contours. For the real objects in the last two columns, the Haar-like high-pass graph filtering based local variation (2) also works well.

For a large-scale point cloud, a building with $381,903$ 3D points, the result is shown in Fig. 1. Fig. 1 (a) shows the resampled points by using uniform resampling and Fig. 1 (b) shows the resampled points by using high-pass graph filtering based resampling. We see that Fig. 1 (b) enhances more contours than Fig. 1 (a). The entire computation process, including graph construction, local variation computation and resampling, was running on Matlab in a desktop and took around 30 seconds.

5. CONCLUSIONS

We propose a resampling strategy to reduce the number of points in a 3D point cloud while preserving contours. To avoid costly computation, such as mesh construction or surface normal calculation, we construct a general graph to represent 3D point clouds and propose graph-based methods for resampling. We detect contours by using a high-pass graph filter and formulate an optimization problem to preserve contours and obtain the optimal resampling distribution. Finally, we visually validate the effectiveness of the proposed resampling strategy on both toy and large-scale examples; results show that the contours are preserved during resampling. A future application of contour-enhanced resampling is point cloud registration. We can resample a small number of key points from two point clouds for matching, which reduces both computational and storage costs. The performance can be quantified by registration error.

6. REFERENCES

[1] R. B. Rusu and S. Cousins, “3D is here: Point cloud library (PCL),” in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, CN, May 2011.

[2] M. Shahzad and X. Zhu, “Robust reconstruction of building facades for large areas using spaceborne tomosar point clouds,” *IEEE Trans. Geoscience and Remote Sensing*, vol. 53, no. 2, pp. 752–769, 2015.

[3] W. Luo and H. Zhang, “Visual analysis of large-scale lidar point clouds,” in *IEEE International Conference on Big Data*, Santa Clara, CA, Nov. 2015, pp. 2487–2492.

[4] J. Ryde and H. Hu, “3D mapping with multi-resolution occupied voxel lists,” *Autonomous Robots*, pp. 169–185, Apr. 2010.

[5] C. T. Loop, C. Zhang, and Z. Zhang, “Real-time high-resolution sparse voxelization with application to image-based modeling,” in *Proceedings of the 5th High-Performance Graphics 2013 Conference HPG ’13*, Anaheim, California, USA, July 2013, pp. 73–80.

[6] J. Peng and C.-C. Jay Kuo, “Geometry-guided progressive lossless 3D mesh coding with octree (OT) decomposition,” *ACM Trans. Graph. Proceedings of ACM SIGGRAPH*, vol. 24, no. 3, pp. 609–616, Jul. 2005.

[7] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, “Octomap: An efficient probabilistic 3D mapping framework based on octrees,” *Autonomous Robots*, pp. 189–206, Apr. 2013.

[8] J. Peng, C-S. Kim, and C.-C. Jay Kuo, “Technologies for 3D mesh compression: A survey,” *J. Vis. Comun. Image Represent.*, vol. 16, no. 6, pp. 688–733, Dec. 2005.

[9] P. Alliez and C. Gotsman, “Recent advances in compression of 3D meshes,” in *In Advances in Multiresolution for Geometric Modelling*. 2003, pp. 3–26, Springer-Verlag.

[10] A. Maglo, G. Lavoué, F. Dupont, and C. Hudelot, “3D mesh compression: Survey, comparisons, and emerging trends,” *ACM Comput. Surv.*, vol. 47, no. 3, pp. 44:1–44:41, Feb. 2015.

[11] D. Thanou, P. A. Chou, and P. Frossard, “Graph-based compression of dynamic 3d point cloud sequences,” *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1765–1778, Feb. 2016.

[12] A. Anis, P. A. Chou, and A. Ortega, “Compression of dynamic 3d point clouds using subdivisional meshes and graph wavelet transforms,” in *ICASSP*, Shanghai, China, Mar. 2016, pp. 6360–6364.

[13] T. Hackel, J. D. Wegner, and K. Schindler, “Contour detection in unstructured 3d point clouds,” in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recogn.*, Las Vegas, Nevada, Jun. 2016.

[14] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz, “Accelerated generative models for 3D point cloud data,” in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recogn.*, Las Vegas, Nevada, Jun. 2016.

[15] A. Sandryhaila and J. M. F. Moura, “Discrete signal processing on graphs,” *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[16] 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, pp. 83–98, May 2013.

[17] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, “Discrete signal processing on graphs: Sampling theory,” *IEEE Trans. Signal Process.*, vol. 63, no. 24, pp. 6510–6523, Dec. 2015.

[18] A. Anis, A. Gadde, and A. Ortega, “Efficient sampling set selection for bandlimited graph signals using graph spectral proxies,” *IEEE Trans. Signal Process.*, vol. 64, pp. 3775–3789, July 2016.

[19] S. Chen, R. Varma, A. Singh, and J. Kovačević, “Signal recovery on graphs: Fundamental limits of sampling strategies,” *IEEE Trans, Signal and Inform. Process. over Networks*, 2016, Submitted.

[20] L. D. Floriani and P. Magillo, “Multiresolution mesh representation: Models and data structures,” in *Tutorials on Multiresolution in Geometric Modelling, Summer School Lecture Notes*, pp. 363–417. Springer, 2002.

[21] M. Vetterli and J. Kovačević, *Wavelets and Subband Coding*, Prentice Hall, Englewood Cliffs, NJ, 1995, http://waveletsandsubbandcoding.org/.

[22] J. Kovačević, V. K. Goyal, and M. Vetterli, *Fourier and Wavelet Signal Processing*, Cambridge University Press, Cambridge, 2016, http://www.fourierandwavelets.org/.

[23] Y. Loannou, B. Taati, R. Harrap, and M. A. Greenspan, “Difference of normals as a multi-scale operator in unorganized point clouds,” in *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission, Zurich, Switzerland, October 13-15, 2012*, 2012, pp. 501–508.