Fast Resampling of 3D Point Clouds via Graphs

Chen, S.; Tian, D.; Feng, C.; Vetro, A.; Kovacevic, J.

TR2017-215 November 09, 2017

Abstract

To reduce the cost of storing, processing and visualizing a large-scale point cloud, we propose a randomized resampling strategy that selects a representative subset of points while preserving application-dependent features. The strategy is based on graphs, which can represent underlying surfaces and lend themselves well to efficient computation. We use a general feature-extraction operator to represent application-dependent features and propose a general reconstruction error to evaluate the quality of resampling; by minimizing the error, we obtain a general form of optimal resampling distribution. The proposed resampling distribution is guaranteed to be shift-, rotation- and scale-invariant in the 3D space. We then specify the featureextraction operator to be a graph filter and study specific resampling strategies based on allpass, lowpass, highpass graph filtering and graph filter banks. We validate the proposed methods on three applications: large-scale visualization, accurate registration and robust shape modeling demonstrating the effectiveness and efficiency of the proposed resampling methods.

IEEE Transactions on Signal Processing

Fast Resampling of 3D Point Clouds via Graphs

Siheng Chen, Dong Tian, *Senior Member, IEEE*, Chen Feng, *Member, IEEE*, Anthony Vetro, *Fellow, IEEE*, Jelena Kovačević, *Fellow, IEEE*

***Abstract*—To reduce the cost of storing, processing and visualizing a large-scale point cloud, we propose a randomized resampling strategy that selects a representative subset of points while preserving application-dependent features. The strategy is based on graphs, which can represent underlying surfaces and lend themselves well to efficient computation. We use a general feature-extraction operator to represent application-dependent features and propose a general reconstruction error to evaluate the quality of resampling; by minimizing the error, we obtain a general form of optimal resampling distribution. The proposed resampling distribution is guaranteed to be shift-, rotation- and scale-invariant in the 3D space. We then specify the feature-extraction operator to be a graph filter and study specific resampling strategies based on allpass, lowpass, highpass graph filtering and graph filter banks. We validate the proposed methods on three applications: large-scale visualization, accurate registration and robust shape modeling demonstrating the effectiveness and efficiency of the proposed resampling methods.**

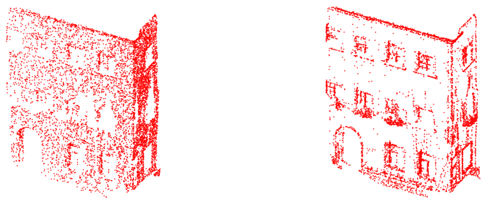
***Index Terms*—3D point clouds, sampling,graph signal processing, graph filtering, contour detection, visualization, registration, shape modeling**

I. Introduction

With the advent of 3D sensing technologies, one can now represent objects and surrounding environments by 3D point clouds—collections of large numbers of 3D points on an object's surface measured by a sensing device. These 3D point clouds have become popular representation tools in applications such as virtual reality, mobile mapping, scanning of historical artifacts, 3D printing and digital elevation modeling [2], among others.

A 3D point cloud is described by its 3D coordinates as well as attributes such as color, temperature and texture. Based on the storage order and spatial connectivity among 3D points, we distinguish between two types of point clouds: *organized*, such as those collected by a camera-like 3D sensors or 3D laser scanners and arranged on a grid, and *unorganized*, such as those that, due to their complex structure, are scanned from multiple view points and are subsequently merged leading to the loss of ordering of indices. Organized point clouds are easier to process as the underlying grids produce a natural spatial connectivity and reflect the sensing order. For generality, we here consider unorganized point clouds.

3D point cloud processing has become an important component in many 3D imaging and vision systems. It broadly includes compression [3], [4], [5], [6], [7], visualization [8], [9], surface reconstruction [10], [11], rendering [12], [13], editing [14], [15] and feature extraction [16], [17], [18], [19], [20]. A particular challenge is how to handle a large number of incoming 3D points [21], [22]; for example, real-time sensing systems generate millions of data points per second while in digital documentation of historical buildings and terrain visualization, we need to store billions of 3D points, making storage and subsequent processing challenging.



(a) Uniform resampling. (b) Contour-enhanced resampling.

Fig. 1: Proposed resampling framework enhances important information based on user's preferences, for example, enhancing contours of a 3D point cloud. Plots (a) and (b) show clouds with 2% resampled out of 381,903 total points from a 3D point cloud of a building using two different strategies. Because of contour enhancement, Plot (b) is more visually pleasing and caries more information than Plot (a).

An approach to solving the problem is to consider efficient data structures to represent 3D point clouds. For example, [23], [24] partition the 3D space into voxels and then discretize point clouds over those voxels; to achieve fine resolution, however, a dense grid is required causing storage challenges. [25], [26] use an octree representation of point clouds, which is space efficient but suffers from discretization errors, while [27], [28] propose a probabilistic generative model to model the distribution of point clouds; such parametric models may not capture the true surface, however, and may be inefficient in terms of inferring model parameters [29], [30].

Another approach is to consider reducing the number of points through mesh simplification by constructing a triangular or polygonal mesh where nodes are 3D points (not necessarily input points) and edges are connectivities between those points respecting certain restrictions (belonging to a manifold, for exampe) [31], [32], [33]. The mesh is then simplified by reducing the number of nodes or edges; for example, several nodes may be merged into one with local structure preserved. Mesh construction requires costly computation, however, and mesh simplification displaces original points, causing distortion.

We here *resample* 3D point clouds, keeping a subset of points from the original object and reducing the number of

S. Chen is with Uber Advanced Technologies Group, Pittsburgh, PA, USA. Email: sihengc@uber.com. D. Tian, C. Feng and A. Vetro are with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA USA. Emails: tian,cfeng,avetro@merl.com. J. Kovačević is with the Departments of Electrical and Computer Engineering and Biomedical Engineering, Carnegie Mellon University, Pittsburgh, PA, USA. Email: jelenak@cmu.edu.

The work has been done while Siheng Chen was an intern at MERL during the summer of 2016. Parts of this paper appears in [1].

3D points without changing the locations of the original ones in the process. We then design application-dependent resampling strategies to preserve application-crucial information. For example, conventional contour detection in 3D point clouds requires careful and costly computation to obtain surface normals and classification models [34], [28]. Instead, we efficiently resample a small subset of points that carries the required contour information, making the subsequent processing cheaper without losing accuracy; see Figure 1 for an example. Note that after resampling, we unavoidably lose information from the original 3D point cloud.

The proposed method is rooted in graph signal processing, a framework to explore the interaction between signals and graph structure [35], [36]. We use a graph to capture local dependencies among points, representing a discrete version of the surface of an original object. The advantage of using a graph is that one is able to capture both local and global structure of point clouds. Each of the 3D coordinates and other attributes associated with 3D points constitute a graph signal indexed by the nodes of the underlying graph. We thus formulate a resampling problem as graph signal sampling. As graph sampling methods usually select samples in a deterministic fashion necessitating solving nonconvex optimization problems to obtain samples sequentially and requiring costly computation [37], [38], [39], [40], we aim to reduce computational complexity. We thus propose an efficient randomized resampling strategy to select a subset of points; we choose subsamples according to a nonuniform resampling distribution, which provably preserves application-dependent information in the original 3D point cloud and is computational efficient (since the computational complexity scales linearly with the number of input 3D points).

• We start by proposing a feature-extraction based resampling framework. We use a general feature-extraction operator to represent application-dependent information, based on which we quantify the quality of resampling by using a simple, yet general reconstruction error, where we can derive the mean-squared error (MSE). We obtain the resampling distribution by optimizing that MSE; this distribution is guaranteed to be shift-/rotation-/scale-invariant.

• Next, we specify the feature-extraction operator to be a graph filter and study optimal resampling distributions based on allpass, lowpass and highpass graph filtering. In each case, we derive an optimal resampling distribution and validate the performance on both simulated and real data. The total computational cost to obtain the optimal resampling distribution scales linearly with the number of 3D points. We further combine all the proposed techniques into an efficient surface reconstruction system based on graph filter banks, which enables us to enhance features in a 3D point cloud.

• Finally, we validate these methods on three applications: (1) For large-scale visualization, we use the highpass graph filtering based resampling strategy to highlight the contours of buildings and streets in an urban scene, which avoids saturation problems in visualization. (2) For accurate registration, we use the highpass graph filtering based resampling strategy to extract the key points of a sofa, which makes the registration precise. (3) For robust shape modeling, we use the lowpass graph filtering based resampling strategy to reconstruct a surface, which makes the reconstruction robust to noise. The performances in these three applications validate the effectiveness and efficiency of the proposed resampling methods.

Contributions. In many large-scale 3D point cloud processing tasks including commercial software packages, resampling point clouds uniformly is widely used as a preprocessing step; despite this, this step is often approached heuristically. We instead consider 3D point cloud resampling from a *novel theoretical perspective*. For example, Theorem 4 shows that uniform resampling is a suboptimal resampling distribution when all 3D points are associated with the same feature value.

The main contributions of the paper are thus:

- Section III: We propose a *novel framework for 3D point cloud resampling*. We obtain the exact MSE and optimal resampling distribution for invariant features (Theorems 1, 2) and an upper bound on the MSE and corresponding resampling distribution for variant features (Theorems 3, 4).
- Section IV: We propose a *novel feature-extraction operator for 3D point clouds* based on graph filtering. We use Haar-like low/highpass graph filters, which is both effective and efficient (scales linearly with the number of 3D points).
- Section V: We *validate* the proposed resampling strategies on both simulated data and real point clouds, including accurate registration, large-scale visualization and robust shape modeling.

We discuss a number of possible future directions in Section VI, such as efficient 3D point cloud compression system based on graph filter banks, surface reconstruction based on arbitrary graphs and robust metric to evaluate the visualization quality of a 3D point cloud.

II. Problem Formulation and Background

We now cover the background material necessary for the rest of the paper. We start by formulating the task of resampling a 3D point cloud. We then introduce graph signal processing, which lays the foundation for our proposed methods.

A. Resampling a Point Cloud

Consider a matrix representation of a point cloud with N points and K attributes,

$$\mathrm{X} = \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \dots & \mathbf{s}_K \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix} \in \mathbb{R}^{N \times K}, \qquad (1)$$

where $\mathbf{s}_i \in \mathbb{R}^N$ denotes the ith attribute and $\mathbf{x}_j \in \mathbb{R}^K$ denotes the jth point; depending on the sensing device, attributes can be 3D coordinates, RGB colors, textures, and many others. To distinguish 3D coordinates from other attributes, we store them in the first three columns of X and call that submatrix $\mathrm{X_c} \in \mathbb{R}^{N\times 3}$ while storing the rest of the attributes in the last $K-3$ columns of X and call that submatrix $\mathrm{X_o} \in \mathbb{R}^{N\times (K-3)}$.

Our task is resampling the original 3D point cloud while peserving application-crucial information by selecting M ($M <$

N) original points (M rows from the point cloud matrix X). Let the resampled point cloud be $\mathrm{X}_{\mathcal{M}} = \Psi \mathrm{X} \in \mathbb{R}^{M \times K}$, where $\mathcal{M} = (\mathcal{M}_1, \ldots, \mathcal{M}_M)$ denotes the sequence of resampled indices, called *resampled set*, $\mathcal{M}_i \in \{1, \ldots, N\}$ with $|\mathcal{M}| = M$ and the resampling operator Ψ is a linear mapping from $\mathbb{R}^N$ to $\mathbb{R}^M$, defined as

$$\Psi_{i,j} = \begin{cases} 1, & j = \mathcal{M}_i; \\ 0, & \text{otherwise}. \end{cases} \tag{2}$$

As the efficiency of the proposed resampling strategy is critical, we consider a randomized resampling strategy by choosing resampled indices from a resampling distribution. Let $\pi \in \mathbb{R}^N$ be a resampling distribution, where π_i denotes the probability of selecting the ith sample in each random trial. Our goal is to find a resampling distribution that preserves information in the original point cloud.

The invariance properties of the proposed resampling strategy are also critical. When we shift, rotate or scale a point cloud, the intrinsic distribution of 3D points does not change and neither should the proposed resampling strategy.

Definition 1. A resampling strategy is called *shift-invariant* when, for any shift vector $\mathbf{a} \in \mathbb{R}^3$, input 3D point cloud $\mathrm{X} = \begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix}$ and corresponding resampling distribution $\pi \in \mathbb{R}^N$, the corresponding resampling distribution to $\begin{bmatrix} \mathrm{X_c} + \mathbf{1}\mathbf{a}^T & \mathrm{X_o} \end{bmatrix}$ is π.

Definition 2. A resampling strategy is called *rotation-invariant* when, for any rotation matrix $\mathrm{R} \in \mathbb{R}^{3 \times 3}$, input 3D point cloud $\mathrm{X} = \begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix}$ and corresponding resampling distribution $\pi \in \mathbb{R}^N$, the corresponding resampling distribution to $\begin{bmatrix} \mathrm{X_c}\,\mathrm{R} & \mathrm{X_o} \end{bmatrix}$ is π.

Definition 3. A resampling strategy is called *scale-invariant* when, for any scaling factor $c \in \mathbb{R}$, input 3D point cloud $\mathrm{X} = \begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix}$ and corresponding resampling distribution $\pi \in \mathbb{R}^N$, the corresponding resampling distribution to $\begin{bmatrix} c\,\mathrm{X_c} & \mathrm{X_o} \end{bmatrix}$ is π.

Our aim is to guarantee that the proposed resampling strategy is shift-, rotation- and scale-invariant.

B. Graph Signal Processing for Point Clouds

A graph is a natural and efficient way of representing a 3D point cloud as a discretized version of an original surface. In computer graphics, a class of graphs with particular connectivity restrictions called polygon meshes are extensively used to represent the shape of an object [41]; mesh construction usually requires sophisticated geometry analysis, however, such as calculating surface normals, and the mesh representation may not be suitable for point clouds because of connectivity restrictions. We here extend the ideas behind polygon meshes to general graphs by relaxing those connectivity restrictions, gaining flexibility to capture geometry information and simplifying construction in the process.

Graph signal processing (GSP) is a theoretical framework that extends classical signal processing concepts to the analysis of high-dimensional data with complex, irregular structure [36], [42]. In particular, GSP models such data by graphs and graph signals and generalizes the concepts and tools, such as filtering and Fourier transform, from classical discrete signal processing to graph signal processing. In our work, we model all pairwise proximities between 3D points as a graph and each of 3D coordinates and attributes as a graph signal, allowing us to leverage tools from graph signal processing to deal with 3D point clouds.

Graph Construction. The graph consists of a set of N nodes, our input 3D points, and a set of edges among them that we construct by encoding the local geometry information through an adjacency matrix $\mathrm{W} \in \mathbb{R}^{N \times N}$. Let $\mathbf{x}_i^{(\mathrm{c})} \in \mathbb{R}^3$ be the 3D coordinates of the ith point (the ith row of $\mathrm{X_c}$). We assign the edge weight between two points $\mathbf{x}_i^{(\mathrm{c})}$ and $\mathbf{x}_j^{(\mathrm{c})}$ as

$$\mathrm{W}_{i,j} = \begin{cases} e^{-\frac{\left\| \mathbf{x}_i^{(\mathrm{c})} - \mathbf{x}_j^{(\mathrm{c})} \right\|_2^2}{\sigma^2}}, & \left\| \mathbf{x}_i^{(\mathrm{c})} - \mathbf{x}_j^{(\mathrm{c})} \right\|_2 \leq \tau; \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

where variance σ and threshold τ are parameters. Equation (3) shows that when the Euclidean distance of two points is smaller than a threshold τ, we connect these two points by an edge with the edge weight depending on the distance between the two points in the 3D space (the closer they are, the higher the weight). The threshold τ controls the sparsity of the graph. When τ is small, only a few connections exist, which may not carry enough geometric information; when τ is large, a large number of connections esist, which may not capture local geometric information and makes it hard to process the entire graph. Since the intrinsic resolution of a point cloud can be either given as per the spacing of a 3D sensor or estimated directly from the point cloud, we choose τ as three times that intrinsic resolution. The advantage of this construction is that we need to compute the weights only when points are within a certain range. This can be efficiently implemented by using Octree [25], [43], [26], an efficient, hierarchical data structure to represent 3D point clouds. While we here only use the 3D coordinates to construct a graph, one could also take other attributes into account.

Given this graph, the attributes of point clouds $\mathbf{s}_i$ (the columns of X) are called *graph signals*.

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 nontrivial graph filter. Some common choices for a graph shift operator include the adjacency matrix W from (3), the transition matrix $\mathrm{D}^{-1}\,\mathrm{W}$ (D is the weighted degree matrix, a diagonal matrix with $\mathrm{D}_{i,i} = \sum_j \mathrm{W}_{i,j}$ reflecting the density around the ith point), the graph Laplacian matrix $\mathrm{D} - \mathrm{W}$, and many other structure-related matrices. 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 (an attribute of a point cloud). Every linear, shift-invariant graph filter is a polynomial in the graph shift [35],

$$h(\mathrm{A}) = \sum_{\ell=0}^{L-1} h_\ell\, \mathrm{A}^\ell = h_0\, \mathrm{I} + h_1\, \mathrm{A} + \ldots + h_{L-1}\, \mathrm{A}^{L-1}, \tag{4}$$

where h_ℓ, $\ell = 0, 1, \ldots, L-1$ are filter coefficients and L is the graph filter length. Its output is given by the matrix-vector product $\mathbf{y} = h(\mathrm{A})\mathbf{s}$. For more details on graph filtering, see [44], [45].

Graph Fourier Transform. The eigendecomposition of a graph shift operator A is [46]

$$\mathrm{A} \ = \ \mathrm{V} \Lambda \mathrm{V}^{-1}, \tag{5}$$

where the eigenvectors of A form the columns of matrix V, and the eigenvalue matrix $\Lambda \in \mathbb{R}^{N \times N}$ is the diagonal matrix of corresponding eigenvalues $\lambda_1, \ldots, \lambda_N$ of A. These eigenvalues are called *frequencies* on the graph [46]. By convention, for $\mathrm{A} = \mathrm{W}$ or $\mathrm{D}^{-1}\mathrm{W}$, the eigenvalues are arranged in descending order; for $\mathrm{A} = \mathrm{D} - \mathrm{W}$, the eigenvalues are arranged in ascending order. In either case, we here ensure that λ_1 is the lowest frequency and λ_N is the highest frequency. In other words, $\mathbf{v}_1$ captures the smallest variations in the graph and $\mathbf{v}_N$ captures the highest variations in the graph.

In the above, V is called *graph Fourier basis* and the *graph Fourier transform* of a graph signal $\mathbf{s} \in \mathbb{R}^N$ is defined as

$$\widehat{\mathbf{s}} \ = \ \mathrm{V}^{-1} \mathbf{s}. \tag{6}$$

The *inverse graph Fourier transform* is $\mathbf{s} = \mathrm{V}\widehat{\mathbf{s}} = \sum_{k=1}^{N} \widehat{s}_k \mathbf{v}_k$, where $\mathbf{v}_k$ is the kth column of V and $\widehat{s}_k$ is the kth component in $\widehat{\mathbf{s}}$. The vector $\widehat{\mathbf{s}}$ in (6) represents the signal's expansion in the eigenvector basis and describes the frequency components of the graph signal $\mathbf{s}$. The inverse graph Fourier transform reconstructs the graph signal by combining graph frequency components. For more details on the graph Fourier transform, see [47], [48], [49].

Graph-based Sampling. The interest in sampling of graph signals has increased in the last few years. Sampling theory for graph signals in [50], [51], [37], [52] showed perfect recovery for bandlimited graph signals based on a few subsamples; the sampling and recovery framework in [53] is based on local-set-based sampling; [54] proposed local-aggregated sampling; [55] proposed percolation from selected seeding nodes; [56] studied greedy sampling of bandlimited graph signals; [57] explored the connection between sampling and sketching to handle streaming data. These sampling methods select samples in a greedy fashion and thus, the complexity does not scale linearly with the number of nodes making it impractical for handling large-scale graphs. [58], [59] proposed randomized sampling strategies for approximately bandlimited graph signals; [60] considered fast spectral clustering based on random sampling of bandlimited graph signals; [61] showed that active sampling does not asymptotically outperform experimentally designed sampling; however, all these results are rooted in the assumption that graph signals are smooth.

Instead, here we intend to sample 3D points according to the salient features, which are not necessarily smooth over the constructed graph. Further, in terms of geometric properties of 3D point clouds, we want to ensure that the proposed resampling strategy is shift-/rotation-/scale-invariant.

III. Resampling Based on Feature Extraction

During resampling, the loss of information is unavoidable; our goal is thus to design an application-dependent resampling strategy that preserves application-crucial information. For example, to efficiently detect the contour in a large-scale 3D point cloud, we want to reduce the number of points while preserving the contour information. We thus select more points along the contour and fewer points outside of the contour; see Figure 1. We consider such application-dependent information as features and call this process *feature-based resampling*. To preserve as many features as possible, we formulate an optimization problem for designing a resampling operator. To adapt to the geometric nature of 3D point clouds, we also guarantee that the proposed resampling strategy is shift-/rotation-/scale-invariant no matter which features are given.

Let $f(\mathrm{X}) \in \mathbb{R}^{N \times K}$ be a feature-extraction operator that extracts application-crucial information from a point cloud[1] $\mathrm{X} \in \mathbb{R}^{N \times K}$. Depending on the application, those features can be edges, key points and flatness [17], [18], [19], [62], [20]. In this section, we consider feature-extraction operators in an abstract level and use graph filters to implement specific feature-extraction operation the next section.

We consider both invariant and variant features.

Definition 4. A feature-extraction operator $f(\cdot)$ is called *shift-invariant* when, for any shift vector $\mathbf{a} \in \mathbb{R}^3$, input 3D point cloud $\mathrm{X} = \begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix}$ and corresponding feature $f(\begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix})$, the corresponding feature to $\begin{bmatrix} \mathrm{X_c} + \mathbf{1}\mathbf{a}^T & \mathrm{X_o} \end{bmatrix}$ is $f(\begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix})$

Definition 5. A feature-extraction operator $f(\cdot)$ is called *rotation-invariant* when, for any 3D rotation matrix $\mathrm{R} \in \mathbb{R}^{3 \times 3}$, input 3D point cloud $\mathrm{X} = \begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix}$ and corresponding feature $f(\begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix})$, the corresponding feature to $\begin{bmatrix} \mathrm{X_c}\mathrm{R} & \mathrm{X_o} \end{bmatrix}$ is $f(\begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix})$

Definition 6. A feature-extraction operator $f(\cdot)$ is called *scale-invariant* when, for any scaling factor $c \in \mathbb{R}$, input 3D point cloud $\mathrm{X} = \begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix}$ and corresponding feature $f(\begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix})$, the corresponding feature to $\begin{bmatrix} c\,\mathrm{X_c} & \mathrm{X_o} \end{bmatrix}$ is $f(\begin{bmatrix} \mathrm{X_c} & \mathrm{X_o} \end{bmatrix})$

In what follows, we discuss resampling strategies based on each type of features.

A. Resampling Based on Invariant Features

To evaluate the performance of a resampling operator, we quantify how many features are lost during resampling; that is, we sample features, interpolate to get back the original features and compute the recovery error. The features are considered to reflect the targeted information contained in each 3D point.

We resample a point cloud M times. At 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 from (2) and $\mathrm{S} \in \mathbb{R}^{N \times N}$ be a diagonal rescaling matrix with $\mathrm{S}_{i,i} = 1/(M\pi_i)$. We quantify the performance of a resampling operator through

$$D_{f(\mathrm{X})}(\Psi) \ = \ \left\| \mathrm{S}\, \Psi^T \Psi f(\mathrm{X}) - f(\mathrm{X}) \right\|_F^2, \tag{7}$$

where $\left\|\cdot\right\|_F$ is the Frobenius norm. $\Psi^T\Psi \in \mathbb{R}^{N \times N}$ is a zero-padding operator, a diagonal matrix with diagonal elements $(\Psi^T\Psi)_{i,i} > 0$ when the ith point is sampled and 0, otherwise, which ensures that the resampled and the original point clouds are of the same size. S is used to compensate for nonuniform

[1]While for simplicity, we consider the number of features to be the same as the number of attributes, this is not a limitation of the method.

weights during resampling. $\mathrm{S}\,\Psi^T$ is the 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. We consider the interpolation operator $\mathrm{S}\,\Psi^T$ for two reasons. (1) The formulation is simple and leads to a closed-form solution that will be shown in Theorems 2 and 4. (2) To ensure the robustness of the resampling operator, we want to minimize the worst-case interpolation error; this interpolation operator provides a reasonable upper bound on the interpolation error.

Note that in practice, practical interpolation operators may consider reconstructing the underlying surface, instead of original 3D points (that may not be known exactly). Because 3D points may not live on a grid, it would be hard to interpolate all points from the resampled ones without knowing the graph; however, storing the graph could be even more expensive than storing all 3D points. This is why, in practice, we prefer to reconstruct the underlying surface from resampled points and compare it with the surface reconstructed by using all 3D points. We leave this surface setting to future work.

Lemma 1 shows that S aids in providing an unbiased estimator; the proof is given in Appendix A.

Lemma 1. Let $f(\mathrm{X}) \in \mathbb{R}^{N\times K}$ be features extracted from a point cloud X. Then,

$$\begin{aligned}\mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi f(\mathrm{X})\right) &\propto \pi \odot f(\mathrm{X}),\\ \mathbb{E}_{\Psi\sim\pi}\left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right) &= f(\mathrm{X}),\end{aligned}$$

with $\mathbb{E}_{\Psi\sim\pi}$ the expectation over samples, which are generated from a distribution π independently and randomly and $\odot$ row-wise multiplication.

The evaluation metric $D_{f(\mathrm{X})}(\Psi)$ measures the interpolation error, that is, how much feature information is lost during resampling, while $\mathbb{E}_{\Psi\sim\pi}\left(D_{f(\mathrm{X})}(\Psi)\right)$ is the expected error caused by resampling and quantifies the performance of a resampling distribution π.

Here $f(\cdot)$ is shift-/rotation-/scale-invariant, (7) does not change through shifting, rotation or scaling, leading to a shift-/rotation-/scale-invariant resampling strategy. Our goal is thus to minimize the MSE of the objective function (7), $\mathbb{E}_{\Psi\sim\pi}\left(D_{f(\mathrm{X})}(\Psi)\right)$ over π to obtain an optimal resampling distribution.

We now derive that MSE; the proof is given in Appendix B.

Theorem 1. Let $f(\cdot)$ be a rotation-invariant feature-extraction operator. The MSE of the objective function (7) is

$$\mathbb{E}_{\Psi\sim\pi} D_{f(\mathrm{X})}(\Psi) \;=\; \mathrm{Tr}\left(f(\mathrm{X})\,\mathrm{Q}\,f(\mathrm{X})^T\right), \tag{8}$$

where $\mathrm{Q} \in \mathbb{R}^{N\times N}$ is a diagonal matrix with the ith element $\mathrm{Q}_{i,i} = (1-\pi)/(M\pi)$.

We now derive the optimal resampling distributions by minimizing the MSE in (8); the proof is given in Appendix D.

Theorem 2. Let $f(\cdot)$ be a rotation-invariant feature-extraction operator. The resampling distribution π^* minimizing (8) is,

$$\pi_i^* \;\propto\; \|f_i(\mathrm{X})\|_2\,, \tag{9}$$

where $f_i(\mathrm{X}) \in \mathbb{R}^K$ is the ith row of $f(\mathrm{X})$.

The optimal resampling distribution is proportional to the magnitude of the features; that is, points with higher magnitudes have higher probability to be selected, while points with smaller magnitudes have smaller probability to be selected. The intuition is that the response after appying the feature-exaction operator reflects the information contained in each 3D point and determines the resampling probability for each.

B. Resampling Based on Variant Features

When $f(\cdot)$ is shift-/rotation-/scale-variant, (7) may change through shifting, rotation or scaling, leading to a shift-/rotation-/scale-variant resampling strategy.

Before processing, we first recenter the point cloud to the origin to handle shift variance; that is, we normalize the mean of the 3D coordinates to zeros. We then normalize the magnitude of the 3D coordinates to handle scale variance; that is, we normalize the spectral norm $\|\mathrm{X}_c\|_2 = c$ with constant $c > 0$. From now, we will perform this normalization to guarantee the shift/scale invariance of any 3D point cloud. The choice of c depends on users' preference and we will show that c is a trade-off between 3D coordinates and the values of other attributes. Note that those normalization procedures may not perfectly remove the influence of shift and scale-variance due to noise; we thus still prefer to work with invariant features.

To handle rotation variance, a straightforward approach is to align principal directions of point clouds; this procedure cannot, however, handle ball-shaped point clouda. We here consider a rotation-invariant evaluation metric,

$$\begin{aligned} D_f(\Psi) &= \max_{\mathrm{X}'_c : \|\mathrm{X}'_c\|_2 = c} D_{f([\mathrm{X}'_c\;\mathrm{X}_o])}(\Psi) \qquad (10)\\ &= \max_{\mathrm{X}'_c : \|\mathrm{X}'_c\|_2 = c} \left\| \left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I}\right) f\left(\left[\mathrm{X}'_c \quad \mathrm{X}_o\right]\right)\right\|_F^2, \end{aligned}$$

where constant $c = \|\mathrm{X}_c\|_2$ is the normalized spectral norm of 3D coordinates.

Unlike the evaluation metric $D_{f(\mathrm{X})}(\Psi)$ from (7), the one $D_f(\Psi)$ from (10) removes the influence of rotation by considering the worst possible reconstruction error caused by rotation. In (10), we consider 3D coordinates as variables due to rotation. We constrain their spectral norm because it does not change during rotation (a rotation matrix is orthonormal). We then minimize $\mathbb{E}_{\Psi\sim\pi}\left(D_f(\Psi)\right)$ to obtain a resampling strategy that is rotation-invariant even when $f(\cdot)$ is not.

For simplicity, we show the derivation for linear feature-extraction operators only, those of the form of $f(\mathrm{X}) = \mathrm{F}\,\mathrm{X}$, where X is a 3D point cloud and $\mathrm{F} \in \mathbb{R}^{N\times N}$ is a feature-extraction matrix; the proof is given in Appendix C.

Theorem 3. Let $f(\mathrm{X}) = \mathrm{F}\,\mathrm{X}$ be a linear, rotation-variant feature-extraction operator, with $\mathrm{F} \in \mathbb{R}^{N\times N}$. The MSE of the objective function (10) is

$$\begin{aligned} &\mathbb{E}_{\Psi\sim\pi}\left(D_f(\Psi)\right)\\ =\;& c^2\mathbb{E}_{\Psi\sim\pi}\left(\sum_{k=1}^{3}\sigma_k^2\left(\left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I}\right)\mathrm{F}\right)\right)\\ &+\mathrm{Tr}\left(\mathrm{F}\,\mathrm{X}_o\,\mathrm{Q}(\mathrm{F}\,\mathrm{X}_o)^T\right) \qquad (11)\\ \le\;& c^2\mathrm{Tr}\left(\mathrm{F}\,\mathrm{Q}\,\mathrm{F}^T\right) + \mathrm{Tr}\left(\mathrm{F}\,\mathrm{X}_o\,\mathrm{Q}(\mathrm{F}\,\mathrm{X}_o)^T\right), \qquad (12) \end{aligned}$$

where $\sigma_k\left(\left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I}\right)\mathrm{F}\right)$ is the kth singular value of $\left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I}\right)\mathrm{F}$ in a descending order and $\mathrm{Q} \in \mathbb{R}^{N\times N}$ is a diagonal matrix with the ith element $\mathrm{Q}_{i,i} = (1-\pi)/(M\pi)$.

Given a rotation-variant linear feature-extraction operator, it is computational expensive to minimize the exact form (11); instead, we minimize its upper bound (12); the proof is given in Appendix E.

Theorem 4. Let $f(\mathrm{X}) = \mathrm{F}\,\mathrm{X}$ be a linear, rotation-variant feature-extraction operator, with $\mathrm{F} \in \mathbb{R}^{N\times N}$. The resampling strategy π^* minimizing (12) is,

$$\pi_i^* \;\propto\; \sqrt{c^2\left\|\mathrm{F}_i\right\|_2^2 + \left\|(\mathrm{F}\,\mathrm{X}_{\rm o})_i\right\|_2^2}, \tag{13}$$

where constant $c = \left\|\mathrm{X}_{\rm c}\right\|_2$, F_i is the ith row of F and $(\mathrm{F}\,\mathrm{X}_{\rm o})_i$ is the ith row of $\mathrm{F}\,\mathrm{X}_{\rm o}$.

We see that the proposed resampling distribution is also proportional to the magnitude of features. The tuning parameter c in (13) is the normalized spectral norm used to remove the scale variance. The choice of c trades off the contribution from 3D coordinates from the other attributes.

Theorems 2 and 4 show that regardless of whether the feature-extraction operator is rotation-invariant or not, the proposed resampling operator is designed to be rotation-invariant.

C. Computational Complexity

To obtain the rotation-invariant feature-based resampling strategy (9), we compute the row norms of $f(\mathrm{X})$; thus, the computational complexity is $O(NK)$, which scales linearly with the number of 3D points. To obtain the rotation-variant feature-based resampling strategy (13), we compute the row norms of F and $\mathrm{F}\,\mathrm{X}_{\rm o}$; thus, the computational complexity is $O\left(\left\|\mathrm{vec}\left(\mathrm{F}\right)\right\|_0 + N)K\right)$, with $\left\|\mathrm{vec}\left(\mathrm{F}\right)\right\|_0$ the number of nonzero elements in F. In Section IV, we show that the proposed feature-extraction matrix F is sparse, ensuring that the computational complexity still scales linearly with the number of 3D points.

IV. Resampling Based on Graph Filtering

The previous section studied resampling based on an arbitrary feature-extraction operator. In this section, we extract features from a point cloud by using specific graph filters (4), sparse matrices that ensure low computational complexity.

Let features extracted from a point cloud X be

$$f(\mathrm{X}) \;=\; h(\mathrm{A})\,\mathrm{X} \;=\; \sum_{\ell=0}^{L-1} h_\ell\,\mathrm{A}^\ell\,\mathrm{X}\,.$$

A graph filter is a linear operator; thus, the corresponding resampling distribution follows from Theorems 2 and 4 by setting $\mathrm{F} = \sum_{\ell=0}^{L-1} h_\ell\,\mathrm{A}^\ell$. All graph filtering-based feature-extraction operators are scale-variant due to linearity, and thus, as discussed earlier, we normalize the spectral norm of the 3D coordinates. We will see that by carefully choosing the graph shift A and filter coefficients h_is, a graph filtering-based feature-extraction operator may be shift or rotation invariant.

Similarly to filter design in classical signal processing, we design a graph filter either in the graph vertex domain or in the graph spectral domain. In the graph vertex domain, for each point, a graph filter outputs a linear combination of the attributes of its neighboring points. For example, the output of the ith point, $f_i(\mathrm{X}) = \sum_{\ell=0}^{L-1} h_\ell\left(\mathrm{A}^\ell\,\mathrm{X}\right)_i$ is a weighted average of the attributes of points that are within L hops away from the ith point. The ℓth graph filter coefficient, h_ℓ, quantifies the contribution from the ℓth-hop neighbors. We design the filter coefficients to change the weights in local averaging.

In the graph spectral domain, we first design a desired spectral distribution and then use graph filter coefficients to fit this distribution. For example, an L-length graph filter is

$$\begin{aligned} h(\mathrm{A}) \;&=\; \mathrm{V}\,h(\Lambda)\,\mathrm{V}^{-1} \\ &=\; \mathrm{V}\begin{bmatrix} \sum_{\ell=0}^{L-1} h_\ell\lambda_1^\ell & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \sum_{\ell=0}^{L-1} h_\ell\lambda_N^\ell \end{bmatrix}\mathrm{V}^{-1}, \end{aligned}$$

where V is the graph Fourier basis and λ_i are graph frequencies (5). When we want the response of the ith graph frequency to be c_i, we set

$$h(\lambda_i) \;=\; \sum_{\ell=0}^{L-1} h_\ell\lambda_i^\ell \;=\; c_i,$$

and solve a set of linear equations to obtain the graph filter coefficients h_ℓ. It is also possible to use the Chebyshev polynomial to design graph filter coefficients [63]. We now consider some special cases of graph filters.

A. Allpass Graph Filtering

Let $h(\lambda_i) = 1$ for all i; that is, $h(\mathrm{A}) = \mathrm{I}$ is an identity matrix with $h_0 = 1$ and $h_i = 0$ for $i = 1, \ldots, L-1$. This is the setting where the original point cloud is trustworthy and all points are uniformly sampled from an object without noise, reflecting the true geometric structure of the object. We want to preserve all the information and thus, the features are the original attributes themselves. Since $f(\mathrm{X}) = \mathrm{X}$, the feature-extraction operator $f(\cdot)$ is rotation-variant. Based on Theorem 4, the corresponding resampling strategy is

$$\pi_i^* \;\propto\; \sqrt{c^2 + \left\|(\mathrm{X}_{\rm o})_i\right\|_2^2}. \tag{14}$$

Here the feature-extraction matrix F in (9) is an identity matrix and the norm of each row of F is 1. When we only preserve 3D coordinates and ignore other attributes, $\mathrm{X}_{\rm o}$, we obtain a constant resampling probability for each point, meaning that uniform resampling is the resampling strategy that preserves the overall geometry information. We compute the row norms of $\mathrm{X}_{\rm o}$ to obtain the optimal resampling strategy in (14); thus, the computational complexity is $O(NK)$, which scales linearly with the number of 3D points.

B. Highpass Graph Filtering

In image processing, a highpass filter is used to extract edges and contours; similarly, we here use a highpass graph filter to extract contours in a point cloud. We only consider the 3D coordinates as attributes ($\mathrm{X} = \mathrm{X_c} = \mathbb{R}^{N \times 3}$), but the proposed method can be easily extended to other attributes as well.

A critical question is how to define a contour in a 3D point cloud; some authors propose sophisticated geometry-related computations, such as surface normals, to detect contours [34]. We instead use the response of highpass graph filtering to measure local variation on graphs to detect contour points; that local variation at the ith point is

$$f_i(\mathrm{X}) = \left\| (h(\mathrm{A})\,\mathrm{X})_i \right\|_2^2, \quad (15)$$

where $h(\mathrm{A})$ is a highpass graph filter. The local variation $f(\mathrm{X}) \in \mathbb{R}^N$ quantifies the energy of the response after highpass graph filtering; when the local variation at a point is high, its 3D coordinates cannot be well approximated from those of its neighboring points, and the point is thus likely to be a contour point.

The following theorem characterizes the invariance of the local variation; the proof is given in Appendix F.

Theorem 5. Let $f(\mathrm{X}) = \operatorname{diag}\left(h(\mathrm{A})\,\mathrm{X}\,\mathrm{X}^T\, h(\mathrm{A})^T\right) \in \mathbb{R}^N$, where $\operatorname{diag}(\cdot)$ extracts the diagonal elements. $f(\mathrm{X})$ is rotation invariant and shift invariant unless $h(\mathrm{A})\mathbf{1} = \mathbf{0} \in \mathbb{R}^N$.

To guarantee shift invariance of local variation without recentering the cloud, we can use a transition matrix as a graph shift operator; that is, $\mathrm{A} = \mathrm{D}^{-1}\,\mathrm{W}$, where D is the diagonal degree matrix. This is because $\mathbf{1} \in \mathbb{R}^N$ is the eigenvector of a transition matrix, $\mathrm{A}\,\mathbf{1} = \mathrm{D}^{-1}\,\mathrm{W}\,\mathbf{1} = \mathbf{1}$. Thus,

$$h(\mathrm{A})\mathbf{1} = \sum_{\ell=0}^{N-1} h_\ell\, \mathrm{A}^\ell\, \mathbf{1} = \sum_{\ell=0}^{N-1} h_\ell \mathbf{1} = \mathbf{0},$$

when $\sum_{\ell=0}^{N-1} h_\ell = 0$; an example is a Haar-like highpass filter

$$\begin{aligned} h_{\mathrm{HH}}(\mathrm{A}) &= \mathrm{I} - \mathrm{A} \quad (16) \\ &= \mathrm{V} \begin{bmatrix} 1-\lambda_1 & 0 & \cdots & 0 \\ 0 & 1-\lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1-\lambda_N \end{bmatrix} \mathrm{V}^{-1}. \end{aligned}$$

Note that $\lambda_{\max} = \max_i |\lambda_i| = 1$, where λ_i are eigenvalues of A, because the graph shift operator is a transition matrix. Then, $h_0 = 1$, $h_1 = -1$ and $h_i = 0$ for all $i > 1$, and $\sum_{\ell=0}^{N-1} h_\ell = 0$. Thus, a Haar-like highpass graph filter is both shift- and rotation-invariant. The graph frequency response of the filter is $h_{\mathrm{HH}}(\lambda_i) = 1 - \lambda_i$. Since the eigenvalues are arranged in descending order, $1 - \lambda_i \le 1 - \lambda_{i+1}$, meaning low frequencies are attenuated and high frequencies are amplified.

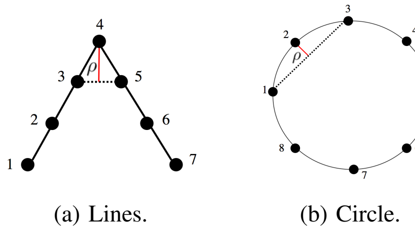


Fig. 2: Red line shows the local variation.

In the graph vertex domain, the response of the ith point is

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

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 convex combination of its neighbors. The geometric interpretation of the proposed local variation is the Euclidean distance between the original point and the convex combination of its neighbors, reflecting how much information we can glean about a point from its neighbors. When the local variation at a point is large, the Euclidean distance between this point and the convex combination of its neighbors is large and the point provides a large amount of variation.

We validate the proposed local variation on simple examples.

Example 1. When a point cloud forms a line in 3D, the two endpoints belong to the contour.

Example 2. When a point cloud forms a polygon/polyhedron in 3D, the vertices (corner points) and the edges (line segment connecting two adjacent vertices) belong to the contour.

Example 3. When a point cloud forms a circle/sphere in 3D, there is no contour.

When the points are uniformly spread along the defined shape, the proposed local variation (15) satisfies Examples 1, 2 and 3 from the geometric perspective. In Figure 2(a), Point 2 is the convex combination of Points 1 and 3, and the local variation of Point 2 is thus zero. However, Point 4 is not the convex combination of Points 3 and 5 and the length of the red line indicates the local variation of Point 4. Only Points 1, 4 and 7 have nonzero local variation, as expected. In Figure 2(b), all the nodes are evenly spread on a circle and have the same amount of variation, which is represented as a red line. Similar arguments show that the proposed local variation (15) satisfies Examples 1, 2 and 3.

The feature-extraction operator $f(\mathrm{X}) = \|h_{\mathrm{HH}}(\mathrm{A})\,\mathrm{X}\|_F^2$ is shift and rotation-invariant. Based on Theorem 2, the optimal resampling distribution is

$$\pi_i^* \;\propto\; \left\| (h_{\mathrm{HH}}(\mathrm{A})\,\mathrm{X})_i \right\|_2^2 \;=\; \left\| \mathbf{x}_i - \sum_{j \in \mathcal{N}_i} \mathrm{A}_{i,j}\,\mathbf{x}_j \right\|_2^2, \quad (17)$$

where $\mathrm{A} = \mathrm{D}^{-1}\,\mathrm{W}$ is a transition matrix. We also recognize that $h_{\mathrm{HH}}(\mathrm{A}) = \mathrm{I} - \mathrm{D}^{-1}\,\mathrm{W}$ is the random-walk graph Laplacian matrix [64].

Note that the standard graph Laplacian matrix is commonly used to measure variation. Let $\mathrm{L} = \mathrm{D} - \mathrm{W} \in \mathbb{R}^{N \times N}$ be a graph Laplacian matrix [64]. The graph Laplacian based total variation is

$$\operatorname{Tr}\left(\mathrm{X}^T\,\mathrm{L}\,\mathrm{X}\right) \;=\; \sum_i \sum_{j \in \mathcal{N}_i} \mathrm{W}_{i,j} \left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2, \quad (18)$$

where $\mathcal{N}_i$ denotes the neighbors of the ith node and the variation contributed by the ith point is

$$f_i(\mathrm{X}) \;=\; \sum_{j \in \mathcal{N}_i} \mathrm{W}_{i,j} \left\|\mathbf{x}_i - \mathbf{x}_j\right\|_2^2. \quad (19)$$

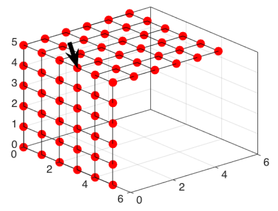
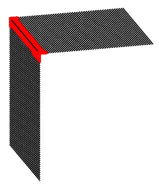


Fig. 3: The pairwise difference based local variation cannot capture the contour points connecting the two faces.

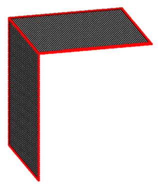
The variation here is defined based on the accumulation of pairwise differences. We call (19) pairwise difference based local variation; it cannot, however, capture geometry change, for example, it violates Example 2. Figure 3 shows another example. The points are uniformly spread along the faces of a cube with each point connecting to its adjacent four points with the same edge weight. The pairwise difference based local variations of all points are the same, implying the absence of a contour; clearly, however, this is not the case (see, for example, the point to which the black arrow points in the figure).

Computational Complexity. To obtain the resampling strategy in (17), we compute the row norms of $h_{\rm HH}({\rm A})\,{\rm X}$; the total computational complexity is thus $O(K\left\|{\rm A}\right\|_0)$, where $\left\|{\rm A}\right\|_0$ is the number of nonzero elements in the graph shift ${\rm A}$. Because we construct a sparse graph in (3), $\left\|{\rm A}\right\|_0$ scales linearly with the number of 3D points. Note that the proposed Haar-based highpass graph filter simply considers the first-hop neighbors with a low computational complexity; higher-order highpass graph filters could also be used at the expense of higher computational complexity.

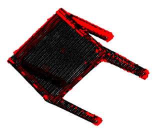
Experimental Validation. Figure 4 compares the Haar-like highpass graph filtering based local variation (15) (second column) with that computed from the difference of normals (DoN) method (first column) [65], which is used to analyze point clouds for segmentation and contour detection. As a contour detection technique, DoN computes the difference between surface normals calculated at two scales. In each plot, we highlight the points that have tje top 10% largest DoN scores or local variations. In Figure 4(a), we see that DoN cannot find the boundary in the plane because the surface normal does not change. The performance of DoN is also sensitive to predetermined radius. For example, the DoN cannot capture precise contours in the hinge. On the other hand, local variation captures all the contours precisely in Figure 4(b). We see similar results in Figures 4(c) and (d). Further, for each 3D point, DoN computes the first principle component of the neighboring points, a computationally inefficient procedure. The local variation only involves a sparse matrix and vector multiplication, a computationally efficient procedure.



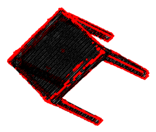
(a) Hinge: Difference of normals.



(b) Hinge: Local variation.



(c) Table: Difference of normals.



(d) Table: Local variation.

Fig. 4: Haar-like highpass graph filtering based local variation (15) outperforms the DoN method.

Figure 5 shows the local variation based resampling distribution on example point clouds—hinge, cone, table, chair and trash container. 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 (19) and Haar-like highpass graph filtering based local variation (15). The two resampled versions have the same number of points—10% of points in the original point cloud.

For the two simulated objects, the hinge and the cone (first two columns), the pairwise difference based local variation (19) fails to detect the contours while the Haar-like highpass graph filtering based local variation (15) detects all of the contours. For the real objects, the Haar-like highpass graph filtering based resampling (15) also outperform the pairwise difference based local variation (19). In summary, the Haar-like highpass graph filtering based local variation (15) detects object contours using only 10% of points.

The highpass graph filtering based resampling strategy can be easily extended to detect transient changes in other attributes. Figure 6(a) simulates a hinge with two different textures. Black points have the same texture with value 0 and the green-circle points have a different texture with value 1. We consider texture as a new attribute in the point cloud matrix ${\rm X} \in \mathbb{R}^{N \times 4}$, where the first three columns are 3D coordinates and the fourth column is the texture. We resample 10% of points based on the highpass graph filtering based local variation (15). Figure 6(b) shows the resampled point cloud, which clearly detects both the geometric contour and the texture contour.

C. Lowpass Graph Filtering

In classical signal processing, a lowpass filter is used to capture a rough shape of a smooth signal and reduce noise. Similarly, we here use a lowpass graph filter to capture rough shape of a point cloud and reduce sampling noise during resampling. Since we use the 3D coordinates of points to construct a graph (3), these 3D coordinates are naturally smooth on the graph (two adjacent points in the graph have similar coordinates in the 3D space). When a 3D point cloud is corrupted by noise and outliers, a lowpass graph filter, as a denoising operator, uses local neighboring information to approximate a true position for each point. Since the output after lowpass graph filtering is a denoised version of the original point cloud, it is more appropriate to resample from denoised points than from original points.

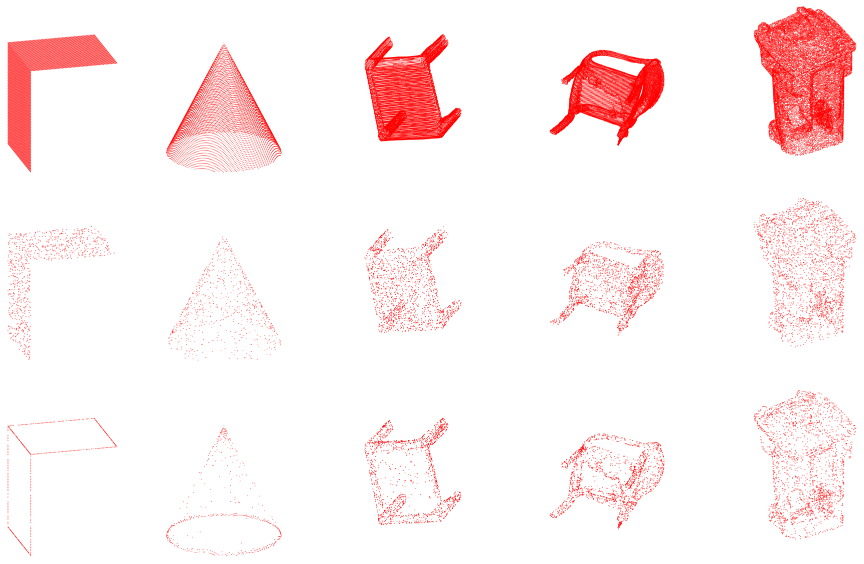


Fig. 5: Haar-like highpass graph filtering based local variation (15) outperforms pairwise difference based local variation (19). We use local variation to capture the contour. The first row shows the original point clouds; the second and third rows show the resampled versions with respect to two local variations: pairwise difference based local variation (19) and Haar-like highpass graph filtering based local variation (15). Two resampled versions have the same number of points, which is 10% of points in the original point cloud.

1) Ideal Lowpass Graph Filter: A straightforward choice is an ideal lowpass graph filter, which completely eliminates all graph frequencies above a given one while passing those below unchanged. An ideal lowpass graph filter with bandwidth b is

$$\begin{aligned} h_{\rm IL}({\rm A}) &= {\rm V} \begin{bmatrix} {\rm I}_{b\times b} & \mathbf{0}_{b\times(N-b)} \\ \mathbf{0}_{(N-b)\times b} & \mathbf{0}_{(N-b)\times(N-b)} \end{bmatrix} {\rm V}^{-1} \\ &= {\rm V}_{(b)}\, {\rm V}_{(b)}^T \in \mathbb{R}^{N\times N}, \end{aligned}$$

where ${\rm V}_{(b)}$ denotes the first b columns of V, and the graph frequency response is

$$h_{\rm IL}(\lambda_i) = \begin{cases} 1, & i \leq b; \\ 0, & \text{otherwise}. \end{cases} \quad (20)$$

The ideal lowpass graph filter $h_{\rm IL}$ projects an input graph signal onto a bandlimited subspace [37] and $h_{\rm IL}({\rm A})\mathbf{s}$ is a bandlimited approximation of the original graph signal $\mathbf{s}$. We show an example in Figure 7. Figure 7(b)—(d) shows that the bandlimited approximation of the 3D coordinates of a teapot gets better as the bandwidth b increases. We see that the bandwidth influences the shape of the teapot rapidly: with only ten graph frequencies, we obtain a rough structure of the teapot. Figure 7(e) shows that the main energy is concentrated in the lowpass graph frequency band.

The feature-extraction operator $f({\rm X}) = {\rm V}_{(b)}\, {\rm V}_{(b)}^T\, {\rm X}$ is shift and rotation-varying. Based on Theorem 4, the corresponding resampling strategy is

$$\begin{aligned} \pi_i^* &\propto \sqrt{c^2 \left\| \left({\rm V}_{(b)}\right)_i \right\|_2^2 + \left\| \left({\rm V}_{(b)}\, {\rm V}_{(b)}^T\, {\rm X}_{\rm o} \right)_i \right\|_2^2} \quad (21) \\ &= \sqrt{c^2 \left\| \mathbf{v}_i \right\|_2^2 + \left\| {\rm X}_{\rm o}{}^T\, {\rm V}_{(b)}\, \mathbf{v}_i \right\|_2^2}, \end{aligned}$$

where $\mathbf{v}_i \in \mathbb{R}^b$ is the ith row of ${\rm V}_{(b)}$. When we ignore other attributes ${\rm X}_{\rm o}$, the resampling strategy (21) is the same as the randomized resampling strategy designed for bandlimited graph signals [58], [59].

Computational Complexity. A direct way to obtain $\|\mathbf{v}_i\|_2$ requires the truncated eigendecomposition (6), whose computational cost is $O(Nb^2)$, where b is the bandwidth. It is potentially possible to approximate the leverage scores through a fast algorithm [66], [67] by using randomized techniques to avoid the eigendecomposition; the computational cost is $O(N\log(N))$. One can also partition the graph into several subgraphs and obtain leverage scores for each. Since this ideal lowpass graph filter may lead to high computational complexity, we now consider a simple, Haar-like lowpass graph filter.

2) Haar-like Lowpass Graph Filter: A Haar-like lowpass graph filter is

$$\begin{aligned} h_{\rm HL}({\rm A}) &= {\rm I} + \frac{1}{|\lambda_{\max}|}{\rm A} \quad (22) \\ &= {\rm V} \begin{bmatrix} 1+\frac{\lambda_1}{|\lambda_{\max}|} & 0 & \cdots & 0 \\ 0 & 1+\frac{\lambda_2}{|\lambda_{\max}|} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1+\frac{\lambda_N}{|\lambda_{\max}|} \end{bmatrix} {\rm V}^{-1}, \end{aligned}$$

where $\lambda_{\max} = \max_i |\lambda_i|$ (with λ_i eigenvalues of A) helps avoid magnitude amplification. We denote ${\rm A}_{\rm norm} = {\rm A}/|\lambda_{\max}|$ for simplicity. The graph frequency response is $h_{\rm HL}(\lambda_i) = 1 + \lambda_i/|\lambda_{\max}|$. Since the eigenvalues are arranged in a descending order, we have $1+\lambda_i \geq 1+\lambda_{i+1}$, meaning low frequencies are amplified while high frequencies are attenuated.

In the graph vertex domain, the response of the ith point is $(h_{\rm HL}({\rm A})\,{\rm X})_i = \mathbf{x}_i + \sum_{j\in\mathcal{N}_i} ({\rm A}_{\rm norm})_{i,j}\mathbf{x}_j$, with $\mathcal{N}_i$ its neighborhood; $h_{\rm HL}({\rm A})$ averages the attributes of each point and its neighbors to provide a smooth output.

The feature-extraction operator $f({\rm X}) = h_{\rm HL}({\rm A})\,{\rm X}$ is shift and rotation-varying. Based on Theorem 4, the corresponding resampling strategy is

$$\pi_i^* \propto \sqrt{c^2 \left\|({\rm I} + {\rm A}_{\rm norm})_i\right\|_2^2 + \left\|(({\rm I} + {\rm A}_{\rm norm})\,{\rm X}_{\rm o})_i\right\|_2^2}. \quad (23)$$

Note that we can replace ${\rm A}_{\rm norm}$ by either ${\rm D}^{-\frac{1}{2}}\,{\rm W}\,{\rm D}^{-\frac{1}{2}}$ or ${\rm D}^{-1}\,{\rm W}$; in either case, the largest eigenvalue is one, and thus ${\rm A} = {\rm A}_{\rm norm}$.

Computational Complexity. Similarly to the Haar-based highpass graph filter, the Haar-based lowpass graph filter also uses first-hop neighbors, with low computational cost; we could consider higher-order graph filters at the expense of more expensive computation. To obtain the resampling distribution (23), we compute the largest magnitude eigenvalue $\lambda_{\max}$ at $O(N)$ and $\|({\rm I}+{\rm A}_{\rm norm})_i\|_2^2$ and $\|(({\rm I}+{\rm A}_{\rm norm})\,{\rm X}_{\rm o})_i\|_2^2$ for each row at $O(\|{\rm vec}({\rm A})\|_0)$ with $\|{\rm vec}({\rm A})\|_0$ the nonzero elements in the A, which still scales linearly with the number of 3D points.

Experimental Validation. We use a lowpass graph filter to handle a noisy point cloud. Figure 8(a) shows a fitness ball point cloud that contains 62,235 points collected from a Kinect device. In this noiseless case, the surface of the fitness ball can be modeled by a sphere. Figure 8(b) fits a green sphere to the fitness ball [2]. The radius and the central point of this sphere are 0.3171 and (0.0954, 0.2077, 1.3498), respectively. To simplify computation, we resample and then fit another sphere to the resampled points. We want the spheres generated by the original point cloud and the resampled point cloud to be as close as possible.

In many real-world problems, the point cloud is noisy; to simulate it, we add Gaussian noise with mean zero and variance 0.02 to each point. Figures 9(a) and (b) show a noisy point cloud and its resampled version based on uniform resampling, respectively. Figures 9(c) and (d) show a denoised point cloud and its resampled version, respectively. Since ideal lowpass graph filtering tends to be over-smooth leading to distortion (see Figure 7), we use the Haar-like lowpass graph filtering (22) to obtain the denoised point cloud. Figure 10 compares its denoising performance with the partial differential equation (PDE)-based method [68], a standard point cloud denoising algorithm; the Haar-like lowpass graph filtering achieves a slightly better performance and is 18 times faster than the PDE-based method.

The resampling strategy can be designed based on either the Haar-like (22) or the ideal (21) lowpass graph filtering. When we set the graph shift ${\rm A} = {\rm D}^{-1}\,{\rm W}$, the corresponding Haar-like lowpass graph filtering based resampling strategy is exactly uniform sampling. Here we compare uniform sampling and the resampling strategy based on the ideal lowpass graph filtering. We fit a sphere to each of the four point clouds and show the statistics in Table I with relative errors, defined as Error $= |(x - \hat{x})/x|$, where x is the ground truth, and $\hat{x}$ is the estimation, in parentheses. The denoised ball (fourth column) significantly outperforms the noisy ball (third column), as the estimated radius and central point are closer to the original radius and central point. Based on the denoised ball, the ideal lowpass graph filtering based resampling (sixth column) slightly outperforms uniform sampling (fifth column). This the proposed resampling strategy as the one that provides robust shape modeling for noisy point clouds.

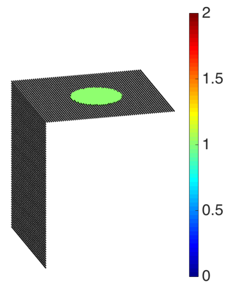
D. Graph Filter Banks

In classical signal processing, a filter bank is an array of bandpass filters that break an input signal in multiple subbands and synthesize the original signal from all of the subbands [69], [70], [71]. We use a similar idea to analyze a 3D point cloud: separate an input 3D point cloud into multiple components via different resampling operators, allowing us to enhance its different components. For example, we can resample both contour points and noncontour points to reconstruct the original surfaces; we do need more contour points to emphasize contours.

Figure 11 shows a surface reconstruction system for a 3D point cloud based on graph filter banks. In the analysis part, we separate a 3D point cloud X into k subbands. In each subband, the information preserved is determined by a specific graph filter and we resample a subset of 3D points according to (9) and (13). The number of samples in each subband is determined by a sampling ratio α. We have flexibility to use either the original 3D points or the 3D points after graph filtering. In the synthesis part, we use the resampled points to reconstruct the surface. (A good literature review on surface reconstruction algorithms can be found in [72].) Since each surface reconstruction algorithm has its own specific set of assumptions, different surface reconstruction algorithms perform differently on the same set of 3D points.

We measure the overall performance of a surface reconstruction system by reconstruction error—the difference between the surface reconstructed from resampled points and the original surface. This leads to a rate-distortion like tradeoff: when we resample more points, we encode more bits and the reconstruction error is smaller; when we resample fewer points, we encode fewer bits and the reconstruction error is larger. The overall goal is: given a reconstruction error tolerance, we use as few samples as possible to reconstruct

[2]We use CloudCompare, a public software, to generate Figure 8(b).



(a) Hinge with textures.

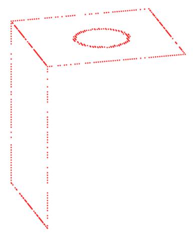
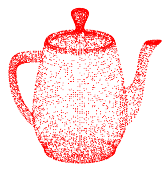
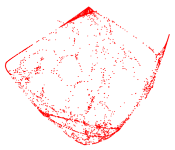
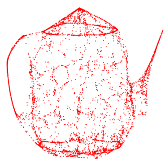
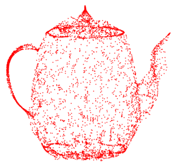

(b) Resampled version.

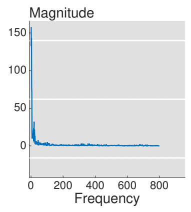
Fig. 6: Highpass graph filtering based resampling strategy detects both the geometric contour and the texture contour.


(a) Teapot.


(b) Approximation with 10 graph frequencies.


(c) Approximation with 100 graph frequencies.


(d) Approximation with 500 graph frequencies.

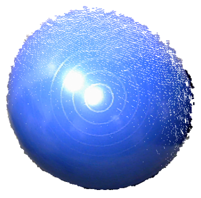


(e) Graph spectral distribution.

Fig. 7: Lowpass approximation represents the main shape of the original point clouds. Plot (a) shows a point cloud with 8,000 points representing a teapot. Plots (b)—(d) show the approximations with 10, 100 and 500 graph frequencies. We see that the approximation with 10 graph frequencies shows a rough structure of a teapot; the approximation with 100 graph frequencies can be recognized as a teapot; the approximation with 500 graph frequencies show some details of the teapot. Plot (e) shows the graph spectral distribution, which clearly shows that most energy is concentrated in the lowpass band.

	Original ball (Figure 8(a))	Noisy ball (Figure 9(a))	Denoised ball (Figure 9(b))	Uniform resampling (Figure 9(c))	Lowpass graph filtering based resampling (Figure 9(d))
Radius	0.3171	0.3312 (4.4485%)	0.3156(0.4847%)	0.3147 (0.7452%)	**0.3176** (**0.1476**%)
Center-x	0.0954	0.0875 (8.2184%)	0.0969 (1.5797%)	**0.0952** (**0.1816**%)	0.0956 (0.2933%)
Center-y	0.2077	0.2115 (1.8557%)	**0.2083** (**0.3053**%)	0.2086 (0.4627%)	0.2086 (0.4603%)
Center-z	1.3498	1.3702 (1.5061%)	**1.3516** (**0.1326**%)	1.3518 (0.1482%)	1.3517 (0.1356%)

TABLE I: Proposed resampling strategy with lowpass graph filtering provides a robust shape modeling for a fitness ball. The first column is the ground truth. The relative error is shown in the parentheses. Best results are marked in bold.


(a) Fitness ball.

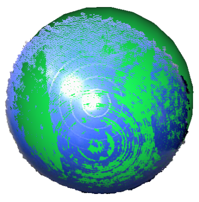
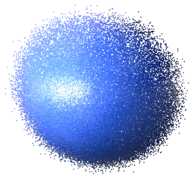
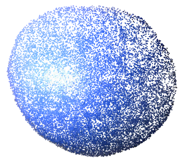
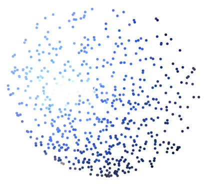

(b) Sphere fitting.

Fig. 8: Shape modeling for a fitness ball.


(a) Noisy ball.


(b) Denoised ball.


(c) Uniform sampling.

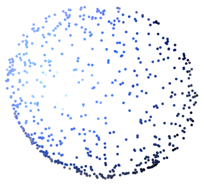

(d) Lowpass graph based resampling.

Fig. 9: Denoising and resampling of a noisy fitness ball. Plot (b) denoises Plot (a). Plots (c) and (d) resample from Plot (b) according to uniform sampling and resampling strategy (21), respectively.

a surface by carefully choosing a graph filter and sampling ratio in each subband. Such a surface reconstruction system will benefit a 3D point cloud storage and compression because we only need to store a few resampled points. Since a surface reconstruction system is application-dependent, the design details are beyond the scope of this paper.

V. Applications

In this section, we apply the proposed resampling strategies to three applications: accurate registration, large-scale visualization, and robust shape modeling. Due to the limited space, robust shape modeling is presented in Appendix H.

A. Large-Scale Visualization

We use the proposed resampling strategy (17) to efficiently visualize large-scale urban scenes. Since our visual system is sensitive to the contours of buildings and streets in a urban scene, instead of showing an entire point cloud, we only show a selected subset of points, which is useful for large-scale visualization and efficient data summarization. We consider a large-scale dataset [3], which involves several natural scenes with over 3 billion points in total and covers a range of diverse urban scenes: churches, streets, railroad tracks,

[3]http://semantic3d.net/view_dbase.php?chl=1

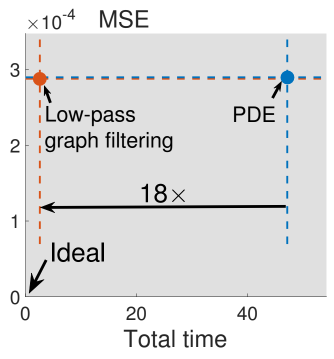


Fig. 10: Comparison of denoising performance. The proposed lowpass graph filtering is 18 times faster than PDE-based denoising algorithm.

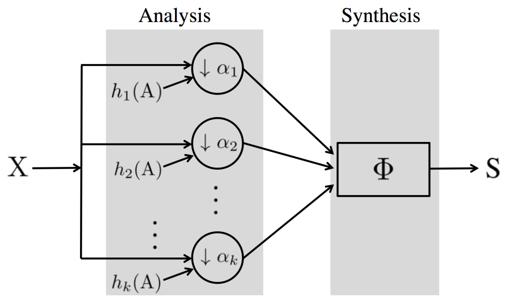


Fig. 11: Graph filter bank analysis for 3D point clouds. In the analysis part, we separate a 3D point cloud into multiple subbands. In each subband, we resample a subset of 3D points based on a specific graph filter $h(\mathrm{A})$. The number of samples in each subband is determined by a sampling ratio α. In the synthesis part, we use all of the resampled points to reconstruct a surface via a reconstruction operator Φ.

squares, villages, soccer fields, castles; see more details in Appendix V-A.

Figure 12(a) shows a point cloud of *domfountain3* with 15,105,667 points (we ignore the points on the ground). We compare the resampled point clouds based on two resampling strategies: uniform resampling in Figure 12(b) and highpass graph filtering based resampling in Figure 12(c), each with 151,057 (1%) points. Contours in (c) are much clearer demonstrating the potential of highpass graph filtering based resampling strategy for large-scale urban scene visualization. The entire computation, including graph construction, local variation computation and resampling (in Matlab on a desktop) took less than 1,000 seconds. To test scalability, we randomly select a subset of this point cloud, run the same resampling process, and record the total elapsed time. Figure 13 shows that the total computational complexity scales linearly with the number of input 3D points.

We do acknowledge that our conclusions are based soley on visual evaluation; we lack proper models of human perception to provide quantitative results. For example, while we do know that our eyes are more sensitive to contours, a reliable perception model for 3D point clouds is unknown. We leave the quality evaluation for 3D point clouds for future work.

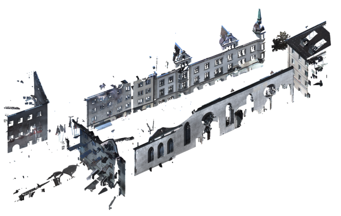
	RMSE	$\mathrm{Error}_{\mathrm{shift}}$	$\mathrm{Error}_{\mathrm{rotation}}$
All points	4.22	8.76	2.30×10^{-3}
Uniform resampling	4.27	9.38	3.76×10^{-3}
Highpass graph filtering based resampling (17)	**1.49**	**0.01**	$\mathbf{4.29 \times 10^{-5}}$

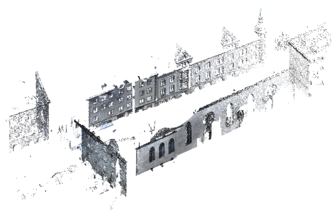
TABLE II: Proposed highpass graph filtering based resampling strategy provides an accurate registration for a sofa. Best results are marked in bold. Highpass graph filtering based resampling chooses key points and provides the best registration performance.

B. Contour-Based Registration

In this task, we use the proposed resampling strategy (17) to register two point clouds efficiently and accurately. Figure 14(a) shows a point cloud of a sofa with 1,204,055 points collected from a Kinect based SLAM system [62]. We split the original point cloud into two overlapping point clouds marked in red and blue, respectively (Figure 14). We intentionally shift and rotate the red part. The task is to invert the process and retrieve the shift and rotation. We use the iterative closest point (ICP) algorithm to register the two point clouds—a standard algorithm to rotate and shift different scans into a consistent coordinate frame [73]. The ICP algorithm iteratively revises the rigid body transformation (combination of shift and rotation) needed to minimize the distance from the source to the reference point cloud. Figures 14(b) and (c) show the registered sofa and the details of the overlapping part after registration, respectively. We see that the registration process recovers the overall structure of the original point cloud; some mismatch is still seen at a detailed level.

As it is inefficient to register two large-scale point clouds, we resample a subset of 3D points from each point cloud and then register. We compare the registration performance between a uniformly resampled point cloud and highpass graph filtering based resampled point cloud. Note that highpass graph filtering based resampling can enhance the contours and key points. Figures 14(d) and (g) show the resampled point clouds based on uniform resampling and highpass graph filtering based resampling, respectively. The two resampled versions have the same number of points (5% of points in the original point cloud). Figure 14(g) shows more contours than Figure 14(d). Based on the uniformly resampled version in Figureq 14(d), Figures 14(e) and (f) show the registered sofa and the details of the overlapping part after registration, respectively. Based on the contour-enhanced resampled version Figure 14(g), Figures 14(h) and (i) show the registered sofa and the details of the overlapping part after registration, respectively. The registration based on highpass graph filtering based resampling precisely recovers the original point cloud, even at a detailed level. The intuition is that the highpass graph filtering based resampling enhances the contour points, which emphasizes the skeletons of both the source point cloud and the target point cloud, and reduces the interior points to remove redundant and misleading information; thus, the overall registration becomes easier. The quantitative results are shown in Table II, with the root mean-squared error RMSE


(a) Original.


(b) Uniform resampling ($\downarrow$ 100).

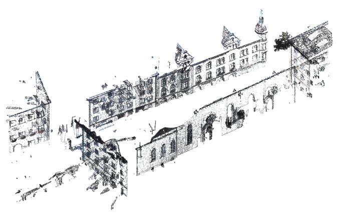

(c) Highpass graph filtering based resampling ($\downarrow$ 100).

Fig. 12: Proposed resampling strategy (17) helps efficiently visualize a large-scale urban scene. (a) A point cloud of Station3 of domfountain. (b) The resampled point clouds based on uniform resampling with 151,057 (1%) points. (c) The resampled point cloud based on highpass graph filtering based resampling with 151,057 (1%) points. Highpass graph filtering based resampling provides clear contours.

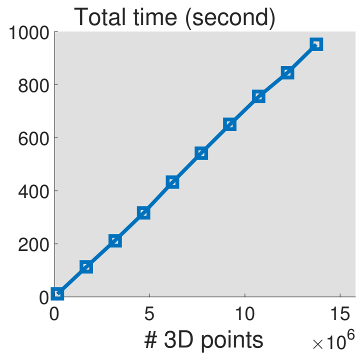


Fig. 13: Proposed resampling strategy scales linearly with the number of input 3D points. We process 15 million 3D points within 1,000 seconds by using Matlab on a standard desktop (2.3 GHz Intel Core i7, 16 GB memory); no parallel processing or code optimization was used.

$= \sqrt{\sum_{i=1}^{N} \min_{j=1,\ldots,N} \|\widehat{\mathbf{x}}_i - \mathbf{x}_j\|_2^2}$ in the first column, the shift error $\mathrm{Error}_{\mathrm{shift}} = \|\widehat{\mathbf{a}} - \mathbf{a}\|_2$ in the second column and the rotation error $\mathrm{Error}_{\mathrm{rotation}} = \left\|\widehat{\mathrm{R}} - \mathrm{R}\right\|_{\mathrm{Frobenius}}$ in the third column. Here, $\widehat{\mathbf{x}}_i$, $\widehat{\mathbf{a}}$ and $\widehat{\mathrm{R}}$ are the 3D coordinates of the ith point, recovered shift vector and rotation matrix after registration, respectively; $\mathbf{x}_i$, $\mathbf{a}$ and R are the ground-truth 3D coordinates of the ith point, shift vector and recovered rotation matrix. Highpass graph filtering based resampled point cloud uses 20 times fewer points and achieves even better results than using all of the points. The shift and rotation errors of using highpass graph filtering based resampling are significantly smaller than those of using all the points or using uniform resampling.

VI. Conclusions

We proposed a resampling framework to extract application-dependent features from a subset of points and reduce the subsequent computation in a large-scale point cloud. We formulated an optimization problem to obtain the optimal resampling distribution, which is also guaranteed to be shift-/rotation-/scale-invariant. We then specified the feature-extraction operator to be a graph filter and studied the resampling strategies based on allpass, lowpass and highpass graph filtering. A surface reconstruction system based on graph filter banks was introduced to compress 3D point clouds. Three applications, including large-scale visualization, accurate registration and robust shape modeling were presented to validate the effectiveness and efficiency of the proposed resampling methods. We also discussed future directions, such as efficient 3D point cloud compression system based on graph filter banks, surface reconstruction based on arbitrary graphs and a robust metric to evaluate the quality of a 3D point cloud.

References

[1] S. Chen, D. Tian, C. Feng, A. Vetro, and J. Kovačević, "Contour-enhanced resampling of 3D point clouds via graphs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, New Orleans, LA, Mar. 2017.

[2] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, May 2011.

[3] J. Kammerl, N. Blodow, R. B. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *Proc. IEEE Int. Conf. Robot. Autom.*, St. Paul, MN, May 2012, pp. 778–785.

[4] C. Zhang, D. Florencio, and C. T. Loop, "Point cloud attribute compression with graph transform," in *Proc. IEEE Int. Conf. Image Process.*, Paris, Oct. 2014, pp. 2066–2070.

[5] 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.

[6] A. Anis, P. A. Chou, and A. Ortega, "Compression of dynamic 3D point clouds using subdivisional meshes and graph wavelet transforms," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Shanghai, Mar. 2016, pp. 6360–6364.

[7] E. Pavez, P. A. Chou, R. L. de Queiroz, and A. Ortega, "Dynamic polygon cloud compression," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, New Orleans, LA, Mar. 2017.

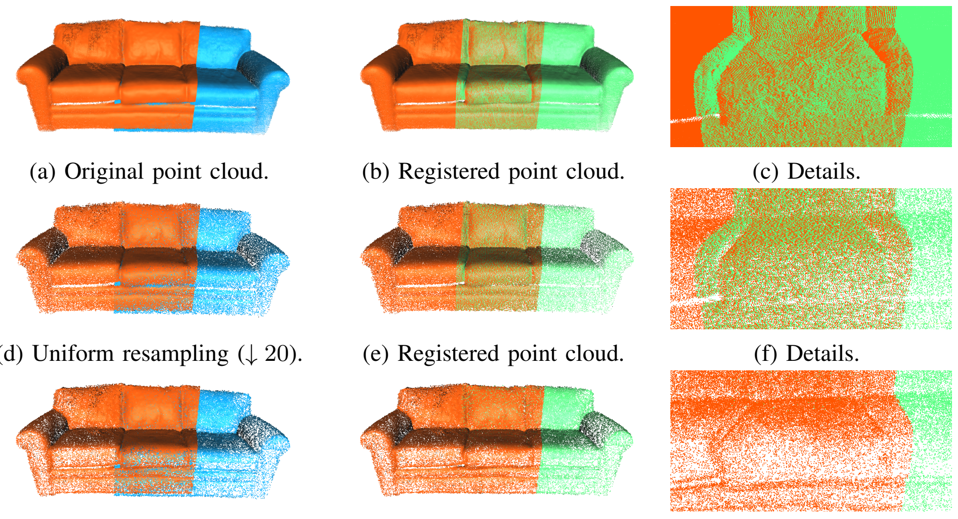
[8] P. Oesterling, C. Heine, H. Jänicke, G. Scheuermann, and G. Heyer, "Visualization of high-dimensional point clouds using their density distribution's topology," *IEEE Trans. Vis. Comput. Graphics*, vol. 17, no. 11, pp. 1547–1559, 2011.

[9] S. Pecnik, D. Mongus, and B. Zalik, "Evaluation of optimized visualization of LiDAR point clouds, based on visual perception," in *Proc. Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data*, Maribor, July 2013, pp. 366–385.

[10] B. F. Gregorski, B. Hamann, and K. I. Joy, "Reconstruction of B-spline surfaces from scattered data points," in *Proc. Comput. Graphics Int.l*, Geneva, June 2000, pp. 163–170.

[11] A. Golovinskiy, V. G. Kim, and T. A. Funkhouser, "Shape-based recognition of 3D point clouds in urban environments," in *Proc. IEEE Int. Conf. Comput. Vis.*, Kyoto, Sept. 2009, pp. 2154–2161.

[12] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE Trans. Vis. Comput. Graphics*, vol. 9, no. 1, pp. 3–15, Jan. 2003.



(a) Original point cloud. (b) Registered point cloud. (c) Details.

(d) Uniform resampling (↓ 20). (e) Registered point cloud. (f) Details.

(g) Highpass graph filtering based graph filtering (↓ 20). (h) Registered point cloud. (i) Details.

Fig. 14: Accurate registration for a sofa point cloud. The figure shows the original point cloud (first row), the uniformly resampled point cloud (second row) and the highpass graph filtering based resampled point cloud (17) (third row) as well as the point clouds before (first column) and after (second column) registration and the details around the overlapping area (third column). We see that highpass graph filtering-based resampling provides more precise registration with fewer points.

[13] F. Ryden, S. N. Kosari, and H. J. Chizeck, "Proxy method for fast haptic rendering from time varying point clouds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, San Francisco, CA, Sept. 2011, pp. 2614–2619.

[14] I. Guskov, W. Sweldens, and P. Schröder, "Multiresolution signal processing for meshes," in *Proc. ACM SIGGRAPH*, New York, NY, July 1999, vol. 24, pp. 325–334.

[15] M. Wand, A. Berner, M. Bokeloh, A. Fleck, M. Hoffmann, P. Jenke, B. Maier, D. Staneker, and A. Schilling, "Interactive editing of large point clouds," in *Proc. Symp. Point Based Graphics*, Prague, 2007, pp. 37–45.

[16] K. Demarsin, D. Vanderstraeten, T. Volodine, and D. Roose, "Detection of closed sharp feature lines in point clouds for reverse engineering applications," in *Proc. Geometric Modeli. Process.*, Pittsburgh, PA, 2006, pp. 571–577.

[17] C. Weber, S. Hahmann, and H. Hagen, "Sharp feature detection in point clouds," in *Proc. Int. Conf. Shape Modell.*, Aix-en-Provence, June 2010, pp. 175–186.

[18] J. Daniels, L. K. Ha, T. Ochotta, and C. T. Silva, "Robust smooth feature extraction from point clouds," in *Proc. IEEE Int. Conf. Shape Model. Applic.*, Lyon, June 2007, pp. 123–136.

[19] C. Choi, A. J. Trevor, and H.I. Christensen, "RGB-D edge detection and edge-based registration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Tokyo, Nov. 2013, pp. 1568–1575.

[20] C. Feng, Y. Taguchi, and V. Kamat, "Fast plane extraction in organized point clouds using agglomerative hierarchical clustering," in *Proc. IEEE Int. Conf. Robot. Autom.*, Hong Kong, May 2014, pp. 6218–6225.

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

[22] W. Luo and H. Zhang, "Visual analysis of large-scale LiDAR point clouds," in *Proc. IEEE Int. Conf. Big Data*, Santa Clara, CA, Nov. 2015, pp. 2487–2492.

[23] J. Ryde and H. Hu, "3D mapping with multi-resolution occupied voxel lists," *Autonom. Robots*, pp. 169–185, Apr. 2010.

[24] C. T. Loop, C. Zhang, and Z. Zhang, "Real-time high-resolution sparse voxelization with application to image-based modeling," in *Proc. High-Perform. Graphics*, Anaheim, CA, July 2013, pp. 73–80.

[25] J. Peng and C.-C. Jay Kuo, "Geometry-guided progressive lossless 3D mesh coding with octree (OT) decomposition," *ACM Trans. Graphics — Proc. ACM SIGGRAPH*, vol. 24, no. 3, pp. 609–616, July 2005.

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

[27] B. Eckart and A. Kelly, "REM-Seg: A robust em algorithm for parallel segmentation and registration of point clouds," in *Proc. IEEE Int. Conf. Intell. Robots Systems*, Tokyo, June 2013, pp. 4355–4362.

[28] 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, NV, June 2016.

[29] J. Du, S. Ma, Y-C.Wu, S. Kar, and J. M. F. Moura, "Convergence analysis of distributed inference with vector-valued Gaussian belief propagation," *J. Mach. Learn. Res., Submitted*, 2016, arXiv:0704.3434v3 [cs.IT].

[30] J. Du, S. Ma, Y-C.Wu, S. Kar, and J. M. F. Moura, "Convergence analysis of the information matrix in gaussian belief propagation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, New Orleans, LA, Mar. 2017.

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

[32] P. Alliez and C. Gotsman, "Recent advances in compression of 3D meshes," in *Adv. in Multires. for Geom. Modelling*. 2003, pp. 3–26, Springer-Verlag.

[33] 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.

[34] 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, NV, June 2016.

[35] 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.

[36] 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.

[37] 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.

[38] 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, no. 14, pp. 3775–3789, July 2016.

[39] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Sampling of graph signals with successive local aggregations," *IEEE Trans. Signal Process.*, vol. 64, no. 7, pp. 1832–1843, Apr. 2016.

[40] M. Tsitsvero, S. Barbarossa, and P. D. Lorenzo, "Signals on graphs:

Uncertainty principle and sampling," *IEEE Trans. Signal Process.*, vol. 64, pp. 4845–4860, Sept. 2016.

[41] 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.

[42] A. Sandryhaila and J. M. F. Moura, "Big data processing with signal processing on graphs," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, Sept. 2014.

[43] J. Kammerl, "Point cloud compression," http://pointclouds.org/documentation/tutorials/compression.php#octree-compression, [Online; accessed 2015-11-09].

[44] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, "Signal denoising on graphs via graph filtering," in *Proc. IEEE Glob. Conf. Signal Information Process.*, Atlanta, GA, Dec. 2014, pp. 872–876.

[45] S. Segarra, G. Mateos, A. G. Marques, and A. Ribeiro, "Blind identification of graph filters," *IEEE Trans. Signal Process.*, vol. 65, no. 5, pp. 1146–1159, 2017.

[46] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, June 2014.

[47] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Graph Fourier transform," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Vancouver, May 2013, pp. 6167–6170.

[48] W. Huang, L. Goldsberry, N. F. Wymbs, S. T. Grafton, D. S. Bassett, and A. Ribeiro, "Graph frequency analysis of brain signals," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 7, pp. 1189–1203, 2016.

[49] S. Sardellitti, S. Barbarossa, and P. D. Lorenzo, "On the graph Fourier transform for directed graphs," *IEEE J. Sel. Topics Signal Process.*, vol. 11, Sept. 2017.

[50] I. Z. Pesenson, "Sampling in Paley-Wiener spaces on combinatorial graphs," *Trans. Am. Math. Soc.*, vol. 360, no. 10, pp. 5603–5627, May 2008.

[51] A. Anis, A. Gadde, and A. Ortega, "Towards a sampling theorem for signals on arbitrary graphs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Florence, May 2014, pp. 3864–3868.

[52] 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, no. 14, pp. 3775–3789, July 2015.

[53] X. Wang, J. Chen, and Y. Gu, "Local measurement and reconstruction for noisy graph signals," *Signal Process.*, vol. 129, pp. 119–129, 2016.

[54] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Sampling of graph signals with successive local aggregations," *IEEE Trans. Signal Process.*, vol. 64, no. 7, pp. 1832–1843, Apr. 2016.

[55] S. Segarra, A. G. Marques, G. Leus, and A. Ribeiro, "Reconstruction of graph signals through percolation from seeding nodes," *IEEE Trans. Signal Process.*, vol. 64, pp. 4363–4378, Aug. 2016.

[56] L. F. Chamon and A. Ribeiro, "Greedy sampling of graph signals," *arXiv:1704.01223*, 2016.

[57] F. Garna, A. G. Marques, G. Mateos, and A. Ribeiro, "Rethinking sketching as sampling: Efficient approximate solution to linear inverse problems," in *Proc. IEEE Glob. Conf. Signal Information Process.*, Washington, DC, Dec. 2016, pp. 390–394.

[58] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Signal recovery on graphs: Random versus experimentally designed sampling," in *Proc. Sampling Theory Appl.*, Washington, DC, May 2015, pp. 337–341.

[59] G. Puy, N. Tremblay, R. Gribonval, and P. Vandergheynst:, "Random sampling of bandlimited signals on graphs," *Appl. Comput. Harmon. Anal.*, June 2016.

[60] N. Tremblay, G. Puy, R. Gribonval, and P. Vandergheynst, "Compressive spectral clustering," in *Proc. Int. Conf. Mach. Learn.*, New York, NY, June 2016, vol. 48, pp. 1002–1011.

[61] 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*, vol. 2, no. 4, pp. 539–554, Dec. 2016.

[62] Y. Taguchi, Y-D Jian, S. Ramalingam, and C. Feng, "Point-plane slam for hand-held 3D sensors," in *Proc. IEEE Int. Conf. Robot. Autom.*, Karlsruhe, May 2013.

[63] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Appl. Comput. Harmon. Anal.*, vol. 30, pp. 129–150, Mar. 2011.

[64] F. R. K. Chung, *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92)*, Am. Math. Soc., 1996.

[65] Y. Loannou, B. Taati, R. Harrap, and M. A. Greenspan, "Difference of normals as a multi-scale operator in unorganized point clouds," in *Proc. Int. Conf. 3D Imaging, Model., Process., Visual., Transm.*, Zurich, Oct. 2012, pp. 501–508.

[66] P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. Woodruff, "Fast approximation of matrix coherence and statistical leverage," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 3475–3506, 2012.

[67] D. P. Woodruff, "Sketching as a tool for numerical linear algebra," *Found. Trends Theor. Comput. Sci.*, vol. 10, no. 1–2, pp. 1–157, Oct. 2014.

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

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

[70] M. Vetterli, J. Kovačević, and V. K. Goyal, *Foundations of Signal Processing*, Cambridge University Press, Cambridge, 2014, http://foundationsofsignalprocessing.org.

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

[72] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, G. Guennebaud, J. A. Levine, A. Sharf, and C. T. Silva, "A survey of surface reconstruction from point clouds," *Comput. Graphics Forum*, 2016.

[73] P. J. Besl and N. D. McKay, "A method for registration of 3D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992.

Appendix

A. Proof of Lemma 1

Proof. Let $\mathcal{M}, \Psi$ be the resampling set under the resampling distribution π and the corresponding resampling operator, respectively. Let $\delta_{i=\mathcal{M}_j}$ be a Bernoulli random variable indicating whether the ith point is selected in the jth selection. According to the resampling distribution π, we have

$$\delta_{i=\mathcal{M}_j} = \begin{cases} 1, & \text{with probability } \pi_i; \\ 0, & \text{with probability } 1-\pi_i. \end{cases}$$

According to (2), $\Psi^T\Psi$ is a diagonal, zero-padding matrix, whose ith element along the diagonal is

$$\left(\Psi^T\Psi\right)_{i,i} = \sum_{\mathcal{M}_j \in \mathcal{M}} \delta_{i=\mathcal{M}_j},$$

which accumulates the number of times that the ith point is selected; in other words, $\left(\Psi^T\Psi\right)_{i,i}$ is a Binomial random variable, following $B(M, \pi_i)$. We thus have

$$\mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi\right)_{i,i} = M\pi_i, \tag{24}$$

$$\mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi\right)^2_{i,i} = M^2\pi_i^2 + M\pi_i(1-\pi_i). \tag{25}$$

For the nonweighted version, the ith row of $\mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi f(\mathrm{X})\right)$ is

$$\begin{aligned} \mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi f(\mathrm{X})\right)_i &\overset{(a)}{=} \mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi\right)_{i,i} f_i(\mathrm{X}) \\ &\overset{(b)}{=} M\pi_i f_i(\mathrm{X}), \end{aligned}$$

where (a) follows from that $\Psi^T\Psi$ is a diagonal matrix and (b) follows from (24). Thus, $\mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi f(\mathrm{X})\right) = M\pi \odot f(\mathrm{X})$, with $\odot$ the row-wise multiplication.

For the reweighted version, $\mathrm{S} \in \mathbb{R}^{N\times N}$ is a diagonal rescaling matrix, whose the ith element along the diagonal is $\mathrm{S}_{i,i} = 1/(M\pi_i)$. The ith row of $\mathbb{E}_{\Psi\sim\pi}\left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right)$ is

$$\begin{aligned} \mathbb{E}_{\Psi\sim\pi}\left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right)_i &= \mathrm{S}_{i,i}\,\mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi\right)_{i,i} f_i(\mathrm{X}) \\ &= \mathrm{S}_{i,i}\, M\pi_i f_i(\mathrm{X}) \;=\; f_i(\mathrm{X}). \end{aligned}$$

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

B. Proof of Theorem 1

Proof. The mean square error of the estimator $\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})$ is

$$\begin{aligned}
& \mathbb{E}_{\Psi\sim\pi} \left\| \mathrm{S}\,\Psi^T\Psi f(\mathrm{X}) - f(\mathrm{X}) \right\|_F^2 \\
\overset{(a)}{=} \;& \mathbb{E}_{\Psi\sim\pi} \left[\mathrm{Tr} \left(f(\mathrm{X})^T \left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right)^T \left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right) f(\mathrm{X}) \right) \right] \\
= \;& \mathrm{Tr} \left(f(\mathrm{X})^T \mathbb{E}_{\Psi\sim\pi} \left[\left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right)^T \left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right) \right] f(\mathrm{X}) \right).
\end{aligned}$$

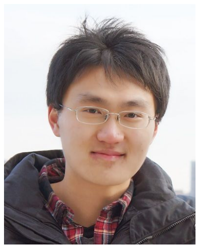
where (a) follows from the property of the Frobenius norm. We next want to show

$$\mathbb{E}_{\Psi\sim\pi} \left[\left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right)^T \left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right) \right] = \mathrm{Q}\,.$$

Since both S and $\Psi^T\Psi$ are diagonal matrices, $\left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right)^T \left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right)$ is also a diagonal matrix, whose $i$the element along the diagonal is

$$\begin{aligned}
& \mathbb{E}_{\Psi\sim\pi} \left[\left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right)^T \left(\mathrm{S}\,\Psi^T\Psi - \mathrm{I} \right) \right]_{i,i} \\
\overset{(a)}{=} \;& \mathbb{E}_{\Psi\sim\pi} \left(\mathrm{S}_{i,i} (\Psi^T\Psi)_{i,i} - 1 \right)^2 \\
= \;& \mathrm{S}_{i,i}^2 \, \mathbb{E}_{\Psi\sim\pi} (\Psi^T\Psi)_{i,i}^2 - 2\,\mathrm{S}_{i,i}\, \mathbb{E}_{\Psi\sim\pi} (\Psi^T\Psi)_{i,i} + 1 \\
\overset{(b)}{=} \;& \frac{M^2\pi_i^2 + M\pi_i(1-\pi_i)}{M^2\pi_i^2} - 2\frac{M\pi_i}{M\pi_i} + 1 \\
= \;& \frac{1-\pi}{M\pi} \;=\; \mathrm{Q}_{i,i}\,.
\end{aligned}$$

where (a) follows from the property of the diagonal matrix and (b) follows from (24) and (25). Thus, the mean square error is $\mathbb{E}_{\Psi\sim\pi} \left\| \mathrm{S}\,\Psi^T\Psi f(\mathrm{X}) - f(\mathrm{X}) \right\|_F^2 = \mathrm{Trace} \left(f(\mathrm{X})^T \mathrm{Q}\, f(\mathrm{X}) \right)$. □



Siheng Chen received his B. Eng degree in Optoelectronics Engineering in 2011 from Beijing Institute of Technology, the M.S degrees in Electrical and Computer Engineering and Machine learning in 2012 and 2016, and the Ph.D degree in Electrical and Computer Engineering in 2016 from Carnegie Mellon University. He was a postdoctoral research associate in Electrical and Computer Engineering at Carnegie Mellon University. His research interests include signal processing, machine learning, and graph mining.

Dong Tian received the Ph.D. degree at Beijing University of Technology in 2001, and the M.Eng. and B.Eng. degrees on automation from the University of Science and Technology of China (USTC) in 1998 and 1995, respectively.

He is Senior Principal Member Research Scientist in the Multimedia Group of Mitsubishi Electric Research Laboratories (MERL) at Cambridge, MA. Prior to joining MERL, he worked with Thomson Corporate Research at Princeton, NJ for over 4 years, where he was devoted to H.264/MPEG AVC encoder optimization and 3D video coding/processing, especially to the standards of Multiview Video Coding (MVC) and later on 3D Video Coding (3DV) within MPEG. From Jan 2002 to Dec 2005, he was a postdoc at Tampere University of Technology in Finland for a Nokia funded project and made contributions on video coding standards and applications for mobile environments. His current research interests include graph signal processing, point cloud processing, deep learning, image/video coding and processing. Besides academic publications, he has over 20 US-granted patents. He is a senior member of IEEE.

Chen Feng received his B.Eng degree in Geospatial Engineering in 2010 from Wuhan University, an M.S. degree in Construction Engineering in 2012, an M.S. degree in Electrical Engineering in 2013, and the Ph.D. degree in Civil Engineering in 2015 from University of Michigan, Ann Arbor. His research interests include computer vision, robotics, and construction automation.

Anthony Vetro (S'92–M'96–SM'04–F'11) received the B.S., M.S., and Ph.D. degrees in electrical engineering from Polytechnic University, Brooklyn, NY. He joined Mitsubishi Electric Research Labs, Cambridge, MA, in 1996. He is currently a Deputy Director of the lab and also manages a group that is responsible for research on video coding and image processing, information security, sensing technologies, and speech/audio processing. He has published more than 200 papers in these areas. He has also been an active member of the ISO/IEC and ITU-T standardization committees on video coding for many years, where he has served as an ad-hoc group chair and editor for several projects. He was a key contributor to 3D and multiview extensions of the AVC and HEVC standards, and served as Head of the U.S. delegation to MPEG. Dr. Vetro is also active in various IEEE conferences, technical committees, and editorial boards. He served as an Associate Editor for IEEE Transactions on Image Processing, IEEE Transactions on Circuits and Systems for Video Technology, and IEEE Signal Processing Magazine, and as a member of the Editorial Boards of IEEE Multimedia, IEEE Signal Processing Magazine, IEEE Journal on Selected Topics in Signal Processing, IEEE Journal on Emerging and Selected Topics in Circuits and Systems, and IEEE Transactions on Consumer Electronics. He served as Chair of the Technical Committee on Multimedia Signal Processing of the IEEE Signal Processing Society and on the Steering Committees for ICME and the IEEE Transactions on Multimedia. He was a General Co-Chair of ICIP 2017 in Beijing and ICME 2015 in Torino, and a Technical Program Co-chair of ICME 2016 in Seattle. He has received several awards for his work on transcoding, including the 2003 IEEE Circuits and Systems CSVT Transactions Best Paper Award, and is a Fellow of IEEE.

Jelena Kovačević (S'88–M'91–SM'96–F'02) received the Dipl. Electr. Eng. degree from the EE Department, University of Belgrade, Yugoslavia, in 1986, and the M.S. and Ph.D. degrees from Columbia University, New York, in 1988 and 1991, respectively. From 1991–2002, she was with Bell Labs, Murray Hill, NJ. She was a co-founder and Technical VP of xWaveforms, based in New York City and an Adjunct Professor at Columbia University. In 2003, she joined Carnegie Mellon University, where she is Hamerschlag University Professor and Head of Electrical and Computer Engineering, Professor of Biomedical Engineering, and was the Director of the Center for Bioimage Informatics at Carnegie Mellon University. Her research interests include wavelets, frames, graphs, and applications to bioimaging and smart infrastructure.

Dr. Kovačević coauthored the books Wavelets and Subband Coding (Prentice Hall, 1995) and Foundations of Signal Processing (Cambridge University Press, 2014), a top-10 cited paper in the Journal of Applied and Computational Harmonic Analysis, and the paper for which A. Mojsilović received the Young Author Best Paper Award. Her paper on multidimensional filter banks and wavelets was selected as one of the Fundamental Papers in Wavelet Theory. She received the Belgrade October Prize in 1986, the E.I. Jury Award at Columbia University in 1991, and the 2010 CIT Philip L. Dowd Fellowship Award from the College of Engineering at Carnegie Mellon University and the 2016 IEEE SPS Technical Achievement Award. She is a past Editor-in-Chief of the IEEE Transactions on Image Processing, served as a guest co-editor on a number of special issues and is/was on the editorial boards of several journals. She was a regular member of the NIH Microscopic Imaging Study Section and served as a Member-at-Large of the IEEE Signal Processing Society Board of Governors. She is a past Chair of the IEEE Signal Processing Society Bio Imaging and Signal Processing Technical Committee. She has been involved in organizing numerous conferences. She was a plenary/keynote speaker at a number of international conferences and meetings.