Graph-Based Depth Denoising & Dequantization for Point Cloud Enhancement

Xue Zhang, *Member, IEEE*, Gene Cheung, *Fellow, IEEE*, Jiahao Pang, *Member, IEEE*, Yash Sanghvi, *Student Member, IEEE*, Abhiram Gnanasambandam, *Member, IEEE*, and Stanley H. Chan, *Senior Member, IEEE*

Abstract— A 3D point cloud is typically constructed from depth measurements acquired by sensors at one or more viewpoints. The measurements suffer from both quantization and noise corruption. To improve quality, previous works denoise a point cloud *a posteriori* after projecting the imperfect depth data onto 3D space. Instead, we enhance depth measurements directly on the sensed images *a priori*, before synthesizing a 3D point cloud. By enhancing near the physical sensing process, we tailor our optimization to our depth formation model before subsequent processing steps that obscure measurement errors. Specifically, we model depth formation as a combined process of signal-dependent noise addition and non-uniform log-based quantization. The designed model is validated (with parameters fitted) using collected empirical data from a representative depth sensor. To enhance each pixel row in a depth image, we first encode intra-view similarities between available row pixels as edge weights via feature graph learning. We next establish inter-view similarities with another rectified depth image via viewpoint mapping and sparse linear interpolation. This leads to a maximum a posteriori (MAP) graph filtering objective that is convex and differentiable. We minimize the objective efficiently using accelerated gradient descent (AGD), where the optimal step size is approximated via Gershgorin circle theorem (GCT). Experiments show that our method significantly outperformed recent point cloud denoising schemes and state-of-the-art image denoising schemes in two established point cloud quality metrics.

Index Terms— 3D point cloud, depth sensing, signal-dependent noise, non-uniform quantization, graph signal processing.

I. INTRODUCTION

POINT Cloud (PC) is a collection of discrete geometric samples of the surface of a physical object in 3D space,
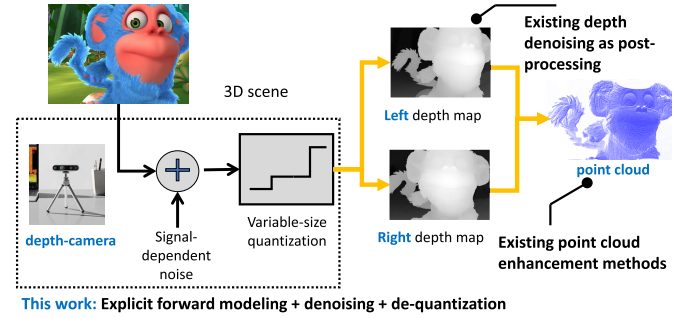
Fig. 1. Depth measurements from a consumer sensor suffer from signal-dependent noise and variable-size quantization. We enhance multi-view depth measurements as close to the physical sensing process as possible, resulting in higher-quality subsequent 3D point cloud construction.

useful for a range of imaging applications such as immersive communication and virtual/augmented reality (AR/VR) [1], [2], [3], [4]. With the ubiquity of inexpensive active sensors like Microsoft Kinect and Intel RealSense, one common method to generate a PC is to deploy one or more sensors at multiple viewpoints to capture depth measurements (in the form of images) of an object, then project these measurements to 3D space to synthesize a PC [5], [6]. However, limitations in the depth acquisition process mean that the acquired depth measurements suffer from both imprecision (due to quantization) and additive noise. This results in a noisy synthesized PC, and previous works focus on denoising PCs using a variety of methods: low-rank prior, low-dimensional manifold model (LDMM), surface smoothness priors expressed as graph total variation (GTV), graph Laplacian regularizer (GLR) and feature graph Laplacian regularizer (FGLR), Moving Robust Principal Components Analysis (MRPCA), data-driven learning, etc [7], [8], [9], [10], [11], [12].

However, all the aforementioned denoising methods enhance a PC *a posteriori*, *i.e.*, after a PC is synthesized from corrupted depth measurements. Recent works in image denoising [13], [14] have shown that by denoising raw sensed RGB measurements directly on the Bayer-patterned grid *before* demosaicking, contrast boosting and other steps typical in an imaging pipeline [15], [16] that obscure acquisition noise, one can dramatically improve performance compared to denoising the constructed image *after* the pipeline (up to 15dB in PSNR).

Inspired by these works, we propose to *enhance* depth measurements in sensed images *a priori*, before projecting to the 3D space to synthesize a PC. By "enhancement",
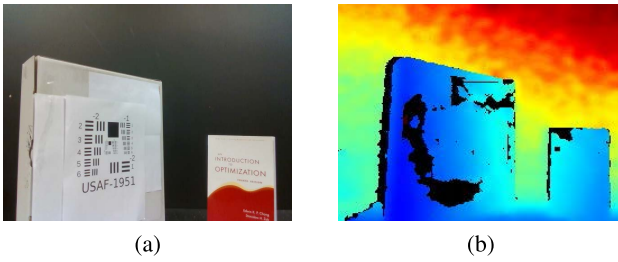
Fig. 2. Example of collected images using the *Intel RealSense^TM D435*: (a) the color image, (b) the corresponding depth map with missing pixels.

here we mean performing joint denoising and dequantization based on our proposed depth formation model. In our case, by enhancing near the physical sensing process before steps in a PC synthesis pipeline including projection, registration, stitching, and filtering [17], we can tailor our optimization by parameterizing our proposed depth formation model to a specific sensor. As illustrated in Fig. 1, this means fitting model parameters for *signal-dependent noise* and *variable-size quantization* that are unique for individual sensors. This precise modeling of sensed depth measurements is in stark contrast to the grossly inaccurate i.i.d. Gaussian noise[1] typically assumed in the PC denoising literature [7], [8], [9], [10], [11], [12].

Mathematically, we first model depth pixel acquisition as a combined process of signal-dependent noise addition and non-uniform log-based quantization. We verify the validity of this model and fit model parameters through an empirical study using the *Intel RealSense^TM D435* camera—a representative depth sensor that employs stereo correspondence technology on a pair of IR images to improve depth quality. We parameterize the signal-dependent noise variance as a function of depth from empirical data using the golden-section (GS) search [18] in a *maximum likelihood* (ML) formulation.

To enhance a row of available pixels in a left depth image—often interleaved with missing pixels[2] as shown in Fig. 2—we first encode *intra-view* similarities between neighboring available pixels as edge weights in a graph via *feature graph learning* [9]. Specifically, estimated 3D features per depth pixel—*i.e.*, 3D coordinate and surface normal—are used to compute *feature distance* d_{ij} between pixel pair (i, j), so that edge weight $w_{ij} = \exp(-d_{ij})$ can be computed using rich 3D structure information to construct a similarity graph.

We next establish *inter-view* similarities with a rectified right depth image via viewpoint mapping [20] and sparse linear interpolation. Appropriate approximations via Taylor series expansion lead to a *maximum a posteriori* (MAP) graph filtering objective that is convex and differentiable. We minimize the objective efficiently using *accelerated gradient descent* (AGD) [21], [22], where the optimal step size

is efficiently approximated via *Gershgorin circle theorem* (GCT) [23]. Experimental results show that by enhancing depth measurements in the image domain prior to PC synthesis, our method significantly outperformed several recent PC denoising algorithms [7], [10], [11], [12], [24], [25] and representative image denoising schemes[3] [26], [27], [28] in two commonly used PC quality metrics [29], [30] for three different types of depth datasets.

We summarize our technical contributions as follows:
  (i) To improve PC construction quality near the physical sensing process, we enhance depth measurements in the image domain, where we can parameterize our depth formation model for a specific sensor.
  (ii) We design the depth formation model by combining *both* signal-dependent noise addition *and* non-uniform log-based quantization process, validated with collected empirical data from a representative depth sensor.
  (iii) We encode intra-view similarities for *available* pixel pairs in a depth image row as edge weights via feature graph learning, and inter-view similarities via viewpoint mapping and sparse linear interpolation.
  (iv) We minimize the resulting convex and differentiable MAP graph filtering objective via AGD, where the important step size is approximated speedily via GCT.

We stress that, while the general approach to denoise/restore observed data as close to the physical layer as possible is not new, our specialization for 3D point cloud is novel. In particular, we are the first in the 3D point cloud literature to propose a joint non-uniform quantization/signal-dependent additive noise model, and a corresponding restoration algorithm designed for the proposed model.

The outline of the paper is as follows. We first overview related works in Section II. We then describe our depth capturing system and the depth formulation model in Section III. We also present our empirical study based on collected depth data. The formulation of our optimization problem and the corresponding algorithm to solve it are discussed in Section IV. We discuss the graph learning procedure to capture inter-pixel similarities in Section V. Experimental results and conclusion are presented in Section VI and VII, respectively.

II. RELATED WORKS

We review two categories of related works: depth image denoising/enhancement and 3D point cloud denoising.

A. Depth Image Denoising/Enhancement

1) Depth Image Denoising: Unlike natural image denoising [28], [31], [32], existing literature in depth image denoising is dominated by model-based schemes [26], [33], [34], [35], [36], [37], [38], [39], [40]. Specifically, it is common to perform depth denoising using an assumed signal prior or filter [33], [34], [35], [36], [37], [38]. For example, [33] used a sparsity prior for each pixel patch in the graph Fourier domain,

---

[1]For example, a corrupted depth pixel is erred *perpendicular* to the image plane, which is very different from a projected 3D point corrupted by i.i.d. Gaussian noise along its three coordinates as assumed in existing works.

[2]A depth pixel from a structured light sensor is missing if an occlusion or optical interference occurs [19]. A depth pixel from a time-of-flight sensor is missing if the emitted laser arrives at an incident angle such that it is not reflected and returned to the sensor [19].

[3]Our paper is a non-trivial journal extension of our earlier work [26] that assumed signal-independent noise and uniform quantization. We compare against [26] experimentally in Section VI.

while [35] employed a regularized shock filter. Both methods exploited the known piecewise smooth (PWS) characteristic of depth images [41] for better performance. When the color image associated to the depth map is available, denoising performance can be further boosted. For example, [39] built neighborhood graphs from color images for depth denoising.

However, these works modeled the depth degradation process with *signal-independent* noise—*e.g.*, i.i.d. additive Gaussian noise [33], [37], [38]—for simplicity. As discussed in [42], [43] and also verified in our work, depth measurements are noisier for larger depth values; *i.e.*, the noise variance is *signal-dependent*. Moreover, existing depth image denoising algorithms offer no explicit procedures to handle missing depth values that are common in real-world sensed depth images, as shown in Fig. 2. In contrast, our proposed depth formation model accounts for noise variance's signal dependency, which we parameterize accurately using collected empirical data of an actual depth sensor in Section III-C. Further, our graph-based optimization enhances available pixels around missing ones by capturing inter-pixel similarity via feature graph learning [9]. We compare against several representative image-denoising schemes [26], [27], [28] in Section VI.

2) Depth Image Dequantization: Depth images are quantized by depth sensors for storage or transmission purposes [44]. To enhance depth precision, previous works adopted different strategies. Given several depth maps from different views, [45] used the *Projection On Convex Set* (POCS) procedure to enhance precision of two depth images simultaneously, while [26], [41], [46] employed a graph-signal smoothness prior to enhance a single depth image.

Previous research all assumed that depth images are quantized uniformly. However, in practical sensors, larger quantization bin are used as depth values increase, as confirmed by our empirical data in Section III-D. Thus, we propose a non-uniform log-based quantization process, so that larger depth values have coarser quantization.

B. 3D Point Cloud Denoising

PC denoising [47] is necessary prior to 2D viewpoint image rendering or another down-stream task. Related model-based methods can be roughly categorized into four types: moving least square (MLS)-based methods [25], [48], [49], locally optimal projection (LOP)-based methods [50], [51], sparsity-based methods [52], [53] and nonlocal-based methods [7], [54], [55]. Among these works, with the exception of [53] that related the noise variance to surface normal, they all assumed independent additive noise. For instance, [7], [52], [55] adopted the additive i.i.d. Gaussian noise model to simplify analysis. As discussed earlier, we model noise on each measurement in a depth image as signal-dependent.

Beyond the signal-dependent nature of acquisition noise, another drawback of existing methods is the assumption that the 3D coordinates of each PC point are omnidirectionally corrupted by additive noise. This is a gross oversimplification: depth measurements are first corrupted by additive noise and quantization perpendicular to the image plane, before 3D projection to synthesize a PC—the resulting distortion in the
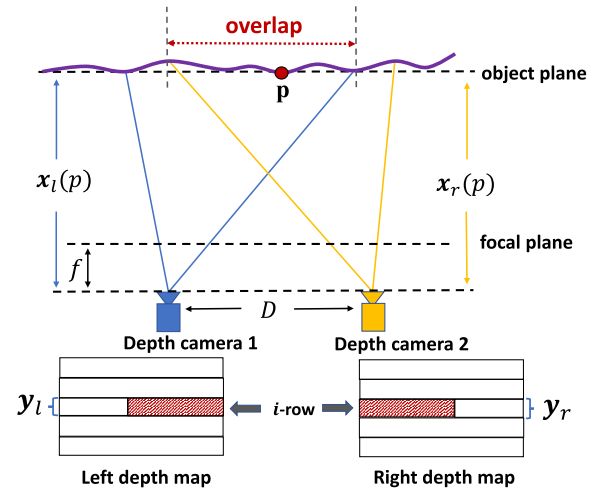

Fig. 3. Example of the depth camera system. Two depth cameras with focal length f are placed distance D apart capturing the same 3D scene. Assuming that the corresponding depth images are rectified, a left pixel row $\mathbf{x}_l$ maps to a right pixel row $\mathbf{x}_r$ via a view-to-view mapping described in Section IV-B, $\mathbf{y}_l$ and $\mathbf{y}_r$ are corrupted observations.

projected 3D space is far from omnidirectional. In contrast, we enhance depth measurements on an image before 3D projection.

Recent developments of *deep neural networks* (DNN) have revolutionized different areas in computer vision and image processing, including PC denoising [11], [12]. Beyond the inaccurate assumption of each PC corrupted by i.i.d. Gaussian noise, these learning-based approaches are purely data-driven, and a large amount of labeled data is required for training [56]. However, it is difficult to collect noiseless ground-truth data for supervised learning, since the PCs acquired from depth sensors are always noise-corrupted. One option to circumvent this issue is to synthesize training data via computer graphics [57]. However, it still suffers from inaccurate modeling of noise and missing pixels of actual depth sensors. Different from the learning-based methods, by optimizing only a small set of parameters, our model-based method can easily adapt to new depth sensors with different specifications. Moreover, compared to complex deep models acting like black boxes, the derived filters of our methods are interpretable [58].

III. FORWARD MODEL AND VALIDATION

A. System Overview

We consider a depth-sensing system consisting of two depth sensors from different viewpoints, which are separated by a distance D.[4] For systems with two sensors (or one sensor placed at two locations at different time instants), there exists an overlapping *field of view* (FoV), where the same object surface is observed twice from two different viewpoints as illustrated in Fig. 3. The output of each depth sensor is a depth map of resolution $H \times N$. Each pixel is a noise-corrupted observation of the physical distance between the camera and the object, and is non-uniformly quantized to a finite B-bit

---

[4]The extension to the more general multiple-camera (more than two) case is straightforward. We omitted this discussion for brevity.
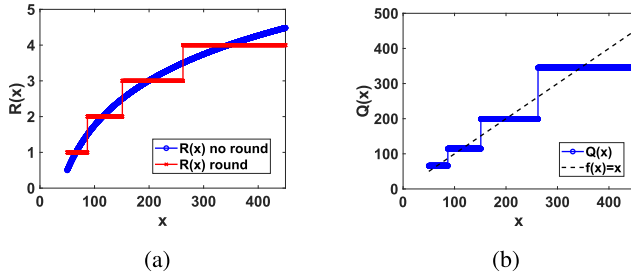
Fig. 4. Illustration of (a) quantization mapping $R(x)$ in (1) and (b) non-uniform quantization function $Q(x)$ in (3), with $\theta = 2$, $\rho = 1$, $x_{\min} = 50$, $x_{\max} = 450$.

representation. Without access to the underlying hardware directly, we assume that the depth map is the "rawest" signal we can acquire from the sensor. To simplify later discussion, we assume that two viewpoint depth maps are *rectified*; *i.e.*, pixel row h in the left view is capturing the same slice of the object as pixel row h in the right view. Rectification is a well-studied problem, and a known procedure [59] can be executed prior to our depth enhancement algorithm.

We now present the image formation model of a depth pixel. Our model consists of two parts: non-uniform quantization and signal-dependent noise.

B. Non-Uniform Quantization

Typically, a depth camera quantizes an input depth value into a finite B-bit representation. For example, Microsoft Kinect version 1.0 has a 11-bit per-pixel representation, while version 2.0 has a 13-bit representation. Such a quantization is common for depth sensors in the consumer market, where limited precision is allocated more for objects closer to the camera than ones further away.

To model non-uniform quantization, we define our non-uniform quantizer with wider quantization bins for larger input depth values.

Definition 1 (Quantization Mapping): Let $x \in [x_{\min}, x_{\max}]$ be the input value. We define the quantization mapping function $R(x)$ using the natural log and rounding functions as

$$R(x) \triangleq round\left[\phi \ln(\theta x + \rho) - R_0 + 0.5 \right], \quad (1)$$

where ϕ, θ and ρ are parameters, and $R_0 \triangleq \phi \ln(\theta x_{\min} + \rho)$. The rounding operation in (1) implies that the quantization mapping (QM) $R(x) \in \mathbb{Z}^+$ is piecewise constant.

The quantization mapping $R(x)$ maps the input depth value x with range $[x_{\min}, x_{\max})$ to one of 2^B discrete values:[5] $\{1, \ldots, 2^B\}$. To make sense of this definition, we note that $\theta > 0$ and $\theta x_{\min} + \rho \geq 1$. Since $\theta x + \rho \geq 1$, we use only the non-negative part of the log function in (1). The rounding operation also means that a smaller slope of function $\ln(\theta x + \rho)$ would lead to a coarser quantization of x; *i.e.*, a larger range of x would map to the same rounded integer $R(x)$. Thus, concavity of the log function means that larger x

has coarser quantization, as desired. See Fig. 4(a) for an illustration.

We note that the use of log function in (1) is similar to the μ-law companding algorithm in ITU-T's G.711 standard for PCM digital communication,[6] where non-uniform quantization is used for audio encoding.

The parameter ϕ is chosen so that $R(x) \in \{1, \ldots, 2^B\}$ can represent 2^B quantization bins given available B-bit representation per pixel. Specifically, ϕ is computed as

$$\phi = \frac{2^B}{\ln(\theta x_{\max} + \rho) - \ln(\theta x_{\min} + \rho)}. \quad (2)$$

In contrast, θ and ρ are fitting parameters to fit characteristics of a particular depth sensor (to be discussed in Section III-D). Fig. 4(a) shows $R(x)$ when $B = 2$, and there are $2^B = 4$ quantization bins, *i.e.*, $R(x) \in \{1, 2, 3, 4\}$.

Given quantization mapping $R(x)$ in (1), we can now define *quantization function $Q(x)$*.

Definition 2 (Quantization Function): We define the quantization function $Q(x)$ as

$$Q(x) = \frac{1}{\theta}\left(\exp\left\{ \frac{R(x) + R_0 - 0.5}{\phi} \right\} - \rho \right). \quad (3)$$

The quantization function in (3) essentially reverses the operations in quantization mapping $R(\cdot)$ in (1) to recover input x. Given that $R(x)$ is piecewise constant, $Q(x)$ is also piecewise constant, as shown in Fig. 4(b). We verify that $Q(x)$ is indeed a quantization function—the centers of each quantization step coincide with the identity function $f(x) = x$—while the quantization bin size is increasing with x as intended.

C. Signal-Dependent Noise Model

Before quantization $Q(\cdot)$, we assume that each measurement x is first corrupted by *signal-dependent* noise $n \sim \mathcal{N}(0, \sigma^2)$ following a zero-mean Gaussian distribution. Noise variance $\sigma^2 \in \mathbb{R}^+$ is dependent on signal x. Specifically, as done in [42] and [43], we assume that the *standard deviation* (SD) σ is a quadratic function of signal $\mathbf{x}$, *i.e.*,

$$\sigma = \alpha(x + \mu)^2 + \kappa, \quad \text{for } x \geq -\mu \quad (4)$$

where parameters $\alpha > 0$, $\mu < 0$ and $\kappa > 0$. It means that σ increases quadratically with signal magnitude.

D. Validation of the Model

To provide a concrete example of the quantization and noise model we just presented, we use the *Intel RealSense^{TM} D435* camera to validate this model and fit model parameters through an empirical study. Our chosen Intel sensor is representative of popular depth sensors in the market and shares similar characteristics. Although the working principles of various types of 3D sensors, *stereo, structured light*, and *time of flight (ToF)*, are slightly different, existing works [60], [61], [62] commonly assumed non-uniform quantization and [43], [63], [64] assumed signal-dependent noise in their depth sensor models. We stress, however, that we are

---

[5]A depth sensor typically specifies a range of depth sensitivity. For example, the range for MS Kinect 2.0 is 0.5m to 4.5m.

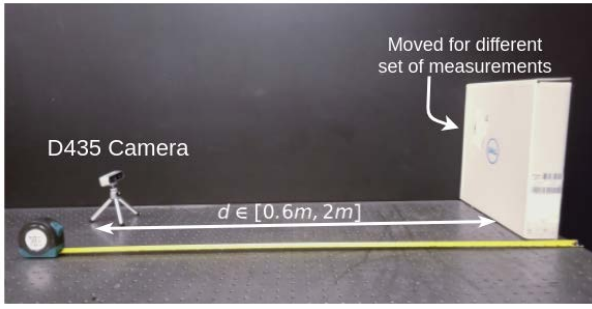[6]https://www.itu.int/rec/T-REC-G.711

Fig. 5. Experimental Setup for obtaining set of measurements for different depths. The *Intel RealSense^{TM} D435* camera is placed in front of the cardboard box and the distance between them is varied in steps of 100mm from 615mm to 1525mm.
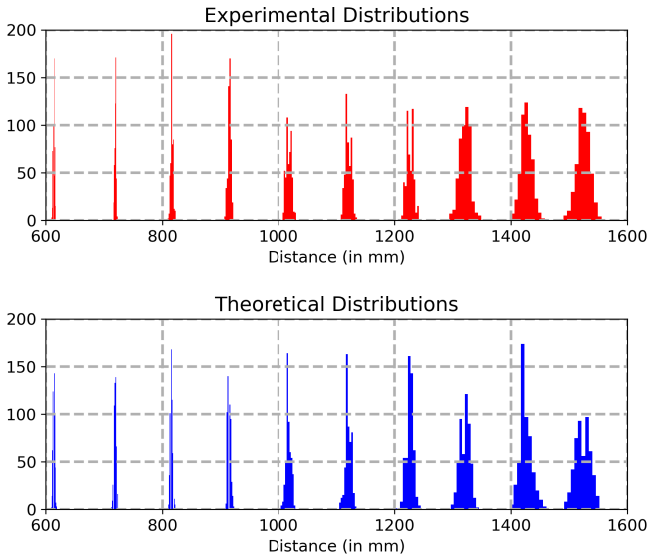

Fig. 6. Histogram of experimental data *(top row)* and corresponding histogram generated using image formation model described in Section III B) *(bottom row)*. For each set of experimental measurements obtained at a fixed distance (differentiated by colour) in the top row, the corresponding theoretical distribution is obtained by generating samples from equation (3) with mean $x_{l,i}^o$ and standard deviation $\sigma_{l,i}$ are chosen as the mean and variance of the corresponding samples from actual data. The variable quantization parameter $R(x)$, which ultimately determines the quantization output $Q(x)$, was set using trial and error to match the quantization bins for each set.

the first to combine *both* non-uniform quantization *and* signal-dependent noise into the same forward model.

Our experimental setup is shown in Fig. 5, where a cardboard box was placed at a measurable distance from the camera, while ensuring that both the camera and the box axes were aligned to each other. From each frame of the depth camera stream, five randomly selected measurements were collected from a 5×5 window around the center of the frame until 100 measurements for a single distance were obtained. This process was repeated for multiple clusters of measurements, with distances between the camera and the box ranging from 615mm to 1525mm in steps of approximately 100mm. The obtained measurements are shown as a histogram in the top half of Fig. 6.

1) Empirical Distribution: From the top half of Fig. 6, two observations can be made. First, the variance of measurements
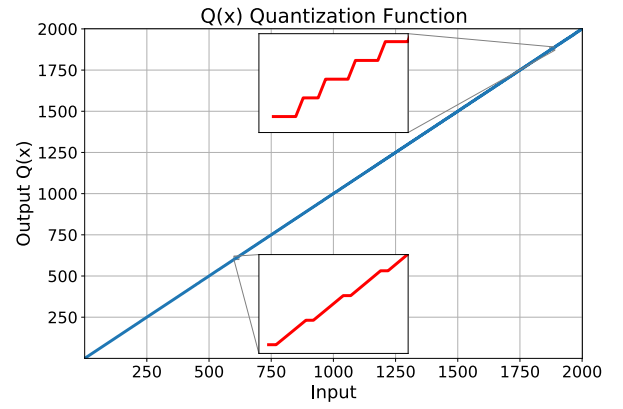

Fig. 7. Quantization Function $Q(x)$ used to generate second row of histograms in Figure 6. As expected, the quantization function stays close to the $y = x$ line but on zooming in, we notice it is a staircase function with size of the steps of the steps increasing as the argument of the function increases.

for a measurement cluster *increases* as distance increases. This suggests that (4) is a reasonable assumption, since the noise SD σ increases[7] with true distance x of a cluster. Second, the quantization step size *increases* as distance increases. Specifically, at 0.6m, measurements varied in step of 1mm, while at 1.5m, measurements varied in step of 3mm. This justifies the use of a non-uniform quantization function.

2) Theoretical Distribution: We synthesized the theoretical statistics using the image formation model described in (3) to match the corresponding experimental measurements. To generate these synthetic measurements, we estimated the ground truth to be equal to the mean of the corresponding experimental distribution. Then, zero-mean Gaussian noise of variance σ^2 was added to the estimated ground truth followed by quantization using function $Q(x)$. The SD of the noise σ varied quadratically as described in (4), and the parameters were set as $\alpha = 1 \times 10^{-5}$, $\mu = -528$ and $\kappa = 1.4$. Quantization mapping $R(x)$ as defined in (1) was determined similarly to qualitatively match the bins of experimental distributions. After data fitting, the parameters were found to be $\phi = 500, \theta = 500, \rho = 200, x_{\min} = 10$. Given $R(x)$, the corresponding $Q(x)$ was determined and was plotted in Fig. 7. The figure shows that the function remains close to the $y = x$ line but on closer inspection, it is a staircase function with quantization bins of increasing size.

The lower half of Fig. 6 shows the histogram of the synthetic measurements. Based on the similarity between the top and bottom halves of Fig. 6, we can conclude that the image formation model described in previous subsection provides a good fit for measurements taken from a real depth sensor.

E. Parameter Estimation for Noise Variance

We next discuss how we compute noise variance σ^2 given observed depth measurements in a cluster in Fig. 6. Denote by $\mathcal{Y} = \{y_1, y_2, \ldots, y_M\}$ a set of noise-corrupted and quantized observations of ground-truth depth x^*. For each observation

---

[7]Note that the variance of measurements and noise variance σ^2 are fundamentally two different quantities, though the former is determined by the latter, and thus are positively correlated.

$y_i \in \mathbb{R}^+$, $R(y_i) \in \mathbb{Z}^+$ is the corresponding quantization mapping value. Thus, $x^* + n_i$ must reside in the quantization bin indexed by $R(y_i)$, *i.e.*,

$$R(y_i) - \frac{1}{2} \le \phi \ln(\theta(x^* + n_i) + \rho) - R_0 + \frac{1}{2} < R(y_i) + \frac{1}{2}. \quad (5)$$

Thus, we can derive that additive noise n_i is lower- and upper-bounded as follows:

$$n^-(y_i, x^*) \le n_i < n^+(y_i, x^*),$$
$$n^-(y_i, x^*) \triangleq \underbrace{\theta^{-1}\left(\exp\left(\frac{R(y_i) + R_0 - 1}{\phi}\right) - \rho\right)}_{z_i^-} - x^*,$$
$$n^+(y_i, x^*) \triangleq \underbrace{\theta^{-1}\left(\exp\left(\frac{R(y_i) + R_0}{\phi}\right) - \rho\right)}_{z_i^+} - x^*. \quad (6)$$

Note that z_i^- and z_i^+ are both function of y_i.

Since the y_i's in $\mathcal{Y}$ are independent, we formulate a *maximum likelihood estimation* (MLE) problem for variance σ^2:

$$\max_{\sigma^2} \Pr(\mathcal{Y}|\sigma^2) = \prod_{i=1}^{M} \Pr(y_i|\sigma^2)$$
$$= \prod_{i=1}^{M} \Pr\left(n^-(y_i, x^*) \le n_i < n^+(y_i, x^*)\big|\sigma^2\right). \quad (7)$$

Since $n_i \sim \mathcal{N}(0, \sigma^2)$, we can write

$$\Pr(\mathcal{Y}|\sigma^2) = \prod_{i=1}^{M} \int_{\mathcal{R}_i} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{n_i^2}{2\sigma^2}\right) d\,n_i \triangleq p(\sigma) \quad (8)$$

where the region of integration $\mathcal{R}_i$ is

$$\mathcal{R}_i = \left\{n_i \mid n^-(y_i, x^*) \le n_i < n^+(y_i, x^*)\right\}. \quad (9)$$

The probability $p(\sigma)$ in (8) is difficult to maximize over σ for two reasons. First, each term in the product is of the form $\int_{\mathcal{R}_i} e^{-n_i^2/2\sigma^2} dn_i$, which has no closed-form expression. Second, (8) involves a product of M terms, each integrating over a different region $\mathcal{R}_i$. Thus, we propose the following fast search procedure to find a near-optimal σ.

For simplicity, first we assume there is only one observation, *i.e.*, $M = 1$. Then the optimization problem becomes

$$\sigma_1^* = \arg\max_{\sigma} \left(\Phi\left(n^+(y_1, x^*)\right) - \Phi\left(n^-(y_1, x^*)\right)\right)$$
$$\approx \arg\max_{\sigma} \left(\left(\frac{n^-(y_1, x^*)\sigma}{(n^-(y_1, x^*))^2 + \sigma^2}\right) \exp\left(-\frac{(n^-(y_1, x^*))^2}{2\sigma^2}\right)\right.$$
$$\left. - \left(\frac{n^+(y_1, x^*)\sigma}{(n^+(y_1, x^*))^2 + \sigma^2}\right) \exp\left(-\frac{(n^+(y_1, x^*))^2}{2\sigma^2}\right)\right)$$
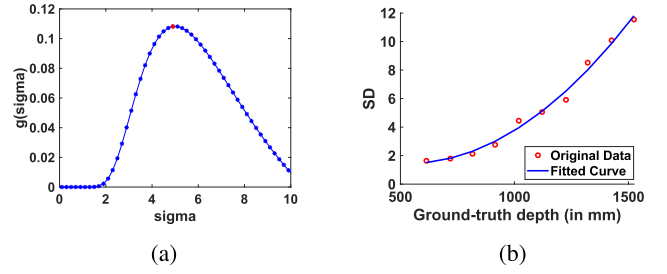$$\triangleq \tilde{g}(\sigma) \quad (10)$$

Fig. 8. Illustration of (a) unimodal objective $\tilde{g}(\sigma)$ in (10) with a single local maximum at σ_1^* and (b) parameterization of σ in (4).

where $\Phi(x)$ is the *cumulative distribution function* (CDF) of n_1, and in (10) we use the approximation of $\Phi(x)$ in [65], *i.e.*,

$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{n_1^2}{2\sigma^2}\right) dn_1$$
$$\approx 1 - \frac{1}{\sqrt{2\pi}} \frac{x\sigma}{x^2 + \sigma^2} \exp\left(-\frac{x^2}{2\sigma^2}\right). \quad (11)$$

One can show that $\tilde{g}(\sigma)$ in (10) is a unimodal objective with a single local maximum at σ_1^*, as shown in Fig. 8(a). This means that σ_1^* can be computed using numerical methods like the Newton's method [18] to locate σ_1^* where $\tilde{g}'(\sigma_1^*) = 0$.

More generally, when there are $M > 1$ observations, the approximate objective $\tilde{g}(\sigma)$ becomes:

$$\tilde{g}(\sigma) = \prod_{i=1}^{M} \left(\left(\frac{n^-(y_i, x^*)\sigma}{(n^-(y_i, x^*))^2 + \sigma^2}\right) \exp\left(-\frac{(n^-(y_i, x^*))^2}{2\sigma^2}\right)\right.$$
$$\left. - \left(\frac{n^+(y_i, x^*)\sigma}{(n^+(y_i, x^*))^2 + \sigma^2}\right) \exp\left(-\frac{(n^+(y_i, x^*))^2}{2\sigma^2}\right)\right). \quad (12)$$

The optimal σ^* in this general $M > 1$ case is lower- and upper-bounded as follows

$$\min_{i \in \{1, \dots, M\}} (\sigma_i^*) \le \sigma^* \le \max_{i \in \{1, \dots, M\}} (\sigma_i^*). \quad (13)$$

Thus, we can first compute the lower/upper bounds, then use the *Golden-section* (GS) search [18] to find σ^*. GS finds the optimal value σ^* that maximizes/minimizes $\tilde{g}(\sigma^*)$ via an iterative search, given $\tilde{g}(\sigma)$ is unimodal.[8] See [18] for details.

Finally, given computed σ^*'s for different clusters with corresponding ground-truth depth x^*'s, we can parameterize $\alpha = 1 \times 10^{-5}$, $\mu = -528$ and $\kappa = 1.4$ in (4) via nonlinear least-squares algorithm [66] as shown in Fig. 8(b).

IV. APPROXIMATION AND OPTIMIZATION

The image formation model in Section III-B is difficult to optimize directly; we first present a practical approximation. We then introduce a mapping function from left-view pixels to right-view pixels assuming FoV overlapped. We next define a likelihood term and a signal prior for individual pixel rows in depth images. Finally, we formulate a MAP optimization problem and derive a corresponding algorithm based on AGD to reconstruct a target pixel row.

[8]Though in general the multiplication of unimodal functions is not necessarily unimodal, in our experiments we found this to be the case.
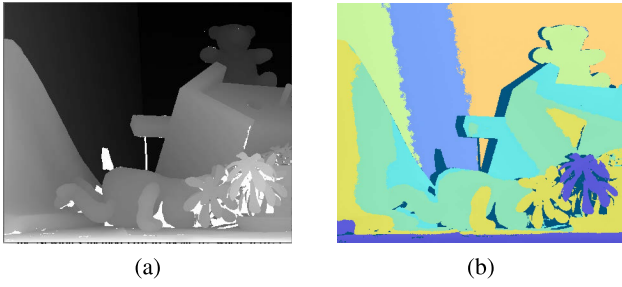
Fig. 9. Example of depth map segmentation: (a) the ground-truth disparity map, (b) the segmented eight layers of pre-filtered depth map, each of approximately the same depth with the same noise variance.

A. Approximation of Image Formation Model

Additive noise in our image formation model, as discussed in Section III-C, is signal-dependent, which is difficult to address directly. Thus, as a pre-processing step, we first segment a depth image into non-overlapping *layers*, each of approximately the same depth, and then assume the same fixed noise variance in each layer. This constitutes a reasonable approximation in practice, as a depth image is typically composed of a static indoor background plus one or more foreground objects, each with roughly the same distance to the depth sensor.

To achieve robust image segmentation, we first segment the corrupted depth image into layers using the k-means algorithm, where the optimal number of layers is determined by the Elbow method [67]. We then pre-filter it using *bilateral filter* (BF) [27] layer by layer, where for each layer we compute the SD of the layer for the range filter parameter, while fixing the domain filter parameter at 3 for all layers. Finally, we perform segmentation again on the pre-filtered depth map. See Fig. 9 for an example.

After segmentation, for each layer ζ we compute an average depth $\bar{x}_\zeta$ given observed depth pixels of the layer, then use (4) to compute a noise variance $\bar{\sigma}_\zeta^2$ for the layer. In the sequel, we will assume a constant and pre-computed noise variance $\bar{\sigma}_\zeta^2$ for each depth layer ζ.

B. View-to-View Mapping

Consider two depth images of adjacent viewpoints with FoV overlapped, as shown in Fig. 3. Pixels in the left and right rectified images corresponding to overlapping FoV are projections of the same object surface onto two different camera planes, and thus are related. We optimize a left pixel row of a layer exploiting this inter-view redundancy as follows. (Optimization of a right pixel row is similar and thus omitted.) Specifically, denote by $\mathbf{x}_l$ a row with N available pixels in a layer of the left view, and by $\mathbf{x}_r$ a corresponding *sub-row* of M pixels, where $M \leq N$, capturing the overlapping spatial region. For simplicity, we assume that there is no occlusion in the overlapping region between $\mathbf{x}_l$ and $\mathbf{x}_r$.

Given that the two depth images are rectified, we employ a known 1D warping procedure [20] to relate $\mathbf{x}_l$ and $\mathbf{x}_r$. For the i-th pixel in the left view, $x_{l,i}$, its (non-integer) horizontal
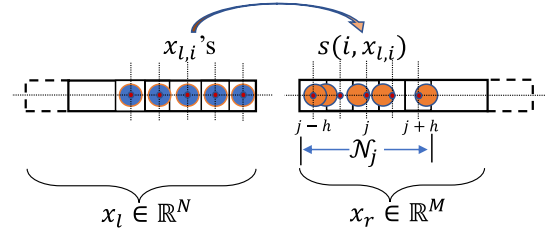

Fig. 10. Illustration of view-to-view mapping: using left pixels $x_{l,i}$'s with mapped (non-integer) positions $s(i, x_{l,i})$ falling within right pixel $x_{r,j}$'s *neighborhood* $\mathcal{N}_j \triangleq (j - h, j + h)$ to interpolate right pixel $x_{r,j}$.

position $s(i, x_{l,i}) \in \mathbb{R}$ in the right view after projection is

$$s(i, x_{l,i}) = i - \delta(x_{l,i}),$$
$$\delta(x_{l,i}) = \frac{f D}{x_{l,i}} \tag{14}$$

where $\delta(x_{l,i}) \in \mathbb{R}$ is the *disparity* of $x_{l,i}$, $f \in \mathbb{R}^+$ is the camera focal length, and $D \in \mathbb{R}^+$ is the baseline between the two capturing cameras. Note that $s(\cdot)$ is a function of *both* left pixel's integer horizontal position i *and* depth value $x_{l,i}$. Here, we use the pre-filtered depth values $\hat{x}_{l,i}$ to compute reliable disparities $\delta(\hat{x}_{l,i})$ for view-to-view mapping.

1) Sparse BF Interpolation: Assuming that the object surface is smooth, we interpolate each right pixel $x_{r,j}$ in sub-row $\mathbf{x}_r$ as a *sparse* weighted combination of left pixels $x_{l,i}$'s; *i.e.*, each $x_{r,j}$ is linearly interpolated using $x_{l,i}$'s with mapped positions $s(i, x_{l,i})$ falling within $x_{r,j}$'s *neighborhood* $\mathcal{N}_j \triangleq (j - h, j + h)$, where $h \in \mathbb{Z}^+$ is a parameter that determines neighborhood size. See Fig. 10 for an illustration, where a left pixel $x_{l,i}$ maps to a location within neighborhood $\mathcal{N}_j$ of right pixel $x_{r,j}$, and thus will be used for linear interpolation of $x_{r,j}$.

We use linear interpolation for two practical reasons. First, it has been commonly used in the multiview-plus-depth imaging literature and shown to obtain high quality rendering [68], [69]. Second, linear interpolation will ease computation of our optimization problem, to be formulated in Section IV-E.

In matrix form, we write the interpolation as

$$\mathbf{x}_r = \mathbf{W}(\mathbf{x}_l) \mathbf{x}_l = \mathbf{g}(\mathbf{x}_l) \tag{15}$$

where $\mathbf{W}(\mathbf{x}_l) \in \mathbb{R}^{M \times N}$ is a sparse weight matrix. We model the weight w_{ij} between right pixel $x_{r,j}$ and left pixel $x_{l,i}$ using a Gaussian kernel based on the spatial proximity between the projected position $s(i, x_{l,i})$ of left pixel i and the target right pixel position j in $x_{r,j}$, *i.e.*,

$$w_{ij} = \frac{1}{\bar{w}_j} \exp\left(-\frac{(s(i, x_{l,i}) - j)^2}{\sigma_s^2}\right), \tag{16}$$

$$\bar{w}_j = \sum_{m \mid s(m, x_{l,m}) \in \mathcal{N}_j} \exp\left(-\frac{(s(m, x_{l,m}) - j)^2}{\sigma_s^2}\right) \tag{17}$$

where $\bar{w}_j$ is a normalization term so that $\sum_i w_{ij} = 1$. Gaussian kernel is commonly used in image filtering schemes such as BF [27]. In words, weight w_{ij} is large if the distance between location $s(i, x_{l,i})$ of mapped left pixel $x_{l,i}$ and location j of target right pixel $x_{r,j}$ is small. To simplify (16),
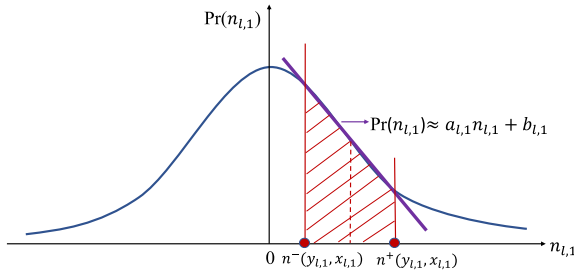
Fig. 11. Linear approximation of the Gaussian pdf.

we assume $\bar{w}_j$ is a constant. Combining with (14), (16) is rewritten as

$$w_{ij} = \bar{w}_j^{-1} \exp\left(-\frac{(i - fDx_{l,i}^{-1} - j)^2}{\sigma_s^2}\right). \qquad (18)$$

Given (18), the interpolation of $\mathbf{g}(\mathbf{x}_l)$ in (15) is differentiable w.r.t. $\mathbf{x}_l$. We use the first-order Taylor series expansion around $\hat{\mathbf{x}}_l$, the pre-filtered estimate of $\mathbf{x}_l$, to obtain a linear approximation. Thus,

$$\begin{aligned}\mathbf{g}(\mathbf{x}_l) &\approx \mathbf{g}(\hat{\mathbf{x}}_l) + \mathbf{g}'(\hat{\mathbf{x}}_l)(\mathbf{x}_l - \hat{\mathbf{x}}_l) \\ &= \mathbf{g}'(\hat{\mathbf{x}}_l)\mathbf{x}_l + \mathbf{g}(\hat{\mathbf{x}}_l) - \mathbf{g}'(\hat{\mathbf{x}}_l)\hat{\mathbf{x}}_l \\ &= \mathbf{H}\mathbf{x}_l + \mathbf{e} \end{aligned} \qquad (19)$$

where $\mathbf{H} = \mathbf{g}'(\hat{\mathbf{x}}_l) = \left[\frac{\partial g_j(\hat{\mathbf{x}}_l)}{\partial x_{l,i}}\right]_{j,i} \in \mathbb{R}^{M \times N}$ is the *Jacobian matrix* (first-order partial derivatives) of $\mathbf{g}(\mathbf{x}_l)$ at $\mathbf{x}_l = \hat{\mathbf{x}}_l$, and $\mathbf{e} = \mathbf{g}(\hat{\mathbf{x}}_l) - \mathbf{g}'(\hat{\mathbf{x}}_l)\hat{\mathbf{x}}_l$ is a constant vector in $\mathbb{R}^M$. Note that row j of $\mathbf{H}$ contains only non-zero entries $H_{j,i}$ corresponding to left pixels $x_{l,i}$ that map to positions $s(i, x_{l,i})$ within right pixel $x_{r,j}$'s neighborhood $\mathcal{N}_j$. Thus, $\mathbf{H}$ is sparse, where sparsity depends on the number of left pixels $x_{l,i}$ mapping to $\mathcal{N}_j$.

C. Likelihood Term

Since every entry of the zero-mean additive noise $\mathbf{n}_l \in \mathbb{R}^N$ follows an independent Gaussian distribution, the probability density function (pdf) of $\mathbf{n}_l$ is

$$\Pr(\mathbf{n}_l) = \prod_{i=1}^{N} \Pr(n_{l,i}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\,\sigma_{l,i}} \exp\left(-\frac{n_{l,i}^2}{2\sigma_{l,i}^2}\right) \qquad (20)$$

where $\sigma_{l,i}^2$ is the noise variance for left pixel i given that it belongs to a known layer ζ during image segmentation, as discussed in Section IV-A. Recall that $\sigma_{l,i}$ is computed via (4), using the average depth $\bar{x}_\zeta$ of layer ζ.

Given observation $\mathbf{y}_l$, the likelihood term $\Pr(\mathbf{y}_l|\mathbf{x}_l)$ is

$$\Pr(\mathbf{y}_l|\mathbf{x}_l) = \prod_{i=1}^{N} \int_{\mathcal{R}_{l,i}} \Pr(n_{l,i})\, d\,n_{l,i} \qquad (21)$$

where the region of integration $\mathcal{R}_{l,i}$ is

$$\mathcal{R}_{l,i} = \left\{n_{l,i} \mid n^-(y_{l,i}, x_{l,i}) \leq n_{l,i} < n^+(y_{l,i}, x_{l,i})\right\}. \qquad (22)$$

The lower and upper bounds, $n^-(y_{l,i}, x_{l,i}) = z_{l,i}^- - x_{l,i}$ and $n^+(y_{l,i}, x_{l,i}) = z_{l,i}^+ - x_{l,i}$, are defined in (6).

Each integration in (21) is over a Gaussian pdf, which has no closed-form expression. For ease of later optimization, we approximate each Gaussian function $\Pr(n_{l,i})$ as a linear function over the region of integration $\mathcal{R}_{l,i} = \left[n^-(y_{l,i}, x_{l,i}), n^+(y_{l,i}, x_{l,i})\right]$, i.e.,

$$\Pr(n_{l,i}) \approx a_{l,i} n_{l,i} + b_{l,i}, \quad n_{l,i} \in \mathcal{R}_{l,i}. \qquad (23)$$

$a_{l,i}$ and $b_{l,i}$ are constant scalars computed using Taylor series expansion at the initial estimate $n_{l,i}^0 = y_{l,i} - \tilde{x}_{l,i}$, where the recovered $\tilde{x}_{l,i}$ is updated throughout the algorithm iterations (to be discussed in detail later). See Fig. 11 for an illustration.

The linear approximation (23) is good if i) the integration region $\mathcal{R}_i$ is narrow, or ii) the Gaussian function $\Pr(n_{l,i})$ is flat. When the captured depth pixel $x_{l,i}$ is near, the corresponding quantization bin is small, and hence $\mathcal{R}_i$ is narrow. On the other hand, when $x_{l,i}$ is far, the noise variance $\sigma_{l,i}^2$ is large, and hence $\Pr(n_{l,i})$ is flat. As we will demonstrate in Section VI, in either case the linear approximation is sufficiently accurate.

Denote by $\mathbf{1}_i$ a suitably long *canonical vector* of all zeros except entry i is 1. We now rewrite (21) as

$$\begin{aligned}\Pr(\mathbf{y}_l|\mathbf{x}_l) &\approx \prod_{i=1}^{N} \int_{\mathcal{R}_{l,i}} \left(a_{l,i}n_{l,i} + b_{l,i}\right) d\,n_{l,i} \\ &= \prod_{i=1}^{N} (\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i}) \end{aligned} \qquad (24)$$

where $\tilde{\mathbf{a}}_{l,i}^\top = a_{l,i}(z_{l,i}^- - z_{l,i}^+)\mathbf{1}_i^\top$ and $\tilde{b}_{l,i} = \frac{a_{l,i}}{2}((z_{l,i}^+)^2 - (z_{l,i}^-)^2) + b_{l,i}(z_{l,i}^+ - z_{l,i}^-)$. (24) is proven in the Appendix.

Similarly, for noise $\mathbf{n}_r \in \mathbb{R}^M$, we can write

$$\begin{aligned}\Pr(\mathbf{y}_r|\mathbf{x}_l) &\approx \prod_{j=1}^{M} \int_{\mathcal{R}_{r,j}} \left(a_{r,j}n_{r,j} + b_{r,j}\right) d\,n_{r,j} \\ &\approx \prod_{j=1}^{M} (\bar{\mathbf{a}}_{r,j}^\top \mathbf{H}\mathbf{x}_l + \bar{b}_{r,j}) \end{aligned} \qquad (25)$$

where $\bar{\mathbf{a}}_{r,j}^\top = a_{r,j}(z_{r,j}^- - z_{r,j}^+)\mathbf{1}_j^\top$ and $\bar{b}_{r,j} = \bar{\mathbf{a}}_{r,j}^\top \mathbf{e} + \frac{a_{r,j}}{2}((z_{r,j}^+)^2 - (z_{r,j}^-)^2) + b_{r,j}(z_{r,j}^+ - z_{r,j}^-)$. (25) is proven in the Appendix.

D. Signal Prior

As done in graph-based image processing work [70], [71], [72], we model the similarities of pixel pairs in $\mathbf{x}_l$ using a graph Laplacian matrix $\mathbf{L}_l$, and thus prior $\Pr(\mathbf{x}_l)$ can be written as

$$\Pr(\mathbf{x}_l) = \exp\left(-\frac{\mathbf{x}_l^\top \mathbf{L}_l \mathbf{x}_l}{\sigma_p^2}\right) \qquad (26)$$

where $\sigma_p > 0$ is a weight parameter that is a function of noise variance. σ_p determines the importance of prior $\Pr(\mathbf{x}_l)$ in a MAP formulation. In particular, we define σ_p as

$$\sigma_p^2 = \frac{1}{g_1 \bar{\sigma}_\zeta^2 + g_2} \qquad (27)$$

where $g_1 > 0$ and g_2 are empirically fitted constants for a dataset. Recall that $\bar{\sigma}_\zeta^2$ is the noise variance for the layer ζ in which pixels in $\mathbf{x}_l$ reside. In words, (27) states that a larger

noise variance $\bar{\sigma}_\zeta^2$ leads to a smaller σ_p^2, making prior $\Pr(\mathbf{x}_l)$ more important in a noisy setting.

We assume that the previous K pixel rows in the left depth image have been enhanced first. Assuming in addition that the next row i follows a similar image structure, $\mathbf{L}_l$ can be learned from the previous K rows. See Section V for details.

E. MAP Formulation

We now formulate a MAP problem for $\widetilde{\mathbf{x}}_l$ as follows.

$$\max_{\mathbf{x}_l} \ \Pr(\mathbf{y}_l, \mathbf{y}_r | \mathbf{x}_l, \mathbf{x}_r) \, \Pr(\mathbf{x}_l, \mathbf{x}_r) \tag{28}$$

$$= \Pr(\mathbf{y}_l, \mathbf{y}_r | \mathbf{x}_l, \mathbf{g}(\mathbf{x}_l)) \, \Pr(\mathbf{x}_l, \mathbf{g}(\mathbf{x}_l)) \tag{29}$$

$$= \Pr(\mathbf{y}_l | \mathbf{x}_l) \Pr(\mathbf{y}_r | \mathbf{g}(\mathbf{x}_l)) \, \Pr(\mathbf{x}_l) \Pr(\mathbf{g}(\mathbf{x}_l)) \tag{30}$$

$$\approx \prod_{i=1}^{N} (\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i}) \prod_{j=1}^{M} (\bar{\mathbf{a}}_{r,j}^\top \mathbf{H} \mathbf{x}_l + \bar{b}_{r,j})$$

$$\times \exp\left(-\frac{\mathbf{x}_l^\top \mathbf{L}_l \mathbf{x}_l}{\sigma_p^2}\right) \exp\left(-\frac{\mathbf{g}(\mathbf{x}_l)^\top \mathbf{L}_r \mathbf{g}(\mathbf{x}_l)}{\sigma_p^2}\right) \tag{31}$$

where in (29) we substituted $\mathbf{g}(\mathbf{x}_l)$ for $\mathbf{x}_r$, and in (30) we split up the first term since left and right noise, $\mathbf{n}_l$ and $\mathbf{n}_r$, are independent. Note that parameters $\tilde{\mathbf{a}}_{l,i}$, $\tilde{b}_{l,i}$, $\bar{\mathbf{a}}_{r,j}$ and $\bar{b}_{r,j}$ are all computed from Taylor series expansion using the updated $\tilde{\mathbf{x}}_l$ in each algorithm iteration.

To ease optimization, we minimize the negative log of (31):

$$\min_{\mathbf{x}_l} -\ln \prod_{i=1}^{N} (\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i}) - \ln \prod_{j=1}^{M} (\bar{\mathbf{a}}_{r,j}^\top \mathbf{H} \mathbf{x}_l + \bar{b}_{r,j})$$

$$+ \mathbf{x}_l^\top \mathbf{L}_l \mathbf{x}_l + \mathbf{g}(\mathbf{x}_l)^\top \mathbf{L}_r \mathbf{g}(\mathbf{x}_l)$$

$$= -\sum_{i=1}^{N} \ln(\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i}) - \sum_{j=1}^{M} \ln(\bar{\mathbf{a}}_{r,j}^\top \mathbf{H} \mathbf{x}_l + \bar{b}_{l,i})$$

$$+ \frac{1}{\sigma_p^2} \left(\mathbf{x}_l^\top \mathbf{L}_l \mathbf{x}_l + \mathbf{x}_l^\top \mathbf{H}^\top \mathbf{L}_r \mathbf{H} \mathbf{x}_l + 2 \, \mathbf{e}^\top \mathbf{L}_r \mathbf{H} \mathbf{x}_l \right). \tag{32}$$

(32) is an unconstrained convex and differentiable objective, but has no closed-form solution. Thus, we solve for its minimum efficiently using AGD [21], [22]. AGD is an extension of gradient descent (GD) that provably achieves a convergence rate of $1/t^2$ after t steps in the convex scenario. To execute AGD efficiently, selecting an appropriate step size is important. We next overview AGD and our choice of step size in our optimization.

F. Optimization of $\mathbf{x}_l$

For notation simplicity, we rewrite the objective in (32) as

$$\min_{\mathbf{x}_l} f(\mathbf{x}_l) = -\sum_{i=1}^{N} \ln(\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i}) - \sum_{j=1}^{M} \ln(\bar{\mathbf{a}}_{r,j}^\top \mathbf{H} \mathbf{x}_l + \bar{b}_{r,j})$$

$$+ \frac{1}{\sigma_p^2} \left(\mathbf{x}_l^\top \mathcal{L} \mathbf{x}_l + 2 \mathbf{h}^\top \mathbf{x}_l \right) \tag{33}$$

where $\mathcal{L} = \mathbf{L}_l + \mathbf{H}^\top \mathbf{L}_r \mathbf{H}$ and $\mathbf{h}^\top = \mathbf{e}^\top \mathbf{L}_r \mathbf{H}$.

Algorithm 1 Accelerated Gradient Descent

Input: Convergence parameter ϵ, smooth parameter β.

1: Initialize $\mathbf{c}_l^1 \leftarrow \mathbf{x}_l^1$, $t \leftarrow 1$ and $\eta^0 \leftarrow 0$;
2: **while** $\|\nabla f(\mathbf{x}_l^t)\|^2 \geq \epsilon$ **do**
3: $\quad \mathbf{c}_l^{t+1} \leftarrow \mathbf{x}_l^t - \frac{1}{\beta} \nabla f(\mathbf{x}_l^t)$,
4: $\quad \eta^t \leftarrow \frac{1 + \sqrt{1 + 4(\eta^{t-1})^2}}{2}$
5: $\quad \gamma^t \leftarrow \frac{1 - \eta^t}{\eta^{t+1}}$
6: $\quad \mathbf{x}_l^{t+1} \leftarrow (1 - \gamma^t) \, \mathbf{c}_l^{t+1} + \gamma^t \mathbf{c}_l^t$
7: $\quad t \leftarrow t + 1$.
8: **endwhile**

Output: $\mathbf{x}_l^{t+1}$.

We summarize AGD in Algorithm 1. Suppose objective $f(\mathbf{x}_l)$ is β-*smooth* (i.e., β-Lipschitz),[9] with gradient denoted by $\nabla f(\mathbf{x}_l)$. From (33), $\nabla f(\mathbf{x}_l)$ is

$$\nabla f(\mathbf{x}_l) = -\sum_{i=1}^{N} \frac{\tilde{\mathbf{a}}_{l,i}}{\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i}} - \sum_{j=1}^{M} \frac{\mathbf{H}^\top \bar{\mathbf{a}}_{r,j}}{\bar{\mathbf{a}}_{r,j}^\top \mathbf{H} \mathbf{x}_l + \bar{b}_{r,j}}$$

$$+ \frac{2}{\sigma_p^2} (\mathcal{L} \mathbf{x}_l + \mathbf{h}). \tag{34}$$

At each iteration t, AGD first takes a greedy step in the negative gradient direction $-\nabla f(\mathbf{x}_l^t)$ from previous solution $\mathbf{x}_l^t$ to $\mathbf{c}_l^{t+1}$ with step size $1/\beta$. Then, new solution $\mathbf{x}_l^{t+1}$ is a convex combination of $\mathbf{c}_l^{t+1}$ and previously computed $\mathbf{c}_l^t$ in iteration $t - 1$, given parameter γ^t determined by AGD.

The only remaining question is how to best estimate smoothness β of $f(\mathbf{x}_l)$ for step size $1/\beta$. As discussed in [22], β is the upper bound of the largest eigenvalue of the Hessian matrix $\mathbf{\Psi}$ of $f(\mathbf{x}_l)$. Thus, we first write the Hessian matrix $\nabla^2 f(\mathbf{x}_l)$ of $f(\mathbf{x}_l)$ as

$$\nabla^2 f(\mathbf{x}_l) = \sum_{i=1}^{N} \frac{\tilde{\mathbf{a}}_{l,i} \tilde{\mathbf{a}}_{l,i}^\top}{(\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i})^2} + \sum_{j=1}^{M} \frac{\mathbf{H}^\top \bar{\mathbf{a}}_{r,j} \bar{\mathbf{a}}_{r,j}^\top \mathbf{H}}{(\bar{\mathbf{a}}_{r,j}^\top \mathbf{H} \mathbf{x}_l + \bar{b}_{r,j})^2}$$

$$+ \frac{2}{\sigma_p^2} \mathcal{L}. \tag{35}$$

Recall that $\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i}, \forall i$ are linear functions in (24) that approximate Gaussian integrals over quantization bins. In contrast, matrix $\tilde{\mathbf{a}}_{l,i} \tilde{\mathbf{a}}_{l,i}^\top$ in the numerator of the first summation has a single non-zero entry proportional to $(z_{l,i}^+ - z_{l,i}^-)^2$, which is small when compared to $(\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i})^2$. (A similar argument can be made for matrix $\tilde{\mathbf{a}}_{r,j} \tilde{\mathbf{a}}_{r,j}^\top$ of the second summation.) Hence, the two summations in (35) are small compared to $(2/\sigma_p^2)\mathcal{L}$, and we can approximate $\nabla^2 f(\mathbf{x}_l) \approx (2/\sigma_p^2)\mathcal{L}$.

Instead of using computation-expensive eigen-decomposition, we consider *Gershgorin Circle Theorem* (GCT) [23] to compute an upper bound of the largest eigenvalue $\lambda_{\max}$ of $\mathcal{L}$. By GCT, each eigenvalue λ of $\mathbf{\Psi}$ resides in at least one *Gershgorin disc* corresponding to row i of $\mathcal{L}$, with center $o_i = \mathcal{L}_{i,i}$ and radius $r_i = \sum_{j \neq i} |\mathcal{L}_{i,j}|$.

---

[9]Though log functions are used in (33), each argument is an approximation of Gaussian density integral over a quantization bin in (24), and thus is sufficiently larger than 0. Hence, the log function slopes are upper-bounded.

Thus, $\lambda_{\max}$ must satisfy

$$\lambda_{\max} \leq \max_i (o_i + r_i)$$

$$= \max_i \left(\mathcal{L}_{i,i} + \sum_{j \neq i} |\mathcal{L}_{i,j}| \right) \triangleq \lambda_{\max}^+(\mathcal{L}). \quad (36)$$

$\lambda_{\max}^+(\mathcal{L})$ can be computed efficiently given $\mathcal{L}$ is a sparse matrix. Thus, we can finally compute $\tilde{\beta} \triangleq (2/\sigma_p^2)\lambda_{\max}^+(\mathcal{L})$.

G. Computation Complexity Analysis

We analyze the computation complexity of our algorithm. Assuming metric matrix $\mathbf{M}$ is optimized via feature graph learning (see Section V for details), we compute sparse symmetric graph Laplacian matrices $\mathbf{L}_l \in \mathbb{R}^{N \times N}$ and $\mathbf{L}_r \in \mathbb{R}^{M \times M}$, where $M \leq N$, with non-zero entries (i, j) iff $i-2 \leq j \leq i+2$. From Section IV-B, we know $\mathbf{H} \in \mathbb{R}^{M \times N}$ is also sparse, with non-zero entries (j, i) iff $s(i, x_{l,i}) \in \mathcal{N}_j$. Thus, assuming $h \geq 2$, matrix product $\mathbf{L}_r \mathbf{H}$ has non-zero entries (i, j) iff $s(i, x_{l,i}) \in \bigcup_{k=j-2}^{j+2} \mathcal{N}_k = \mathcal{N}_j^+$. Since each i of N left pixels is mapped to $\mathcal{O}(1)$ neighborhoods $\mathcal{H}_j^+$ of right pixels j, $\mathbf{L}_r \mathbf{H}$ is sparse and has $\mathcal{O}(N)$ non-zero entries. Similarly, we can conclude $\mathbf{H}^\top \mathbf{L}_r \mathbf{H}$ is also sparse and contains $\mathcal{O}(N)$ non-zero entries.

Thus, $\mathcal{L} = \mathbf{L}_l + \mathbf{H}^\top \mathbf{L}_r \mathbf{H}$ is also sparse and contains $\mathcal{O}(N)$ non-zero entries, computed in $\mathcal{O}(N)$. Similarly, $\mathbf{h}^\top = \mathbf{e}^\top \mathbf{L}_r \mathbf{H}$ contains $\mathcal{O}(N)$ non-zero entries. To compute $\lambda_{\max}^+(\mathcal{L})$ in (36), we compute all candidates i, which requires accessing each non-zero entry in $\mathcal{L}$ exactly once, and thus complexity is $\mathcal{O}(N)$. Thus, computing $\tilde{\beta}$ is also $\mathcal{O}(N)$.

Given $\mathcal{L}$, $\mathbf{h}$ and $\tilde{\beta}$, we execute AGD iteratively. In each iteration, we calculate gradient $\nabla f(\mathbf{x}_l)$ in (34). Recalling that $\bar{\mathbf{a}}_{r,j}$ has only one single non-zero entry and $\mathbf{H}$ is sparse, the cost of computing $\mathbf{H}^\top \bar{\mathbf{a}}_{r,j}$ and $\bar{\mathbf{a}}_{r,j}^\top \mathbf{H} \mathbf{x}_l$ in the second summation can be omitted. Computing $\mathcal{L} \mathbf{x}_l$ has complexity $\mathcal{O}(N)$ since $\mathcal{L}$ has $\mathcal{O}(N)$ non-zero entries. Thus, computing $\nabla f(\mathbf{x}_l)$ has complexity $\mathcal{O}(N + M + N) = \mathcal{O}(N)$. Since the number of iterations of AGD is $\mathcal{O}(1/\sqrt{\epsilon})$ [22], executing AGD has $\mathcal{O}\left(\frac{N}{\sqrt{\epsilon}}\right)$ computation. To summarize, combining the computation cost of $\mathcal{L}$, $\mathbf{h}$, β and AGD, we get $\mathcal{O}\left(N + N + N + \frac{N}{\sqrt{\epsilon}}\right) = \mathcal{O}(\frac{N}{\sqrt{\epsilon}})$, which is linear time. Linear-time complexity essentially means accessing each datum once, which constitutes a complexity lower bound for serial data processing. Thus, our depth enhancement algorithm is practical and can potentially be implemented in real-time.

V. FEATURE GRAPH LEARNING

A. Learning Metric for Graph Construction

When pixel row i of the left view is optimized, we assume that the previous K rows, $i - 1, \ldots i - K$, have already been enhanced into $\tilde{\mathbf{x}}_l^{i-1}, \ldots \tilde{\mathbf{x}}_l^{i-K}$. Using these K enhanced rows, we compute graph Laplacian $\mathbf{L}_l$ to define prior $\Pr(\mathbf{x})$ in (26). Given $K < N$ in practice, estimating $\mathbf{L}_l \in \mathbb{R}^{N \times N}$ reliably using only K signal observations is difficult. In particular, statistical graph learning algorithms such as *graphical lasso* (GLASSO) [73] that compute a sparse precision matrix using

as input a reliable empirical covariance matrix estimated from many observations do not work in our scenario.

Instead, we construct an appropriate similarity graph via *metric learning* [9]. We first assume that associated with each pixel (graph node) i in $\mathbf{x}_l$ is a length-F relevant *feature vector* $\mathbf{f}_i \in \mathbb{R}^F$ (to be discussed). The *feature distance* d_{ij} between two nodes i and j is computed using a real, symmetric and PD *metric matrix* $\mathbf{M} \in \mathbb{R}^{F \times F}$ as

$$d_{ij} = (\mathbf{f}_i - \mathbf{f}_j)^\top \mathbf{M} (\mathbf{f}_i - \mathbf{f}_j). \quad (37)$$

(37) is also called the *Mahalanobis distance* in the machine learning literature [74]. Since $\mathbf{M}$ is PD, $d_{ij} > 0$ for $\mathbf{f}_i - \mathbf{f}_j \neq \mathbf{0}$. The edge weight u_{ij} between nodes i and j is then computed using a Gaussian kernel:

$$u_{ij} = \exp(-d_{ij}). \quad (38)$$

Note that, to reduce computation complexity, we construct a sparse graph where each pixel i is only connected to its four closest neighbors $i \pm 1$ and $i \pm 2$. Thus, Laplacian $\mathbf{L}$ is sparse with $\mathcal{O}(1)$ non-zero entries per row/column.

To optimize $\mathbf{M}$, we minimize the *graph Laplacian regularizer* (GLR) evaluated using K previous pixel rows, *i.e.*,

$$\min_{\mathbf{M} \succ 0} \sum_{k=1}^K \left(\tilde{\mathbf{x}}_l^k\right)^\top \mathbf{L}_l^k(\mathbf{M}) \tilde{\mathbf{x}}_l^k \quad (39)$$

$$= \sum_{k=1}^K \sum_{i,j} u_{ij}^k \left(\tilde{x}_{l,i}^k - \tilde{x}_{l,j}^k\right)^2 \quad (40)$$

where edge weights u_{ij}^k in Laplacian $\mathbf{L}_l^k(\mathbf{M})$ is computed using features $\mathbf{f}_i^k$ and $\mathbf{f}_j^k$ of the k-th observation $\tilde{\mathbf{x}}_l^k$ via (37) and (38). To optimize $\mathbf{M}$ in (39), [9] proposed a fast algorithm to optimize the diagonal and off-diagonal entries of $\mathbf{M}$ alternately. See [9] for details.

B. Feature Selection for Metric Learning

To construct a feature vector $\mathbf{f}_i$ for each pixel i in $\mathbf{x}_l$, we first compute the pixel's corresponding *surface normal* $\mathbf{n}_i \in \mathbb{R}^3$ by projecting it to 3D space and computing it using its neighboring points via a method in [53]. Then, together with depth value x_i and location $\mathbf{l}_i \in \mathbb{R}^2$ in the 2D grid, we construct $\mathbf{f}_i \in \mathbb{R}^6$. Because $\mathbf{M}$ is symmetric, the number of matrix entries we need to estimate is only 21.

VI. EXPERIMENTATION

A. Experimental Setup

We conducted simulations with three types of datasets:

(i) Public synthetic dataset: in [57], three synthetic datasets, `FlyingThings`, `Monkaa` and `Driving`, were introduced. They are generated using Blender,[10] and the ground-truth depth images of both views are provided. As a pre-processing, we projected the depth images to 3D space, leading to PCs with around 1 million points;

[10] https://www.blender.org/

(ii) Public real dataset: the Middlebury dataset is collected with structured-light sensor [75], which also provides the ground-truth depth images of both views. We adopted five representative cases for our test, *i.e.*, `Adir`, `Playtable`, `Recycle`, `Teddy`, and `ArtL`. After projecting the depth images into 3D space, the PCs of `Adir`, `Playtable` and `Recycle` have roughly 60K points, while `Teddy` and `ArtL` have roughly 32K and 17K points, respectively;

(iii) In-house real dataset: we collected three pairs of raw depth images using the *Intel RealSense^{TM} D435* for three different scenes, `Scene0`, `Scene1` and `Scene2`. The associated projected PCs have roughly 17k points. This dataset has no ground-truths.

For the datasets (i) and (ii), we first added signal-dependent *Gaussian* noise (SDGN) to both views according to (4), then they were quantized using (3). In addition, instead of Gaussian noise, we added signal-dependent *Laplacian* noise (SDLN) [8] to both views according to (4) in order to evaluate the robustness of our approach. For dataset (iii), the collected data was already corrupted by real noise.

When learning the metric for graph construction, we considered the previous $K = 30$ pixel rows. To reduce computation complexity, the same optimized $\mathbf{M}$ was used for the right view when enhancing the left view, and vice versa. Given feature vector in $\mathbf{y}^i$ of the current row i, we computed the corresponding Laplacian $\mathbf{L}^i(\mathbf{M})$.

We compared our 3D PC enhancement method against three model-based image denoising schemes, BF [27], BM3D [28] and the early version of this work, SINUQ — short for Signal-Independent Noise Uniform Quantization [26]. Similar to our workflow, we applied these three schemes on the noise-corrupted and quantized depth image pairs before the PC synthesis steps. We also compared our work with six representative PC denoising algorithms, APSS [24], RIMLS [25], MRPCA [10], GLR [7], PCN [11], and DMR [12]. This set of methods was applied to the PCs projected from the corrupted left and right views. Among these six methods, PCN and DMR are based on deep learning, where we directly applied their released trained models to our test datasets.

To make different PCs comparable, we re-centered each PC to the origin then scaled it inside the unit sphere. For datasets (i) and (ii) with ground-truth PCs, we adopted two commonly used PC evaluation metrics for objective evaluation, *i.e.*, the parameter-free point-to-point (C2C) error [29] and point-to-plane (C2P) error [30], where plane normal in C2P was computed using six neighboring points (the only parameter). For dataset (iii) without ground-truths, we employed a no-reference metric (*i.e.*, Pseudo MOS) using sparse convolutional neural network, designed specifically for quality assessment of 3D PCs [76]. We include also visual comparisons.

B. Experimental Results

1) Linear Approximation of the Gaussian Pdf: To verify the accuracy of the linear approximation (23) of the Gaussian pdf, we constructed two extreme cases, as shown in Fig. 12. We first consider a close depth pixel $x_{l,1} = 0.5$m and with a
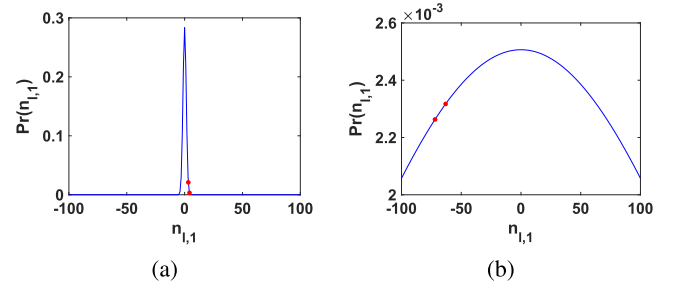

Fig. 12. Gaussian pdf of (a) $x_{l,1} = 0.5$m and (b) $x_{l,1} = 4.5$m, where the red dots are pdf of the lower and upper bounds $n^-(y_{l,1}, x_{l,1})$ and $n^+(y_{l,1}, x_{l,1})$ in the region of integration $\mathcal{R}_{l,1}$, respectively, which is an empirical demonstration corresponding to Fig. 11.

TABLE I
C2C($\times 10^{-3}$) AND C2P($\times 10^{-5}$) RESULTS OF COMPETING METHODS FOR THE PCs OF THE PUBLIC SYNTHETIC DATASET CORRUPTED BY SDGN

C2P \ C2C	FlyingThings	Monkaa	Driving	Average
BF [27]	1.83 / 2.3	0.88 / 0.2	2.06 / 6.3	1.59 / 2.9
BM3D [28]	4.77 / 8.3	1.15 / 0.3	4.11 / 16.8	3.34 / 8.5
SINUQ [26]	1.44 / 2.9	1.03 / 0.3	2.09 / 10.2	1.52 / 4.5
APSS [24]	2.67 / 2.1	1.00 / 0.2	2.94 / 5.9	2.20 / 2.7
RIMLS [25]	2.65 / 2.1	0.99 / 0.2	2.95 / 5.9	2.20 / 2.7
MRPCA [10]	2.69 / 2.1	0.95 / 0.2	2.93 / 5.2	2.19 / 2.5
GLR [7]	2.77 / 2.2	1.03 / 0.2	3.02 / 5.5	2.27 / 2.6
PCN [11]	2.13 / 1.1	0.98 / 0.2	1.62 / 2.2	1.58 / 1.2
DMR [12]	2.70 / 2.0	0.96 / 0.2	2.98 / 4.9	2.21 / 2.4
Proposed	**1.12 / 0.5**	**0.59 / 0.1**	**1.22 / 1.1**	**0.98 / 0.5**

small quantization bin, which results in a very narrow region of integration $\mathcal{R}_1$, as shown in Fig. 12(a). We observe that the pdf $\Pr(n_{l,1})$ within $\mathcal{R}_1$ is roughly linear. Specifically, the likelihood term $\Pr(y_{l,1}|x_{l,1})$ in (21) is 0.0098, and its linear approximation in (23) is 0.0083. In contrast, for a far depth pixel $x_{l,1} = 4.5$m, and a large noise variance $\sigma_{l,1}^2$, the pdf $\Pr(n_{l,1})$ becomes relatively flat, see Fig. 12(b). We observe that $\Pr(n_{l,1})$ within $\mathcal{R}_1$ is roughly linear as well. In this case, the likelihood term $\Pr(y_{l,1}|x_{l,1})$ in (21) is 0.0209, and its linear approximation in (23) is 0.0209. Thus, we can conclude that, in both cases, the linear approximation was sufficiently accurate.

2) Comparison for Public Dataset: In the case of the depth views corrupted by SDGN, quantitative results of different methods in terms of C2C and C2P errors are shown in Table I and II, where the former one provides results for the PCs of the synthetic datasets, and the latter one is for the PCs of the Middlebury dataset. Overall, our method achieved by far the best performance under both metrics for both datasets. For synthetic datasets in Table I, our proposal outperformed the second-best algorithm SINUQ by 0.54 in terms of C2C ($\times 10^{-3}$) and PCN by 0.7 in terms of C2P ($\times 10^{-5}$) on average, respectively. For the Middlebury
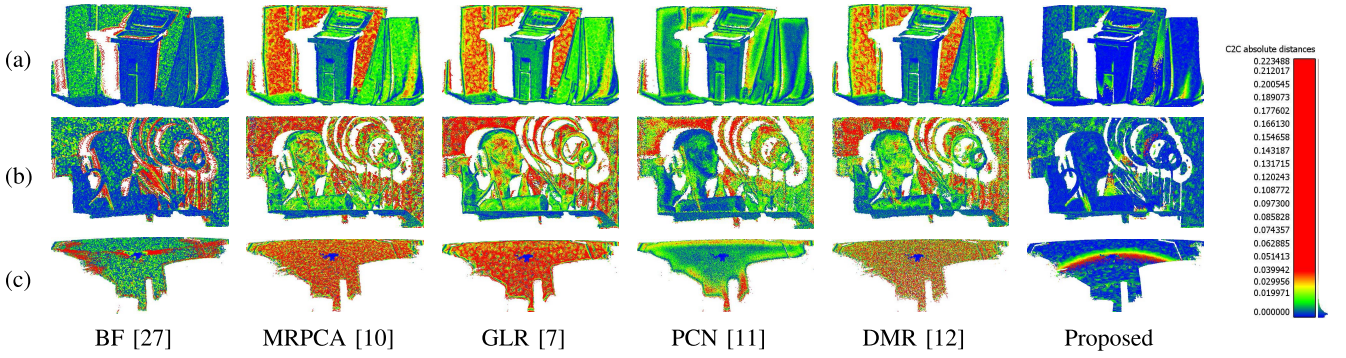
Fig. 13. Visual comparisons using (a) `Recycle` and (b) `ArtL` in the Middlebury dataset [75] and (c) `Driving` in the synthetic dataset [57] in terms of C2C errors, where the depth views were corrupted by SDGN. From blue to red, C2C absolute errors gradually become larger. More blue points are noticeably included in the proposed method.

TABLE II

C2C($\times 10^{-3}$) AND C2P($\times 10^{-5}$) RESULTS OF COMPETING METHODS FOR THE PCs OF THE PUBLIC REAL DATASET CORRUPTED BY SDGN

C2C / C2P	Adir	ArtL	Teddy	Recycle	Playtable	Average
BF [27]	4.88 / 4.2	7.76 / 8.7	4.81 / 2.9	2.75 / 1.0	3.11 / 1.8	4.66 / 3.7
BM3D [28]	2.92 / 2.9	8.19 / 11.1	5.22 / 8.0	2.37 / 2.9	3.80 / 6.2	4.50 / 6.2
SINUQ [26]	3.12 / 2.4	7.96 / 10.8	4.16 / 2.4	2.68 / 1.3	3.44 / 2.5	4.27 / 3.9
APSS [24]	10.64 / 13.8	12.05 / 12.5	9.13 / 7.8	6.91 / 6.3	6.23 / 5.8	8.99 / 9.2
RIMLS [25]	10.41 / 13.7	11.74 / 12.3	8.86 / 7.0	6.69 / 6.1	5.98 / 5.6	8.74 / 8.9
MRPCA [10]	10.18 / 15.9	11.26 / 13.5	8.56 / 8.8	6.30 / 5.2	5.48 / 4.4	8.36 / 9.6
GLR [7]	10.02 / 11.9	11.11 / 9.5	8.39 / 6.7	6.03 / 4.6	5.00 / 3.7	8.11 / 7.3
PCN [11]	7.03 / 5.5	9.03 / 6.8	5.51 / 2.7	4.00 / 1.8	4.23 / 2.4	5.96 / 3.8
DMR [12]	9.60 / 11.1	9.90 / 8.0	7.86 / 5.7	5.63 / 3.9	4.26 / 2.9	7.45 / 6.3
Proposed	**2.60** / **0.9**	**4.11** / **1.7**	**3.71** / **1.4**	**2.18** / **0.7**	**2.58** / **1.2**	**3.04** / **1.2**

TABLE III

C2C($\times 10^{-3}$) AND C2P($\times 10^{-5}$) RESULTS OF THE PCs OF THE PUBLIC SYNTHETIC DATASET AND REAL DATASET CORRUPTED BY SDLN

C2C / C2P	FlyingThings	Driving	Adir	ArtL	Teddy	Average
BF [27]	1.83 / 2.3	2.03 / 6.0	4.12 / 2.8	7.71 / 8.7	4.19 / 2.3	3.98 / 4.4
BM3D [28]	4.78 / 8.4	4.10 / 16.9	2.92 / 3.0	8.20 / 11.0	5.23 / 8.6	5.05 / 9.6
SINUQ [26]	1.37 / 2.3	2.01 / 9.4	3.12 / 2.4	6.63 / 9.2	4.15 / 2.3	3.46 / 5.1
APSS [24]	2.44 / 1.7	2.70 / 6.1	9.44 / 13.5	10.72 / 12.1	8.15 / 7.5	6.69 / 8.2
RIMLS [25]	2.39 / 1.7	2.62 / 5.8	9.19 / 13.4	10.39 / 11.8	7.89 / 7.2	6.50 / 7.9
MRPCA [10]	2.29 / 1.8	2.53 / 5.1	8.62 / 13.8	9.47 / 10.9	7.33 / 7.6	6.05 / 7.8
GLR [7]	2.32 / 1.8	2.45 / 5.2	8.67 / 11.3	9.61 / 8.7	7.42 / 6.2	6.09 / 6.6
PCN [11]	1.86 / 0.9	1.26 / 1.1	5.57 / 4.7	7.45 / 6.3	4.71 / 2.3	4.17 / 3.1
DMR [12]	2.23 / 1.6	2.82 / 5.0	7.31 / 9.5	7.22 / 5.8	6.39 / 5.0	5.19 / 5.4
Proposed	**1.12** / **0.4**	**1.21** / **1.1**	**2.66** / **1.0**	**4.10** / **1.7**	**3.67** / **1.4**	**2.55** / **1.1**

The synthetic datasets had relatively small depth values, meaning that the signal-dependent additive noise was small. Thus, the performance of the six PC denoising algorithms were comparable to that of the three image denoising schemes. However, for the Middlebury dataset capturing real-world scenes, they had relatively large depth values, and thus the additive noise was large.

Table III shows C2C and C2P results of the PCs from both the public synthetic dataset and real dataset, corrupted by SDLN. Similarly, our proposal outperformed nine competitors in both metrics, with C2C ($\times 10^{-3}$) error reduced by 0.91 compared with the second-best algorithm SINUQ, and C2P ($\times 10^{-5}$) error reduced by 2.0 compared with the second-best algorithm PCN, respectively.

Visual results of the denoised PCs `Recycle` and `ArtL` from the Middlebury dataset, and a PC `Driving` from the synthetic dataset for the additive SDGN case are shown in Fig. 13, where we colored the point clouds according to the C2C absolute distances between the ground truth points and their closest denoised points. For the result of `Driving`, we only show its background for better visualization. From Fig. 13, it is obvious that our proposal achieved smaller C2C errors compared to the competitors.

Overall, one can observe that the six PC denoising algorithms were worse than the image denoising schemes. The reason is that these PC denoising algorithms assume that i.i.d. Gaussian noise are added to the point coordinates in 3D space. This assumption is inaccurate in practice, because a corrupted depth pixel causes errors only in the depth dimension perpendicular to the image plane, which is quite different from i.i.d. noise in 3D space. Thus, enhancing depth measurements before synthesizing a PC is more sensible. In contrast, the three selected image denoising schemes performed poorly under signal-dependent noise and non-uniform quantization. Further, it was difficult for BF [27] and BM3D [28] to handle existing holes throughout the images, leading to locally poor performance.

In contrast, our proposal enhances depth measurements before projecting to 3D space to synthesize a PC. It allows us to tailor the optimization specifically for our depth formation model which complies with the physical acquisition process. Particularly, we model the combination of signal-dependent noise addition and non-uniform log-based quantization. Further, by employing feature graph learning, we can
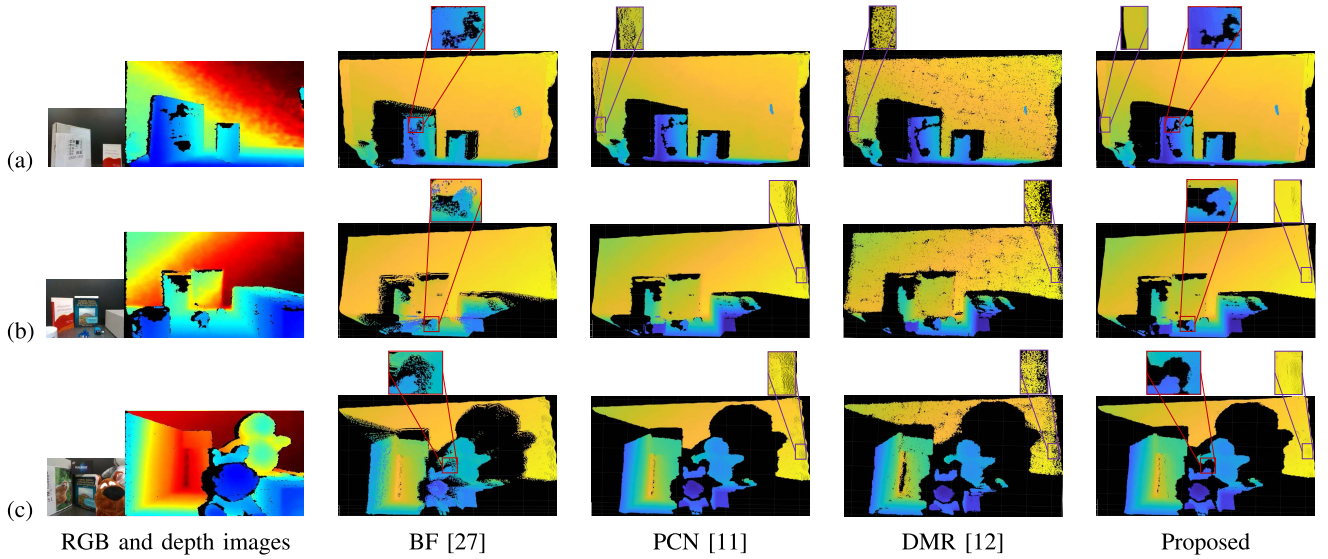
dataset in Table II, in terms of C2C ($\times 10^{-3}$), our method outperformed the second-best algorithm SINUQ by 1.23, while in terms of C2P ($\times 10^{-5}$), our method outperformed the second-best algorithm BF by 2.5 on average.

Fig. 14. Visual comparisons for the denoised PCs (a) `Scene0`, (b) `Scene1` and (c) `Scene2`, projected from our in-house raw depth image pairs collected using *Intel RealSense*^{TM} *D435*. Smoother surface silhouette implies a higher denoising quality. Note that the collected RGB images and depth images are for reference without alignment, and thus are different view images captured by slightly offset cameras. The resulting PCs are shown from a particular chosen viewpoint.

TABLE IV
PSEUDO MOS RESULTS OF COMPETING METHODS FOR
THE PCs OF THE IN-HOUSE REAL DATASET

Pseudo MOS	Scene0	Scene1	Scene2	Average
BF [27]	2.96	2.73	2.55	2.75
BM3D [28]	2.72	2.31	2.15	2.39
SINUQ [26]	2.88	2.47	2.47	2.61
APSS [24]	2.94	2.88	3.06	2.96
RIMLS [25]	2.99	2.80	2.94	2.91
MRPCA [10]	3.11	2.82	2.84	2.92
GLR [7]	2.84	2.65	2.97	2.82
PCN [11]	3.10	3.02	3.16	3.09
DMR [12]	3.05	2.59	2.84	2.83
Proposed	**3.13**	**3.15**	**3.35**	**3.21**

flexibly enhance only the available pixels around missing ones in a depth image. For these reasons, our scheme achieved noticeable performance gains.

3) Comparison for In-House Dataset: All the above experiments were conducted using artificial noise. To see the effectiveness of our method on real sensor noise, we computed Pseudo MOS scores of no-reference metric [76] for different methods using PCs from our in-house dataset. When MOS score equals 5, it means that no discernible distortion is perceived in the PC. MOS score equaling 3 implies that distortion slightly obstructs viewing. Note that this no-reference metric considers both color and geometry attributes, while our optimization restores only geometric information, and thus generally lowering the scores. However, we observe that our method still outperformed three model-based image denoising schemes and six PC denoising algorithms.

We next show visual comparisons for our in-house collected PCs as perceived from the right depth image in Fig. 14. For the denoised PCs, a smoother surface silhouette basically implies a higher restoration quality. Note that BF reconstructed the intensity of each pixel with a weighted average of intensity values from nearby pixels. Consequently, given an image with many missing pixels, this method diffused the available pixels to the missing pixels and propagated errors from missing pixels

to the target pixels, as shown in the enlarged regions in the second column. Due to the i.i.d. Gaussian noise assumption on PCs, distortion in 3D points stemming from the real formation process were challenging for the two deep learning based methods, PCN and DMR, leading to poor restoration results. In contrast, our proposal provided restoration with noticeably better visual quality, which has smoother surface and fewer noisy points.

VII. CONCLUSION AND DISCUSSION

Point clouds are typically synthesized from finite-precision depth measurements that are noise-corrupted. In this paper, we improve the quality of a synthesized point cloud by jointly enhancing multiview depth images—the "rawest" signal we can acquire from an off-the-shelf sensor—prior to steps in a typical point cloud synthesis pipeline that obscure acquisition noise. We formulate a graph-based MAP optimization that specifically targets an image formation model accounting for both signal-dependent noise addition and non-uniform log-based quantization. We validate the designed model using collected empirical data from an actual depth sensor. We optimize the objective efficiently using AGD with GCT-aided optimal step size determination. Simulation results show that our proposed scheme outperforms competing schemes that denoise point clouds after the synthesis pipeline and representative image denoising schemes.

APPENDIX
PROOF OF MULTIPLE INTEGRAL

We first prove (24) by induction.

$$\Pr(\mathbf{y}_l|\mathbf{x}_l) \approx \prod_{i=1}^{N} \int_{\mathcal{R}_{l,i}} \left(a_{l,i} n_{l,i} + b_{l,i} \right) d\, n_{l,i}$$

$$= \prod_{i=1}^{N} \int_{z_{l,i}^- - x_{l,i}}^{z_{l,i}^+ - x_{l,i}} \left(a_{l,i} n_{l,i} + b_{l,i} \right) d\, n_{l,i}$$

$$= \prod_{i=1}^{N} \frac{a_{l,i}}{2} n_{l,i}^2 + b_{l,i} n_{l,i} \Big|_{z_{l,i}^- - x_{l,i}}^{z_{l,i}^+ - x_{l,i}}$$

$$= \prod_{i=1}^{N} a_{l,i}(z_{l,i}^- - z_{l,i}^+)x_{l,i} + \frac{a_{l,i}}{2}((z_{l,i}^+)^2 - (z_{l,i}^-)^2)$$

$$+ b_{l,i}(z_{l,i}^+ - z_{l,i}^-). \tag{41}$$

Denote by $\mathbf{1}_i$ an one-hot vector, which is a suitably long *canonical vector* of all zeros except entry i is 1, $\tilde{\mathbf{a}}_{l,i}^\top = a_{l,i}(z_{l,i}^- - z_{l,i}^+)\mathbf{1}_i^\top$ and $\tilde{b}_{l,i} = \frac{a_{l,i}}{2}((z_{l,i}^+)^2 - (z_{l,i}^-)^2) + b_{l,i}(z_{l,i}^+ - z_{l,i}^-)$. We now rewrite (41) as

$$\Pr(\mathbf{y}_l|\mathbf{x}_l) \approx \prod_{i=1}^{N} (\tilde{\mathbf{a}}_{l,i}^\top \mathbf{x}_l + \tilde{b}_{l,i}). \tag{42}$$

Given (41), (42) and (19), we now prove (25) for noise $\mathbf{n}_r$.

$$\Pr(\mathbf{y}_r|\mathbf{x}_l) \approx \prod_{j=1}^{M} \int_{\mathcal{R}_{r,j}} (a_{r,j}n_{r,j} + b_{r,j})\, d\,n_{r,j}$$

$$= \prod_{j=1}^{M} (\tilde{\mathbf{a}}_{r,j}^\top \mathbf{x}_r + \tilde{b}_{r,j}) \approx \prod_{j=1}^{M} (\tilde{\mathbf{a}}_{r,j}^\top (\mathbf{H}\mathbf{x}_l + \mathbf{e}) + \tilde{b}_{r,j}). \tag{43}$$

Denote by $\bar{\mathbf{a}}_{r,j}^\top = \tilde{\mathbf{a}}_{r,j}^\top = a_{r,j}(z_{r,j}^- - z_{r,j}^+)\mathbf{1}_i^\top$ and $\bar{b}_{r,j} = \tilde{\mathbf{a}}_{r,j}^\top \mathbf{e} + \tilde{b}_{r,j} = \bar{\mathbf{a}}_{r,j}^\top \mathbf{e} + \frac{a_{r,j}}{2}((z_{r,j}^+)^2 - (z_{r,j}^-)^2) + b_{r,j}(z_{r,j}^+ - z_{r,j}^-)$. We now rewrite (43) as

$$\Pr(\mathbf{y}_r|\mathbf{x}_l) \approx \prod_{j=1}^{M} (\bar{\mathbf{a}}_{r,j}^\top \mathbf{H}\mathbf{x}_l + \bar{b}_{r,j}). \tag{44}$$

□

REFERENCES

[1] R. Mekuria, K. Blom, and P. Cesar, "Design, implementation, and evaluation of a point cloud codec for tele-immersive video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 4, pp. 828–842, Apr. 2016.

[2] S. N. Gunkel, H. M. Stokking, M. J. Prins, N. van der Stap, F. B. T. Haar, and O. A. Niamut, "Virtual reality conferencing: Multi-user immersive VR experiences on the web," in *Proc. 9th ACM Multimedia Syst. Conf.*, 2018, pp. 498–501.

[3] H. Yuan, D. Zhang, W. Wang, and Y. Li, "A sampling-based 3D point cloud compression algorithm for immersive communication," *Mobile Netw. Appl.*, vol. 25, no. 5, pp. 1863–1872, Oct. 2020.

[4] X. Zhang, G. Cheung, Y. Zhao, P. Le Callet, C. Lin, and J. Z. G. Tan, "Graph learning based head movement prediction for interactive 360 video streaming," *IEEE Trans. Image Process.*, vol. 30, pp. 4622–4636, 2021.

[5] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[6] X. Huang et al., "The apolloscape dataset for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2018, pp. 954–960.

[7] J. Zeng, G. Cheung, M. Ng, J. Pang, and Y. Cheng, "3D point cloud denoising using graph Laplacian regularization of a low dimensional manifold model," *IEEE Trans. Image Process.*, vol. 29, pp. 3474–3489, 2019.

[8] C. Dinesh, G. Cheung, and I. V. Bajić, "Point cloud denoising via feature graph Laplacian regularization," *IEEE Trans. Image Process.*, vol. 29, pp. 4143–4158, 2020.

[9] W. Hu, X. Gao, G. Cheung, and Z. Guo, "Feature graph learning for 3D point cloud denoising," *IEEE Trans. Signal Process.*, vol. 68, pp. 2841–2856, 2020.

[10] E. Mattei and A. Castrodad, "Point cloud denoising via moving RPCA," *Comput. Graph. Forum*, vol. 36, no. 8, pp. 123–137, 2017.

[11] M.-J. Rakotosaona, V. La Barbera, P. Guerrero, N. J. Mitra, and M. Ovsjanikov, "PointCleanNet: Learning to denoise and remove outliers from dense point clouds," *Comput. Graph. Forum*, vol. 39, no. 1, pp. 185–203, 2020.

[12] S. Luo and W. Hu, "Differentiable manifold reconstruction for point cloud denoising," in *Proc. 28th ACM Int. Conf. Multimedia*, Oct. 2020, pp. 1330–1338.

[13] A. Punnappurath and M. S. Brown, "Learning raw image reconstruction-aware deep image compressors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 4, pp. 1013–1019, Apr. 2019.

[14] R. M. H. Nguyen and M. S. Brown, "RAW image reconstruction using a self-contained sRGB-JPEG image with only 64 KB overhead," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1655–1663.

[15] S. Farsiu, M. Elad, and P. Milanfar, "Multiframe demosaicing and super-resolution of color images," *IEEE Trans. Image Process.*, vol. 15, no. 1, pp. 141–159, Jan. 2005.

[16] D. Sahu, A. Bhargava, and P. Badal, "Contrast image enhancement using various approaches: A review," *J. Image Process. Pattern Recognit. Prog.*, vol. 4, no. 3, pp. 39–45, 2017.

[17] D. S. Alexiadis, D. Zarpalas, and P. Daras, "Real-time, full 3-D reconstruction of moving foreground objects from multiple consumer depth cameras," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 339–358, Nov. 2012.

[18] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing*, 3rd ed. Cambridge, U.K.: Cambridge Univ. Press, 2007.

[19] A. Kadambi, A. Bhandari, and R. Raskar, "3D depth cameras in vision: Benefits and limitations of the hardware," in *Computer Vision and Machine Learning with RGB-D Sensors*. Cham, Switzerland: Springer, 2014, pp. 3–26.

[20] J. Jin, A. Wang, Y. Zhao, C. Lin, and B. Zeng, "Region-aware 3-D warping for DIBR," *IEEE Trans. Multimedia*, vol. 18, no. 6, pp. 953–966, Jun. 2016.

[21] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate $\mathcal{O}(1/k^2)$," *Doklady Akademii Nauk SSSR*, vol. 269, no. 3, pp. 543–547, 1983.

[22] S. Bubeck, "Convex optimization: Algorithms and complexity," *Found. Trends Mach. Learn.*, vol. 8, nos. 3–4, pp. 231–357, 2015.

[23] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. press, 2012.

[24] G. Guennebaud and M. Gross, "Algebraic point set surfaces," *ACM Trans. Graph.*, vol. 26, no. 3, p. 23, 2007.

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

[26] X. Zhang, G. Cheung, J. Pang, and D. Tian, "3D point cloud enhancement using graph-modelled multiview depth measurements," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 3314–3318.

[27] C. Tomasi and zR. Manduchi, "Bilateral filtering for gray and color images," in *Proc. IEEE Int. Conf. Comput. Vis.*, Bombay, India, 1998, pp. 839–846.

[28] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[29] D. Girardeau-Montaut, M. Roux, R. Marc, and G. Thibault, "Change detection on points cloud data acquired with a ground laser scanner," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 36, no. 3, p. W19, 2005.

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

[31] M. Nagahama, K. Yamada, Y. Tanaka, S. H. Chan, and Y. C. Eldar, "Graph signal denoising using nested-structured deep algorithm unrolling," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 5280–5284.

[32] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-play ADMM for image restoration: Fixed-point convergence and applications," *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 84–98, Mar. 2017.

[33] W. Hu, X. Li, G. Cheung, and O. Au, "Depth map denoising using graph-based transform and group sparsity," in *Proc. IEEE 15th Int. Workshop Multimedia Signal Process. (MMSP)*, Sep. 2013, pp. 1–6.

[34] S. Lu, X. Ren, and F. Liu, "Depth enhancement via low-rank matrix completion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 3390–3397.

[35] J. Xie, C.-C. Chou, R. Feris, and M.-T. Sun, "Single depth image super resolution and denoising via coupled dictionary learning with local constraints and shock filtering," in *Proc. IEEE Int. Conf. Multimedia Expo. (ICME)*, Jul. 2014, pp. 1–6.

[36] B. Ham, M. Cho, and J. Ponce, "Robust image filtering using joint static and dynamic guidance," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 4823–4831.

[37] J. Pang and G. Cheung, "Graph Laplacian regularization for image denoising: Analysis in the continuous domain," *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 1770–1785, Apr. 2017.

[38] S. Gu, W. Zuo, S. Guo, Y. Chen, C. Chen, and L. Zhang, "Learning dynamic guidance for depth image enhancement," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3769–3778.

[39] J. Yang, X. Ye, K. Li, C. Hou, and Y. Wang, "Color-guided depth recovery from RGB-D data using an adaptive autoregressive model," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3443–3458, Aug. 2014.

[40] L.-K. Liu, S. H. Chan, and T. Q. Nguyen, "Depth reconstruction from sparse samples: Representation, algorithm, and sampling," *IEEE Trans. Image Process.*, vol. 24, no. 6, pp. 1983–1996, Jun. 2015.

[41] W. Hu, G. Cheung, and M. Kazui, "Graph-based dequantization of block-compressed piecewise smooth images," *IEEE Signal Process. Lett.*, vol. 23, no. 2, pp. 242–246, Feb. 2016.

[42] C. V. Nguyen, S. Izadi, and D. Lovell, "Modeling Kinect sensor noise for improved 3D reconstruction and tracking," in *Proc. 2nd Int. Conf. 3D Imag., Modeling, Process., Vis. Transmiss.*, Oct. 2012, pp. 524–530.

[43] T. Mallick, P. P. Das, and A. K. Majumdar, "Characterizations of noise in Kinect depth images: A review," *IEEE Sensors*, vol. 14, no. 6, pp. 1731–1740, Jun. 2014.

[44] I. Ideses, L. Yaroslavsky, I. Amit, and B. Fishbain, "Depth map quantization—How much is sufficient?" in *Proc. 3DTV Conf.*, May 2007, pp. 1–4.

[45] P. Wan, G. Cheung, P. A. Chou, D. Florencio, C. Zhang, and O. C. Au, "Precision enhancement of 3-D surfaces from compressed multiview depth maps," *IEEE Signal Process. Lett.*, vol. 22, no. 10, pp. 1676–1680, Oct. 2015.

[46] Y. Wang, A. Ortega, D. Tian, and A. Vetro, "A graph-based joint bilateral approach for depth enhancement," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2014, pp. 885–889.

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

[48] 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.

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

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

[51] H. Hui, L. Dan, Z. Hao, U. Ascher, and D. Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–7, 2009.

[52] Y. Sun, S. Schaefer, and W. Wang, "Denoising point sets via L_0 minimization," *Comput. Aided Geometric Des.*, vols. 35–36, pp. 2–15, May 2015.

[53] H. Avron, A. Sharf, C. Greif, and D. Cohen-Or, "ℓ_1-sparse reconstruction of sharp point set surfaces," *ACM Trans. Graph.*, vol. 29, no. 5, pp. 1–12, Oct. 2010.

[54] G. Rosman, A. Dubrovina, and R. Kimmel, "Patch-collaborative spectral point-cloud denoising," *Comput. Graph. Forum*, vol. 32, no. 8, 2013, pp. 1–12.

[55] R.-F. Wang, W.-Z. Chen, S.-Y. Zhang, Y. Zhang, and X.-Z. Ye, "Similarity-based denoising of point-sampled surfaces," *J. Zhejiang Univ.-Sci. A*, vol. 9, no. 6, pp. 807–815, 2008.

[56] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[57] N. Mayer et al., "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 4040–4048.

[58] J. Zeng, J. Pang, W. Sun, and G. Cheung, "Deep graph Laplacian regularization for robust denoising of real images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 1–10.

[59] C. Loop and Z. Zhang, "Computing rectifying homographies for stereo vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 1, Jun. 1999, pp. 125–131.

[60] C. Fehn, K. Hopf, and B. Quante, "Key technologies for an advanced 3D TV system," *Proc. SPIE*, vol. 5599, pp. 66–80, Oct. 2004.

[61] S. Mehrotra, Z. Zhang, Q. Cai, C. Zhang, and P. A. Chou, "Low-complexity, near-lossless coding of depth maps from Kinect-like depth cameras," in *Proc. IEEE 13th Int. Workshop Multimedia Signal Process.*, Oct. 2011, pp. 1–6.

[62] A. Tontini, L. Gasparini, and M. Perenzoni, "Numerical model of SPAD-based direct time-of-flight flash LIDAR CMOS image sensors," *Sensors*, vol. 20, no. 18, p. 5203, Sep. 2020.

[63] A. Dermanis, A. Grün, and F. Sansò, Eds., *Geomatic Methods for the Analysis of Data in the Earth Sciences*, vol. 95. Berlin, Germany: Springer, 2000.

[64] Y. S. Kim et al., "Parametric model-based noise reduction for ToF depth sensors," *Proc. SPIE*, vol. 8290, pp. 91–98, Jan. 2012.

[65] H. Pishro-Nik, *Introduction to Probability, Statistics, and Random Processes*. Blue Bell, PA, USA: Kappa, 2014.

[66] T. F. Coleman and Y. Li, "An interior trust region approach for nonlinear minimization subject to bounds," *SIAM J. Optim.*, vol. 6, no. 2, pp. 418–445, 1996.

[67] P. Bholowalia and A. Kumar, "EBK-means: A clustering technique based on elbow method and k-means in WSN," *Int. J. Comput. Appl.*, vol. 105, no. 9, pp. 17–24, 2014.

[68] P. Merkle et al., "The effect of depth compression on multiview rendering quality," in *Proc. IEEE 3DTV Conf., True Vis.-Capture, Transmiss. Display 3D Video*, May 2008, pp. 245–248.

[69] B. Motz, G. Cheung, A. Ortega, and P. Frossard, "Re-sampling and interpolation of DIBR-synthesized images using graph-signal smoothness prior," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA)*, Dec. 2015, pp. 1077–1084.

[70] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. 19th Int. Conf. Artif. Intell. Statist.*, 2016, pp. 920–929.

[71] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 825–841, Sep. 2017.

[72] Y. Bai, G. Cheung, X. Liu, and W. Gao, "Graph-based blind image deblurring from a single photograph," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1404–1418, Mar. 2018.

[73] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, Jul. 2008.

[74] P. C. Mahalanobis, "On the generalized distance in statistics," Nat. Inst. Sci. India, Kolkata, India, Tech. Rep., 1936, vol. 2, no. 1, pp. 49–55.

[75] D. Scharstein et al., "High-resolution stereo datasets with subpixel-accurate ground truth," in *Proc. German Conf. Pattern Recognit.*, 2014, pp. 31–42.

[76] Y. Liu, Q. Yang, Y. Xu, and L. Yang, "Point cloud quality assessment: Dataset construction and learning-based no-reference metric," *ACM Trans. Multimedia Comput., Commun., Appl.*, early access, Jul. 2022.
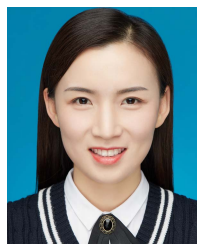
Xue Zhang (Member, IEEE) received the Ph.D. degree in signal and information processing from Beijing Jiaotong Unversity (BJTU), Beijing, China, in 2019. From 2015 to 2017, she was a Visiting Ph.D. student at the Signal Processing Laboratory (LTS4), Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. In 2018, she was a Visiting Ph.D. student at the National Institute of Informatics (NII), Tokyo, Japan. She was a Postdoctoral Fellow with York University, Toronto, Canada, from 2019 to 2022. She is currently an Assistant Professor with the Shandong University of Science and Technology (SDUST), Qingdao, China. Her research interests include 3D image/video processing, interactive media navigation, and graph signal processing.

Gene Cheung (Fellow, IEEE) received the B.S. degree in electrical engineering from Cornell University in 1995, and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California at Berkeley, Berkeley, in 1998 and 2000, respectively.

He was a Senior Researcher at the Hewlett-Packard Laboratories Japan, Tokyo, Japan, from 2000 to 2009. He was an Assistant then Associate Professor at the National Institute of Informatics (NII), Tokyo, from 2009 to 2018. He is currently a Professor with York University, Toronto, Canada. His research interests include 3D imaging and graph signal processing. He is the coauthor of several paper awards and nominations, including the Best Student Paper Finalist in ICASSP 2021; the Best Student Paper Award in ICIP 2013, ICIP 2017, and IVMSP 2016; the Best Paper Runner-up Award in ICME 2012; and IEEE Signal Processing Society (SPS) Japan Best Paper Award 2016.

Dr. Cheung was a recipient of the Canadian NSERC Discovery Accelerator Supplement (DAS) 2019. He has served as Associate Editor for multiple journals, including IEEE TRANSACTIONS ON MULTIMEDIA (2007–2011), IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (2016–2017), and IEEE TRANSACTIONS ON IMAGE PROCESSING (2015–2019). He currently serves as Senior Associate Editor for IEEE SIGNAL PROCESSING LETTERS since 2021. He served as a member for the Multimedia Signal Processing Technical Committee (MMSP-TC) in IEEE Signal Processing Society (2012–2014), and a member of the Image, Video, and Multidimensional Signal Processing Technical Committee (IVMSP-TC) (2015–2017, 2018–2020).

Jiahao Pang (Member, IEEE) received the B.Eng. degree from the South China University of Technology, Guangzhou, China, in 2010, and the M.Sc. and Ph.D. degrees from the Hong Kong University of Science and Technology, Hong Kong, China, in 2011 and 2016, respectively. He was a Senior Researcher at SenseTime Group Ltd., Hong Kong, from 2016 to 2019. He is currently a Senior Staff Engineer with InterDigital, New York, NY, USA. He has over 40 publications in international journals, conferences, and book chapters. His research interests include 3D computer vision, image processing, graph signal processing, and deep learning. He also serves as a regular reviewer for IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE CVPR, and NeurIPS.

Yash Sanghvi (Student Member, IEEE) received the B.Tech. and M.Tech. degrees in electrical engineering from the Indian Institute of Technology, Bombay, in 2018. He is currently pursuing the Ph.D. degree with the Intelligent Imaging Laboratory at Electrical and Computer Engineering, Purdue University, West Lafayette. Since 2019, he is currently a Graduate Research Assistant with the Intelligent Imaging Laboratory at Electrical and Computer Engineering, Purdue University. He was working on the inverse scattering problem as a Project Scientist at the Indian Institute of Technology Madras. His research interests include inverse problems, computational imaging, and deep learning.

Abhiram Gnanasambandam (Member, IEEE) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology Madras in 2017, and the Ph.D. degree from the School of Electrical and Computer Engineering, Purdue University, in 2022. His Ph.D. dissertation dealt with computational imaging, image processing, deep learning, and computer vision. He is currently a Senior Engineer with Samsung Research America.

Stanley H. Chan (Senior Member, IEEE) received the B.Eng. degree (Hons.) in electrical engineering from the University of Hong Kong in 2007, the M.A. degree in mathematics, and the Ph.D. degree in electrical engineering from the University of California at San Diego in 2009 and 2011, respectively.

From 2012 to 2014, he was a Postdoctoral Research Fellow at Harvard University. He joined Purdue University, West Lafayette, IN, USA, in 2014, where he is currently an Elmore Associate Professor of electrical and computer engineering, with a joint appointment with the Department of Statistics. He is the author of a popular undergraduate textbook *Introduction to Probability for Data Science* (Michigan Publishing, 2021). The book is part of the University of Michigan's free textbook initiative to disseminate high-quality educational materials to students and families around the world. Since the launch of the book in Fall 2021, the book has saved more than $3 million USD for families globally. His research interests include single-photon imaging, imaging through atmospheric turbulence, and computational photography.

Dr. Chan was a recipient of the Best Paper Award of IEEE International Conference on Image Processing 2016, IEEE Signal Processing Cup 2016 Second Prize, Purdue College of Engineering Exceptional Early Career Teaching Award 2019, Purdue College of Engineering Outstanding Graduate Mentor Award 2016, and Eta Kappa Nu (Beta Chapter) Outstanding Teaching Award 2015. Since 2018, he has been an Associate Editor of the IEEE TRANSACTIONS ON COMPUTATIONAL IMAGING, where he is recognized as an Outstanding Editorial Board Member in 2021. He also served as an Associate Editor of the OSA Optics Express from 2016 to 2018 and an Elected Member of the IEEE Signal Processing Society Technical Committee in Computational Imaging from 2015 to 2020.