An Efficient Sampling Algorithm With a K-NN Expanding Operator for Depth Data Acquisition in a LiDAR System

Xuan Truong Nguyen, Hyun Kim, *Member, IEEE*, and Hyuk-Jae Lee, *Member, IEEE*

***Abstract*— The spatial resolution of a depth-acquisition device, such as a Light Detection and Ranging (LiDAR) sensor, is limited because of the slow acquisition. To accurately reconstruct a depth image from limited spatial resolution, a two-stage sampling process has been widely used. However, two-stage sampling uses an irregular sampling pattern for the sampling operation, which requires complex computation for reconstruction and additional memory space for storage. A mathematical formulation of a LiDAR system demonstrates that two-stage sampling does not satisfy its timing constraint for practical use. To overcome the drawbacks of two-stage sampling, this paper proposes a new sampling method that reduces the computational complexity and memory requirements by generating the optimal representatives of a sampling pattern in down-sample data. A sampling pattern can be derived from a k-NN expanding operation from the down-sampled representatives. The proposed algorithm is designed to preserve the object boundary by restricting the expansion-operation only to the object boundary or complex texture. In addition, the proposed algorithm runs in linear-time complexity and reduces the memory requirements using a down-sampling ratio. The experimental results demonstrate that the proposed sampling outperforms grid sampling by at most 7.92 dB. Consequently, the proposed sampling achieves reconstructed quality similar to that of optimal sampling, while substantially reducing the computation time and memory requirements.**

***Index Terms*— Compressive and non-uniform sampling, compressive sensing, depth data acquisition, light detection and ranging (LiDAR), sparse representation.**

I. Introduction

DEPTH-DATA-ACQUISITION devices have been a focus of intensive study in recent years, owing to vast applications in autonomous driving [1], [2], remote sensing [3], and robotics [4]. The two major classes of depth acquisition techniques include computational procedures and hardware solutions. The class of computational methods known as disparity-estimation algorithms [5]–[7] estimate depth by computing the disparities in stereo images using their corresponding matching features. Disparity-estimation methods generally function optimally under specific conditioned environments. However, they are sensitive to illumination, noise, and other related factors. Dense disparity estimation is a highly complex computational task, and therefore sampling of input data is critical to accelerating the estimation procedure and selecting the effective number of reliable features. On the other hand, an alternative solution is to use scanning systems equipped with active sensors such as time-of-flight camera [8] and Light Detection and Ranging (LiDAR) sensors [9]. While being capable of producing high-quality depth maps, the data-acquisition time is relatively long, which limits the capturing speed. For example, the frame rate of a LiDAR is 10 fps as opposed to 60 fps in a standard camera [10]. Accelerating the data-acquisition time involves a trade-off with spatial resolution [11]. Efficient and accurate sampling is eventually required to reduce the spatial resolution of such scanning systems.

For a broad use in depth-data acquisition systems, a sampling method should have the following properties:

1. Perceptually capture groups or regions that generally reflect the global aspects of a depth image. Given a sample budget, a sampling method should be capable of capturing details in the object boundary while omitting details in the smooth areas. The definition of sampling is formulated to represent these properties for an enhanced understanding of the method and to facilitate the comparison of different techniques.
2. Being computationally efficient implies having a computational complexity of $O(n)$, where n is the number of image pixels. For practical use, sampling methods must run at a speed that is similar to that of gradient computation or other low-level visual-processing methods, implying approximately linear time with low constant factors.

Uniform random or grid sampling is the most straightforward approach that is highly efficient and satisfies the second property [9], [11]. However, this method is generally inefficient in capturing perceptually critical non-local properties

Manuscript received August 22, 2019; accepted December 23, 2019. Date of publication January 1, 2020; date of current version December 4, 2020. This article was recommended by Associate Editor V. Monga. This work was supported in part by the Institute for Information and communications Technology Promotion grant through the Korea Government (MSIT) under Grant 2017-0-00721-001 (Development of intelligent semiconductor technology for vision recognition signal processing for vehicle based on multi-sensor fusion) and in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant NRF-2019R1A6A1A03032119. *(Corresponding author: Hyuk-Jae Lee.)*

Xuan Truong Nguyen is with the Department of Electrical and Computer Engineering, College of Engineering, Seoul National University, Seoul 151-744, South Korea.

Hyun Kim is with the Department of Electrical and Information Engineering, Seoul National University of Science and Technology, Seoul 01811, South Korea.

Hyuk-Jae Lee is with the Inter-University Semiconductor Research Center, Seoul National University, Seoul 151-744, South Korea, and also with the Department of Electrical Engineering, Seoul National University, Seoul 151-744, South Korea (e-mail: hyuk_jae_lee@capp.snu.ac.kr).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2019.2963448

of an image such as the object boundary. Therefore, reconstruction quality is relatively poor and does not satisfy the quality requirements in several applications. On the other hand, a non-uniform sampling strategy can significantly enhance the signal to noise ratio (SNR) [12]–[16]. An effective sampling approach is proposed in [12] under the assumption that the global properties of a depth image (e.g. its gradients) are available. This method efficiently samples the depth image along edges. Unfortunately, the gradient is not available prior to sampling, which renders the assumption unrealistic in practical uses. Nonetheless, the sampling method provides strong evidence of the feasibility of a more effective sampling method to capture the global aspects of an image; thereby enhancing the SNR or reconstruction quality. To obtain the global properties, several sampling schemes are proposed in [13]–[15] according to which the sampling operations are performed in two stages. In the first stage, the scheme adopts a uniform strategy using only a part of the sample budget. The sampled data are used to reconstruct an image and then extract global information such as gradient map [13] and object saliency map [14], [15]. Non-uniform sampling approaches are used in the refinement stage, and then the sampling result is merged into those of the first one.

Although previous two-step sampling methods significantly enhance the SNR, they exhibit two drawbacks. First, they invoke an intermediate reconstruction that is complicated, which makes it challenging to reconstruct an image in real time even though numerous approaches for efficient reconstruction such as convex optimization and greedy methods [12], [13], [16]–[20] have been extensively studied. Second, an irregular sampling pattern usually requires additional storage space or transmission bandwidth, which must be included in the budget of samples and therefore reduces the number of feasible samples. These two challenging issues limit the use of a two-step approach to practical applications such as data acquisition or laser measurement systems, which strictly require an efficient sampling method.

To address the above two drawbacks, this paper proposes a new mathematical formulation of the constraints for a practical sampling method in a LiDAR system. Based on the proposed mathematical formulation, it is shown that existing two-stage sampling approaches are not suitable for a practical LiDAR system. Therefore, this paper presents a novel sampling method to efficiently perform non-uniform random sampling. The proposed algorithm extends the two-step method in the previous designs [13]–[15] to reduce the computational complexity and the requirements of additional storage or bandwidth while still achieving high SNR quality. The proposed method performs uniform sampling at the pilot stage and non-uniform sampling at the refinement stage. However, unlike in the previous methods, the proposed technique efficiently derives non-uniform sampling based on the gradient of the down-sampled image. Consequently, the proposed method follows implicit global properties notwithstanding decision-making using a greedy approach. More critically, the proposed method for computing the gradient and refinement-sampling map is substantially faster than other methods because it does not require intermediate reconstruction. Consequently, it is computationally efficient with O(n) complexity for n image pixels. In addition, the proposed method reduces the requirement of additional memory (or bandwidth) to store (or transmit) the sampling pattern. To this end, the proposed method outperforms grid sampling by at most 7.92 dB. As a result, the proposed sampling achieves a reconstructed quality that is similar to the optimal sampling in the previous design, while substantially reducing the computation time and memory requirements.

The rest of this paper is organized as follows. Section II briefly introduces a sampling model, a sampling optimization problem, and the previous approaches. In Section III, timing and memory-space constraints in the LiDAR system are presented and three variations of the sampling problem are described. Section IV presents a graph-based representation of a down-sampling operator, including the definition of a new k-expanding operator and a description of the proposed sampling algorithm and its properties. Experimental results are presented in Section V, and Section VI concludes the paper.

II. Sampling Problem and Related works

This section briefly describes the definition of a sampling problem and introduces previous studies on gradient-based sampling and two-step sampling [12], [13], which are the most relevant sampling approaches.

A. Sampling Problem Definition

Let $x \in R^N$ be a $N \times 1$ vector representing the depth map of an entire scene in a field of view (FOV) of a capturing device such as LiDAR. For straightforwardness, x is normalized such that $0 \leq x_i \leq 1$ for $i = 1, \ldots, N$. In general, a sensor device cannot acquire data for all the locations in the FOV such that the depth map of the entire FOV is reconstructed from the sampled data. Let M denote the number of samples that a sensor device can acquire. The *sampling problem* is an optimization problem of selecting the samples in the FOV to minimize the reconstruction error with the constraint that the number of the samples satisfies the target budget M. For mathematical formulation, let $\{1, \ldots, N\}$ denotes the set of indexes that correspond to the locations of the entire FOV, while $\{i_1, \ldots, i_{\mathrm{M}}\}$ represents the set of the indexes that correspond to the sample locations among $\{1, \ldots, N\}$.

Problem 1 (Sampling Problem): The sampling problem is to derive $\{i_1, \ldots, i_{\mathrm{M}}\}$ to minimize the following objective function

$$\min_{i_1,\ldots,i_M} \frac{1}{N} \sum_{j=1}^{N} \left(x_j - \widetilde{x_j}\right)^2 \tag{1}$$

where $x_1, \ldots, x_N$ are real values and $\widetilde{x_1}, \ldots, \widetilde{x_N}$ are the values that are estimated from M measurements $x_{i_1}, \ldots, x_{i_M}$.

Because it is not feasible to obtain a solution in a brute-force search manner, a heuristic method is most likely used. The next subsection presents a heuristic algorithm called Oracle random sampling or gradient-based sampling, which is derived in [13].

B. Gradient-Based Sampling

In [13], a probabilistic model is used to represent the sampling problem. For N locations in an FOV, a diagonal matrix $S \in R^{N\times N}$ is used to represent the sampling operation with the $(i, i)^{\text{th}}$ entry of S being

$$S_{ii} = \begin{cases} 1, & \text{with probability } p_i, \\ 0, & \text{with probability } 1 - p_i, \end{cases} \tag{2}$$

where $0 \leq p_i \leq 1$ for $i = 1, \ldots, N$ is a sequence of predefined probabilities.

Given S, the sampled data $b \in R^{N\times 1}$ is defined by:

$$b = Sx \tag{3}$$

where the i^{th} entry b_i is zero if $S_{ii} = 0$.

The target budget is defined by the target sampling ratio ξ with $0 < \xi < 1$, which represents the average sampling frequency. The following constraint is then obtained as follows:

$$\frac{1}{N}\sum_{i=1}^{N} p_i = \xi. \tag{4}$$

For a large N, the standard concentration inequality guarantees that the average number of ones in S is approximately ξN (i.e., $\xi N = M$) [13].

Similar to [13], this paper uses the gradient-based sampling method (called Oracle random sampling in [13]) to identify the edges or the highly textured areas of a given depth image. Let $a = [a_1, \ldots, a_N]^T$ be a vector that represents the magnitude of the gradient of the depth map:

$$a = \nabla x = \sqrt{(D_x x)^2 + \left(D_y x\right)^2}. \tag{5}$$

The intuition of the gradient-based sampling method is that the average gradient computed by all N samples is similar to the average gradient computed from a subset of ξN samples [13]. Let $\left\{p_j\right\}_{j=1}^{N}$ be the optimal sampling probability for defining the sampling map S. For a specified sampling ratio ξ and a gradient map, the derivation of the optimal sampling probability $\left\{p_j\right\}_{j=1}^{M}$ is formulated as the following optimization problem:

$$\min_{p_1,\ldots,p_N} \frac{1}{N}\sum_{j=1}^{N}\frac{a_j^2}{p_j} \tag{6}$$

subject to $\frac{1}{N}\sum_{j}^{N} p_j = \xi$and0 $\leq p_j \leq 1$. In [12], the solution is formulated as follows:

$$p_j = \min\left(\tau a_j, 1\right). \tag{7}$$

where τ is the solution of $g(\tau) = 0$ and $g(\tau)$ can be calculated as follows:

$$g(\tau) = \sum_{j}^{N} \min(\tau a_j, 1) - \xi N. \tag{8}$$

Note that $g(\tau)$ is a piecewise linear and monotonically increasing function, with $g(+\infty) = N(1-\xi)$ and $g(0) \leq 0$ [21]. Therefore, τ can be uniquely determined as the root of $g(\tau)$. Moreover, an efficient solution for the derivation of τ is available (see Appendix for details). As the gradient map of an image is not available prior to sampling, practical sampling is generally performed in two stages as described in Section I.

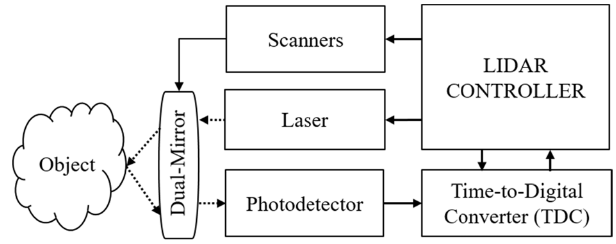


Fig. 1. Block diagram of a LiDAR system.

III. Sampling Problem in LiDAR Imaging With New Timing and Memory-Space Constraints

The sampling model in Section II-A is intuitive. However, it over-simplifies a practical LiDAR system because a timing constraint is not fully considered. The reason is that the derivation of an optimal sampling pattern is time-consuming, which consequently increases the overall data acquisition time even though the number of sampling points is reduced. Furthermore, a practical LiDAR system demands the minimal use of hardware resources such as memory footprint. To address these issues in the design of a practical LiDAR system, this section discusses the constraint required by a LiDAR system and presents a modified formulation of the sampling problem discussed in the previous section.

A. Timing Constraint in a Practical LiDAR System

A LiDAR system usually operates by performing multiple point-wise measurements in a FOV. A block diagram of a LiDAR is illustrated in Fig. 1. A typical measuring procedure of the LiDAR system is described as follows. A controller in the LiDAR system starts by computing a target location in the FOV, which requires a computation time t^{pos}. In the next step, the target position is transmitted to a mechanical scanner that controls motors and mirrors to direct the emitted light. This step requires the communication and motor control time, which is denoted by t^{scan}.After the mirror is aimed at the target, the laser diode in the LiDAR system emits a laser beam in time t^{emit}. Next, the LiDAR waits until the laser reaches an object and its reflected signal arrives at a photodetector. The time interval between the emitted and detected signals is generally referred to as time of flight (TOF) and is denoted as t^{TOF}. Finally, the measurement of t^{TOF} is converted to an electric signal and transmitted to the optical device controller that calculates the TOF from the signal. This time is denoted by t^{calc}. In the last step, the result is transmitted to the main controller that reads the signal in time t^{read}. For a given position, q_k, in an FOV, let t_k denote the time required to measure its distance. Therefore, t_k is a function $f(.)$ of the variables t^{pos}, t^{scan}, t^{emit}, t^{TOF}, t^{calc}, and t^{read}.Obviously, the upper bound of t_k is the summation of all variables when all steps are operated in a sequential manner. Meanwhile, the lower bound of t_k is the maximum among all variables assuming that all steps are operated in a pipelined manner.

To this end, t_k must satisfy both upper and lower bounds.

$$t_k = f\left(t_k^{pos}, t_k^{scan}, t_k^{emit}, t_k^{TOF}, t_k^{calc}, t_k^{read}\right) \tag{9-a}$$

$$t_k \geq \max\left\{t_k^{pos}, t_k^{scan}, t_k^{emit}, t_k^{TOF}, t_k^{calc}, t_k^{read}\right\} \tag{9-b}$$

$$t_k \leq t_k^{pos} + t_k^{scan} + t_k^{emit} + t_k^{TOF} + t_k^{calc} + t_k^{read} \tag{9-c}$$

Three variables t_k^{emit}, t_k^{calc} and t_k^{read} are likely to be fixed as their operations are the same for all measurements. Therefore, t_k usually depends on three remaining variables, t_k^{pos}, t_k^{scan}, and t_k^{TOF}.

The derivation of a sampling pattern affects t_k^{pos}, which indicates the time to determine the sampling point. If a sampling pattern is predetermined, it does not require the time to compute a target location (or $t_k^{pos} = 0$ for all k). On the other hand, the derivation of a complex sampling pattern might require a considerable amount of time so that t_k^{pos} becomes very large.

When the LiDAR system measures at M locations corresponding to indexes $i_1, \ldots, i_M$, the total time is expressed as follows:

$$t_{\sum}^{M} = \sum_{k=1}^{M} f\left(t_{i_k}^{pos}, t_{i_k}^{scan}, t_{i_k}^{emit}, t_{i_k}^{TOF}, t_{i_k}^{calc}, t_{i_k}^{read}\right). \tag{10}$$

Given a time budget T for scanning M locations, the following constraint must be satisfied:

$$t_{\sum}^{M} = \sum_{k=1}^{M} f\left(t_{i_k}^{pos}, t_{i_k}^{scan}, t_{i_k}^{emit}, t_{i_k}^{TOF}, t_{i_k}^{calc}, t_{i_k}^{read}\right) \leq T. \tag{11}$$

In practice, a LiDAR captures an image frame by frame, and therefore, the time budget T is usually set for a single frame. For example, T is 33 milliseconds if 30 frames are captured for every second. For the two-stage sampling in [13], the $M/2$ samples in the first stage are predefined so that they do not require time to calculate their locations. Meanwhile, the $M/2$ remaining samples require complex computation to derive their patterns, which results in longer computation times to generate the pattern (i.e., 20 seconds as reported in [13]). Therefore, this sampling does not satisfy the timing constraint:

$$\begin{aligned} t_{\sum}^{M} &\geq \sum_{k=1}^{M} max\left(t_{i_k}^{pos}, t_{i_k}^{scan}, t_{i_k}^{emit}, t_{i_k}^{TOF}, t_{i_k}^{calc}, t_{i_k}^{read}\right) \\ &\geq \sum_{k=1}^{M} t_{i_k}^{pos} \geq 20s. \end{aligned} \tag{12}$$

Obviously, $t_{\sum}^{M}$ becomes much larger than the time budget T that is, in general, a fraction of second in practice. Consequently, the two-stage sampling in [13] cannot be used for practical LiDAR sampling with the timing constraint considered.

B. Memory-Space Constraint in a Practical LiDAR System

This subsection presents an analysis of memory and bandwidth in a LiDAR system. Consider the practical case in which LiDAR is integrated in a system and a sampling method must satisfy the memory or bandwidth constraint of the system. In particular, let C denote the available memory capacity (or an available transmission bandwidth) for storing (or transmitting) LiDAR data. Meanwhile, let $G(.)$ be a function that represents the amount of stored/transferred data. Therefore, $G(.)$ depends on the sampling budget ξ, depth resolution n, and a sampling pattern S.

$$G(\xi, n, S) = G(M, n, i_1, \ldots, i_M\}. \tag{13}$$

In addition, the amount of data $G(\xi, n, S)$ must satisfy the following inequality:

$$G(\xi, n, S) \leq C \tag{14}$$

The straightforward derivation of $G(\xi, n, S)$ for a given non-uniformly random sampling (i.e., two-stage sampling in [13]) is described as follows. Because each pixel in b consists of n bits, the amount of data becomes $n \times \xi \times N$ bits. In addition, the sampling pattern S is also stored and/or transmitted because it is non-uniform and irregular. Because one bit is necessary for each pixel in the input image of size N, the amount of data for S is N. Therefore, the total amount of data $G(\xi, n, S)$ is derived as follows:

$$G(\xi, n, S) = n \times \xi \times N + N \tag{15}$$

Generally, for a practical capturing device, the memory space inside the device and/or the transmission bandwidth to the external system is limited. Therefore, it is necessary to select the sampling ratio to satisfy the memory and/or bandwidth requirement. By combining (14) and (15), with a given memory and/or bandwidth capacity C, the amount of data $G(\xi, n, S)$ must satisfy the following inequality:

$$n \times \xi \times N + N \leq C \tag{16}$$

From (16), the target sampling ratio ξ is limited by the available memory space C, resolution n, and image size N:

$$\xi \leq \frac{C - N}{n \times N} \tag{17}$$

A new terminology target compression ratio, χ, is defined to represent the ratio of the size of the available memory space C to the size of the input image ($n \times N$):

$$\chi = \frac{C}{n \times N} \tag{18}$$

The compression ratio represents the extent to which the original data should be compressed to satisfy the available memory capacity. The relationship between the sampling ratio ξ and compression ratio χ can be obtained from (17) and (18) and can be expressed as follows:

$$\xi \leq \chi - \mathrm{N}/(n \times N) \tag{19}$$

Uniform grid sampling does not require the storage of the sampling map S because the pattern is fixed. Therefore, the required memory space $G(\xi, n, S)$ is modified as follows:

$$G(\xi, n, S) = n \times \xi \times N \leq C \tag{20}$$

where the second term in (16) is removed. In this case, the sampling ratio becomes

$$\xi \leq \frac{C}{n \times N} = \chi \tag{21}$$

Example 1: A depth image of size 512×512 with 8-bit resolution (data size = 256KB) is sampled and stored in memory of size 64 KB (i.e., $N = 32$ KB, $n = 8$, and $C = 64$ KB). This implies that 32 KB (=64 KB - 32 KB) is used to store the sampled depth map b. If b is obtained from the non-uniform sampling presented in Section II-B, the sampling ratio ξ is limited to the following value:

$$\xi = \frac{C - N}{n \times N} = \frac{64 - 32}{8 \times 32} = 12.5\% \tag{22}$$

On the other hand, the uniform grid uses 64 KB to store b, thereby resulting in the following value of ξ

$$\xi = \frac{C}{n \times N} = \frac{64}{8 \times 32} = 25\% \tag{23}$$

This result demonstrates that the uniform grid stores twice as many samples as the non-uniform grid does. This implies that the uniform grid is likely to achieve higher image quality than that of the non-uniform grid when memory space is limited, which is true in numerous real-world applications.

C. New Sampling Problem With Constraints

Based on those two constraints discussed in the previous two subsections, *Problem 1* in Section II-A is modified as follows:

Problem 2 (New Sampling Problem for LiDAR): The sampling problem is to derive $\{i_1, \ldots, i_M\}$ to minimize the following objective function:

$$\min_{i_1,\ldots,i_M} \frac{1}{N} \sum_{j=1}^{N} \left(x_j - \widetilde{x_j}\right)^2 \tag{24}$$

subject to the following two constraints:

a. (timing constraint)

$$t_{\Sigma}^{M} = \sum_{k=1}^{M} f\left(t_{i_k}^{pos}, t_{i_k}^{scan}, t_{i_k}^{emit}, t_{i_k}^{TOF}, t_{i_k}^{calc}, t_{i_k}^{read}\right) < T$$

b. (memory-space constraint)

$$G(M, n, i_1, \ldots, i_M) \le C$$

where $x_1, \ldots, x_N$ are the real values and $\widetilde{x_1}, \ldots, \widetilde{x_N}$ are values that are estimated from M measurements $x_{i_1}, \ldots, x_{i_M}$. *Problem 2* is modified into two variations by ignoring either the timing or memory constraint. *Problem 2a* is the same problem as *Problem 2* with the removal of the memory-space constraint whereas *Problem 2b* is derived from *Problem 2* by removing the timing constraint.

While the use of non-uniform sampling enhances the image quality of the reconstructed image, it involves considerable computational complexity and additional memory space. This illustrates the trade-off between higher image quality and faster execution time/larger memory requirement. This paper proposes a novel algorithm that improves image quality while reducing computational complexity and memory requirements.

IV. The Proposed Sampling Algorithm and Its Properties

A. Sampling and k-NN Expanding Operator

To reduce the required memory space, the proposed algorithm attempts to reduce the second term on the right side in (15). The concept is explained using an example illustrated in Fig. 2. Fig. 2(a) illustrates the "Ellipse" image. The down-sampled image of Fig. 2(a) (by 3:1 yields) is illustrated in Fig. 2(c) (i.e., one ninth of the original image). Notwithstanding the down-sampling operator, the down-sampled image still perceptually captures object boundaries and textured patterns. These characteristics are shown in Figs. 2(b) and 2(d), which present their gradients of Figs. 2(a) and (c), respectively. It should be noted that in the down-sampled image, object boundaries and textured patterns are perceptually captured in most regions. Fig. 2(e) demonstrates a sampling map generated from the "down-sampled" gradient image in Fig. 2(d) in which sampled points are densely located in objects boundary. It suggests that a "down-sampled" sampling map in Fig. 2(e) can be considerred as an indicator to detect texture areas in a scene.

The proposed algorithm captures the gradient information ∇x from the down-sampled image and then derives sampled depth map b from the down-sampled gradient information. If the image is down-sampled by 3:1 in both the horizontal and vertical directions, the size of the sampled depth map is reduced to 1/9 of the original image. The second term of the right side in (15) is also decreased to $N/9$. On the other hand, the down-sampling results in aliasing artifact in the highly textured or boundary regions. This implies that image quality will likely degrade because of the loss of information by down-sampling. Increasing the number of samples facilitates the capture of more information (e.g., details of the boundary object or highly textured areas). To reduce this artifact, the proposed algorithm uses additional samples in the textured region. In case that a pixel is in a highly textured region, its neighboring pixels are also likely to be in the highly textured region because of the non-local image characteristic. Utilizing this characteristic, the proposed algorithm selects samples in the neighbors of the sampled data observed in a highly-texture region. To achieve this, the algorithm uses the k-NN expanding operation, which is explained in Fig. 3.

Example 2: Figs. 3 (a) and (b) demonstrate a 12×12 image and its sampled points, respectively. Each node in Fig. 3(b) is mapped to a pixel marked with a "gray" color in Fig. 3(a) that can be considered as a representative node of its eight neighbor nodes. The proposed k-NN expanding sampling operator functions as follows. In Fig. 3(c), the central node is marked with label 0, indicating that it is in a smooth area. The proposed k-NN operator predicts that all eight neighbors of this node are also in the smooth area by assigning zero to their labels; and eventually those pixels are skipped during sampling. On the other hand, the central nodes in Figs. 3(d), (e), and (f) are marked as one, indicating that they are in a highly textured area or on the boundary of an object. The k-NN expanding operator sets theirs neighbors as "1" to predict them to be in a highly textured area. In particular, four

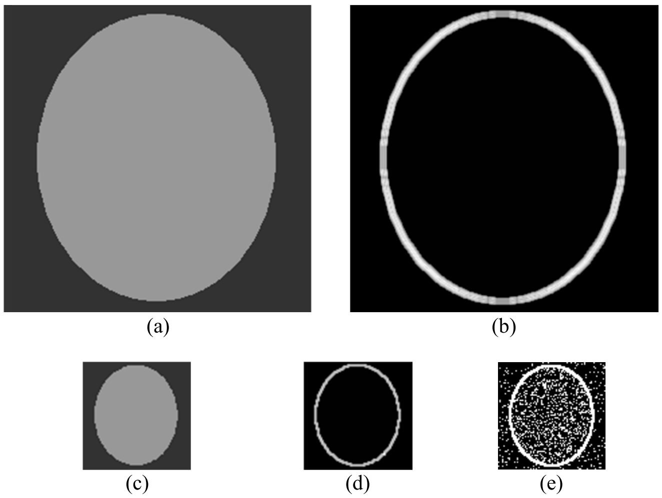


Fig. 2. Example of a down-sampling operator and its graph representation. (a) –(b) "Ellipse" image and its gradient; (c)-(e) a down-sampled image (1:3), its gradients, and its gradient-based sampling map.

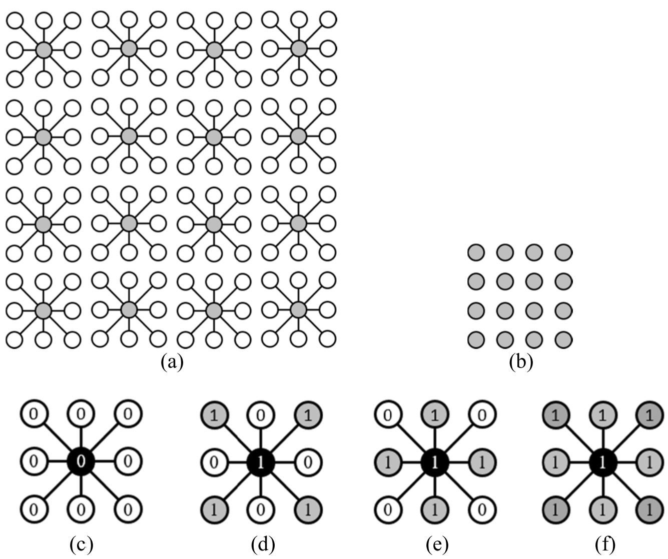


Fig. 3. An example of the k-NN expanding operator: (a)-(b) a 12×12 image. (b) sampled points of the 12×12 image. (c) all neighbors are '0' if center is '1', (d) four neighbors ($k = 4$) forming a **square** is '1' and four others are '1' if center is '1', (e) four neighbors ($k = 4$) forming a **diamond** is '1' and four others are '1' if the node is '1', and (f) eight neighbors ($k = 8$) are '1' if center is '1'.

of eight neighbors ($k = 4$) are predicted as in a highly texture area and marked with "1" in Figs. 3(d) and (e), while eight neighbors ($k = 8$) are marked with "1" in Fig. 3(f). To this end, the pixels marked with "1" are sampled during sampling.

The sampling map is constructed as follows: The sampling map consists of all the representatives and their neighbors marked by non-zero labels. In Fig. 3(c), the four neighbors have not been included in the sampling map as they are marked zero. Meanwhile, the four neighbors marked with one in Figs. 3(d)-(e) and the eight neighbors in Fig. 3(f) are added to the final sampling set.

The following subsection presents the proposed sampling method using the k-NN expanding sampling operator.

B. The Proposed Sampling Scheme

The proposed sampling procedure consists of two stages: a pilot stage to obtain a coarse regular sampling map and a refinement stage to enhance the sampling map. The pilot stage selects the partial ratio α ($0 < \alpha < 1$) from the budget. A uniform grid sampling is used in this stage, resulting in sampling map $S^{(1)}$ with $\alpha \times \xi \times N$ non-zero elements. The sampling period is identical in both the horizontal and vertical directions such that it is straightforwardly defined by step as follows:

$$step = \sqrt{\frac{1}{\alpha \times \xi}} \tag{25}$$

For a specified image, let W and H denote its width and height, respectively. Then, $N = W \times H$ can be obtained. In uniform sampling with the step expressed in (25), $S_{ii} = 1$ if and only if the index i satisfies the following condition:

$$i = \lfloor i_H \times step \rfloor \times W + \lfloor i_W \times step \rfloor \tag{26}$$

where $\lfloor . \rfloor$ represents the floor operation, and i_H and i_W are integer numbers such that the corresponding $\lfloor i_H \times step \rfloor$ and $\lfloor i_W \times step \rfloor$ are the coordinates of the pixel in the 2D image. Apparently, i_H and i_W satisfy the following conditions, $i_H \in \left\{1, 2, \ldots, \left\lfloor \frac{H}{step} \right\rfloor\right\}$ and $i_W \in \left\{1, 2, \ldots, \left\lfloor \frac{W}{step} \right\rfloor\right\}$.

Given $S^{(1)}$, the sampled depth map $b^{(1)}$ is derived as follows:

$$b^{(1)} = S^{(1)} x \tag{27}$$

and its corresponding down-sampled map $x^{(1)}$ is down-sampled by the step in (25). The size of $x^{(1)}$ corresponds to $M_\alpha = \alpha \times \xi \times N$, where M_α represents the size of $x^{(1)}$ hereafter in this paper. It must be noted that this is the main variation between the proposed study and that in [13]. The map $x^{(1)}$ in the proposed study is straightforwardly a down-sampled image of a substantially smaller size than that of the original image. Fig. 4 illustrates a visual comparison between the proposed sampling scheme and that of [13]. Fig. 4(a) illustrates the proposed sampling scheme where $x^{(1)}$ is directly derived by down-sampling operation. Meanwhile, the $x^{(1)}$ in [13] is the reconstructed image derived from $S^{(1)}$ as illustrated in Fig. 4(b).

In the second stage, the down-sampled image $x^{(1)}$ is used as a guide to compute the gradient $\nabla x^{(1)}$. The gradient map $\nabla x^{(1)} = \left[a_1, \ldots, a_{M_\alpha}\right]^T$ is derived as explained in Section II-B. By formulating the optimization problem of (6), the optimal probability and sampling map S_η are obtained. By applying the k-NN expanding operator to S_η, where S_η is used as the set of representatives for the original image, $S^{(2)}$ can be obtained. The sampling ratio η is separate from ξ, and it is derived as follows.

Recall that the size of S_η, the set of representatives for the original image, is equal to that of down-sampled image $x^{(1)}$. Based on the k-NN expanding operator in Section IV-A, for each representative, k neighboring points are added into the final sampling map. Note that the number of remaining samples is $(1 - \alpha) \times \xi \times N$ because $M_\alpha = \alpha \times \xi \times N$ samples are used in the pilot stage. Using the k-NN expanding operator,

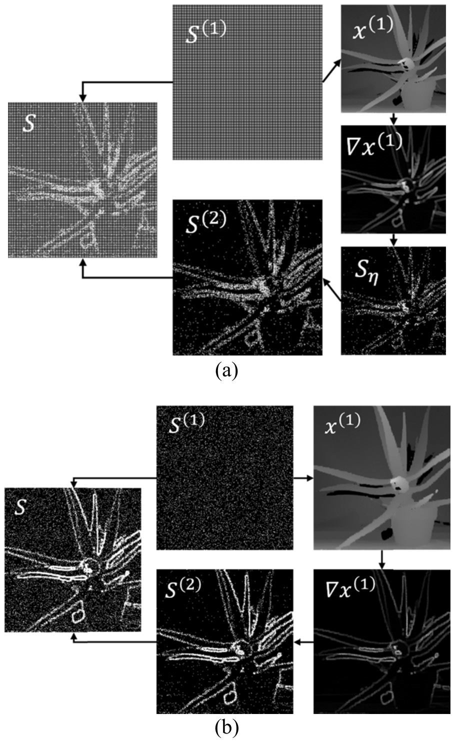


Fig. 4. Comparison of two sampling algorithms: (a) the proposed sampling algorithm and (b) two-stage algorithm in [13].

Alg. 1: Proposed Sampling Algorithm

Input: $\mathrm{N}, \xi, b, k, \alpha$
Output: S

Pilot Stage
Obtain $S^{(1)}$ by a uniform grid sampling with a budget ratio $\alpha \times \xi$.
Compute $b^{(1)}$ from $S^{(1)}$.
Compute $x^{(1)}$ from $b^{(1)}$.

Refinement Stage
Compute $\nabla x^{(1)}$.
Conduct the optimal sampling on $\nabla x^{(1)}$ to obtain the set S_η with the sampling ratio $\eta = \frac{1-\alpha}{k \times \alpha}$.
Compute $S^{(2)}$by k-NN expanding of S_η.
Compute $S = S^{(1)} + S^{(2)}$.

Fig. 5. Proposed sampling algorithm with a k-NN expanding operator.

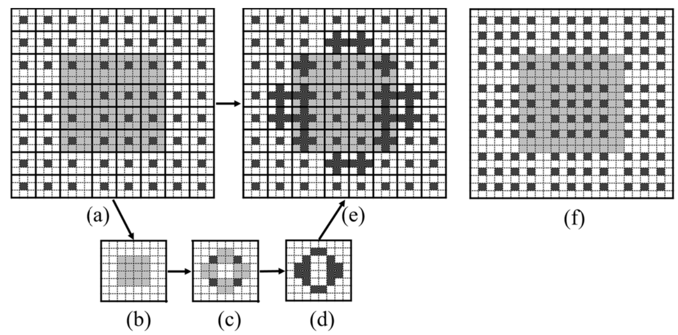


Fig. 6. An example of the operation results by the proposed algorithm: (a) $S^{(1)}$ (b) $x^{(1)}$ (c) $\nabla x^{(1)}$ (d) S_η (e) $S = S^{(1)} + S^{(2)}$ and (f) the grid sampling.

each of the representatives are extended to their k neighboring pixels. Consequently, the number of representatives added in this refinement stage can be derived by dividing the remaining samples by k and can be expressed as follows:

$$\frac{(1-\alpha) \times \xi \times N}{k} \tag{28}$$

This results in the sampling ratio η for the down-sampled image $x^{(1)}$, which can be expressed as follows:

$$\eta = \frac{\frac{(1-\alpha)\times\xi\times N}{k}}{M_\alpha} = \frac{(1-\alpha) \times \xi \times N}{k \times \alpha \times \xi \times N} = \frac{1-\alpha}{k \times \alpha} \tag{29}$$

Given S_η, the proposed k-NN expanding operator in Section IV is applied for the derivation of the refined sampling map $S^{(2)}$. As each representative expands only to its neighbors, $S^{(1)}$ and $S^{(2)}$ are exclusive. Therefore, the final sampling map S, is obtained by $S = S^{(1)} + S^{(2)}$. The algorithm is summarized in Fig. 5.

Example 3: Assume that the target sampling ratio is 20% ($\xi = 0.2$) and half of the budget is used at the first stage ($\alpha = 0.5$). Then, the size of the down-sampled image in the first stage includes 10% of the original image's size. The refinement stage applies the straightforward four-node pattern ($k = 4$). To obtain the remaining 10% of the samples, 2.5% of the representatives must be selected from the down-sampled image. From (28), the sampling ratio η in the down-sampled image is selected as 25%.

$$\eta = \frac{1-0.5}{4 \times 0.5} = 0.25. \tag{30}$$

C. An Example Using Synthetic Data

An example of the implementation of the proposed sampling algorithm is illustrated in Fig. 6. The synthetic image has 25×25 pixels, and the sampling ratio is 20% ($\xi = 0.2$). Fig. 6(a) illustrates the sampling map $S^{(1)}$ when 10% is used by a uniform grid sampling ($\alpha = 0.5$). From (25), the sampling period is derived as $step = \sqrt{10}$. Without loss of generality, the starting index is selected as 2 (=1 + $step/2$) such that a pattern is derived (see Fig. 6(a)). Therefore, the 8×8 down-sampled image $x^{(1)}$ is obtained in Fig. 6(b). For convenience, $x^{(1)}$ exhibits a straightforward shape having a smooth 4×4 square in its center. The gradient $\nabla x^{(1)}$ is illustrated in Fig. 6(c). The refinement stage applies the straightforward 4-NN pattern ($k = 4$) in Section IV. 16 representatives are then selected from the available 8×8 gradient image. The feasible solution of (6) is displayed in set S_η in Fig. 6(d), wherein the black pixels represent highly-textured areas, and the white pixels indicate smooth ones. It is random in a general case. However, a feasible solution is selected as an example because the randomness is not likely to be true if the number of samples is substantially small. The use of a random solution will likely result in bias such that the selected samples are not located on the object boundary. Finally, an expanding operator is applied to S_η and then the derived sampling $S^{(2)}$ is merged to $S^{(1)}$ to form the final sample S, which is illustrated in Fig. 6(e).

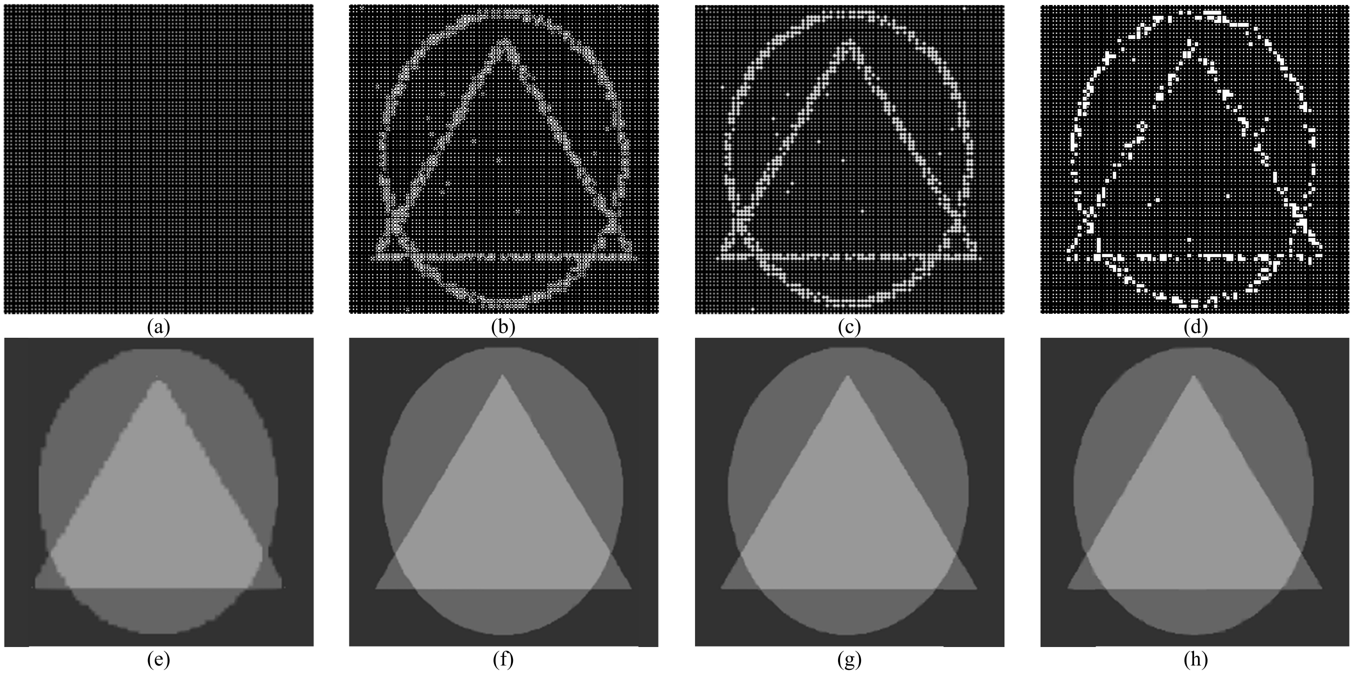


Fig. 7. Example of sampling patterns with the proposed k-NN expanding sampling operator. (a) Grid sampling pattern, (b), (c), (d) Sampling patterns by adding 5% sampling points along gradients into the pattern in (a). Similar to Fig. 3, the expanding patterns in (b), (c) and (d) are "4-NN, square", "4-NN, diamond" and "8-NN", respectively. Figs. (e), (f), (g), and (h) are reconstructed images corresponding to sampling patterns in (a), (b), (c) and (d), respectively.

In S, the white squares are omitted while the remaining ones are sampled. Meanwhile, the uniform grid-sampling pattern is illustrated in Fig. 6(f) as the entire sampling budget is used to derive the pattern. The sampling pattern in Fig. 6(e) preserves more points in the boundary area than that in Fig. 6(f). Note that the difference between $S^{(1)}$ and $S^{(2)}$ is that $S^{(1)}$ is the representative obtained from $S\eta$ and $S^{(2)}$ represents the pixels expanded from $S^{(1)}$, which does not include $S^{(1)}$. For example in the figure below, $S^{(1)}$ represents the shaded pixels in Fig. 6(a) and $S^{(2)}$ represents the neighboring pixels around the shaded pixels in Fig. 6(e). Because $S^{(2)}$ does not include the shaded pixels, $S^{(2)}$ is exclusive to $S^{(1)}$.

D. Impact of the Proposed k-NN Expanding Operator

This subsection demonstrates the impact of the proposed k-NN expanding operator. Fig. 7 illustrates an example of patterns with the k-NN expanding sampling operator. The grid-sampling pattern at $\xi = 0.1$ (i.e., 10%) is shown in Fig. 7(a). Patterns in Figs. 7(b), (c), and (d) are derived by adding 5% sampling points along gradients into the sampling pattern in Fig. 7(a) with the k-NN expanding operator. Similar to Fig. 3, expanding patterns in Figs. 7 (b), (c) and (d) are "4-NN, square", "4-NN, diamond," and "8-NN", respectively. Figs. 7(e), (f), (g), and (h) demonstrate the reconstructed images corresponding to the sampling patterns in Figs.7(a), (b), (c) and (d), respectively. It is clearly shown that in all three cases, the k-NN expanding sampling operator creates additional sampling points in the texture region along gradient, and consequently enhances the reconstruction quality. Visually, thanks to the proposed k-NN expanding sampling operator, the texture regions along gradient of the reconstructed images

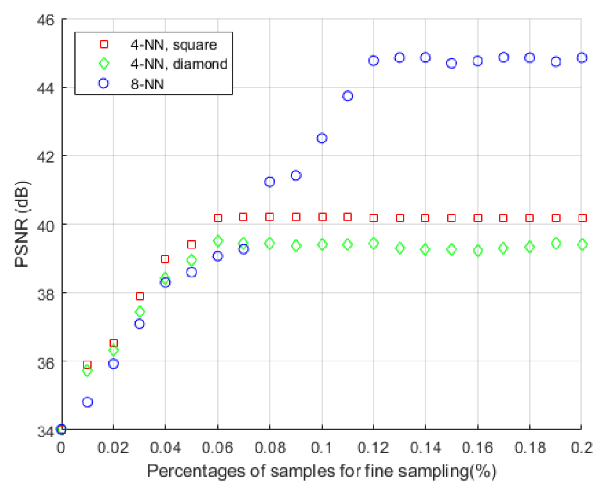


Fig. 8. Comparison of the reconstruction quality among sampling patterns. The construction methods are an alternating direction method of multipliers (ADMM) with wavelet dictionary.

in Figs. 7(f), (g), (h) are nicely captured. The detail reconstruction performance is reported in Fig. 8. The experiment environment is built as follows. The sampling procedure starts with 10% samples in a grid sampling pattern, which is the pilot stage in the algorithm in Fig. 5. By using the k-NN expanding operator, it creates additional sampling points in the texture region along gradient. The percentage of additional sampling points is set from 1% to 10%. The experimental results demonstrate that adding only 1% sampling points along gradient by the proposed method can significantly enhance the reconstruction performance. For example, compared to the grid

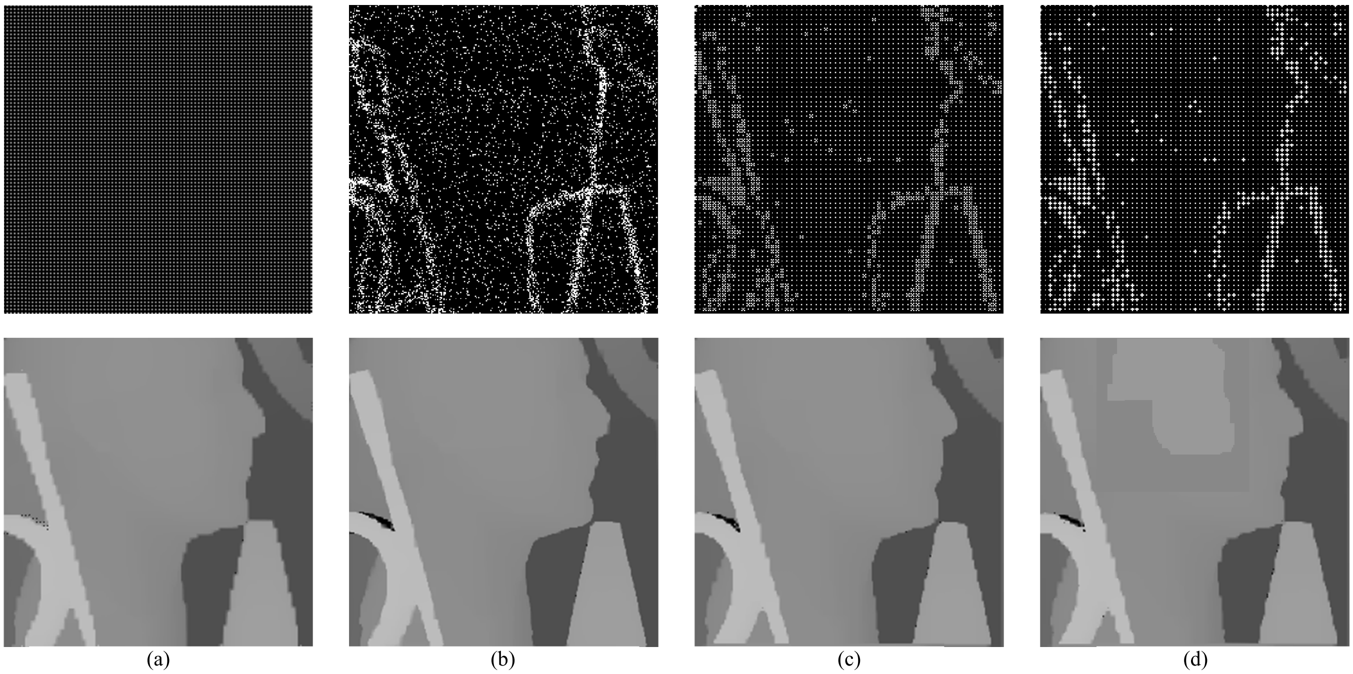


Fig. 9. Comparison between four sampling patterns. (a) Uniform grid, (b) Two-stage sampling in [13]; (c) Proposed sampling with a 4-NN square-expanding operator; (d) Proposed sampling with a 4-NN diamond-expanding operator. A Monte-Carlo simulation with 32 independent trials is conducted. The averages of PSNRs are presented in the Table II.

sampling at the pilot stage, the sampling with the proposed k-NN expanding operator has about 1.58 dB improvement with 1% additional sampling points, and can be further improved by almost 8.45 dB with 10% additional sampling points. It is observed that "4-NN, squared" sampling consistently performs slight better than "4-NN, diamond" with 0.54 dB improvement on average, ranging from 0.23 dB to 0.82 dB. Moreover, it is also observed that the reconstruction performance of "4-NN, diamond" and "4-NN, square" methods become saturated during creating additional sampling points. In particular, for two cases, no clear improvement is observed after the percentage of adding points is over 6%. Meanwhile, it is observed that the 8-NN case enhances the performance almost linearly when the percentage of adding points is up to 12%. Different from 4-NN expanding cases, the 8-NN case takes advantages of both patterns of "4-NN, diamond" and "4-NN, square". Consequently, it is observed that its performance keeps increasing when the percentage of adding points increases. To this end, because of the k-NN expanding sampling operator, the text region along gradient is sampled more frequently than the other region, which significantly improves the reconstruction performance.

TABLE I

AVERAGE RUNNING TIME OF PROPOSED ALGORITHM IN MILLISECONDS. A MONTE-CARLO SIMULATION WITH 100 INDEPENDENT TRIALS IS CONDUCTED

Test Image	Running time for sampling ratio (ms)				
	5%	10%	15%	20%	25%
Aloe	7.113	8.209	10.537	12.247	14.132
Art	6.054	8.058	10.968	12.299	13.837
Baby	6.314	7.907	11.322	12.140	13.512
Dolls	6.131	7.978	10.382	12.214	13.615
Moebius	6.178	8.032	10.392	12.646	13.753
Rocks	6.170	8.541	10.162	12.261	13.725
Average	**6.327**	**8.121**	**10.627**	**12.301**	**13.762**

TABLE II

COMPARISON OF THE AVERAGE PSNRS

Method	Actual sampling ratio	Average PSNR(dB)	PSNR difference(dB)
Uniform grid	0.1128	30.2726	-
Two-stage sampling [13]	0.1002	33.7707	+3.50
Proposed w/ 4-NN square	0.0997	33.1115	+2.84
Proposed w/ 4-NN diamond	0.0995	31.4459	+1.17

E. Timing and Memory Constraints

This subsection discusses the memory requirements and the time complexity of the proposed algorithm to test whether the proposed algorithm satisfies the two constraints in *Problem 2*.

1) Timing Constraint: Among the timing parameters discussed in Section III-A, t^{pos} indicates the time for generating a refinement sampling pattern $S^{(2)}$. The derivation of $x^{(1)}$ in the pilot stage (lines from 3 to 6 in Fig. 5) is not time-consuming because it is a result of uniform grid sampling. Next, the operation in line 8 in Fig. 5 is a simple derivation of a gradient and takes less than 10 cycles in most hardware circuits. In addition, this computation can be executed on-the-fly if the input is received in the raster-scan order. The computation of the line 9 in Fig. 5 aims to obtain a root of $g(\tau)$ that is a piecewise linear and monotonically increasing function, with $g(+\infty) = N(1-\xi)$ and $(0) \leq 0$. Therefore, the root is unique and its derivation is performed in an iterative way. This step, in practice, is likely to obtain a root within a

TABLE III
THE PSNR OF THE RECONSTRUCTED IMAGE

Test Images	Sampling Methods	Target compression ratio (χ) 5%	10%	15%	20%	25%
Aloe	Grid	25.32	28.91	30.09	31.30	32.35
	[13] w/ RGB	27.60	31.39	33.37	36.41	38.63
	[13]	27.87	31.36	34.12	36.31	39.33
	k=4 square	28.34	31.52	34.09	35.89	36.50
	k=4 diamond	27.70	30.98	33.61	34.78	36.18
	k=8	26.54	30.75	33.75	36.54	40.91
Art	Grid	27.52	28.95	30.84	32.52	33.71
	[13] w/ RGB	30.87	34.15	37.28	42.97	48.00
	[13]	29.55	33.04	35.77	37.54	39.60
	k=4 square	28.75	33.11	35.29	36.47	37.85
	k=4 diamond	28.22	32.19	34.64	35.90	37.00
	k=8	27.86	31.97	35.71	38.18	43.38
Baby	Grid	34.44	36.80	37.67	39.05	40.07
	[13] w/ RGB	39.70	44.90	48.66	52.50	52.00
	[13]	38.55	43.05	46.51	49.17	54.59
	k=4 square	36.13	39.38	42.42	43.49	44.32
	k=4 diamond	35.54	39.07	41.49	43.53	44.19
	k=8	35.77	39.99	44.73	48.08	53.58
Dolls	Grid	28.49	29.05	30.09	30.81	31.67
	[13] w/ RGB	29.51	32.53	34.00	36.27	37.65
	[13]	29.32	31.52	32.92	36.88	38.20
	k=4 square	29.00	31.03	32.40	34.24	35.13
	k=4 diamond	28.77	30.57	32.06	33.73	34.47
	k=8	28.84	31.37	33.48	35.72	37.32
Moebius	Grid	27.69	28.72	29.85	31.17	32.24
	[13] w/ RGB	31.07	35.11	37.76	39.92	41.89
	[13]	29.95	32.86	36.11	38.44	40.79
	k=4 square	28.93	31.88	34.00	35.36	36.21
	k=4 diamond	28.63	30.86	33.29	34.65	35.79
	k=8	28.89	32.16	35.05	37.55	41.10
Rocks	Grid	27.69	28.72	29.85	31.17	32.24
	[13] w/ RGB	31.07	35.11	37.76	39.92	41.89
	[13]	29.95	32.86	36.11	38.44	40.79
	k=4 square	28.57	32.06	34.28	36.33	37.01
	k=4 diamond	27.98	30.93	33.45	35.36	36.70
	k=8	28.77	32.76	35.99	40.15	46.74
Average	Grid	28.52	30.19	31.40	32.67	33.71
	[13] w/ RGB	31.63	35.53	38.14	41.33	43.34
	[13]	30.87	34.12	36.92	39.46	42.22
	k=4 square	29.95	33.16	35.41	36.96	37.84
	k=4 diamond	29.47	32.43	34.76	36.32	37.39
	k=8	29.44	33.17	36.45	39.37	43.84

small number of iterations (i.e., 64 iterations) and therefore, it can be processed within 1,000 cycles in most hardware. For more information, a flowchart of the iterative algorithm is given in Appendix B. Note that experimental results also demonstrate that the number of iterations is less than 32 for all locations to be derived. The last step in Line 10 is simply an expanding operator. Given a sampling pattern in line 9 in Fig. 5, it is straightforward that this step can executed within a single hardware cycle. In summary, the proposed algorithm can be implemented in hardware within 1,000 cycles, taking less than 10 microseconds for an operating clock frequency of 100MHz.

In general, the time complexity of the proposed algorithm can be analyzed as follows. A sequential visit of sampling points with the steps defined above permits the simultaneous derivation of $S^{(1)}$, $b^{(1)}$, and $x^{(1)}$. This stage is executed in linear time, $O(M)$, with respect to the number of pixels. In the refinement stage, the gradient computation and the gradient-based optimization are executed in linear time, $O(M)$. These steps are performed using the down-sampled image, which further reduces the complexity. The expanding step is also fast and executes in linear time, $O(M)$. In summary, the proposed sampling algorithm satisfies the timing constraints of the LiDAR system.

2) Memory-Space Constraint: The memory space (or communication bandwidth) required in the proposed algorithm can be analyzed in a manner similar to that in Section III-B. The sampling map, $S^{(1)}$ in the pilot stage is uniform grid such that it does not require a memory space for storage. Meanwhile, the sampling map, S_η, in the refinement stage must be stored because it is a non-uniform pattern. The size of S_η is equal to that of the down-sampled image, which requires $\alpha \times \xi \times N$ bits. In addition, the sampled depth map requires memory space of size, $n \times \xi \times N$, where n indicates the resolution of the depth map. Therefore, the total memory space can be expressed as follows:

$$G(\xi, n, S) = n \times \xi \times N + \alpha \times \xi \times N \tag{31}$$

Given the available memory space C, as the sampling ratio is limited, it can be expressed by the following inequality:

$$\xi \leq \frac{C}{(n+\alpha) \times N} \tag{32}$$

When compared with the sampling ratio ξ in (17), the value in (31) is significantly bigger because α is substantially smaller than n, and N is substantially larger than n and α. This analytical comparison demonstrates that the proposed algorithm has a sampling ratio ξ, which is substantially larger than that of the previous algorithm in [13]. This illustrates that the proposed algorithm has a larger number of pixels in the sampled depth map, thereby resulting in higher reconstruction quality.

Example 1 (Continued): For an equivalent memory space and the depth image specified in Example 1, the proposed algorithm is used to derive the sampled depth map. If half of the samples is selected in the first stage (i.e., $\alpha = 0.5$), the sampling ratio is derived from (31):

$$\xi \leq \frac{64}{(8+0.5) \times 32} = 23.5\% \tag{33}$$

This value is much larger than that for [13] in Example 1 ($\xi = 12.5\%$) and is close to the value for the uniform grid sampling ($\xi = 25\%$).

V. EXPERIMENTAL RESULTS

This section presents an evaluation of the proposed sampling method in comparison to three reference algorithms, uniform grid sampling, gradient-based optimal sampling in [12], and two-stage sampling in [13]. Section V-A presents the results for the conventional sampling problem without considering both the memory-space and timing constraints. Section V-B shows the results for *Problem 2b*, which is derived from *Problem 2* by maintaining the memory space constraint but ignoring the timing constraint. On the other hand, Section V-C compares the proposed and uniform grid sampling approaches that satisfy the timing constraint.

A. Evaluation for the Conventional Sampling Problem

The conventional sampling problem is formulated to find a sampling pattern without timing and memory-space constraints. This section compares the running time to output a sampling pattern of the proposed sampling scheme and the two-stage sampling [13]; and then check how the proposed sampling can close the performance gap between the grid sampling and the two-stage sampling [13].

1) Run-Time Evaluation: For fair evaluation, both algorithms are tested under the same platform of MATLAB 2018b / 64-bit Windows 10 / Intel core I5 / CPU 3.2 GHz (single thread) / 8 GB RAM. Because the hardware setting in the proposed method is different from that in [13], it is observed that the two-stage sampling method takes approximately 90 seconds due to the reconstruction time to derive a rough image. Meanwhile, for the proposed sampling, a Monte-Carlo simulation by repeating 100 independent trials is conducted, and the averages of running time are reported in Table I. The average running time of six test images is only about 6.327, 8.121, 10.627, 12.301, and 13.762 milliseconds for sampling ratios of 5%, 10%, 15%, 20%, and 25%, respectively. By eliminating time-consuming reconstruction and using a simple yet effective k-NN expanding operator, the proposed scheme only consumes a few milliseconds, which can meet a timing constraint in a LiDAR system.

2) Subjective Comparison: As a comparison between sampling patterns, this experiment considers a disparity map shown in Fig. 9. Setting $\xi = 0.1$ (i.e., 10%), four sampling patterns including two versions of the proposed method are evaluated. A Monte-Carlo simulation by repeating 32 independent trials is conducted, and the averages of peak signal-to-noise ratio (PSNR) values are reported in Table II. The results shown in Fig. 9(c) are generated using the sampling scheme with a 4-NN square expanding operator, whereas the results shown in Fig. 9(d) are generated by using a 4-NN diamond-expanding operator. These results indicate that for the same sampling ratio ξ, the choice of the sampling pattern has a strong influence to the reconstruction quality. For example, as compared to the grid sampling, the two-stage sampling [13] has about 3.50 dB improvement. It is noteworthy that this performance gap is achieved by using a reconstructed image to derive a sampling pattern, which usually consumes a considerable amount of time. Meanwhile, the k-NN sampling method gives a simple yet effective way to create additional sampling points in the text region, and consequently enhances the reconstruction performance. Compared to the grid sampling, the proposed sampling has about 1.17 dB improvement with a 4-NN diamond expanding sampling operator, and can be further improved by 2.84 dB using a 4-NN square expanding operator. Especially, compared to the sampling scheme [13], the sampling with a 4-NN square expanding operator only degrades by 0.66 dB while substantially reducing the computation time.

3) Quantitative Evaluation: The experiments for quantitative evaluation are conducted using the six testbeds in the Middlebury datasets: Aloe, Art, Baby2, Moebius, Dolls, and Rocks [22], [23]. Table III shows the PSNR values at different sampling ratios and sampling methods. For this evaluation, the alternating direction method of multipliers with the wavelet and contourlet dictionaries is used as the reconstruction algorithm. The details of this method are available in [13], and the toolbox[1] of [13] is publicly provided by the authors. The proposed sampling scheme is evaluated in three expanding patterns as described in Fig. 3: (1) k =4, square, (2) k =4, diamond, and (3) k =8. For fair evaluation, the PSNR results of the two-stage sampling method in [13] are reported in two cases: 1) use a half of sampling budget to sample along the gradient of an RGB image in the first sampling stage as reported in [13] (marked with “RGB”)[2]; and 2) do uniformly random sampling in the first stage. Experimental results demonstrate that the proposed methods outperform the grid sampling[3] with a large margin. In particular, compared to the grid, the proposed method marked with “k =4, square” improves PSNR by 0.93, 2.24, 3.58, 3.76, and 3.87 (dB) corresponding to percentage of samples of 5%, 10%, 15%, 20%, and 25%, respectively. Furthermore, the proposed sampling method marked with “k = 8” achieves 0.34, 1.88, 3.63, 5.03, and 6.53 (dB) improvements when being compared with the grid. Even compared to the state-of-art two-stage ones in [13], the proposed method degrades PNSRs by 2.04, 2.20, 2.35, 2.09, and 2.46 dB for sampling ratios of 5%, 10%, 15%, 20%, and 25%, respectively.

TABLE IV

THE SAMPLING RATIO (ξ) MATHEMATICALLY DERIVED FROM (17), (21), AND (32) FOR A GIVEN TARGET COMPRESSION RATIO (χ)

Sampling methods	Target Compression Ratio (χ) 5%	10%	15%	20%	25%
Uniform grid	5%	10%	15%	20%	25%
[12]	N/A	N/A	2.5%	7.5%	12.5%
[13]	N/A	N/A	2.5%	7.5%	12.5%
This work	4.71%	9.41%	14.12%	18.82%	23.53%

B. Evaluation for the Sampling Problem in a LiDAR System With a Memory-Space Constraint

This subsection evaluates the sampling patterns when the memory-space constraint is considered.

1) Sampling Ratio: The depth images of Middlebury dataset are represented in 8-bit resolution ($n = 8$). The target compression ratios, χ, in (18) are selected as 5%, 10%, 15%, 20%, and 25% of the size of the original image. The sampling ratio, ξ, is derived from (17), (21), and (31), and the results are presented in Table IV. The first column presents the sampling methods. From the second to the sixth columns, the sampling ratios are reported for various target compression ratios (χ). For the uniform grid sampling, ξ and χ are equivalent because no additional data is necessary to store the sampling pattern. On the other hand, the previous sampling methods in [12] and [13] require memory space for their sampling patterns. As discussed in Section II.C, these

[1] http://videoprocessing.ucsd.edu/~leekang/projects.html

[2] This method is named “ADMM+WT+CT (two-stage) in [13].

[3] This method is named “ADMM+WT+CT Grid” in [13].

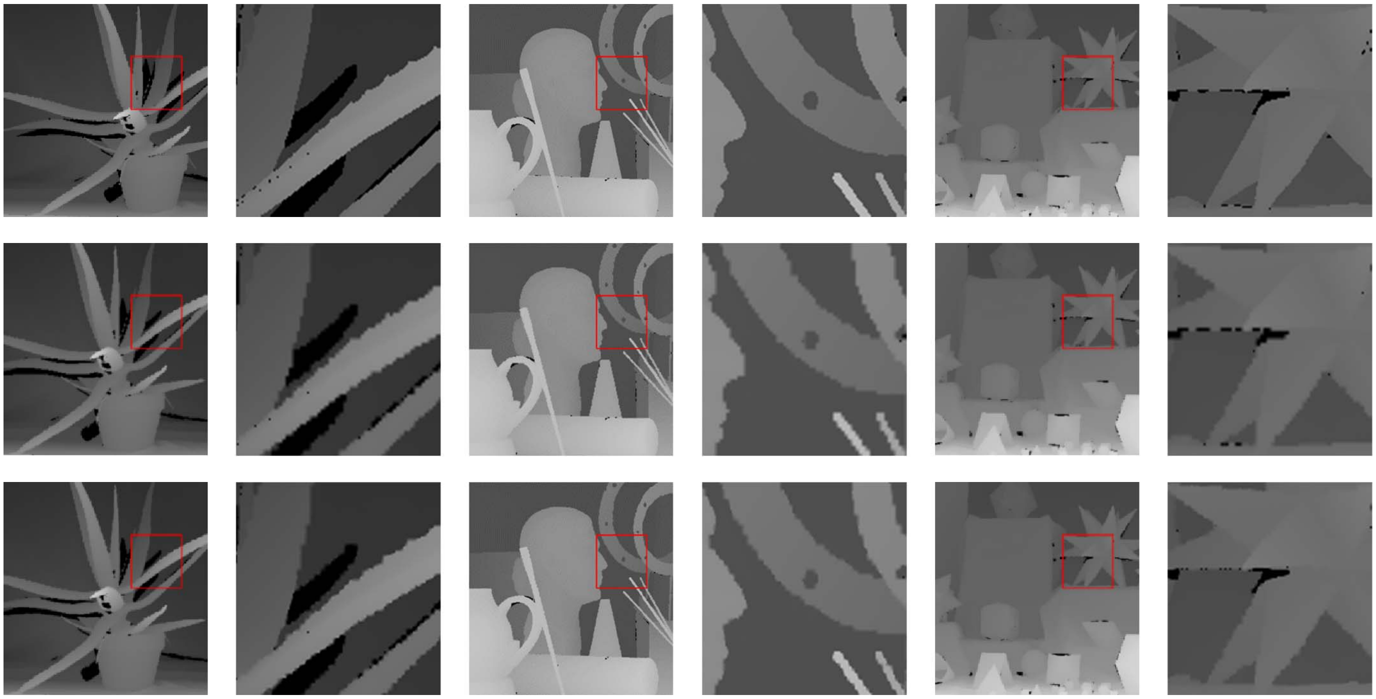


Fig. 10. Subjective comparison of the ground truth (first row), the uniform grid sampling (second row), and the proposed method (third row) on (a) "Aloe", (b) "Art", and (c) "Moebius" test images.

TABLE V

THE MEASURED SAMPLING RATIO (ξ) AVERAGED OVER SIX TEST IMAGES FOR A GIVEN TARGET COMPRESSION RATIO (χ)

Sampling methods	Target Compression Ratio (χ) 5%	10%	15%	20%	25%
Uniform grid	5.0376	10.0319	14.9563	20.0663	25.0566
[12]	N/A	N/A	2.5069	7.5041	12.5043
[13]	N/A	N/A	2.4965	7.4968	12.5102
This work	4.7321	9.5067	14.198	18.8839	23.0648

methods require 12.5% of the space required by the original image when the depth uses 8-bit resolution. Therefore, it is not feasible to use those methods if the target compression ratio is either 5% or 10%. When the target compression ratios are 15%, 20%, or 25%, their sampling ratios are 2.50%, 7.5%, and 12.5%, respectively. The sampling ratios of the proposed sampling method are presented in the last row of Table III, which evidently illustrates that the sampling ratio approaches the available memory space because the amount of data needed to store the sampling pattern is significantly smaller than that for the previous sampling methods.

Because of the randomness in the selection of data samples in (1) and (2), the amount of sampled data may not be equal to the target-sampling ratio ξ. Therefore, the experiment is conducted to demonstrate the extent to which the sampling operation satisfies the target sampling ratio ξ. Table V presents the sampling ratio ξ measured by experiments with Middlebury six test images for a specified target compression ratio. The measurement results are averaged over six images. The numbers in Table V are comparable to those in Table IV. This demonstrates that the sampled data ratios are approximately equal to the target sampling ratios.

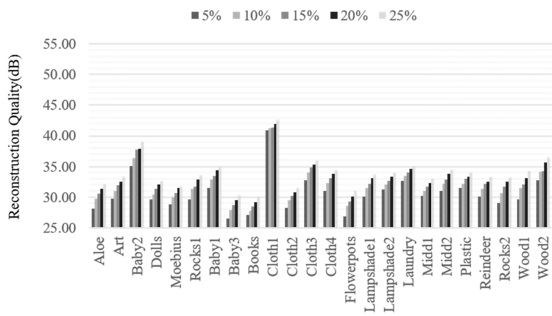
2) Reconstruction Quality: When a memory-space constraint is considered, for a same memory space, a non-uniformly sampling method (i.e., two-stage sampling in [13]) requires an additional storage for storing a sampling map; and therefore a sampling ratio decreased. Eventually, the reconstruction quality of all non-uniformly sampling methods, including the proposed method, decreases. However, thanks to the k-NN expanding sampling operator, the sampling map of the proposed method has a small size (i.e., a down-sampled image's size). Consequently, the proposed method only slight degrades the performance when a memory-space constraint is considered. In addition, the Oracle random sampling in [12] is also included. Table VI demonstrates that the proposed sampling method is superior to grid sampling in terms of the PSNRs at most target compression ratios for all three k-NN expanding patterns. Compared to Table III, the results with the method in [13] are adjusted to solve Problem 2b. Particularly, the results are obtained with the target-sampling ratio ξ instead of target compression ratios (χ), such that they are different from those reported in [13]. For example, given the target memory space of 15%, 20%, and 25% (convert to the sampling ratios of 2.5%, 7.5%, and 12.5% in [13]), 'Rock' image has PSNRs of 27.34, 32.54, and 36.68 dB, respectively. Meanwhile, in [13], 'Rock' image reported PSNRs of 30.7662, 35.3975, and 37.5056 dB for sampling ratios of 5%, 10%, and 15%, respectively.

C. Evaluation for the Sampling Problem in a LiDAR System With a Timing Constraint

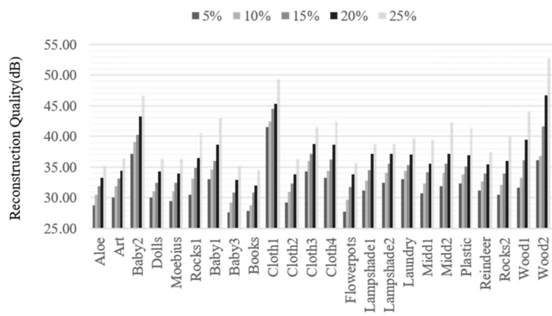
Recall that the two-stage sampling scheme [13] consumes a considerable amount of time so that it does not satisfy

TABLE VI
THE PSNR OF THE RECONSTRUCTED IMAGE FOR SPECIFIED MEMORY SPACE

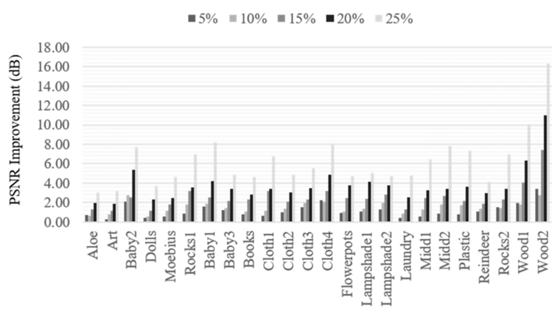
Test Images	Sampling Methods	Target compression ratio (χ) 5%	10%	15%	20%	25%
Aloe	Grid	25.32	28.91	30.09	31.30	32.35
	[13]	N/A	N/A	24.32	30.30	32.51
	[12]	N/A	N/A	25.05	32.72	36.91
	k=4 square	27.71	31.20	33.60	35.23	36.50
	k=4 diamond	27.16	30.67	32.96	34.67	36.16
	k=8	25.33	30.36	33.03	36.00	39.43
Art	Grid	27.52	28.95	30.84	32.52	33.71
	[13]	N/A	N/A	26.27	30.86	34.73
	[12]	N/A	N/A	25.61	32.58	35.64
	k=4 square	28.45	32.64	34.89	36.16	37.85
	k=4 diamond	28.03	32.24	34.53	35.48	36.99
	k=8	27.63	31.88	35.38	38.26	42.93
Baby	Grid	34.44	36.80	37.67	39.05	40.07
	[13]	N/A	N/A	33.02	40.52	44.97
	[12]	N/A	N/A	33.77	38.02	44.22
	k=4 square	35.68	39.94	41.36	43.93	44.31
	k=4 diamond	35.09	39.30	40.99	42.75	44.18
	k=8	35.49	40.32	43.02	47.66	51.80
Dolls	Grid	28.49	29.05	30.09	30.81	31.67
	[13]	N/A	N/A	28.45	30.97	32.77
	[12]	N/A	N/A	29.15	34.57	37.47
	k=4 square	29.19	31.16	32.68	33.78	35.12
	k=4 diamond	29.01	30.69	32.00	33.40	34.31
	k=8	29.15	31.64	33.72	35.02	37.90
Moebius	Grid	27.69	28.72	29.85	31.17	32.24
	[13]	N/A	N/A	28.52	31.47	35.12
	[12]	N/A	N/A	29.15	35.98	41.35
	k=4 square	28.30	30.25	33.52	35.15	36.20
	k=4 diamond	28.06	29.95	32.74	34.09	35.73
	k=8	28.27	30.63	34.34	36.64	40.98
Rocks	Grid	27.69	28.72	29.85	31.17	32.24
	[13]	N/A	N/A	27.34	32.54	36.68
	[12]	N/A	N/A	28.94	38.84	44.57
	k=4 square	28.50	31.34	34.13	35.64	36.99
	k=4 diamond	27.96	30.56	33.11	34.56	36.71
	k=8	28.68	31.93	35.58	38.56	45.92



(a)



(b)



(c)

Fig. 11. PSNR improvements of the proposed method over the uniform grid sampling on Middlebury datasets. (a) Grid sampling; (b) Proposed sampling; and (c) Comparison.

the timing constraint of *Problem 2*. Therefore, the sampling method in [13] is not compared in this subsection, and only the grid sampling and the proposed two-stage sampling are compared. The experiments are conducted with 24 disparity images from the Middlebury datasets [22], [23]. The reconstruction methods[4] in [24] and [25] are selected for the sake of running time and reconstruction quality. The proposed sampling with 4-NN square is used on this experiment.

Fig. 10 demonstrates a comparison of reconstructed images that are obtained from the uniform grid and the proposed sampling schemes with Aloe, Art, and Moebius from the Middlebury sets at a sampling rate of 20%. For each set, the first, second, and third rows present the ground truth images and the reconstructed images obtained from the uniform grid and the proposed samplings, respectively. In the "Art" image, especially in the regions of face and sticks, the proposed sampling pattern produces much better reconstruction quality than in the case of uniform grid sampling. In particular, the region surrounding the face in the reconstructed image of the uniform grid sampling suffers from large artifacts. On the contrary, the proposed sampling efficiently includes more samples in the same area so that its reconstructed image looks better.

Fig. 11 presents the PSNR comparison between the proposed and the uniform grid samplings with the Middlebury datasets. Figs. 11(a)-(b) demonstrate the PSNR results of the uniform grid and the proposed samplings, respectively, while Fig. 9(c) presents the PSNR enhancement by the proposed sampling. Experimental results demonstrate that the proposed sampling consistently outperforms that of the uniform grid for all test images and at all five sampling rates. The best PSNR improvement of about 16 dB is achieved for "Wood2" image at a given sampling rate of 25%. At the sampling rates of 5%, 10%, 15%, 20%, and 25%, the proposed sampling achieves the averaged PSNR improvements of 1.16, 1.50, 2.54, 3.79, and 6.27 dB, respectively.

VI. CONCLUSION

The proposed sampling algorithm includes three main contributions. First, it formulates a new sampling problem for a LiDAR system by considering both timing and memory space

[4]https://github.com/sparse-depth-sensing/sparse-depth-sensing

Alg. 2: Gradient-based sampling

```
% objfunc
function out = objfun(x,b,k)
out = sum(min(1,b*x))-k;
end
% Gradient sampling
function Sout = gradient_sampling(grad, xi)
   [H, W]  = size(grad);
   N       = W*H;
   myfun  = @(v)objfun(v,grad(:),round(N*xi));
   tau     = fzero(myfun,[0,1e16]);
   p       = max(1e-16, min(1,grad
   Sout    = (rand(H,W)<=p);
end
```

Fig. 12. MATLAB code to find a sampling pattern for a given gradient map.

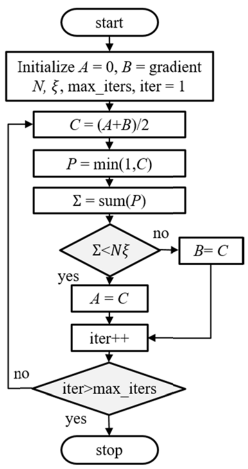


Fig. 13. Flowchart of an efficient solution for the gradient-based sampling.

constraints. Second, it achieves a convenient albeit efficient graph-based representation of a depth image and then proposes the k-NN expanding operator. Third, it proposes an efficient method for selecting samples. The proposed scheme achieves high-quality reconstruction results at a specified sampling budget, and more importantly, it is remarkably rapid in the complexity of linear time. This study also provides the space and time analysis of various sampling methods to be used by the practical LiDAR system. This provides a better understanding of the structures of depth images captured by the LiDAR system in the context of sampling. This study also reveals the underlying relationships between time/memory-space complexity and the reconstruction quality of different sampling techniques. The new sampling scheme proposed in this study is applicable to numerous depth-data-processing tasks for data acquisition, compression, and enhancements applications.

Appendix A

Fig. 12 presents the MATLAB code for finding a sampling pattern for a given gradient map.

Appendix B

A detailed flowchart is illustrated in Fig. 13. In general, the flow chart presents an iterative binary search. Inputs are the number of pixels N, the sampling rate ξ and the number of iterations. To compute an output P, two variables A and B are firstly initialized as a lower bound and an upper bound. On every iteration, a variable C is computed by averaging A and B; and P is computed from C as (35). Note that this flowchart does not include any multiplication so that it can be efficiently implemented in hardware at high clock speed (i.e. 100MHz).

References

[1] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.

[2] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.

[3] S. Foix, G. Alenya, and C. Torras, "Lock-in time-of-flight (ToF) cameras: A survey," *IEEE Sensors J.*, vol. 11, no. 9, pp. 1917–1926, Sep. 2011.

[4] I. R. Nourbakhsh and R. Siegwart, *Introduction to Autonomous Mobile Robots*. Cambridge, MA, USA: MIT Press, 2004, pp. 181–256.

[5] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vis.*, vol. 47, nos. 1–3, pp. 7–42, Apr. 2002.

[6] K. Zhang, J. Lu, and G. Lafruit, "Cross–based local stereo matching using orthogonal integral images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 7, pp. 1073–1079, Jul. 2009.

[7] X. Mei, X. Sun, M. Zhou, S. Jiao, H. Wang, and X. Zhang, "On building an accurate stereo matching system on graphics hardware," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCV Workshops)*, Nov. 2011, pp. 467–474.

[8] B. Schwarz, "LIDAR: Mapping the world in 3D," *Nature Photon.*, vol. 4, no. 7, pp. 429–430, 2010.

[9] C. Niclass, M. Soga, H. Matsubara, S. Kato, and M. Kagami, "A 100-m range 10-frame/s 340×96-pixel time-of-flight depth sensor in 0.18-μm CMOS," *IEEE J. Solid-State Circuits*, vol. 48, no. 2, pp. 559–572, Feb. 2013.

[10] J. Wang, Z. Xiong, Z. Wang, Y. Zhang, and F. Wu, "FPGA design and implementation of Kinect–like depth sensing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 6, pp. 1175–1186, Jun. 2016.

[11] X. T. Nguyen, V. L. Dinh, H.-J. Lee, and H. Kim, "A high–definition LIDAR system based on two–mirror deflection scanners," *IEEE Sensors J.*, vol. 18, no. 2, pp. 559–568, Jan. 2018.

[12] S. Hawe, M. Kleinsteuber, and K. Diepold, "Dense disparity maps from sparse disparity measurements," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2126–2133.

[13] 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.

[14] S. Schwartz, A. Wong, and D. A. Clausi, "Multi–scale saliency–guided compressive sensing approach to efficient robotic laser range measurements," in *Proc. 9th Conf. Comput. Robot Vis.*, May 2012, pp. 1–8.

[15] S. Schwartz, A. Wong, and D. A. Clausi, "Saliency-guided compressive sensing approach to efficient laser range measurement," *J. Vis. Commun. Image Represent.*, vol. 24, no. 2, pp. 160–170, Feb. 2013.

[16] M. Elad and A. Bruckstein, "A generalized uncertainty principle and sparse representation in pairs of bases," *IEEE Trans. Inf. Theory*, vol. 48, no. 9, pp. 2558–2567, Sep. 2002.

[17] E. J. Candes and T. Tao, "Near–optimal signal recovery from random projections: Universal encoding strategies?" *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.

[18] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[19] A. Beck and M. Teboulle, "A fast iterative shrinkage–thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, Jan. 2009.

[20] G. Zhai and X. Yang, "Image reconstruction from random samples with multiscale hybrid parametric and nonparametric modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 11, pp. 1554–1563, Nov. 2012.

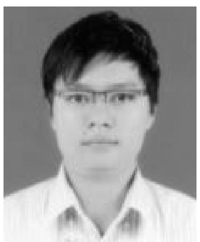
[21] S. H. Chan, T. Zickler, and Y. M. Lu, "Monte Carlo non–local means: Random sampling for large–scale image filtering," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3711–3725, Aug. 2014.

[22] D. Scharstein and C. Pal, "Learning conditional random fields for stereo," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.

[23] H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.

[24] F. Ma, L. Carlone, U. Ayaz, and S. Karaman, "Sparse sensing for resource-constrained depth reconstruction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 96–103.

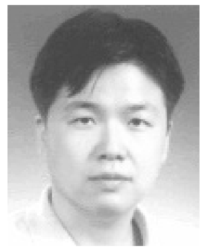
[25] F. Ma, L. Carlone, U. Ayaz, and S. Karaman, "Sparse depth sensing for resource-constrained robots," Mar. 2017, *arXiv:1703.01398*. Accessed: Oct. 31, 2017. [Online]. Available: https://arxiv.org/abs/1703.01398



Xuan Truong Nguyen received the B.S. degree in electrical engineering from the Hanoi University of Science and Technology, Hanoi, Vietnam, in 2011, and the M.S. and Ph.D. degrees in electrical engineering and computer science from Seoul National University, Seoul, South Korea, in 2015 and 2019, respectively. He is currently a Post-Doctoral Fellow from the BK21 PLUS of Electrical and Computer Engineering Department, Seoul National University. His research interests are in the areas of algorithm and SoC design for low-complexity computer vision and multimedia applications.

Hyun Kim (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering and computer science from Seoul National University, Seoul, South Korea, in 2009, 2011, and 2015, respectively.

From 2015 to 2018, he was a B.K. Assistant Professor with the BK21 Creative Research Engineer Development for IT, Seoul National University, Seoul. In 2018, he joined the Department of Electrical and Information Engineering, Seoul National University of Science and Technology, Seoul, where he is currently an Assistant Professor. His research interests are the areas of algorithm, computer architecture, memory, SoC design for low-complexity multimedia applications, and deep neural networks.



Hyuk-Jae Lee (Member, IEEE) received the B.S. and M.S. degrees in electronics engineering from Seoul National University, South Korea, in 1987 and 1989, respectively, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 1996. From 1996 to 1998, he was on the Faculty of the Department of Computer Science, Louisiana Tech University at Ruston, Louisiana. From 1998 to 2001, he was a Senior Component Design Engineer with the Server and Workstation Chipset Division, Intel Corporation, Hillsboro, Oregon. In 2001, he joined the School of Electrical Engineering and Computer Science, Seoul National University, South Korea, where he is currently a Professor. He is a founder of Mamurian Design, Inc., and a fabless SoC design house for multimedia applications. His research interests are in the areas of computer architecture and SoC design for multimedia applications.