Adaptive LiDAR Sampling Based on Generalized Winding Numbers

HANYU CHEN, Carnegie Mellon University, USA

Existing LiDAR systems typically use predetermined sampling schemes that measure depth at pixel locations independent of the observed scene. This limits their ability to accurately represent the geometry of the scene with a limited number of sampels. Recent works have proposed adaptive sampling schemes that focus on the task of depth completion in large-scale environments and rely on depth estimates to adaptively place samples. In this work, we propose an adaptive LiDAR sampling scheme for accurately reconstructing the geometry of a single opaque 3D object. By evaluating the *generalized winding number* along rays sampled in the scene, we determine regions where the *uncertainty* in the object geometry is high and adaptively place more LiDAR samples in those regions. We show that our method achieves a significant improvement in reconstruction quality for both simple and complex 3D objects compared to a baseline sampling scheme.

CCS Concepts: • **Computing methodologies → Computational photography**; **3D imaging**; *Ray tracing*; *Volumetric models.*

ACM Reference Format:
Hanyu Chen. 2023. Adaptive LiDAR Sampling Based on Generalized Winding Numbers. 1, 1 (December 2023), 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Constructing an accurate 3D point cloud representation of a scene is essential to many downstream tasks in computational imaging, graphics, and vision, as they provide a rough estimate of the scene geometry. Light Detection And Ranging (LiDAR) systems have become increasing popular for such tasks due to their ability to accurately measure depth at arbitrary locations of the scene, compared to methods like structure from motion (SfM) [14] and multi-view stereo (MVS) [15] that estimate depth from multiple conventional photos.

Most existing LiDAR systems scan the scene by directing laser pulses in predetermined directions, allowing them to scan the scene along jagged or parallel lines and measure depth at a set of discrete locations along these lines. Such systems do not adapt their sampling scheme to the observed scene, which means that a significant portion of the samples might be wasted on rays that either do not intersect the object of interest or intersect the object in regions of high certainty. More recent LiDAR designs [17] allow them to rapidly scan the scene with programmable scanning patterns and measure depth at arbitrary locations of the scenes. Our work leverages such LiDAR systems to progressively improve the quality of a 3D scan of an object by adaptively adding samples at locations where the uncertainty in the object geometry is high.

Our work is also motivated by recent methods in neural volumetric rendering that model opaque 3D objects as exponential volumes by adopting a probabilistic view of the underlying geometry [13, 18, 19]. Such methods allow volume rendering a probabilistic occupancy field. The generalized winding number[7] allows us to compute an occupancy field from the point cloud representation, and by rendering the occupancy field as a volume, we can determine rays along which the variance in the free-flight distribution is high—providing us with a measure of uncertainty in the scene geometry.

We evaluate our method against a baseline algorithm that uses predetermined scanning patterns by simulating the point cloud acquisition process on several objects with varying geometric complexity—ranging from a simple cube to complex meshes with millions of faces. We show that our adaptive sampling algorithm leads to significantly better reconstruction quality across all objects compared to the baseline algorithm with a limited number of samples.

Author's address: Hanyu Chen, chhy@cmu.edu, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, Pennsylvania, USA, 15213.

ACM XXXX-XXXX/2023/12-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 RELATED WORKS

2.1 Adaptive Depth Sampling

Recent approaches to adaptive LiDAR sampling [2, 5] have focused on the task of depth estimation and depth completion for large-scale scenes. These tasks require estimating a dense depth map from a single RGB image and depth measurements at a sparse set of pixel locations. Many of these methods rely on using a convolutional neural network (CNN) to predict dense depth estimates [3, 4, 11, 16].

Although the particular task of depth completion is less relevant in the context of this work, which estimates the full geometry of a single 3D object, many of the adaptive sampling technique introduced by prior works could still potentially apply. For examples, prior works in depth estimation have proposed a heuristic of allocating more samples to parts of the disparity map where the magnitude of the depth gradient is large [6, 10]. More recent works in depth completion have also proposed methods based on black-box models, such as shifting a predetermined set of samples along a vector field predicted by a neural network [2], or allocating samples based on a variance image computed from the depth maps estimated by an ensemble of depth completion predictors [5].

We note that most of these prior methods redistribute samples on a 2D view of the scene instead of directly sampling rays that shoot into the scene. The former approach is reasonable if we assume the LiDAR system to be fixed in space and thus can only measure the depth of pixels on a fixed 2D plane. However, in the context of producing an accurate 3D reconstruction of a single object, it is desirable to have LiDAR scans of the object from multiple different views, and existing methods in adaptive sampling do not address how to adaptively sample rays that can be arbitrarily placed in 3D space as opposed to on a single 2D image plane. However, heuristics suggested by prior works that also emphasize placing samples in regions of high uncertainty, determined either by the magnitude of the depth gradient [6, 10] or the variance of the depth estimates [5] indeed provide a compelling perspective.

2.2 Volumetric Neural Rendering

The introduction of neural radiance fields (NeRF) [12] led to a large number of works in volumetric neural rendering that model arbitrary 3D scenes as exponential volumes. To account for the fact that these 3D scenes mostly comprise of opaque solid objects, many recent works [13, 18, 19] have proposed methods to model a scene using a probabilistic occupancy field—a function that assigns to each point in space the probability that it lies inside an object of interest. Deriving an expression to convert this occupancy field into attenuation coefficient values provides a way to jointly and differentiably optimize the appearance and the geometry of the scene through volumetric neural rendering.

During volume rendering, we can compute the *free-flight distribution* of any ray originating from the sensor, that is, the probability distribution of the time that it travels before intersecting an object. In the case of modeling opaque solids, we expect the free-flight distribution to approach a Dirac delta centered at the first intersection point. Under a probabilistic view of the object geometry, we can characterize the uncertainty along any given ray by how far its free-flight distribution is from being concentrated at a single point.

2.3 Point Clouds

Point clouds are a common scene representation and the natural output of LiDAR systems. For closed surfaces, the *winding number* is often used to determine whether a given point lies inside or outside the surface. Meanwhile, for point clouds that represent an underlying closed surface, the *generalized winding number* [1, 7] can also be used to robustly answer inside-outside queries.

We can convert this continuous quantity into a measure of occupancy that lies strictly within the $(0, 1)$ range by passing it through a sigmoid and use the computed occupancy for downstream volume rendering tasks. In the context of this work, this method allows us to directly determine from the point cloud regions of the underlying object that have high uncertainty.

3 METHOD

3.1 Volumetric Path Tracing

Given a ray originating from the sensor with origin $\boldsymbol{x}$ and direction $\boldsymbol{\omega}$, we can parameterize the ray as a function of time t as

$$\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv \boldsymbol{x} + t \cdot \boldsymbol{\omega} \tag{1}$$

We can similarly express the attenuation coefficient at any given point along the ray as a function of t. From the attenuation coefficient, we can compute the transmittance from the sensor to any point along the ray and free-flight distribution along the ray as

$$\sigma_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv \sigma_{\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t),\boldsymbol{\omega}} \tag{2}$$

$$T_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv \exp\left(-\int_0^t \sigma_{\boldsymbol{x},\boldsymbol{\omega}}(s)\mathrm{d}s\right) \tag{3}$$

$$p_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv \sigma_{\boldsymbol{x},\boldsymbol{\omega}}(t)T_{\boldsymbol{x},\boldsymbol{\omega}}(t) \tag{4}$$

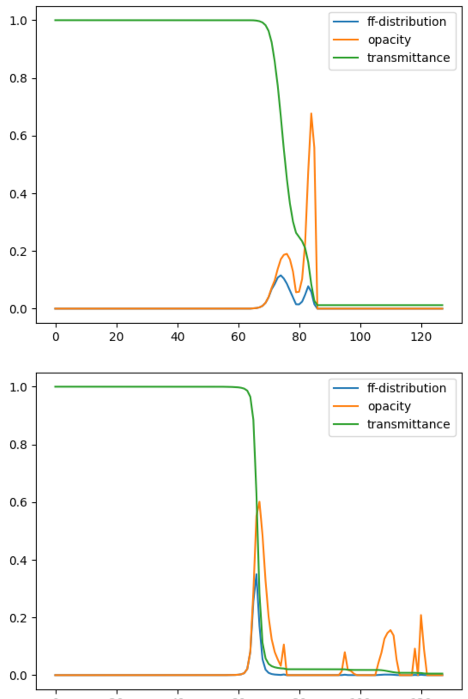

Fig. 1. Visualization of the opacity, transmittance, and free-flight distribution along two rays with high entropy (left) and low entropy (right). We note that less concentrated opacity values does not necessarily translate to higher entropy and high uncertainty.

A visualization of these quantities on two rays with different levels of uncertainty is shown in Figure 1.

We can use the free-flight distribution along the ray to characterize the uncertainty along the ray by considering how far it is from a Dirac delta distribution. A common choice is to directly consider the variance of the distribution, and rays whose free-flight distribution have high variance are identified to be those with high uncertainty. However, this method does not perform well empirically. Therefore, we propose an alternative characterization of the uncertainty of a distribution using its entropy. In the continuous case, one might choose to consider the *differential entropy* of the distribution, which is defined as

$$H(p_{\boldsymbol{x},\boldsymbol{\omega}}) = -\int_0^\infty p_{\boldsymbol{x},\boldsymbol{\omega}}(t)\log p_{\boldsymbol{x},\boldsymbol{\omega}}(t)\mathrm{d}t \tag{5}$$

In other words, rays whose free-flight distribution have high entropy are identified to be those with high uncertainty. We note that the Dirac delta distribution is indeed the continuous distribution with the lowest differential entropy of $-\infty$.

3.2 Discretization

During the volumetric path tracing process, we discretize each ray by uniformly placing a predetermined number of samples along each ray, within some finite time range, where the time steps are denoted with $t^{(1)}, t^{(2)}, \ldots, t^{(M)}$. With this discretization, we denote the samples along a ray as

$$r_{\boldsymbol{x},\boldsymbol{\omega}}^{(i)} \equiv \boldsymbol{x} + t^{(i)} \cdot \boldsymbol{\omega} \tag{6}$$

For each discrete segment along the ray, we adopt the same approximation scheme as used in NeRF [12] and define the discrete *opacity* value on the segment as

$$\alpha_{\boldsymbol{x},\boldsymbol{\omega}}^{(i)} \equiv 1 - \exp\left(-\int_{t^{(i)}}^{t^{(i+1)}} \sigma_{\boldsymbol{x},\boldsymbol{\omega}}(s)\mathrm{d}s\right) \tag{7}$$

Under this approximation scheme, the transmittance and free-flight distribution can be approximated as

$$T_{\boldsymbol{x},\boldsymbol{\omega}}^{(i)} \equiv \prod_{j=0}^{i-1}(1 - \alpha_{\boldsymbol{x},\boldsymbol{\omega}}^{(j)}) \tag{8}$$

$$p_{\boldsymbol{x},\boldsymbol{\omega}}^{(i)} \equiv T_{\boldsymbol{x},\boldsymbol{\omega}}^{(i)}\alpha_{\boldsymbol{x},\boldsymbol{\omega}}^{(i)} \tag{9}$$

In the discrete case, the free-flight distribution becomes a probability mass function defined on the discrete segments. We similarly consider its entropy, defined as

$$H(p_{\boldsymbol{x},\boldsymbol{\omega}}) = \sum_{i=1}^{M} p_{\boldsymbol{x},\boldsymbol{\omega}}^{(i)} \log p_{\boldsymbol{x},\boldsymbol{\omega}}^{(i)} \tag{10}$$

We note that the entropy of a discrete distribution is minimized when the mass is concentrated on a single discrete segment, which is ideally the segment that contains the first intersection point.

In practice, since we only consider finite times, the free-flight distribution is not necessarily a probability distribution when the volumetric representation of the object is not fully opaque. Therefore, we append an additional value to the end of the discretized free-flight distribution so that the sum of the distribution is 1. The additional value essentially represents the amount of light that reaches the background. In the case where the ray misses the underlying object, a significant portion of the free-flight distribution will be assigned to the background. This also results in a low entropy value, which is the expected behavior.

3.3 Point Cloud Rendering

To effectively render a point cloud as a volume, our method involves two separate steps: we compute a measure of occupancy based on the point cloud, and then we convert the occupancy field into attenuation coefficient values for volume rendering.

Computing the generalized winding number for a point cloud can be viewed as solving Laplace's equation with double-sided Dirichlet boundary conditions defined on the point cloud. This corresponds to integrating the Poisson kernel with boundary conditions of 1 and per-point weights set to be the geodesic Voronoi area of each point. Specifically, in an oriented point cloud with points $p_1, p_2, \ldots, p_N$, normals $n_1, n_2, \ldots, n_N$, and area weights $a_1, a_2, \ldots, a_N$, the generalized winding number at a query point q can be computed as

$$w(q) = \sum_{i=1}^{N} a_i \frac{\langle p_i - q, n_i\rangle}{4\pi\|p_i - q\|^3} \tag{11}$$

In practice, instead of explicitly looping over all points in the point cloud, which is extremely inefficient, we use Barnes-Hut approximation to compute the generalized winding number with logarithmic complexity. This involves building a spatial hierarchy such that all points under a node are represented by a multipole—a point computed from a *far-field expansion* of the Poisson kernel.

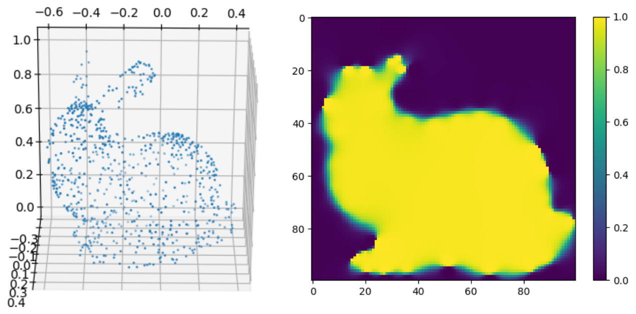

Fig. 2. Visualization of a slice of the occupancy field through the Stanford bunny model and the corresponding point cloud.

Due to the potentially unevenly distributed point cloud, inaccurate normal estimates, and numerical stability issues, it is possible for the computed generalized winding numbers to lie outside of the expected $(0, 1)$ range. Therefore, we enforce them to lie within the $(0, 1)$ and effectively view them as occupancy values by applying a sigmoid function

$$O(q) = \Phi_s(w(q)) \tag{12}$$

where s is a scale parameter and

$$\Phi_s(x) = \frac{1}{1 + \exp(-s \cdot x)} \tag{13}$$

In practice, a value of $s = 10$ enforces the occupancy field to be reasonably binary and the underlying representation to be sufficiently surface-like. A visualization of the occupancy field is shown in Figure 2.

To convert the occupancy field into attenuation coefficient values, we use equations proposed in NeuS [18]. Along a ray $r_{\boldsymbol{x},\boldsymbol{\omega}}$, the attenuation coefficient can be computed as

$$\sigma_{\boldsymbol{x},\boldsymbol{\omega}}(t) = \max\left\{\frac{\langle \nabla O(r_{\boldsymbol{x},\boldsymbol{\omega}}(t)), \boldsymbol{\omega}\rangle}{1 - O(r_{\boldsymbol{x},\boldsymbol{\omega}}(t))}, 0\right\} \tag{14}$$

Moreover, instead of explicitly computing the attenuation coefficients, under the assumption that the attenuation coefficient is monotone along a ray segment, we can directly approximate [18] the opacity as

$$\alpha_{\boldsymbol{x},\boldsymbol{\omega}}^{(i)} = 1 - \min\left\{\frac{1 - O\left(r_{\boldsymbol{x},\boldsymbol{\omega}}^{(t+1)}\right)}{1 - O\left(r_{\boldsymbol{x},\boldsymbol{\omega}}^{(t)}\right)}, 1\right\} \tag{15}$$

In summary, we use Equation 11 to evaluate the generalized winding number at arbitrary positions along a ray, Equation 12 to convert the generalized winding number into a measure of occupancy, Equation 15 to compute the opacity of discrete segments along the ray, Equation 9 to compute the free-flight distribution of a ray, and finally Equation 10 to compute the entropy of the distribution as a measure of uncertainty.

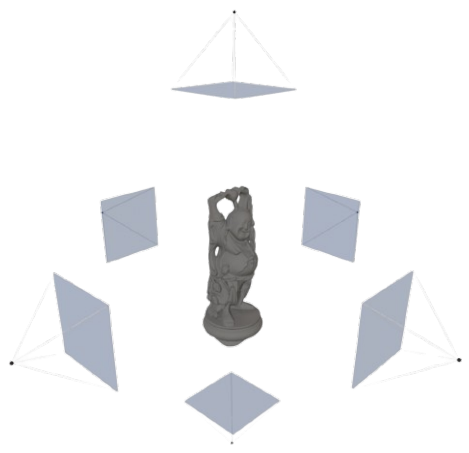

Fig. 3. Visualization of the scene setup with the six viewpoints and the Buddha model. The points indicate sensor locations and the squares correspond to image planes.

4 IMPLEMENTATION

4.1 Scene Setup

For all of the experiments, we consider an object centered at the origin and scaled to lie within an axis-aligned bounding cube of side length 1. The sensor is initially placed at location $(0, 0, 1.5)$ facing the negative z direction. The initial image plane is a square of side length 0.5, centered at $(0, 0, 1.1)$, with its normal direction aligned with the z axis. These values are experimentally determined so that the object fills up most of the camera view, but not all of it. This setup allows us to evaluate the performance of the adaptive sampling scheme without being too unfair to the uniform sampling scheme—if the field of view is too large, then most of the rays from the uniform sampling scheme will miss the object entirely.

We also assume that the LiDAR system is only able to scan the scene from six predetermined viewpoints, corresponding to rotating the initial sensor and image plane locations by multiples of 90 degrees around the three axes. This makes the search space for adaptive sampling significantly smaller, while it also corresponds to a real world scenario where we cannot accurately move the sensor to arbitrary locations for sampling. A visualization of the sensor and image plane locations for the six viewpoints is shown in Figure.

4.2 Baseline Sampler

For the baseline sampler, we assume we have no prior knowledge of the object geometry, and consider uniformly sampling ray directions by sampling pixels on the image plane. The same number of samples are taken for each individual viewpoint. We intersect each of the camera rays with the object, represented as a mesh, and retrieve the location of the intersection and the face normal at the intersection to simulate the LiDAR output. The rays that do not intersect the object—and potentially intersect the background—are simply ignored.

Due to the nature of this sampling method, we typically end up with an oriented point cloud with far fewer points than the number of samples. This is expected behavior given that the geometry of the object is unknown.

4.3 Adaptive Sampler

For the adaptive sampler, we allocate a portion of the available samples to obtain an initial point cloud representation of the scene. Specifically, assuming we have a budget of S rays in total, we uniformly sample $S/4$ initial rays in the same way as in the baseline sampler. This allows us to obtain an initial point cloud that is often extremely sparse.

After obtaining the initial point cloud, in each of the following iterations, we *adaptively* sample $S/8$ rays. This is done based on the heuristic described in the previous section. Specifically, the process can be described as follows:

(1) For each of the six viewpoints, we uniformly sample "virtual" rays with a significantly higher number of samples. These rays are only used for rendering—not as LiDAR samples—so we can sample arbitrarily many of them.
(2) For each ray, we evaluate the generalized winding number at a discrete set of locations along the ray, and compute the free-flight distribution of the ray and its entropy.
(3) We keep rays with entropy values above the 95th percentile as candidate rays. This step is performed for rays across all viewpoints together.
(4) For each viewpoint, we consider the candidate rays associated with the viewpoint, and perform k-means clustering on the ray directions, where k is determined from both the number of available samples $S/8$, as well as the proportion of candidate rays that are associated with the viewpoint.
(5) After obtaining $S/8$ cluster centers, which represent ray directions, we perform the LiDAR scan with the directions represented by the cluster centers and the sensor locations corresponding to the view points. These previous step ensures that these rays are not all clustered together.

We run the adaptive sampling precedure for six iterations so that we have effectively sampled S rays in total. During the experiments, we ensure that both samplers use the same number of rays, although the resulting point cloud may have varying number of points due to rays that do not intersect the object.

4.4 Evaluation

To evaluate the performance of the LiDAR sampling schemes, we run Poisson surface reconstruction [8] on the oriented point clouds generated from the LiDAR scans. Then, to evaluate the quality of each reconstructed mesh, we randomly sample Q points on both the ground truth mesh and the reconstructed mesh, and computed the Chamfer distance between the two set of points. A value of $Q = 2^{16}$ is used for all of our experiments.

5 EXPERIMENTS

5.1 Experiment Setup

The experiments are run on four different meshes of varying geometric complexity: a rotated cube (12 triangles), a cone (64 triangles), the Stanford Bunny model (144,046 triangles), and the Happy Buddha model (1,087,474 triangles) [9]. The meshes are visualized in the

Table 1. Evaluation results on the four meshes. Metric used is Chamfer distance (↓), presented in units of 10^{-5}.

	Cube	Cone	Bunny	Buddha
Uniform	13.24	17.41	5.39	37.17
Adaptive	**5.96**	**7.19**	**0.98**	**7.31**

results section in Figure 4. The cube mesh, in particular, is rotated due to the fact that the uniform sampling scheme draws samples on an axis-aligned grid, which leads to undesirable effects if the edges of the cube are also axis-aligned.

For the first two models, the number of available samples is set to $16 \times 16 \times 6$, corresponding to uniform sampling on a 16×16 grid for each of the six views, while for the last two models, the number of available samples is increased to $32 \times 32 \times 6$ due to their higher geometric complexity.

For the adaptive sampling scheme, this corresponds to taking $8 \times 8 \times 6$ or $16 \times 16 \times 6$ uniform samples in the first iteration and taking 192 or 768 adaptive samples in each following iteration. In each iteration, the number of virtual rays used is set to 256 for each view and each ray is discretized into 256 segments.

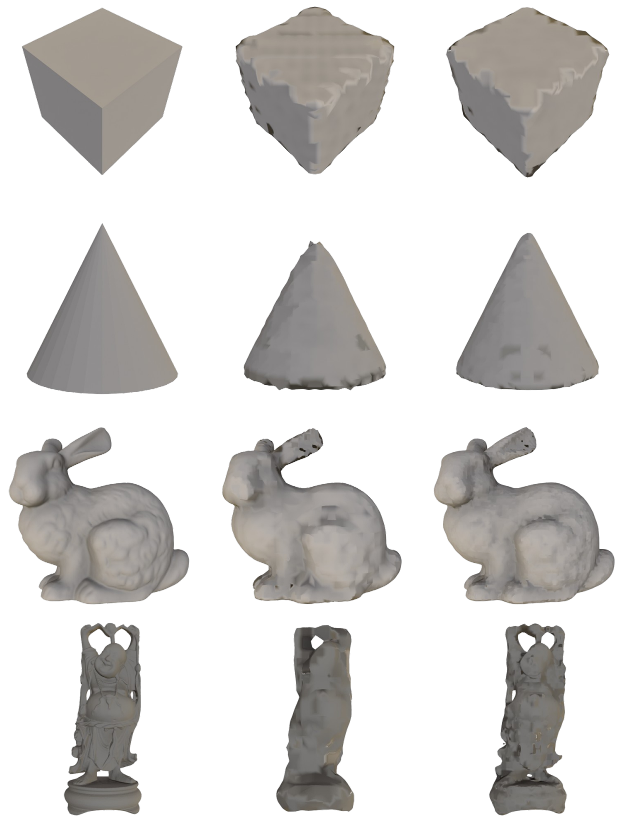
Fig. 4. Front views of the ground truth meshes (left) and reconstructed meshes from point clouds obtained using uniform sampling (middle) and adaptive sampling (right).

Table 2. Ablation results with varying number of samples, computed as $L \times L \times 6$. Metric used is Chamfer distance (↓), presented in units of 10^{-5}.

L	10	16	32	48	64	96
Uniform	286.76	115.78	37.17	30.37	17.90	6.78
Adaptive	**83.84**	**26.05**	**7.19**	**3.04**	**1.68**	**1.34**
Decrease (%)	70.7	77.5	80.6	90.0	90.6	80.2

5.2 Results

The quantitative results are summarized in Table 1. The adaptive sampling scheme outperforms the uniform sampling scheme on all four meshes, while the improvement is more significant on the more complex meshes.

The reconstructed meshes are also visualized in Figure 4. Both sampling schemes do not work particularly well at recovering the sharp edges on the cube model. The adaptive sampling scheme recovers the shape of the cone much better than the uniform sampling scheme, which fails to recover the sharp tip of the cone. For the bunny mesh, since it has roughly the same extent in all three dimension, the uniform sampling scheme performs reasonably well, but the adaptive sampling scheme is still able to recover more details in the shape, such as around the legs of the bunny. For the Buddha mesh, on the other hand, since its height is much larger than its width, most rays from the uniform sampling scheme end up missing the object, and the reconstruction quality is extremely poor. Meanwhile, the adaptive sampling scheme still performs reasonably well since it adaptively places samples in regions where the object roughly lies.

5.3 Ablations

To further investigate the effect of the number of samples on the reconstruction quality, we also run ablations on the Happy Buddha model by varying the total number of samples for both the uniform and adaptive sampling schemes. In particular, the total number of samples is given by $L \times L \times 6$ where $L \in \{10, 16, 32, 48, 64, 96\}$. The quantitative results are summarized in Table 2. Notably, the improvement in reconstruction quality is even more significant when the number of available samples is increased up to $64 \times 64 \times 6$ samples. However, due to the limited number of virtual rays we sample during the adaptive sampling scheme, adding more real samples beyond $64 \times 64 \times 6$ samples do not significantly improve reconstruction quality.

The reconstructed meshes for the ablations are also visualized in Figure 5. In the reconstructions with few samples, the uniform sampling scheme completely fails to recover the geometry of the model. Meanwhile, in the reconstructions with more samples, the adaptive sampling scheme still consistently does better at recovering finer details, such as the facial expression of the Buddha and the patterns on the clothing.

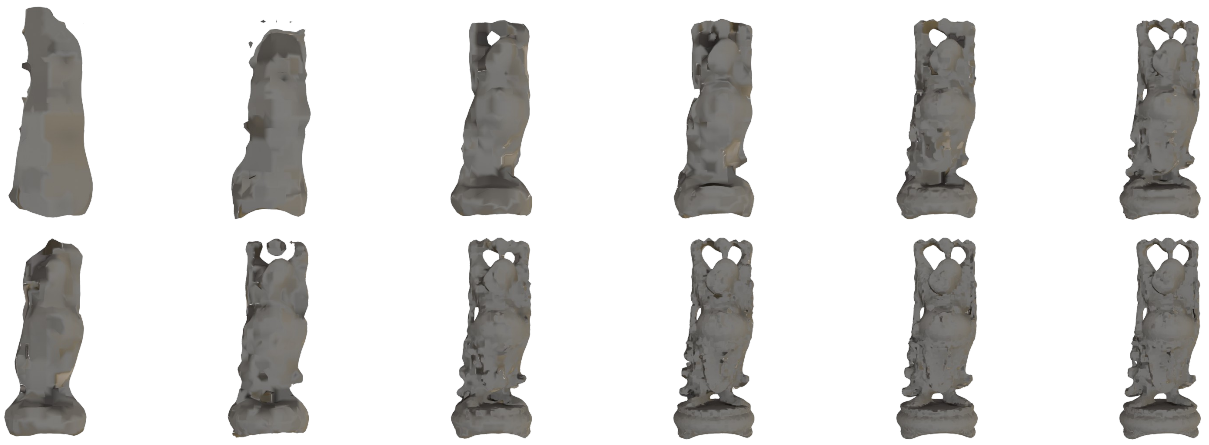

Fig. 5. Front views of the reconstructed meshes. Meshes in the first row use the uniform sampling scheme, and meshes in the second row use the adaptive sampling scheme. Each vertical pair uses the same number of samples, increasing from left to right.

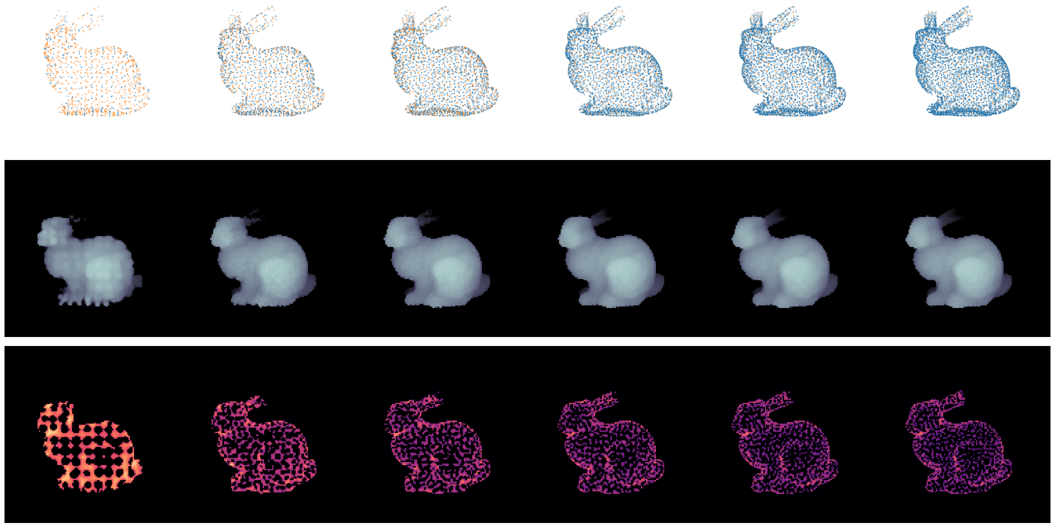

Fig. 6. Visualizations of the ray sampling process for the bunny model. The first row shows the existing points (blue) and added points (orange) in each iteration, the second row shows the average free-flight distance, and the third row shows entropy values of rays with highest entropy (above 90th percentile).

5.4 Ray Selection

To better understand how rays are sampled in the adaptive sampler, we can visualize several intermediate quantities during the ray sampling process for the adaptive sampler. Since the virtual rays are sampled on a regular grid on the image plane, we can directly visualize per-ray quantities, such as the average free-flight distance (or depth) and the entropy as intensity images. We can also visualize how these quantities change across iterations, as shown in Figure 6.

In particular, the rays with highest entropy are always those that are relatively far from existing points. The locations of existing points also coincide with the ball-like structures in the depth plots and the gaps in the entropy plots, especially in the early iterations. This suggests that rays should be sampled to intersect the mesh in regions where there are few existing points, which matches our intuition of adaptive sampling.

6 DISCUSSION

6.1 Limitations and Future Work

We note that the tip of the ear of the bunny is not reconstructed particularly well, while the entropy plots still suggest that the entropy is not very high around that region. This might be due to the lack of initial points sampled in that region, which causes most of the free-flight distribution to be concentrated on the background, leading to low entropy but inaccurate reconstructions in those regions.

One potential way to fix this issue could be that, instead of appending an additional value to the free-flight distribution to represent the background, we directly normalize the free-flight distribution to sum up to 1, and compute the entropy of the normalized distribution. However, one challenge with this method is that, for rays that do not intersect the object, the free-flight distribution computed this way is ill-defined, so we need to filter out those rays using some heuristic.

Another potential limitation of this method is that it relies on an initial set of points to begin the adaptive sampling process. In practice, it might not be feasible to use the LiDAR system to uniformly sample an initial set of points before the adaptive sampling process, as this introduces additional costs.

One alternative way to obtain the initial set of points would be to use pipelines that estimate point clouds based on conventional images, such as SfM and MVS [14, 15]. The initial set of points do not need to be particularly accurate, as we only require it to provide a rough estimate of the object's geometry. The initial set of points can even be discarded further down the adaptive sampling pipeline to ensure the quality of the final point cloud representation.

6.2 Conclusion

In summary, we present a novel method of adaptive LiDAR sampling for the task of scanning and reconstructing a single 3D object, based on the generalized winding numbers and volumetric rendering. Our method achieves significantly better reconstruction quality compared to a baseline uniform sampling scheme on objects with varying geometric complexity. We also present results from an ablation study on the influence of the number of avaible samples on the reconstruction quality and show that it consistently outperforms the baseline algorithm. By visualizing the ray selection process, we also verify that the sampled rays match our intuition of placing samples in regions of high uncertainty in the object geometry.

ACKNOWLEDGMENTS

I would like to thank Prof. Ioannis Gkioulekas for the initial motivation of the project idea and providing helpful suggestions on various design choices.

REFERENCES

[1] Gavin Barill, Nia Dickson, Ryan Schmidt, David I.W. Levin, and Alec Jacobson. 2018. Fast Winding Numbers for Soups and Clouds. *ACM Transactions on Graphics* (2018).

[2] Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. 2020. Deep Adaptive LiDAR: End-to-end Optimization of Sampling and Depth Completion at Low Sampling Rates. In *2020 IEEE International Conference on Computational Photography (ICCP)*. 1–11. https://doi.org/10.1109/ICCP48838.2020.9105252

[3] Xinjing Cheng, Peng Wang, and Ruigang Yang. 2018. Depth estimation via affinity learned with convolutional spatial propagation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 103–119.

[4] Abdelrahman Eldesokey, Michael Felsberg, and Fahad Shahbaz Khan. 2020. Confidence Propagation through CNNs for Guided Sparse Depth Regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 10 (2020), 2423–2436. https://doi.org/10.1109/TPAMI.2019.2929170

[5] Eyal Gofer, Shachar Praisler, and Guy Gilboa. 2021. Adaptive LiDAR Sampling and Depth Completion Using Ensemble Variance. *IEEE Transactions on Image Processing* 30 (2021), 8900–8912. https://doi.org/10.1109/TIP.2021.3120042

[6] Simon Hawe, Martin Kleinsteuber, and Klaus Diepold. 2011. Dense disparity maps from sparse disparity measurements. In *2011 International Conference on Computer Vision*. 2126–2133. https://doi.org/10.1109/ICCV.2011.6126488

[7] Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung. 2013. Robust Inside-Outside Segmentation using Generalized Winding Numbers. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)* 32, 4 (2013), 33:1–33:12.

[8] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. 2006. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, Vol. 7. 0.

[9] Stanford Computer Graphics Laboratory. 2023. The Stanford 3D Scanning Repository. https://graphics.stanford.edu/data/3Dscanrep/ Last accessed 17 December 2023.

[10] Lee-Kang Liu, Stanley H. Chan, and Truong Q. Nguyen. 2015. Depth Reconstruction From Sparse Samples: Representation, Algorithm, and Sampling. *IEEE Transactions on Image Processing* 24, 6 (2015), 1983–1996. https://doi.org/10.1109/TIP.2015.2409551

[11] Fangchang Mal and Sertac Karaman. 2018. Sparse-to-Dense: Depth Prediction from Sparse Depth Samples and a Single Image. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* (Brisbane, Australia). IEEE Press, 1–8. https://doi.org/10.1109/ICRA.2018.8460184

[12] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.

[13] Michael Oechsle, Songyou Peng, and Andreas Geiger. 2021. UNISURF: Unifying Neural Implicit Surfaces and Radiance Fields for Multi-View Reconstruction. In *International Conference on Computer Vision (ICCV)*.

[14] Johannes Lutz Schönberger and Jan-Michael Frahm. 2016. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[15] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. 2016. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*.

[16] Wouter Van Gansbeke, Davy Neven, Bert De Brabandere, and Luc Van Gool. 2019. Sparse and Noisy LiDAR Completion with RGB Guidance and Uncertainty. In *2019 16th International Conference on Machine Vision Applications (MVA)*. IEEE, 1–6.

[17] Dingkang Wang, Conor Watkins, Sanjeev Koppal, Mengyuan Li, Yingtao Ding, and Huikai Xie. 2019. A compact omnidirectional laser scanner based on an electrothermal tripod mems mirror for lidar. In *2019 20th International Conference on Solid-State Sensors, Actuators and Microsystems & Eurosensors XXXIII (TRANSDUCERS & EUROSENSORS XXXIII)*. 1526–1529. https://doi.org/10.1109/TRANSDUCERS.2019.8808659

[18] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. *NeurIPS* (2021).

[19] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. 2021. Volume rendering of neural implicit surfaces. In *Thirty-Fifth Conference on Neural Information Processing Systems*.