VoxNeRF: Bridging Voxel Representation and Neural Radiance Fields for Enhanced Indoor View Synthesis

Sen Wang[1,3] Wei Zhang[2,3] Stefano Gasperini[1,4] Shun-Cheng Wu[1,4] Nassir Navab[1]

***Abstract*— Creating high-quality view synthesis is essential for immersive applications but continues to be problematic, particularly in indoor environments and for real-time deployment. Current techniques frequently require extensive computational time for both training and rendering, and often produce less-than-ideal 3D representations due to inadequate geometric structuring. To overcome this, we introduce VoxNeRF, a novel approach that leverages volumetric representations to enhance the quality and efficiency of indoor view synthesis. Firstly, VoxNeRF constructs a structured scene geometry and converts it into a voxel-based representation. We employ multi-resolution hash grids to adaptively capture spatial features, effectively managing occlusions and the intricate geometry of indoor scenes. Secondly, we propose a unique voxel-guided efficient sampling technique. This innovation selectively focuses computational resources on the most relevant portions of ray segments, substantially reducing optimization time. We validate our approach against three public indoor datasets and demonstrate that VoxNeRF outperforms state-of-the-art methods. Remarkably, it achieves these gains while reducing both training and rendering times, surpassing even Instant-NGP in speed and bringing the technology closer to real-time.**

I. INTRODUCTION

Neural Radiance Fields (NeRFs) have recently gained prominence for their versatility in outdoor settings, such as autonomous driving, visual navigation, and augmented and virtual reality (AR/VR) applications. As a specialized class of implicit scene representations, NeRFs [1] offer the unique capability to recreate 3D environments solely from color images. Their proven efficacy in capturing intricate details, even under challenging conditions such as low lighting [2] or motion blur [3], positions them as an invaluable tool, particularly in sectors that require precise 3D scene understanding.

NeRF has demonstrated remarkable achievements in generating high-fidelity novel view synthesis for outdoor environments[6], [7], object-level scenes [8], and smaller, localized regions. However, its application to large-scale, real-world indoor scenes for robotic navigation and interaction has yielded unsatisfactory outcomes. Several factors contribute to these shortcomings. First, NeRF struggles to generate geometrically consistent models when relying solely on RGB correspondences obtained from a limited set of posed images. This inconsistency is more pronounced in "inside-out" settings than in "outside-in," object-centered ones [9], where the overlap between different perspectives is typically more substantial. Second, the diverse characteristics of indoor scenes, which include varying scales, textures, and lighting conditions, pose a complexity that a single neural network struggles to capture effectively. This variability can result in unreliable data representations, impacting the model's ability to interpret and adapt to its surroundings. Lastly, the heavy architecture of the Multi-Layer Perceptron (MLP) inherent in NeRF results in prolonged training and rendering times, thereby limiting its real-time applicability, which is essential for real-world applications such as robotics.

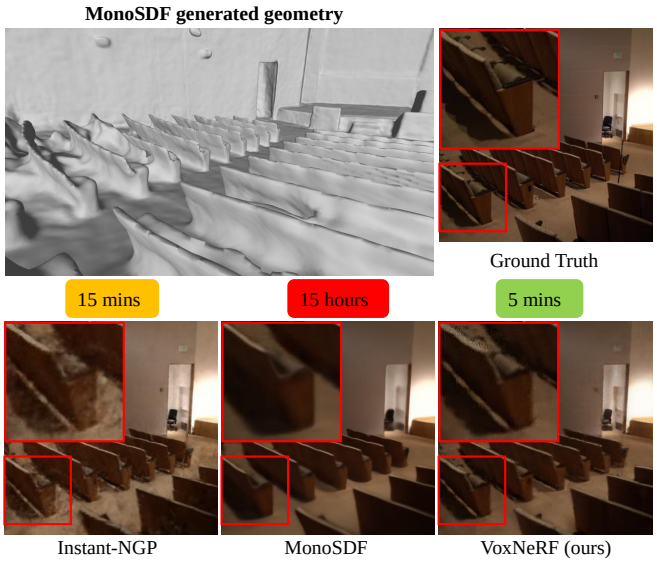

Fig. 1: By exploiting geometric priors [4] and efficient sampling, the proposed VoxNeRF generates better novel views, even faster than Instant-NGP [5].

To address these challenges, several studies have introduced geometric priors derived from COLMAP [10]. These priors can serve either as supervision signals to reduce the number of input views and improve convergence or to improve optimization [9]. In DDPNeRF [9], models learn dense depth priors, which improves NeRF for complex indoor scenes with occlusions and reflections. However, the depth completion network introduces view inconsistency and generalization issues. Similarly, NerfingMVS [11] employs depth prediction as an auxiliary task to aid in optimizing NeRF. Nevertheless, due to the limitation of depth prediction

[1]CAMP, Technical University of Munich, Germany sen.wang, stefano.gasperini, shuncheng.wu, nassir.navab@tum.de
[2]Audiovisual Lab, Huawei Munich Research Center, Germany sen.wang, wei.zhang3@huawei.com
[3]Institute for Photogrammetry, University of Stuttgart, Germany wei.zhang@ifp.uni-stuttgart.de
[4]VisualAIs

network, these methods can only be applied to local regions. Another Avenue of research employs depth and pose as augmentation for sparse view inputs. This addition adds new pseudo-ground truth views as supervisory signals [12], [13], [14], thereby addressing the few-shot input problem. Yet, these methods focus primarily on forward-facing scenery and not focus on large scale indoor scenes. Last but not least, despite the significant progress in overcoming challenges in indoor settings, the limitation on only local area forward-facing scenery, as well as the lengthy computational time required, remain a significant bottleneck for real-world robotic applications.

To address these challenges, we propose a two-stage pipeline that utilizes voxel representations of structured geometry to reconstruct the neural radiance field. In the first stage, we extract implicit geometry to learn the Signed Distance Function (SDF) of the scene surface using only input images. Subsequently, we convert the structured scene geometry into a Sparse Voxel Octree (SVO), and we employ ray casting techniques to approximate the scene's noisy surface [15]. To account for the potential noise in the generated scene geometry, we use a Gaussian distribution to quantify the uncertainty at each hit point, defining an appropriate voxel size to encapsulate this uncertainty. Within these occupied voxels, we introduce a guided sampling strategy specifically designed to expedite the training process. This is achieved by actively forcing the weights along different ray segments to diverge in specific, desired ways. Following guided sampling, the sampled points are incorporated into multi-resolution grids [5]. Here, the features of eight vertices are backpropagated to facilitate the learning of spatial geometry across the entire scene. Consequently, our method not only achieves 66% faster convergence speed but also delivers superior rendering results, improving PSNR by 30% in average. Our method enables the deployment of novel view synthesis in real-time applications, such as robotics, which could not be handled by prior works. By conducting extensive experiments, our method demonstrates improvements not only in qualitative and quantitative metrics but also in training and rendering efficiency.

Our contributions can be summarized as follows:

1) We introduce VoxNeRF, a two-stage, geometry-guided framework for large-scale novel view synthesis in indoor environments
2) We propose an innovative approach that leverages a Sparse Voxel Octree for modeling structured geometry priors to guide the optimization of NeRF.
3) We address the challenges of noisy surfaces with an adaptive Gaussian distribution technique that both accelerates training and reduces rendering time.

II. Related Works

a) Neural Radiance Fields in Robotics: Neural Radiance Fields (NeRF) [1] have gained traction as a versatile tool for addressing a range of challenges in robotics, including Simultaneous Localization and Mapping (SLAM) [16], [17], [18], pose estimation [19], [20], [21], [22], [23], and object or scene reconstruction [24], [25], [4], [26], [27]. For instance, Sucar [16] utilized implicit neural networks to achieve dense mapping and tracking, although their approach struggles in large-scale indoor environments. To circumvent the limitations inherent to a single multi-layer perceptron (MLP), NICE-SLAM [17] employs hierarchical feature grids, thereby allowing the effective modeling of multi-room environments. Capitalizing on traditional SLAM tracking capabilities, Rosinol *et al* [18] integrate poses generated by Droid-SLAM [28] and Instant-NGP [5] to develop a real-time monocular SLAM system. Another emerging application in robotics involves employing NeRF for pose estimation. INeRF [19] inverts an optimized NeRF model to estimate pose, while BARF [20] leverages a bundle adjustment algorithm for the joint optimization of the NeRF model and camera poses. Beyond optimizing NeRF, Lens [22] utilizes an optimized NeRF model to generate synthetic views, thereby training a pose regressor that improves robot localization in challenging environments. Additionally, Adamkiewicz *et al* [21] and Maggio *et al*[23] have employed pre-trained NeRF models as environmental maps, facilitating improved navigation and localization. The utility of NeRF extends to 3D reconstruction, an essential function for robots to understand their surroundings. Approaches like those by Yariv *et al* [25] and Wang *et al* [24] represent surfaces as zero-level sets of a signed distance function (SDF), optimized through volume rendering from multi-view images. MonoSDF [4] further enhances reconstruction quality by incorporating estimated geometric cues as additional supervision signals. On a different note, iSDF [26] proposes a real-time SDF generation system using posed depth images, and SHINE-Mapping [27] leverages point clouds from LiDAR sensors to construct Sparse Voxel Octrees.

b) Faster NeRF: Although NeRF has achieved significant success in various applications, its design suffers from lengthy optimization times for fitting the radiance field encoding. Several works have been proposed to address this issue. NSVF [29] models the object of interest using an octree to avoid free space sampling, which boosts rendering speed by an order of magnitude. Inspired by sparse voxel grids, other works [30], [31], [32], [5] propose storing spatial features in voxel grids to speed up the optimization. Among them, Instant-NGP [5] leverages a spatial hash table and a single-layer MLP, enabling scene fitting in just a few minutes without quality loss. Another branch of works [33], [34], [35], [36] targets the optimization bottleneck attributed to the oversampling of empty spaces, which is a significant concern for robots navigating clutter-free areas. Specifically, DoNeRF [33] and TermiNeRF [34] use depth prediction and spatially conditioned sampling strategies, respectively, to reduce the computational load. AutoInt [37] employs separate networks to predict segment lengths along each ray, further optimizing the computational cost associated with density and color integration. In contrast, our method takes advantage of this sampling strategy while utilizing off-the-shelf geometry modeled into an SVO and storing the features using a spatial hash table, thus achieving rapid rendering

without compromising the quality crucial for robotic perception and interaction.

c) NeRF with Geometric Prior: The utilization of geometric priors has emerged as a significant advancement in NeRF-based scene reconstruction techniques [38], [9], [39], [40], [41], particularly beneficial for robotics where accurate spatial awareness is essential. NerfingMVS [11] trains a monocular depth estimation network to obtain scene-specific depth priors, thereby guiding NeRF sampling. Deng *et al* [38] and Roessle *et al* [9] incorporate depth maps generated by COLMAP to guide the efficient training of NeRF models in few-view settings. While Deng *et al* [38] employ these depth maps as a supervision signal to constrain ray termination, enabling efficient training with few views, Roessle *et al* [9] use an additional network to predict dense depth maps from the previous one, And this dense depths maps are used to inform the placement of sampling points. However, Dense Depth Priors present inherent challenges. The depth completion network, for one, lacks view consistency since each perspective is processed individually. Additionally, it confronts generalization challenges due to its dependency on labeled datasets, like ScanNet.

In contrast to prior approaches [1], [38], [11], [9], the proposed VoxNeRF uses structured geometry to overcome the limitations associated with depth inconsistencies across multiple views. Additionally, as shown across three different datasets, unlike existing approaches using geometric priors [38], [9], our method generalizes robustly.

III. Method

We propose using scene geometry as a prior to accelerate both the training and rendering of NeRF. To this end, we introduce a two-stage framework. Firstly, We take advantage of the recent advanced surface generation approach MonoSDF [4] to generate scene surface (Section III-A). We then represent the scene geometry as a Sparse Voxel Octree (SVO) $\mathcal{S}$, and can trace the path of the ray to account for the noisy surface of the scene geometry. Acknowledging the inherent uncertainty of each intersected point, we employ a Gaussian distribution to quantify this variability. Consequently, we adapt the dimensions of the corresponding voxel to encompass these inaccurate surface points (Section III-B). Building upon this probabilistic framework, we further introduce a voxel integration technique guided by the Gaussian distribution to inform and enhance the sampling strategy (Section III-C).

A. Obtaining Scene Geometry Priors

To obtain the geometry priors for our method, we first perform a 3D reconstruction of the scene using a Signed Distance Field (SDF). To this end, we follow previous works [25], [24], which optimize the surface as the zero-level set of a Signed Distance Function (SDF), represented implicitly with neural networks, from 2D image inputs and monocular predictions of geometry priors (i.e., depth and surface normals) [4].

Given a point in 3D space, a signed distance function f is represented as the point's distance to the closet surface:

$$f : \mathcal{R}^3 \to \mathcal{R} \qquad \mathbf{p} \mapsto s = SDF(\mathbf{p}), \tag{1}$$

Here, $\mathbf{p}$ is the 3D point, and s denotes the corresponding SDF value. For each image pixel corresponding to a ray $\mathbf{r}$, M 3D points $\mathbf{p}_i = \mathbf{o} + t_i\mathbf{d}$ are sampled along each ray, while SDF $\hat{s}_i(\mathbf{r})$ and color value $\hat{\mathbf{c}}_\mathbf{i}(\mathbf{r})$ are also predicted. For volume rendering, the SDF values $\hat{s}_i(\mathbf{r})$ are transformed to density σ_i following [25]:

$$\sigma_\beta(s) = \begin{cases} \frac{1}{2\beta}\exp\left(\frac{s}{\beta}\right) & \text{if } s \le 0 \\ \frac{1}{\beta}\left(1 - \frac{1}{2}\exp\left(-\frac{s}{\beta}\right)\right) & \text{if } s > 0 \end{cases}, \tag{2}$$

where β is a learnable parameter. Following NeRF [1], the *transparency* T_i and the opacity α_i can be acquired via:

$$T_i = \exp\left(-\sum_{j<i}\delta_j\sigma_j\right), \quad \alpha_i = 1 - \exp(-\delta_i\sigma_i), \tag{3}$$

where $\delta_i = t_i - t_{i-1}$ is the distance between two neighboring ray segments. In this way, the color $\mathbf{c}_{out}$ for the current ray $\mathbf{r}$ could be rendered as:

$$\hat{\mathbf{c}}_{\text{out}} = \sum_{i=1}^{N} T_i\alpha_i\mathbf{c}_i. \tag{4}$$

$$\hat{D}_{\text{out}} = \sum_{i=1}^{N} T_i\alpha_i t_i \quad \hat{\mathbf{N}}_{\text{out}} = \sum_{i=1}^{N} T_i\alpha_i\mathbf{n}_i. \tag{5}$$

To optimize the above SDF, the predicted depth and the normals, we use the following losses.

Reconstruction Loss. We utilize the image reconstruction loss from NeRF [1] to compute the predicted RGB observations and apply the posed training images as supervision:

$$\mathcal{L}_{\text{rgb}} = \frac{1}{N}\sum_{\mathbf{r}\in\mathcal{R}}^{N} ||\hat{\mathbf{C}}(\mathbf{r}) - \mathbf{C}(\mathbf{r})||_1. \tag{6}$$

Eikonal Loss. Following the typical method, the Eikonal loss is applied for regularization:

$$\mathcal{L}_{\text{eik}} = \frac{1}{MN}\sum_{\mathbf{x}\in\mathcal{X}}(\|f_\theta(\mathbf{x})\|_2 - 1)^2. \tag{7}$$

Geometry Prior Loss. Following previous works [4], we use the geometry prior loss as these two terms:

$$\mathcal{L}_{\text{depth}} = \frac{1}{N}\sum_{\mathbf{r}\in\mathcal{R}} ||w(\hat{D}(\mathbf{r}) + b) - D(\mathbf{r})||^2, \tag{8}$$

$$\mathcal{L}_{\text{norm}} = \frac{1}{N}\sum_{\mathbf{r}\in\mathcal{R}} ||\hat{\mathbf{N}}(\mathbf{r}) - \mathbf{N}(\mathbf{r})||_1 + ||1 - \hat{\mathbf{N}}(\mathbf{r})^{\mathbf{T}}\mathbf{N}(\mathbf{r})||_1, \tag{9}$$

where the w and q are the scale and shift to match the depth prediction and depth prior, since the depth prior generated by the pretrained model is only up to scale.

The total loss function for the geometry optimization is:

$$\mathcal{L}_{\text{geo}} = \mathcal{L}_{\text{rgb}} + \lambda_1\mathcal{L}_{\text{eik}} + \lambda_2\mathcal{L}_{\text{depth}} + \lambda_3\mathcal{L}_{\text{norm}}. \tag{10}$$

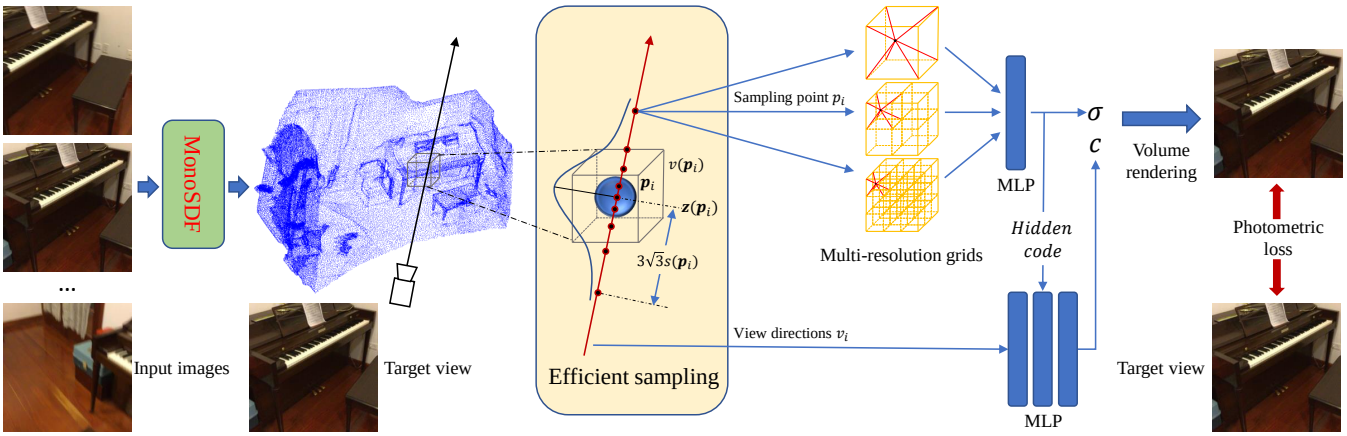

Fig. 2: Pipeline of the proposed VoxNeRF. First, we extract the scene geometry with MonoSDF, which we transform into Octrees. Then, we perform ray casting and sample points efficiently following a Gaussian distribution. Then, we perform ray casting and sample points efficiently following a Gaussian distribution. The models are optimized by minimizing the photometric loss to the ground truth.

B. Representing the Scene with Sparse Volumes

Modeling Geometry Prior. After optimizing the geometry, we can obtain the geometric surface of the scene. Toward this end, we utilize the scene geometry as a prior by encoding it with an SVO [29]. This representation can take various forms, such as a point cloud, a mesh, or a corresponding depth map with known poses. Each voxel in the SVO comprises a center point p_i and an axis-aligned bounding box $\mathcal{V}$ that surrounds this center point. The voxel coordinates are encoded as 8-bit Morton codes, which is known as the Z-order curve. Z-order gives a deterministic ordering of integer coordinates on a 3D grid. And we can convert the 3D coordinates to a non-negative 1D integer coordinate. This significantly reduces memory consumption.

Obtaining accurate scene geometry is an ill-posed problem because the process of acquiring geometric information is inherently noisy. This is due to factors such as inaccuracies in feature matching, camera parameters, or optimization in methods such as COLMAP [42] for visual-only processes. Another source of issues is the presence of small objects in the scene. The geometry priors of MonoSDF are derived from Omnidata [43], where the geometry cues at the edges of the objects could be easily missed, leading to reconstruction errors. To address this issue, we model the scene geometry with Gaussian noise $\mathcal{N}(\mathbf{p}_i, s(\mathbf{p_i})^2)$.

Our 3D geometry representation starts from a 3D point cloud, as shown in Figure 2. Specifically, each point $\mathbf{p}_i$ in the scene geometry is modeled with a sphere $\mathcal{S}$ centered at $\mathbf{p}_i$, where the radius r_i corresponds to the sigma value s of the Gaussian noise. This sphere serves as a guide for setting the leaf voxel size of the SVO. We set the voxel size to the maximum of the sigma values across all scene points:

$$v(\mathbf{p}_i) = \max(r_1, r_2, ..., r_n). \tag{11}$$

Here, n represents the total number of points used for building the SVO.

Scene Representation. To encode the radiance field of the scene, we use the multi-resolution hash grids to encode the spatial features, as suggested in [5]. We first feed the input into the hash tables and pass the concatenated outputs through a single MLP. Specifically, a function $\mathcal{F}_\theta$ is used to predict the volume density σ_i and the radiance $\mathbf{c_i}$ from a position $\mathbf{p}_i$ and a unit-norm viewing direction $\mathbf{v}_i$, as follows:

$$\mathbf{c}_i, \sigma_i = \mathcal{F}_\theta(\mathbf{p}_i, \mathbf{v}_i). \tag{12}$$

The trainable feature vectors are stored in several compact spatial hash tables, each indexed at different resolutions. The interpolated outputs from each hash table are concatenated and passed through a single MLP to generate the density. The one-layer MLP also generates a hidden code, which is then concatenated with view directions encoding and passed through another 3-layer MLP to generate the radiance $\mathbf{c_i}$. The final color of each ray is determined by volume rendering Eq. (4).

It is important to note that the SVO and the hash table are different representations of the scene geometry. The SVO takes in geometry priors and is used to determine the ray-surface intersections, enabling our sampling algorithm (Section III-C). In contrast, the hash grids solely account for encoding features from the input sampling points.

C. Efficient Sampling

In NeRF, the sampling points are distributed throughout the whole space. However, this approach can be time-consuming, as many are at empty locations and do not contribute to the final rendering. Inspired by the observation from Deng *et al* [38] that the ideal weight distribution along the ray is unimodal near the possible surface [38], we propose an efficient sampling strategy. Specifically, we aim to draw more informative sampling near the surface.

Efficient Sampling with Gaussian-Distributed Voxels. We improve efficiency by only sampling a small fraction of points in the empty space. To determine the empty space, we

Method	Geometry	scene 0000		scene 0158		scene 0316		scene 0653	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
NeRF[1]	No	15.76	0.699	29.19	0.928	17.09	0.828	30.89	0.953
Instant-NGP[5]	No	17.78	0.558	12.40	0.575	27.34	0.927	25.08	0.875
NerfingMVS[11]	Depth network	18.07	0.748	30.55	**0.948**	20.88	0.899	30.89	**0.953**
VoxNeRF+SLAM	SLAM [44]	30.45	0.795	33.22	0.905	32.24	0.933	32.60	0.911
VoxNeRF	MonoSDF [4]	**30.13**	**0.791**	**35.78**	0.929	**34.61**	**0.940**	**34.10**	0.930

TABLE I: Comparative experiments on the ScanNet dataset.

Method	Geometry	Courtroom		Auditorium		Museum		Ballroom	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
NeRF[1]	No	19.14	0.625	18.90	0.813	17.45	0.596	23.48	0.709
NerfingMVS[11]	Depth Network	19.77	0.4376	21.95	0.5597	18.62	0.3594	16.88	0.2907
Instant-NGP [5]	No	19.97	**0.753**	19.46	0.744	17.59	0.721	24.39	**0.784**
MonoSDF [4]	Omnidata [43]	20.71	0.671	24.01	0.847	19.16	0.587	22.04	0.650
VoxNeRF	MonoSDF [4]	**24.76**	0.733	**28.62**	**0.850**	**23.91**	**0.770**	**25.40**	0.779

TABLE II: Comparative experiments on the Tanks and Temples dataset.

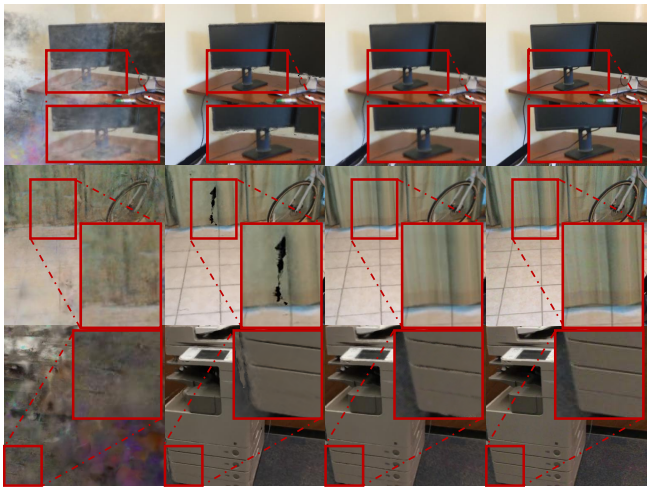

Fig. 3: Renderings of the ScanNet dataset show that our method provides more detailed and clearer outputs, especially at the edge of the objects, compared to the state-of-the-art [11].

consider an octree cube covering the majority of the Gaussian distribution $\mathcal{N}(\mathbf{p}_i, s(\mathbf{p_i})^2)$. Specifically, we set the size of the cube to $\max s(\mathbf{p_i})$. To sample points around the surface, which we call **important sampling points**, we draw points according to the following Gaussian distribution:

$$\mathcal{N}(\mathbf{z}(\mathbf{p}_i), 3s(\mathbf{p_i})^2), \tag{13}$$

where $\mathbf{z}(\mathbf{p}_i)$ is the intersection between the ray and the octree cube, obtained by tracing the ray [15], furthermore, we set $\sqrt{3}s(\mathbf{p_i})$ as the standard deviation to cover the entire voxel octree space. As the probability that a value is beyond 3σ in a Gaussian distribution is only 0.3%, we stop sampling *important points* until we are at $3\sqrt{3}s(\mathbf{p_i})$ from the intersection point. This is shown in Figure 2.

To cover the whole space, we generate samples using the uniform distribution:

$$\begin{aligned} &\mathcal{U}(0, \mathbf{z}(\mathbf{p}_i) - 3\sqrt{3}s(\mathbf{p_i})), \\ &\mathcal{U}(\mathbf{z}(\mathbf{p}_i) + 3\sqrt{3}s(\mathbf{p_i}), \infty). \end{aligned} \tag{14}$$

Method	**Training Time**	**Convergence Iters**	**Rendering Time**
NeRF [1]	24hs	1M	7s
NerfingMVS [11]	3.5hs	300k	7s
Instant-NGP [5]	15mins	300k	0.66s
VoxNeRF	**5mins**	**30k**	**0.5s**

TABLE III: Efficiency comparison on the ScanNet dataset.

IV. Experiments and Results

A. Experimental setup

Dataset We evaluated our VoxNeRF for indoor spaces across three datasets: Replica [45], ScanNet [46], and Tanks and Temples Advanced Indoor Dataset [47]. Replica is a renowned benchmark with an array of scenes characterized by disparate layouts, illumination, and textures. ScanNet encompasses a vast collection of indoor settings captured via RGB-D sensors, providing an extensive set of environments and configurations. Tanks and Temples offer scenes with intricate lighting and large-scale surfaces, which are critical to assess the method's robustness.

Implementation details To generate structured geometric priors, we employ MonoSDF [4]. Images from the three datasets are center cropped and resized to 384×384. For the optimization, we employ ADAM [48] with a learning rate of 0.02 and a cosine annealing schedule to 0.001. We sample 4,096 rays per batch and train for 30,000 iterations for each scenario on a single NVIDIA TITAN 2080 Ti 11GB GPU. For the voxel dimensions, we designate 2cm, 4cm, and 4cm for Replica, ScanNet, and Tanks and Temples advanced datasets, respectively. We evaluate PSNR for the renderings, and SSIM for the high-frequency details.

B. Results

Baselines We compare our methods with various state-of-the-art methods: the standard NeRF [1], NerfingMVS [11] which apply depth priors as guided optimization or regularization for small-scale and local indoor scenes, Instant-NGP [5] for its fast optimization and fair reconstruction result, to validate the benefit of our geometric priors.

Fig. 4: Ablation study of our efficient sampling. As highlighted by the boxes, the surfaces appear less sharp using the uniform sampling method.

ScanNet We adopt the training and testing scheme described in [11] for the ScanNet dataset, using only around 40 images in the whole sequence for training, covering only local areas, while five images of the same area are used as validation set. We evaluate the efficiency of our method on four scenes featured in their paper. We do not compare with DDPNeRF [9] because we could not run it successfully, which could possibly stem from the lack of view consistency inherent in the depth completion network. As shown in Table. I, our method achieves the best rendering quality overall. We also report our method with the geometry from SLAM [44] as VoxNeRF+SLAM. Although SLAM generates less complete geometry compared to MonoSDF, the performance of VoxNeRF+SLAM is on par with VoxNeRF, which employs more compact geometry. This outcome validates the effectiveness of our proposed efficient sampling methods.

Tanks and Temples Advanced Sets For this dataset, we follow the split of MonoSDF [4] and randomly select 30 images for evaluation. The quantitative results are presented in Table II. The scenes are significantly larger than in ScanNet. As a result, methods focusing on local areas, such as NerfingMVS [11], produce less satisfactory results. Instant-NGP [5] demonstrates quick optimization times, it underperforms in sparse-view settings. To ensure a fair comparison, we also include results from MonoSDFF [4]. Our method outperforms others in rendering novel views, as substantiated by the highest PSNR values across the four scenes.

Runtime and Efficiency Thanks to our efficient sampling methods, the training and rendering times are much shorter, and the convergence speed is improved. As shown in Table III, our method significantly enhances performance, achieving an astonishing 30k iterations to convergence, resulting in a substantial reduction training time to only 5 minutes on a relatively old 2080 Ti GPU. This is even 66% faster than Instant-NGP. Furthermore, our method achieves a slightly faster rendering time than Instant-NGP. Such speedup indicates that this is a significant step towards real-time rendering, enabling many real-world robotics operations such as immersive AR or VR experience robotics navigation.

dataset	PSNR	SSIM
w/o mesh	35.677	0.9775
visual mesh	39.362	0.9807
GT mesh	**39.754**	**0.9811**

TABLE IV: Ablation study of the structured geometry on the Replica dataset.

Method	TNT advanced		Replica	
	PSNR	SSIM	PSNR	SSIM
uniform sampling	19.87	0.6506	24.63	**0.8652**
efficient sampling	**25.75**	**0.775**	**28.62**	0.8504

TABLE V: An ablation study of sampling methods on the Tanks and Temples (TNT) and Replica datasets.

C. Ablation Studies

Importance of Structured Geometry One of the key contributions of our work is the utilization of structured geometric priors. We show this with experiments under three conditions on the Replica: a) without any geometric priors, b) with a mesh generated from MonoSDF, which relies solely on visual cues for the mesh generation, and c) with ground truth (GT) meshes. It should be noted that all meshes are first converted to point clouds using their vertices. We conducted these experiments using the same experiment setup as the other experiments setup. The results are presented in Table IV and show that the structured geometry significantly impacts the results. Furthermore, using the visual mesh does not substantially degrade compared to the GT mesh. This is because our Gaussian modeling of the spatial voxels can handle the noise generated during the creation of the visual mesh. Even with very sparse geometry where our sampling degenerates to uniform sampling, the proposed VoxNeRF still robustly achieves results comparable to NeRF, as shown in Tab IV.

Efficient Sampling Another key contribution is the efficient sampling. We demonstrate its importance by comparing it with uniformly sampling the same number of points in the entire space. The results are shown in Table V. Our efficient sampling not only leads to faster convergence speed but also produces better view synthesis. Qualitative results can be seen in Fig. 4. Our efficient sampling helps recover surfaces, which is particularly beneficial for textureless areas.

V. Conclusion

We presented a novel framework that leverages the structured geometry of Sparse Voxel Octree for novel view synthesis in large-scale indoor scenes. By applying a Gaussian distribution to the spatial voxel, we mitigate the minor drawbacks of the generated geometry. We also introduced an efficient sampling method that facilitates training and enhances the quality of novel view synthesis. Through comprehensive experiments on Tanks and Temples advanced, ScanNet, and Replica datasets, we demonstrate that our method attains state-of-the-art performance in novel view synthesis. Our method also exhibits faster training and convergence times and faster rendering times. Future work may further incorporate the geometry alongside the rendering optimization.

REFERENCES

[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *The European Conference on Computer Vision (ECCV)*, 2020.

[2] B. Mildenhall, P. Hedman, R. Martin-Brualla, P. P. Srinivasan, and J. T. Barron, "Nerf in the dark: High dynamic range view synthesis from noisy raw images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 190–16 199.

[3] L. Ma, X. Li, J. Liao, Q. Zhang, X. Wang, J. Wang, and P. V. Sander, "Deblur-nerf: Neural radiance fields from blurry images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 861–12 870.

[4] Z. Yu, S. Peng, M. Niemeyer, T. Sattler, and A. Geiger, "Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction," *arXiv preprint arXiv:2206.00665*, 2022.

[5] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *arXiv preprint arXiv:2201.05989*, 2022.

[6] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar, "Block-nerf: Scalable large scene neural view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8248–8258.

[7] H. Turki, D. Ramanan, and M. Satyanarayanan, "Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 922–12 931.

[8] W.-C. Tseng, H.-J. Liao, L. Yen-Chen, and M. Sun, "Cla-nerf: Category-level articulated neural radiance field," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 8454–8460.

[9] B. Roessle, J. T. Barron, B. Mildenhall, P. P. Srinivasan, and M. Nießner, "Dense depth priors for neural radiance fields from sparse input views," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 892–12 901.

[10] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *European Conference on Computer Vision (ECCV)*, 2016.

[11] Y. Wei, S. Liu, Y. Rao, W. Zhao, J. Lu, and J. Zhou, "Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5610–5619.

[12] M. Niemeyer, J. T. Barron, B. Mildenhall, M. S. Sajjadi, A. Geiger, and N. Radwan, "RegNeRF: Regularizing neural radiance fields for view synthesis from sparse inputs," *arXiv preprint arXiv:2112.00724*, 2021.

[13] D. Chen, Y. Liu, L. Huang, B. Wang, and P. Pan, "Geoaug: Data augmentation for few-shot nerf with geometry constraints," in *European Conference on Computer Vision*. Springer, 2022, pp. 322–337.

[14] M. Kwak, J. Song, and S. Kim, "Geconerf: Few-shot neural radiance fields via geometric consistency," *arXiv preprint arXiv:2301.10941*, 2023.

[15] T. Takikawa, J. Litalien, K. Yin, K. Kreis, C. Loop, D. Nowrouzezahrai, A. Jacobson, M. McGuire, and S. Fidler, "Neural geometric level of detail: Real-time rendering with implicit 3D shapes," 2021.

[16] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "imap: Implicit mapping and positioning in real-time," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6229–6238.

[17] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "Nice-slam: Neural implicit scalable encoding for slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 786–12 796.

[18] A. Rosinol, J. J. Leonard, and L. Carlone, "Nerf-slam: Real-time dense monocular slam with neural radiance fields," *arXiv preprint arXiv:2210.13641*, 2022.

[19] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin, "inerf: Inverting neural radiance fields for pose estimation," pp. 1323–1330, 2021.

[20] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, "Barf: Bundle-adjusting neural radiance fields," in *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[21] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager, "Vision-only robot navigation in a neural radiance world," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4606–4613, 2022.

[22] A. Moreau, N. Piasco, D. Tsishkou, B. Stanciulescu, and A. de La Fortelle, "Lens: Localization enhanced by nerf synthesis," in *Conference on Robot Learning*. PMLR, 2022, pp. 1347–1356.

[23] D. Maggio, M. Abate, J. Shi, C. Mario, and L. Carlone, "Loc-nerf: Monte carlo localization using neural radiance fields," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 4018–4025.

[24] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," *arXiv preprint arXiv:2106.10689*, 2021.

[25] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman, "Volume rendering of neural implicit surfaces," *Advances in Neural Information Processing Systems*, vol. 34, pp. 4805–4815, 2021.

[26] J. Ortiz, A. Clegg, J. Dong, E. Sucar, D. Novotny, M. Zollhoefer, and M. Mukadam, "isdf: Real-time neural signed distance fields for robot perception," *arXiv preprint arXiv:2204.02296*, 2022.

[27] X. Zhong, Y. Pan, J. Behley, and C. Stachniss, "Shine-mapping: Large-scale 3d mapping using sparse hierarchical implicit neural representations," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 8371–8377.

[28] Z. Teed and J. Deng, "DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras," *Advances in neural information processing systems*, 2021.

[29] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt, "Neural sparse voxel fields," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020.

[30] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, "Plenoctrees for real-time rendering of neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5752–5761.

[31] A. Yu, S. Fridovich-Keil, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," *arXiv preprint arXiv:2112.05131*, 2021.

[32] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," *arXiv preprint arXiv:2111.11215*, 2021.

[33] T. Neff, P. Stadlbauer, M. Parger, A. Kurz, J. H. Mueller, C. R. A. Chaitanya, A. Kaplanyan, and M. Steinberger, "Donerf: Towards real-time rendering of compact neural radiance fields using depth oracle networks," in *Computer Graphics Forum*, vol. 40, no. 4. Wiley Online Library, 2021, pp. 45–59.

[34] M. Piala and R. Clark, "Terminerf: Ray termination prediction for efficient neural rendering," in *2021 International Conference on 3D Vision (3DV)*. IEEE, 2021, pp. 1106–1114.

[35] A. Kurz, T. Neff, Z. Lv, M. Zollhöfer, and M. Steinberger, "Adanerf: Adaptive sampling for real-time rendering of neural radiance fields," in *European Conference on Computer Vision*. Springer, 2022, pp. 254–270.

[36] W. Zhang, R. Xing, Y. Zeng, Y.-S. Liu, K. Shi, and Z. Han, "Fast learning radiance fields by shooting much fewer rays," *arXiv preprint arXiv:2208.06821*, 2022.

[37] D. B. Lindell, J. N. Martel, and G. Wetzstein, "Autoint: Automatic integration for fast neural volume rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 556–14 565.

[38] K. Deng, A. Liu, J.-Y. Zhu, and D. Ramanan, "Depth-supervised nerf: Fewer views and faster training for free," *arXiv preprint arXiv:2107.02791*, 2021.

[39] Q. Xu, Z. Xu, J. Philip, S. Bi, Z. Shu, K. Sunkavalli, and U. Neumann, "Point-nerf: Point-based neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5438–5448.

[40] K. Rematas, A. Liu, P. P. Srinivasan, J. T. Barron, A. Tagliasacchi, T. Funkhouser, and V. Ferrari, "Urban radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 932–12 942.

[41] J. Ost, I. Laradji, A. Newell, Y. Bahat, and F. Heide, "Neural point light fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 419–18 429.

[42] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited,"

in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113.

[43] A. Eftekhar, A. Sax, J. Malik, and A. Zamir, "Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 786–10 796.

[44] W. Zhang, S. Wang, X. Dong, R. Guo, and N. Haala, "Bamf-slam: Bundle adjusted multi-fisheye visual-inertial slam using recurrent field transforms," *arXiv preprint arXiv:2306.01173*, 2023.

[45] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou, K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. D. Nardi, M. Goesele, S. Lovegrove, and R. Newcombe, "The Replica dataset: A digital replica of indoor spaces," *arXiv preprint arXiv:1906.05797*, 2019.

[46] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.

[47] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, pp. 1–13, 2017.

[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.