LidaRF: Delving into Lidar for Neural Radiance Field on Street Scenes

Shanlin Sun[1] Bingbing Zhuang[3] Ziyu Jiang[3] Buyu Liu[3]
Xiaohui Xie[1] Manmohan Chandraker[2,3]
[1]University of California, Irvine [2]University of California, San Diego [3]NEC Labs America

Abstract

Photorealistic simulation plays a crucial role in applications such as autonomous driving, where advances in neural radiance fields (NeRFs) may allow better scalability through the automatic creation of digital 3D assets. However, reconstruction quality suffers on street scenes due to largely collinear camera motions and sparser samplings at higher speeds. On the other hand, the application often demands rendering from camera views that deviate from the inputs to accurately simulate behaviors like lane changes. In this paper, we propose several insights that allow a better utilization of Lidar data to improve NeRF quality on street scenes. First, our framework learns a geometric scene representation from Lidar, which are fused with the implicit grid-based representation for radiance decoding, thereby supplying stronger geometric information offered by explicit point cloud. Second, we put forth a robust occlusion-aware depth supervision scheme, which allows utilizing densified Lidar points by accumulation. Third, we generate augmented training views from Lidar points for further improvement. Our insights translate to largely improved novel view synthesis under real driving scenes.

1. Introduction

Photorealistic simulation is needed in many applications like autonomous driving, where it is hard to ensure the diversity and coverage of real data. More critically, extensive verification has to be carried out in simulated environment before road testing, in order to ensure safety. Traditional simulation pipeline typically requires graphic artists to manually create 3D assets and compose into virtual environment of interest. However, the demand on human efforts and expertise has prevented it from being scalable in practice.

Neural Radiance Field (NeRF [22]) has recently emerged as a promising way to approach simulation. NeRF has proven an effective implicit representation of scene radiance, with remarkable abilities to capture and interpolate appearances. However, such good performance often requires dense view coverage in training data, in order for

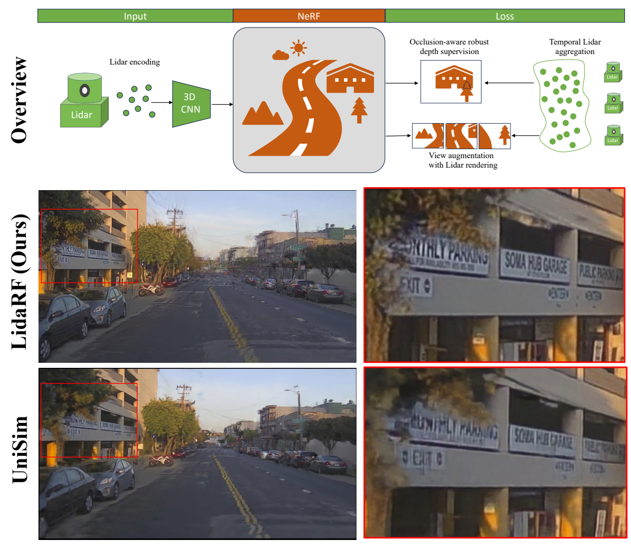

Figure 1. Our framework leverages Lidar to a deep extent to unlock its potential for neural rendering on street scenes, leading to state-of-the-art performance in comparison to UniSim [43].

sufficient constraints to learn accurately the underlying geometry, material properties and illumination. While dense view coverage is not much of a problem in controlled environments, it poses challenges on street scenes – the data collection vehicle typically drives forward along lanes at a potentially high speed, leading to sparse and nearly collinear camera paths. Besides sparsity, forward camera trajectories are well-known [31] to be challenging for 3D reconstruction as it provides much weaker multi-view geometric constraints. Furthermore, road surface is typically low-texture, further introducing ambiguity in scene reconstruction.

To address these challenges, and in view of the implicit representation of NeRF lacking explicit geometric constraint, our key idea lies in leveraging Lidar as an explicit complementary to NeRF. Despite Lidar being used in existing works [27, 40, 43] for street scenes, high-quality rendering remains challenging. Our work delves into Lidar (dubbed LidaRF) and reaps its benefits to a greater extent, yielding largely improved view synthesis quality as shown in Fig. 1.

Our contributions are three-fold. (i) *Fusing Lidar en-*

coding and grid feature for enhanced scene representation. While Lidar has been applied as a natural depth supervision, involving Lidar in the NeRF input offers great potential for geometric inductive bias but remains less straightforward. To this end, we adopt the grid-based representation [24] but fuse features learned from point clouds into the grid, to inherit benefits from the explicit point cloud representation. Inspired by the success of 3D perception frameworks [21, 44], we leverage 3D sparse convolution network as an effective and efficient architecture to extract geometric features from the local and global context of Lidar point clouds. (ii) *Robust occlusion-aware depth supervision.* Similar to existing works [40, 43], we also apply Lidar as a source of depth supervision, but do so to a greater degree. Due to the sparsity of Lidar points limiting its utility, especially in low-texture regions, we densify Lidar points across nearby frames to generate denser depth maps. However, the depth map so obtained does not account for occlusion, yielding ghost depth supervision. Hence, we put forth a robust depth supervision scheme in a curriculum learning fashion – supervising depth from near to far field while gradually filtering out bogus depth as the NeRF trains, leading to more effective learning of depth from Lidar. (iii) *Lidar-based view augmentation.* Furthermore, in view of the view sparsity and limited coverage in the driving scene, we leverage Lidar to denstify training views. That is, we project accumulated Lidar points to novel training views; note they could be views deviating from the driving trajectories to some extent. These views projected from Lidar are added into the pool of training data, but recall that they do not account for occlusion. However, we apply the fore-mentioned supervision scheme to address the occlusion issue, yielding improved performance.

While our insights are also applicable for general scenes, we focus the evaluation on street scenes in this work, which leads to significant improvement compared to prior art, both quantitatively and qualitatively. Our LidaRF also shows advantage in interesting applications such as lane changes that require greater deviation from input views.

In summary, our proposed insights on better incorporation of Lidar lead to significantly improved quality of NeRF in challenging street scene applications.

2. Related Work

NeRF fundamentals. Neural radiance field (NeRF) [22] has become a widely applied scene representation thanks to its remarkable capability for photorealistic novel view synthesis. Since its advent, rapid progress has been made to increase its range of applicability. Notably, given the positional encoding in NeRF does not account for scene scale and causes aliasing when training and testing images are of different resolutions, Mip-NeRF [4] proposes to anti-alias it by casting conical frustums instead of rays, with integrated positional encoding with scale awareness. Mip-NeRF 360 [5] further extends it to handle unbounded scenes by a nonlinear mapping of space for scene contraction. In view of querying large MLPs being the bottleneck for efficiency, Instant-NPG [24] proposes the grid-based scene representation stored using a hash map, with the learned features decoded into radiance by a tiny and fast MLP [23]. We build our framework on top of the successful recipes of Mip-NeRF 360 and Instant-NPG, but makes important advancement to seamlessly integrate the valuable information offered by additional Lidar sensors.

Point-based NeRFs. In contrast to the implicit scene representation, point could is an explicit representation holding the advantage of capturing accurate scene geometry. Point cloud is also widely available, either from structure-from-motion and multi-view stereo, or directly from time-of-flight depth sensors like Kinect or Lidar. This leads to the line of research [3, 26, 42, 47] in rendering images from point clouds or surfaces. Point-NeRF [41] represents one of the pioneering works that assign image features to the point clouds, from which the radiance filed along the rendering ray is decoded by querying features from nearby points. Point2Pix [16] utilizes point encoding to render point clouds in indoor scenes to images. TriVol [15] adopts triple slim volume to encode point cloud efficiently. Pointersect [8] proposes to render point clouds by directly inferring the intersection of the ray with the underlying surface. However, these works only demonstrate results on objects or small-scale indoor scenes. Chang et al. [9] has recently extended Point-NeRF to street scenes, but the rendering remains low-resolution and incomplete due to the sparse nature of Lidar point clouds. In contrast, our LidaRF fuses Lidar encoding with high-resolution grid-based representation for feature learning, yielding results far superior to [9].

Street scene NeRFs. The need for photorealistic simulation in autonomous driving inspires researches [12, 19, 20, 25, 28, 35, 38, 40, 43, 46] to explore NeRFs on unconstrained street scenes. Notably, UniSim [43] demonstrates the promising applicability of NeRF for closed-loop simulation of the autonomy, taking only a real driving log as input. Some works focus on handling sparse view observations [7, 49] or improving geometry of NeRF [12, 33]. Despite these efforts, high-quality rendering of street scenes remains challenging for NeRF. Our work improves NeRF by leveraging Lidar data to a greater extent, with insights on hybrid feature encoding and robust depth supervision with densified Lidar. It is worth noting that while S-NeRF [40] also densifies Lidar with a depth completion network, our strategy distinguishes itself by relying only on actual Lidar frames without additional training data, and is not affected by potential errors in the depth prediction.

Depth-supervised NeRFs. The vanilla NeRF does not enforce any explicit constraint on geometry, often leading to inaccurate depth or surface recovery. This inspires many works to impose depth supervision in various forms [11, 32, 34, 36, 42, 45]. Notably, NeRFs on street scenes [27, 40, 43] typically involve derived depth supervision from Lidar. Our work distinguishes itself by reap benefits from Lidar to a deeper extent, by Lidar aggregation while accounting for occlusions.

3. Preliminaries

Neural Radiance Field (NeRF) [22] represents a radiance field with a continuous neural network $f : (\mathbf{x}, \mathbf{d}) \rightarrow (c, \sigma)$, mapping spatial location $\mathbf{x} = (x, y, z)$ and viewing direction $\mathbf{d} = (\theta, \phi)$ to the RGB color c and volumetric density σ at that point. The network is queried at each point along the rendering ray to estimate color and density, which are then composed into the final pixel color using the volume rendering equation [17]. NeRF is optimized through the loss function $\mathcal{L}_{\text{rgb}}$ defined as the mean squared error between the predicted and true colors of the training RGB images.

Nerfacto is the recommended approach within the open-source project Nerfstudio [29], integrating a variety of recipes that have proven effective for a wide range of real-world data. First, Nerfactor is capable of handling unbound scenes as required by street scenes, by applying the scene contraction strategy as in MipNeRF-360 [5]. For efficiency, it follows [5] to use proposal networks with small MLPs to consolidate the sampled locations along each ray to the regions near the first surface intersection. In terms of NeRF network architecture, it follows Instant-NGP [24] to leverage the grid-based feature representation parameterized by a hash map, which allows to decode color and density with the so-called fused MLPs, *i.e.* small MLPs that admits fast implementation [23].

4. Method

4.1. Overview

We build LidaRF on top of Nerfacto, but develop several insights to integrate the use of Lidar for high-quality view synthesis on street scenes, as illustrated in Fig. 2. Our pipeline takes Lidar point clouds as input (Sec. 4.2), extracts geometric features from the point clouds, and fuses it with the hash-based feature grid to combine their complementary benefits. The hybrid features are fed to MLPs for decoding color and density, followed by standard volume rendering. In the output, we leverage denser Lidar points accumulated across frames as depth supervision while accounting for occlusion. This leads to two extra losses $\mathcal{L}_{\text{ds}}$ (Sec. 4.3) and $\mathcal{L}_{\text{aug}}$ (Sec. 4.4), besides the original losses from Nerfactor $\mathcal{L}_{\text{nerfacto}}$. All together, our loss $\mathcal{L}$ is written as

$$\mathcal{L} = \mathcal{L}_{\text{nerfacto}} + \lambda_1 \cdot \underbrace{\mathcal{L}_{\text{ds}}}_{\text{Sec. 4.3}} + \lambda_2 \cdot \underbrace{\mathcal{L}_{\text{aug}}}_{\text{Sec. 4.4}}, \quad (1)$$

$$\text{where } \mathcal{L}_{\text{nerfacto}} = \mathcal{L}_{\text{rgb}} + \lambda_3 \cdot \mathcal{L}_{\text{dist}} + \lambda_4 \cdot \mathcal{L}_{\text{interval}}. \quad (2)$$

4.2. Hybrid Representation with Lidar Encoding

Motivation. Lidar point clouds holds strong potential for geometric guidance that is highly valuable for NeRF. However, relying on Lidar features alone for scene representation, as done in [9], results in low-resolution rendering, due to the sparse nature of Lidar points despite temporal accumulation. In addition, Lidar does not cover the entire scene due to its limited field of view, e.g. it does not capture building surface above a certain height, yielding blank rendering in those regions as in [9]. Our framework, in contrast, fuses the Lidar features with the high-resolution spatial grid of features to leverage the strength of both, which are jointly learned for high-quality and complete scene rendering.

Lidar feature extraction. We detail here the extraction of geometric features for each Lidar point. Referring to Fig. 2, we first aggregate Lidar point clouds from all frames of the entire sequence to construct a denser set of point clouds. We then voxelize the point clouds into a voxel grid, where the spatial position of points falling into each voxel cell are averaged, yielding a 3-dim feature for each voxel cell. Inspired by its wide success on 3D perception frameworks [21, 44], we encode the scene geometry feature with a 3D sparse UNet [10, 30] on the voxel grid, which permits learning from a more global context of the scene geometry. The 3D sparse UNet takes the voxel grid along with its 3-dim features as input and outputs neural volumetric features, consisting of n-dim feature for each occupied voxel. This yields our Lidar embeddings $P = \{(\mathbf{p}_i, f_i) | i = 1, ..., N\}$, where each point i is located at $\mathbf{p}_i$ and associated with a vector f_i, which is the neural feature of the voxel cell it resides in, encoding the local and global geometry around $\mathbf{p}_i$.

Query of Lidar features. For each sample point $\mathbf{x}$ along the ray to be rendered, we query its Lidar feature if there are at least K nearby Lidar points within a search radius R; otherwise, its Lidar feature is set as empty (*i.e.* all-zero). Specifically, we employ a Fixed Radius Nearest Neighbors (FRNN) approach [14] to search a K-nearest Lidar point index set with respect to $\mathbf{x}$, denoted as $\mathcal{S}_{\mathbf{x}}^K$. Different from [9] where ray sampling is predetermined prior to initiating the training process, our method conducts FRNN searching online as the distribution of the sampled points from our proposal networks shift dynamically towards concentrating on the surface as the NeRF training converges.

Following Point-NeRF [41], our method harnesses an MLP, $\mathcal{F}$, to map Lidar feature from each point to a neural scene description. For the i-th point neighbor of $\mathbf{x}$, $\mathcal{F}$ takes as input the Lidar feature f_i and relative position $\mathbf{x} - \mathbf{p}_i$, and outputs the neural scene description as

$$f_{i,\mathbf{x}} = \mathcal{F}\left([f_i, \mathbf{x} - \mathbf{p}_i]\right), \quad (3)$$

where $[,]$ indicates concatenation. To obtain the final Lidar encoding $\phi_L(\mathbf{x})$ at the sampled location $\mathbf{x}$, we use standard

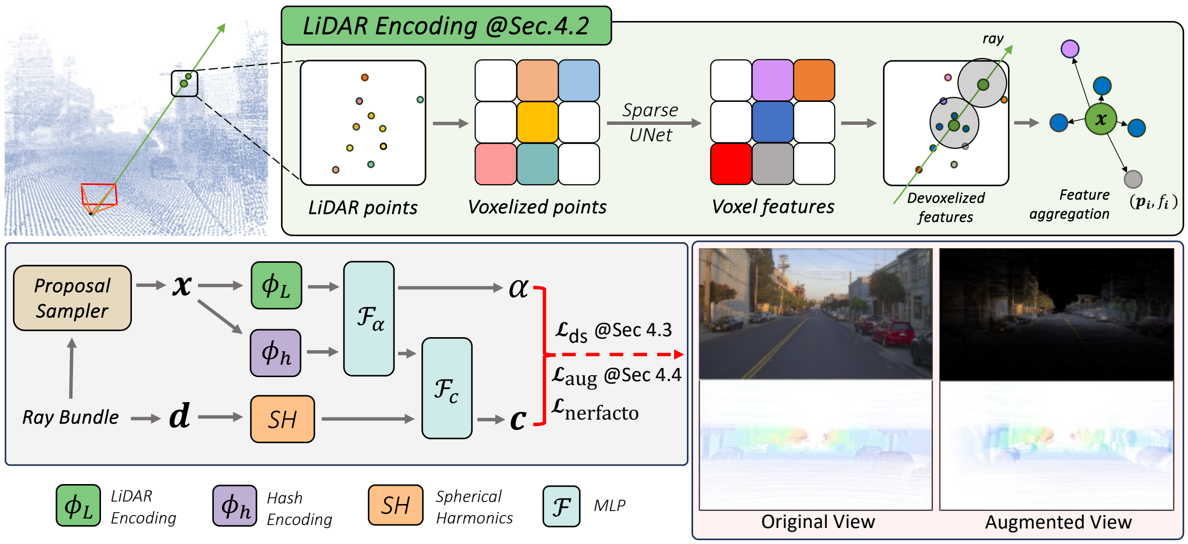

Figure 2. **Overview of LidaRF** – it takes as input the sampled 3D positions $\mathbf{x}$ and ray directions $\mathbf{d}$, and outputs corresponding density α and color $\mathbf{c}$. It incorporates both hash encoding and LiDAR encoding using a sparse UNet. Additionally, augmented training data is generated through LiDAR projections, and the geometry prediction is trained with our proposed robust depth supervision scheme.

inverse-distance weighting to aggregate the neural scene description $f_{i,\mathbf{x}}$ from its K neighboring points,

$$\phi_L(\mathbf{x}) = \begin{cases} \frac{\sum_{i \in \mathcal{S}_\mathbf{x}^K} w_i f_{i,\mathbf{x}}}{\sum_{i \in \mathcal{S}_\mathbf{x}^K} w_i}, & \text{if } \mathcal{S}_\mathbf{x}^K \text{ is not } \varnothing, \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{4}$$

$$\text{and} \quad w_i = \frac{1}{\|\mathbf{p}_i - \mathbf{x}\|}. \tag{5}$$

Feature fusion for radiance decoding. Different from [9, 41] that solely relies on point features, we concatenate the Lidar encoding ϕ_L with hash encoding ϕ_h [24], and apply an MLP $\mathcal{F}_\alpha$ to predict the per-sample density α and density embedding $\mathbf{h}$. Finally, the corresponding color $\mathbf{c}$ is predicted from the spherical harmonics encoding *SH* of the viewing direction $\mathbf{d}$ and density embedding $\mathbf{h}$, via another MLP $\mathcal{F}_\mathbf{c}$:

$$\alpha, \mathbf{h} = \mathcal{F}_\alpha([\phi_L(\mathbf{x}), \phi_h(\mathbf{x})]), \tag{6}$$

$$\mathbf{c} = \mathcal{F}_\mathbf{c}([\mathbf{h}, SH(\mathbf{d})]). \tag{7}$$

4.3. Robust Depth Supervision

Motivation. In addition to the feature encoding, we also derive depth supervision from the Lidar points by projecting them onto the image plane. However, the sparse nature of Lidar points yields limited benefits, that are not sufficient for reconstructing low-texture regions such as road surface. Here, we put forth to accumulate adjacent Lidar frames to increase density. Despite the 3D points accurately capturing the scene structure, one needs to account for inter-points occlusion when projecting them onto the image plane for depth supervision. The occlusion arises due to the increased displacement between the camera and the Lidar from its adjacent frames, yielding bogus depth supervision, as illustrated in Fig. 3. This is non-trivial to handle due to the sparsity of Lidar even after accumulation, making principled graphic techniques such as z-buffering not applicable. In this work, we propose a robust supervision scheme to automatically filter out bogus depth supervision in while training NeRF.

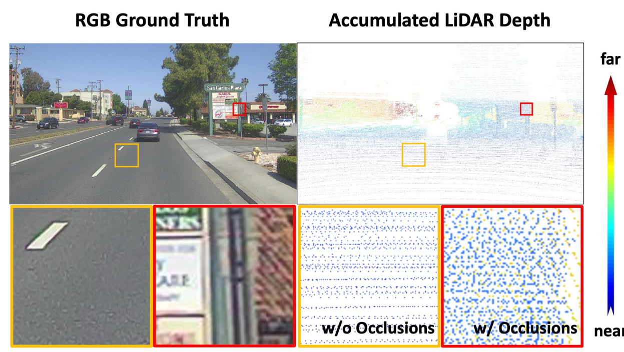

Figure 3. **Illustration of the occlusion issue on the depth map projected from accumulated Lidar points.** Observe that multiple layers of surface points may project to the same region on the image, yielding ghost depth points.

Occlusion-aware robust supervision scheme. We design a curriculum training strategy such that the model initially trains with closer, more reliable depth data, which are less prone to occlusion. As training progresses, the model gradually begins to incorporate more distant depth data. Concurrently, the model develops the capacity to discard depth supervisions that are anomalously distant compared to its predictions. Formally, with the pool of all depth points de-

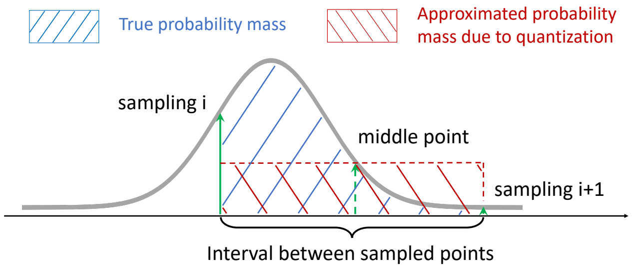

Figure 4. Illustration of the true probability mass and its mid-point approximation.

noted as $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_N\}$, and further denoting the NeRF-rendered depth corresponding to $\mathcal{D}_i$ as $\hat{\mathcal{D}}_i$, we identify the reliable subset of depth points $\mathcal{D}^m_{\text{reliable}}$ in the m-th training iteration as:

$$\mathcal{D}^m_{\text{reliable}} = \{\mathcal{D}_i \mid \mathcal{D}_i {\leq} \epsilon^m_t,\ \ \mathcal{D}_i \leq \hat{\mathcal{D}}_i {+} \epsilon^m_o,\ \ \mathcal{D}_i {\in} \mathcal{D}\}, \quad (8)$$

$$\epsilon^m_t = \min\{\alpha_t \epsilon^{m-1}_t,\ \epsilon_t\}, \quad \alpha_t > 1, \quad (9)$$

$$\epsilon^m_o = \max\{\alpha_o \epsilon^{m-1}_o,\ \epsilon_o\}, \quad \alpha_o < 1. \quad (10)$$

One notices that $\mathcal{D}^m_{\text{reliable}}$ is governed by two scheduled parameters: valid depth threshold ϵ^m_t and valid depth offset ϵ^m_o. The ϵ^m_t serves to filter out depth samples exceeding this threshold, thereby prioritizing nearer depth samples which are less likely to be occluded. As training progresses, ϵ^m_t is exponentially increased at a rate of α_t to involve more depth supervision from further field. Meanwhile, samples exhibiting a depth value far larger than the predicted depth $\hat{\mathcal{D}}_i$ are omitted, as they are likely occluded points. This is thresholded by $\hat{\mathcal{D}}_i + \epsilon^m_o$, with ϵ^m_o decaying exponentially at a rate of α_o, in tandem with the improvement of depth predictions over the course of training.

Lidar Depth Loss For samples in $\mathcal{D}^m_{\text{reliable}}$, we adopt the pixel-level depth loss proposed in URF [27], written as $\mathcal{L}_{\text{ds}} = \mathcal{L}_{\text{depth}} + \mathcal{L}_{\text{sight}}$. In addition to a L_2 loss $\mathcal{L}_{\text{depth}}$ between the rendered depth and the ground truth, a line-of-sight prior $\mathcal{L}_{\text{sight}}$ is applied to further constrain each sampling point individually. But in contrast an approximate computation of $\mathcal{L}_{\text{sight}}$ in NeRFstudio, we implement an exact one. Specifically, we first note that the volume rendering boils down to a weighted sum of the predicted color on sampled points along the ray (see supplementary for equations). Given the weight should ideally concentrate around surfaces, $\mathcal{L}_{\text{sight}}$ enforces the weight distribution to resemble a Gaussian distribution $\mathcal{N}(\hat{\mathcal{D}}_i, \epsilon_n)$ centered at the ground truth depth $\hat{\mathcal{D}}_i$ along the rendering ray, written as

$$\mathcal{L}_{\text{sight}} = \mathbb{E}_{\mathcal{D}_i \in \mathcal{D}^m_{\text{reliable}}} \left[\int_{t_{near}}^{t_{far}} (w(t) - \mathcal{N}(\hat{\mathcal{D}}_i, \epsilon_n))^2 \, dt \right], \quad (11)$$

where w indicates the weight to be integrated along the distance t on the rendering ray from t_{near} to t_{far}. Since the weight w is computed on discrete intervals given by point sampling in NeRF, this loss is discretized to

$$\mathcal{L}_{\text{sight}} = \mathbb{E}_{\mathcal{D}_i \in \mathcal{D}^m_{\text{reliable}}} \left[\sum_i (w_i - \mathcal{N}_i)^2 \right], \quad (12)$$

where $\mathcal{N}_i$ indicates the probability mass within the i-th interval. Here, a possible implementation (*e.g.* in Nerfstudio [29]) to obtain $\mathcal{N}_i$ is by mid-point approximation as illustrated in Fig. 4. However, we note that this approximation is unnecessary and implement differently based on cumulative distribution function (CDF) – the probability mass of a Gaussian distribution can be obtained through its tabulated CDF. We show in supplementary that our exact implementation leads to improved PSNRs.

Methods	Interpolation			Lane Shift	
	PSNR↑	SSIM↑	LPIPS↓	FID↓ @ 2m	FID↓ @ 3.7m
Instant-NGP	24.282	0.733	0.408	140.3	173.2
Mip-NeRF 360	23.693	0.691	0.496	189.4	231.1
Nerfacto	27.122	0.804	0.268	116.7	151.0
UniSim	26.014	0.768	0.342	118.5	141.3
LidaRF (Ours)	27.255	**0.812**	**0.224**	**106.5**	**126.0**

Table 1. **Quantitative comparisons on view synthesis** with state-of-the-art NeRF variants.

4.4. Augmented View Supervision

Recall that a vehicle-mounted camera generates sparse training images with limited view coverage due to its forward motion, posing challenges for NeRF reconstruction, especially when the novel views deviate from vehicle trajectory. Here, we propose to augment training data leveraging Lidar. First, we colorize the point clouds in each Lidar frame by projecting onto its synchronized camera and interpolate the image for RGB values. The colorized point clouds are accumulated as described in Sec. 4.3 and projected onto a set of synthetically augmented views, yielding synthesized images and depth maps as illustrated in Fig. 2. These augmented training views are derived from existing ones, by introducing stochastic perturbations to their camera centers, with the shifting magnitude $\epsilon_a {\in} \mathcal{N}(0, \epsilon_a)$. Nonetheless, such augmented data fails to account for potential occlusions as depicted in Sec. 4.3. Our model, fortified by robust depth supervision, is adept at discerning and excluding occluded Lidar points online. The augmented views are used to train NeRF similarly as the real training views, and we denote the extra loss separately as $\mathcal{L}_{aug}$.

5. Experiments

In this section, we detail our experimental setup and benchmark our method against state-of-the-art NeRF techniques, demonstrating superior photorealism. We further ablate our design choices, underscoring the effectiveness of robust depth supervision, LiDAR encoding, and augmented view supervision in enhancing realism.

5.1. Experimental Setup

Datasets. Following [43], we rely on **Pandaset** [39] as the primary dataset for evaluation, using its front camera and synchronized spinning Lidar. Each scene is consisting of 80 frames captured at 10Hz. We leverage its sensor localization for Lidar accumulation. Since our focus is on static scenes, the dynamic vehicles are masked out during evaluation, similarly as in [9]. We also evaluate on **NuScenes** [6] and **Argoverse2** [37] to compare with S-NeRF [40] and NeRF-LiDAR-cGAN [9], respectively.

Baselines. On Pandaset, we compare our model against several modern implicit-based neural radiance fields methods: **Instant-NGP**, **Mip-NeRF 360**, **Nerfacto** and **UniSim**. Instant-NGP [24] adopts multi-resolution hashing encoding for compact scene representation and efficient rendering. Mip-NeRF 360 [5] adopts integrated position encoding with scene contraction for handling unbounded scenes. Nerfacto [29] combines the compact representation from Instant-NGP and proposal network from Mip-NeRF 360. UniSim [43] is the state-of-the-art simulator for street scenes, reconstructing both the static background and dynamic actors with neural feature grids. In our experiments, we mainly focus on modeling the static background.

Implementation Details. We mask dynamic objects in RGB images using dataset bounding box annotations and an instance segmentation model [13]. Static Lidar points are isolated by omitting points within dynamic objects' 3D bounding boxes. For nearest neighbor searches, we use a CUDA-based FRNN search algorithm [1] to query K=6 nearest Lidar points within a $0.3m$ radius. Our loss weights and scheduling parameters in depth training scheme are given in supplementary. Instant-NGP, Nerfacto, UniSim and our proposed LidaRF use identical size of hash grid and hidden layers in MLPs. Additional details are in the supplementary.

5.2. Novel View Synthesis Results

In our experiments, we assess novel view synthesis under interpolation and extrapolation settings. For interpolation setting, we randomly subsample RGB and Lidar frames, testing on every fourth frame and training on the rest. We follow common practice to report PSNR, SSIM, and LPIPS interpolation views. For extrapolation setting, following [43], we simulate new trajectories by laterally shifting them left or right by 2 or 3.7 meters (lane width of Interstate Highway standards [2]). We report FID at the perceptual level since ground-truth is unavailable,

As shown in Tab. 1, our method outperforms all others in every metric. While methods such as Nerfacto and UniSim show robust performance in the interpolation setting, Mip-NeRF 360 lags behind. The qualitative gap becomes even more significant in extrapolation scenarios. Fig. 5 displays these qualitative differences, where our method exhibits enhanced visual realism compared to the baselines, particularly in rendering fine structures, thanks to our LiDAR encoding. This is especially notable as, even though UniSim also utilizes LiDAR depth supervision, our method more effectively renders low-texture areas like road surfaces, demonstrating the advantage of our deeper utilization of Lidar.

$\mathcal{L}_{\text{ds}}^{1}$	$\mathcal{L}_{\text{ds}}^{10}$	$\mathcal{L}_{\text{ds}}^{\text{HPR}}$	$\mathcal{L}_{\text{ds}}^{\text{robust}}$	ϕ_L	$\mathcal{L}_{\text{aug}}$	PSNR↑	SSIM↑	LPIPS↓	FID↓ @3.7m
☐	☐	☐	☐	☐	☐	27.122	0.804	0.268	151.0
☑	☐	☐	☐	☐	☐	27.016	0.800	0.264	138.2
☐	☑	☐	☐	☐	☐	26.946	0.797	0.264	137.7
☐	☐	☑	☐	☐	☐	27.000	0.799	0.261	139.1
☐	☐	☐	☑	☐	☐	27.090	0.804	0.247	131.7
☐	☐	☐	☑	☑	☐	27.219	0.810	0.228	128.7
☐	☐	☐	☑	☑	☑	**27.254**	**0.812**	**0.223**	**126.0**

Table 2. **Ablation study** of our robust depth supervision, Lidar encoding, and training view augmentation on Pandaset.

5.3. Ablation Study

In our ablation study presented in Tab. 2, we evaluate the impact of three key components – robust depth supervision, Lidar encoding, and augmented view supervision. The baseline for comparison, shown in the first row of Table 2, is the Nerfacto method. We denote different depth supervision strategies as follows: $\mathcal{L}_{ds}^{1}$ for single-frame Lidar depth supervision, $\mathcal{L}_{ds}^{10}$ for depth map supervision using 10 adjacent Lidar frames, and $\mathcal{L}_{ds}$ for the same but within our robust supervision scheme. As another baseline for occlusion handling, we apply hidden point removal (HPR) algorithm [18] implemented in Open3D [48] to remove occluded points as data preprocessing; the supervision after HPR is denoted as $\mathcal{L}_{\text{ds}}^{\text{HPR}}$. Lidar encoding is represented by ϕ_L. Lastly, $\mathcal{L}_{\text{aug}}$ signifies supervision with augmented RGB and depth data derived from Lidar projections.

Effects of Robust Depth Supervision. In Tab. 2, we illustrate that Lidar depth markedly improves LPIPS and FID in lane shift settings with marginal PSNR drop compared to Nerfacto. Notably, our robust supervision scheme with accumulated Lidar points further enhances all metrics. Also note the offline HPR significantly lags behind our more adaptive, NeRF-informed scheme for occlusion handling. Fig. 6 compares different Lidar depth supervision settings. Under the supervision of $\mathcal{L}_{\text{ds}}^{1}$, which utilizes single-frame Lidar depth maps known for their sparsity and reduced occlusions, the rendered depths exhibit high accuracy in texture-rich areas (as indicated by the yellow boxes). However, this accuracy significantly diminishes in regions with thin structures, due to a lack of abundant geometric guidance, as observed in the red boxes. Conversely, the supervision with $\mathcal{L}_{\text{ds}}^{10}$, involving with noisy Lidar depth maps accumulated from 10 adjacent frames, leads to rendered depths that display noticeable noise.

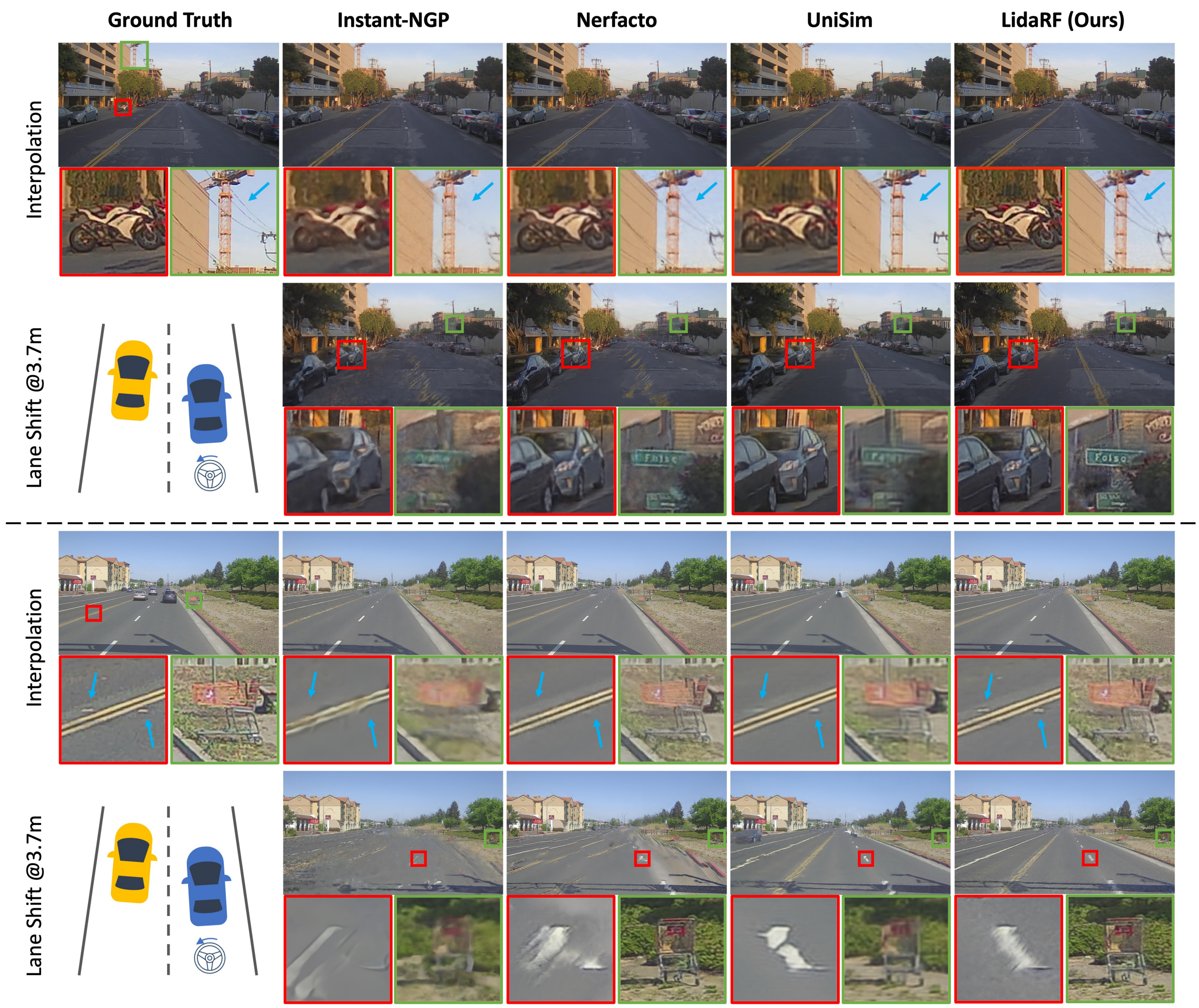

Figure 5. **Qualitative Comparison** on novel view synthesis from different methods. We evaluate on both the interpolation and extrapolation views, the latter of which corresponds to a lane shift. We highlight the performance gap with boxes.

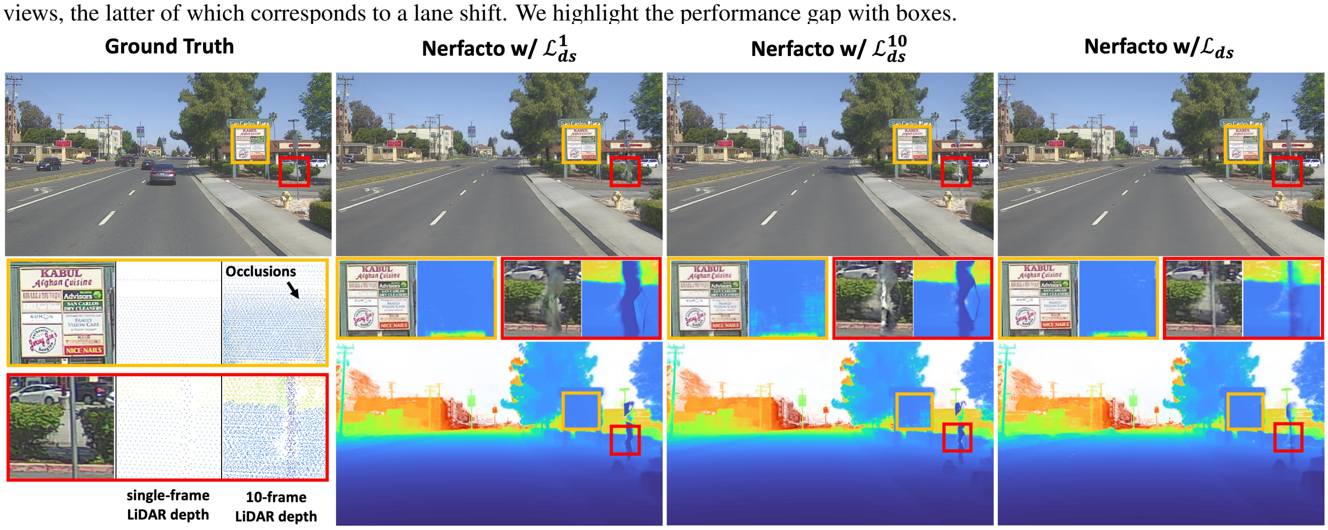

Figure 6. **Qualitative comparison between different LiDAR Depth Supervisions** – With sparse depths, Nerfacto w/ $\mathcal{L}_{\text{ds}}^{i}$ fails to model thin structures. With denser but noisy depths, Nerfacto w/ $\mathcal{L}_{\text{ds}}^{i}$ generate ambiguous depth predictions. Our proposed schemes is robust to occlusions and able to learn delicate structures. Our proposed depth supervision scheme can learn delicate structures with noisy depth map, while the other settings both fail.

Figure 7. **Qualitative comparison w.r.t Lidar encoding** – Lidar encoding is beneficial in modelling sharp textures, *e.g.* power lines.

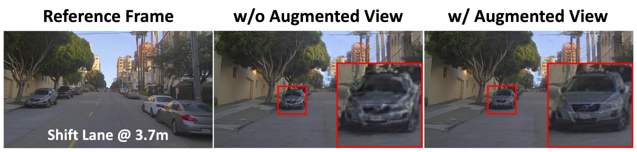

Figure 8. **Qualitative comparison w.r.t augmented view supervision** – it largely improves rendering quality on the regions which are scarcely captured in the raw training data.

Methods	S-NeRF	LidaRF (Ours)			
		w/o $\mathcal{L}_{\text{ds}}$	w/o ϕ_L	w/o $\mathcal{L}_{\text{aug}}$	Full
PSNR↑	29.377	30.629	31.001	31.133	**31.162**
SSIM↑	0.859	0.871	0.873	0.883	**0.884**
LPIPS↓	0.349	0.278	0.237	0.222	**0.211**

Table 3. Evaluation on NuScenes with comparison to S-NeRF.

This is attributed to the prevalent depth ambiguities within the data. Employing our proposed $\mathcal{L}_{ds}^{\text{robust}}$, our methodology effectively leverages denser depth maps for the precise reconstruction of intricate structures (highlighted in red boxes), as well as excludes some occluded depths (noted in yellow boxes). See more examples in the supplementary materials.

Effects of Lidar Encoding. From Tab. 2, it is evident that Lidar encoding contributes to enhancements across all metrics. As shown in Fig. 7, Lidar encoding enables our method to produce sharper textures. This enhancement stems from our sparse convolution-based architecture, which is resilient to Lidar point noise and density variations.

Effects of Augmented View Supervision. While the quantitative benefits of augmented view supervision, as per Tab. 2, appear modest, it consistently enhances performance in extrapolation scenarios across all test scenes. This is particularly notable in scenes where other methodologies fall short. Fig. 8 showcases an instance of this: in scenes with parking cars that are scarcely captured in the raw training data due to the rapid movement of the camera vehicle, our augmented view supervision significantly elevates the quality.

5.4. Results on NuScenes and Argoverse

we evaluate LidaRF on the NuScenes dataset with comparison to S-NeRF, by adapting their method to using the single front camera. We present quantitative results in Tab. 3 and qualitative example in Fig. 9, both showing superior performance from our method over S-NeRF. Ablation study results are also shown in Tab. 3, which further consolidate the efficacy of our proposed components.

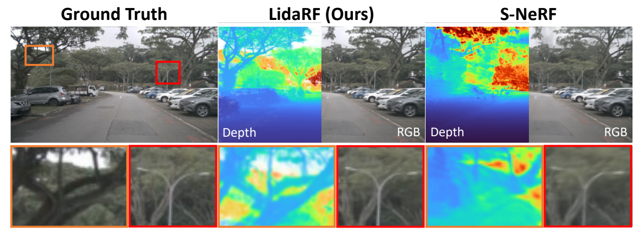

Figure 9. Visual Comparison with S-NeRF on NuScenes.

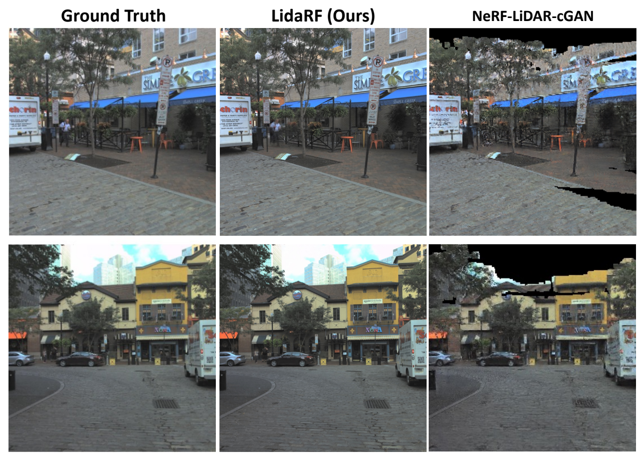

Figure 10. **Qualitative results on Argoverse** with comparison to NeRF-LiDAR-cGAN [9] that solely relies on Lidar encoding, unlike our hybrid scene representation.

Next, we evaluate on Argoverse dataset following the protocol in [9]. We present example qualitative comparisons in Fig. 10 while leaving more evaluations to the supplementary. As can be seen, our LidaRF achieves far superior rendering quality with high resolution. We note the [9]'s blank rendering on regions not covered by Lidar, as they solely rely on Lidar for radiance decoding. This illustrates the advantage of our framework in combining the complementary benefits from Lidar encoding and high-resolution hash grid.

6. Conclusion

In this paper, we focus on unlocking the potential of Lidar to improve NeRF on road scenes, which remain a challenging scenario for novel view synthesis due to highly constrained camera motions. We develop insights on fusing Lidar encoding with high-resolution grid based representation to reap their complementary benefits, and further, extract more robust and extensive depth supervision from Lidar. A limitation of our work is that we currently handle static background only. We envision that the key insights developed in this paper can benefit dynamic objects as well, which remains an interesting future direction to explore.

References

[1] GitHub - lxxue/FRNN: Fixed Radius Nearest Neighbor Search on GPU — github.com. https://github.com/lxxue/FRNN. [Accessed 18-11-2023]. 6

[2] Interstate Highway standards - Wikipedia — en.wikipedia.org. https://en.wikipedia.org/wiki/Interstate_Highway_standards. [Accessed 18-11-2023]. 6

[3] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 696–712. Springer, 2020. 2

[4] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 2

[5] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3, 6

[6] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 6

[7] Alexandra Carlson, Manikandasriram S Ramanagopal, Nathan Tseng, Matthew Johnson-Roberson, Ram Vasudevan, and Katherine A Skinner. Cloner: Camera-lidar fusion for occupancy grid-aided neural representations. *IEEE Robotics and Automation Letters*, 2023. 2

[8] Jen-Hao Rick Chang, Wei-Yu Chen, Anurag Ranjan, Kwang Moo Yi, and Oncel Tuzel. Pointersect: Neural rendering with cloud-ray intersection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8359–8369, 2023. 2

[9] MingFang Chang, Akash Sharma, Michael Kaess, and Simon Lucey. Neural radiance field with lidar maps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17914–17923, 2023. 2, 3, 4, 6, 8

[10] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 3

[11] Kangle Deng et. al. Depth-supervised NeRF: Fewer views and faster training for free. In *CVPR*, 2022. 3

[12] Jianfei Guo, Nianchen Deng, Xinyang Li, Yeqi Bai, Botian Shi, Chiyu Wang, Chenjing Ding, Dongliang Wang, and Yikang Li. Streetsurf: Extending multi-view implicit surface reconstruction to street views. *arXiv preprint arXiv:2306.04988*, 2023. 2

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 6

[14] Rama C Hoetzlein. Fast fixed-radius nearest neighbors: interactive million-particle fluids. In *GPU Technology Conference*, page 2, 2014. 3

[15] Tao Hu, Xiaogang Xu, Ruihang Chu, and Jiaya Jia. Trivol: Point cloud rendering via triple volumes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20732–20741, 2023. 2

[16] Tao Hu, Xiaogang Xu, Shu Liu, and Jiaya Jia. Point2pix: Photo-realistic point cloud rendering via neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8349–8358, 2023. 2

[17] James T Kajiya and Brian P Von Herzen. Ray tracing volume densities. *ACM SIGGRAPH computer graphics*, 1984. 3

[18] Sagi Katz, Ayellet Tal, and Ronen Basri. Direct visibility of point sets. In *ACM SIGGRAPH 2007 papers*, pages 24–es. 2007. 6

[19] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12871–12881, 2022. 2

[20] Jeffrey Yunfan Liu, Yun Chen, Ze Yang, Jingkang Wang, Sivabalan Manivasagam, and Raquel Urtasun. Real-time neural rasterization for large scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8416–8427, 2023. 2

[21] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela L Rus, and Song Han. Bevfusion: Multitask multi-sensor fusion with unified bird's-eye view representation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023. 2, 3

[22] B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, 2020. 1, 2, 3

[23] Thomas Müller. tiny-cuda-nn, 2021. 2, 3

[24] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 2022. 2, 3, 4, 6

[25] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2856–2865, 2021. 2

[26] Julian Ost, Issam Laradji, Alejandro Newell, Yuval Bahat, and Felix Heide. Neural point light fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18419–18429, 2022. 2

[27] Konstantinos Rematas, Andrew Liu, Pratul P Srinivasan, Jonathan T Barron, Andrea Tagliasacchi, Thomas Funkhouser, and Vittorio Ferrari. Urban radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 3, 5

[28] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8248–8258, 2022. 2

[29] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, 2023. 3, 5, 6

[30] Haotian Tang, Shang Yang, Zhijian Liu, Ke Hong, Zhongming Yu, Xiuyu Li, Guohao Dai, Yu Wang, and Song Han. Torchsparse++: Efficient training and inference framework for sparse convolution on gpus. In *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2023. 3

[31] Andrea Vedaldi, Gregorio Guidi, and Stefano Soatto. Moving forward in structure from motion. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE, 2007. 1

[32] Chen Wang, Jiadai Sun, Lina Liu, Chenming Wu, Zhelun Shen, Dayan Wu, Yuchao Dai, and Liangjun Zhang. Digging into depth priors for outdoor neural radiance fields. In *ACMMM*, 2023. 3

[33] Fusang Wang, Arnaud Louys, Nathan Piasco, Moussab Bennehar, Luis Roldão, and Dzmitry Tsishkou. Planerf: Svd unsupervised 3d plane regularization for nerf large-scale scene reconstruction. *arXiv preprint arXiv:2305.16914*, 2023. 2

[34] Guangcong Wang, Zhaoxi Chen, Chen Change Loy, and Ziwei Liu. Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. In *ICCV*, 2023. 3

[35] Zian Wang, Tianchang Shen, Jun Gao, Shengyu Huang, Jacob Munkberg, Jon Hasselgren, Zan Gojcic, Wenzheng Chen, and Sanja Fidler. Neural fields meet explicit geometric representations for inverse rendering of urban scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8370–8380, 2023. 2

[36] Yi Wei, Shaohui Liu, Jie Zhou, and Jiwen Lu. Depth-guided optimization of neural radiance fields for indoor multi-view stereo. *PAMI*, 2023. 3

[37] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*, 2023. 6

[38] Felix Wimbauer, Nan Yang, Christian Rupprecht, and Daniel Cremers. Behind the scenes: Density fields for single view reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9076–9086, 2023. 2

[39] Pengchuan Xiao, Zhenlei Shao, Steven Hao, Zishuo Zhang, Xiaolin Chai, Judy Jiao, Zesong Li, Jian Wu, Kai Sun, Kun Jiang, et al. Pandaset: Advanced sensor suite dataset for autonomous driving. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 3095–3101. IEEE, 2021. 6

[40] Ziyang Xie, Junge Zhang, Wenye Li, Feihu Zhang, and Li Zhang. S-nerf: Neural radiance fields for street views. In *The Eleventh International Conference on Learning Representations*, 2022. 1, 2, 3, 6

[41] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022. 2, 3, 4

[42] Chen Yang, Peihao Li, Zanwei Zhou, Shanxin Yuan, Bingbing Liu, Xiaokang Yang, Weichao Qiu, and Wei Shen. Nerfvs: Neural radiance fields for free view synthesis via geometry scaffolds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16549–16558, 2023. 2, 3

[43] Ze Yang, Yun Chen, Jingkang Wang, Sivabalan Manivasagam, Wei-Chiu Ma, Anqi Joyce Yang, and Raquel Urtasun. Unisim: A neural closed-loop sensor simulator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 1, 2, 3, 6

[44] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021. 2, 3

[45] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *NeurIPS*, 2022. 3

[46] Xiaoshuai Zhang, Abhijit Kundu, Thomas Funkhouser, Leonidas Guibas, Hao Su, and Kyle Genova. Nerflets: Local radiance fields for efficient structure-aware 3d scene representation from 2d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8274–8284, 2023. 2

[47] Yi Zhang, Xiaoyang Huang, Bingbing Ni, Teng Li, and Wenjun Zhang. Frequency-modulated point cloud rendering with easy editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 119–129, 2023. 2

[48] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 6

[49] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Sampling: Scene-adaptive hierarchical multiplane images representation for novel view synthesis from a single image. In *ICCV*, 2023. 2