NeRF-LiDAR: Generating Realistic LiDAR Point Clouds with Neural Radiance Fields

Junge Zhang[1], **Feihu Zhang**[2], **Shaochen Kuang**[3], **Li Zhang**[1]*
[1]Fudan University
[2]University of Oxford
[3]South China University of Technology
jgzhang17@fudan.edu.cn, lizhangfd@fudan.edu.cn

Abstract

Labelling LiDAR point clouds for training autonomous driving is extremely expensive and difficult. LiDAR simulation aims at generating realistic LiDAR data with labels for training and verifying self-driving algorithms more efficiently. Recently, Neural Radiance Fields (NeRF) have been proposed for novel view synthesis using implicit reconstruction of 3D scenes. Inspired by this, we present NeRF-LIDAR, a novel LiDAR simulation method that leverages real-world information to generate realistic LIDAR point clouds. Different from existing LiDAR simulators, we use real images and point cloud data collected by self-driving cars to learn the 3D scene representation, point cloud generation and label rendering. We verify the effectiveness of our NeRF-LiDAR by training different 3D segmentation models on the generated LiDAR point clouds. It reveals that the trained models are able to achieve similar accuracy when compared with the same model trained on the real LiDAR data. Besides, the generated data is capable of boosting the accuracy through pre-training which helps reduce the requirements of the real labeled data. Code is available at https://github.com/fudan-zvg/NeRF-LiDAR

Introduction

LiDAR sensor plays a crucial role in autonomous driving cars for 3D perception and planning. However, labelling the 3D point clouds for training 3D perception models is extremely expensive and difficult. In view of this, LiDAR simulation that aims at generating realistic LiDAR point clouds for different types of LiDAR sensors becomes increasingly important for autonomous driving cars. It can generate useful LiDAR data with labels for developing and verifying the self-driving system.

Many previous works have studied the LiDAR simulation, which can be mainly categorized into two types: the virtual environment creation method and the reconstruction-based method. The former creates the 3D virtual world by graphics-based 3D modeling and then generates the 3D LiDAR point clouds by physics-based simulation (ray tracing). These kinds of works (Dosovitskiy et al. 2017; Koenig and Howard 2004) have natural limitations as it's impossible for 3D modelers to create a virtual world that is the same as the complex real world. The simulated LiDAR points have significant domain differences from the real LiDAR points and cannot be used to train robust deep neural network models. The latter (Manivasagam et al. 2020; Fang et al. 2020) relies on multiple LiDAR scans to densely reconstruct the street background and then place the foreground objects into the background. However, it's expensive to collect dense LiDAR scans which may need special devices (Fang et al.

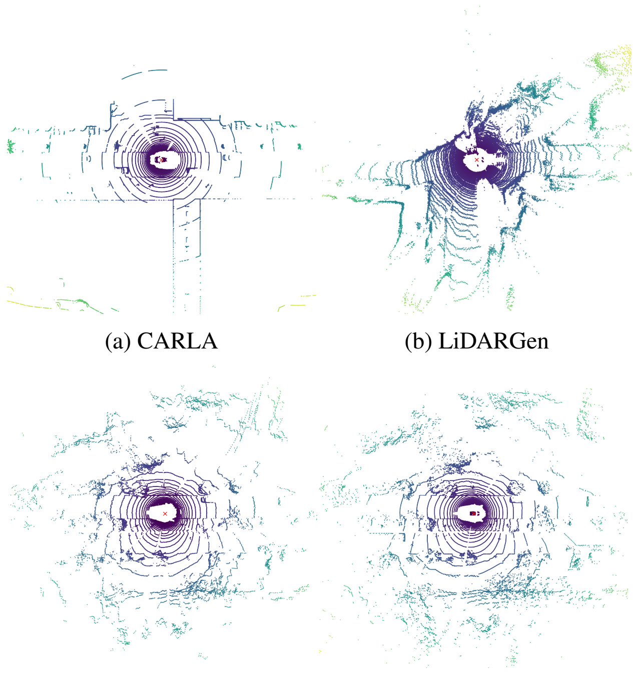

Figure 1: Comparisons of results between our NeRF-LiDAR and other existing LiDAR simulation methods. (a) Method (Dosovitskiy et al. 2017) that creates virtual world for LiDAR simulation. (b) Diffusion model used for LiDAR generation (Zyrianov, Zhu, and Wang 2022). (c) Our NeRF-LiDAR can generate realistic point clouds that is nearly the same as the real LiDAR point clouds (d).

*Li Zhang is the corresponding author with School of Data Science, Fudan University.

2020). Moreover, it's expensive to generate point-wise semantic labels for simulated LiDAR data. It still requires human annotations on the 3D scans.

Recently, Neural Radiance Fields (NeRF) (Mildenhall et al. 2020; Barron et al. 2021) have been proposed for implicit reconstruction of the 3D object/scenes with multiple images as inputs. NeRF can render photo-realistic novel views along with dense depth maps.

Inspired by this, we proposed to learn a NeRF representation for real-world scenes and render LiDAR point clouds along with accurate semantic labels. Different from existing reconstruction-based LiDAR-simulation methods (Manivasagam et al. 2020; Fang et al. 2020) or the virtual world creation (Dosovitskiy et al. 2017) (Fig. 1a), our method takes full use of the multi-view images to implicitly reconstruct the labels and 3D real-world spaces. The multi-view images can assist the simulation system to learn more accurate 3D geometry and real-world details and generate more accurate point labels. The proposed NeRF-LiDAR model consists of two important modules: 1) the reconstruction module that uses NeRF to reconstruct the real world along with labels; 2) the generation module that learns to generate realistic point clouds through a point-wise alignment and a feature-level alignment.

Since our NeRF-LiDAR can generate realistic LiDAR point clouds along with accurate semantic labels, we verify the effectiveness of our NeRF-LiDAR by training different 3D segmentation models on the generated data. The trained 3D segmentation models are shown able to achieve competitive performance when compared with the same model trained on the real LiDAR data which implies that the generated data can be directly used to replace the real labeled LiDAR data. Besides, by using the generated LiDAR data for pre-training and a small number of real data (e.g., 1/10) for fine-tuning, the accuracy can be significantly improved by a large margin which is even better than the model trained on a 10 times larger real LiDAR dataset.

Related Work

LiDAR point-cloud simulation has been studied for many years from the initial engine based rendering to the state-of-the-art real-world reconstruction-based LiDAR rendering.

LiDAR Simulation The first type of the LiDAR simulation method (Dosovitskiy et al. 2017; Koenig and Howard 2004; Yue et al. 2018; Gschwandtner et al. 2011) relies on creating 3D virtual world and rendering the point clouds with physics-based simulation. However, the generated virtual data have large domain gaps with the real data when used for training deep neural networks. This is because the 3d virtual world cannot simulate the complexity and details of the real world. Another point-cloud simulation method (Achlioptas et al. 2018; Luo and Hu 2021; Yang et al. 2019; Zyrianov, Zhu, and Wang 2022) generates 3D point clouds based on generative models. However, these generated data cannot be used to train models as they also have significant domain differences with real point clouds. Moreover, it's difficult to generate labels for the point clouds.

State-of-the-art LiDAR simulation methods (Fang et al. 2020; Manivasagam et al. 2020) first reconstruct the real-world driving scenes into 3D meshes and then run the physics-based simulation. In order to achieve dense accurate reconstruction results, these methods need to scan the street many times using expensive LiDAR devices (Fang et al. 2020). More importantly, it's still expensive to generate point-wise semantic labels for simulated LiDAR data as it requires human annotations on the reconstructed 3D scenes. Instead of simulating whole LiDAR scenes, others, e.g., (Fang et al. 2021) use the real-world 3D scenes and propose a rendering-based LiDAR augmentation framework to enrich training data and boost performance of LiDAR-based 3D object detection. Our method also leverages real-world information for learning LiDAR simulation. Our NeRF-LiDAR creates an implicit neural-radiance-field representation of the real world for both point clouds and label rendering.

Neural Radiance Fields Recently, Neural radiance fields (NeRF) (Mildenhall et al. 2020) have been proposed as an implicit neural representation of the 3D real world for novel view synthesis. NeRFs can take multiple 2D images and their camera-view directions to represent the whole 3D space. However, early NeRFs are only applicable to small object-centric scenes.

Many recent NeRFs have been proposed to address the challenges of large-scale outdoor scenes (Barron et al. 2021, 2022; Tancik et al. 2022; Zhang et al. 2020). There are also some methods (Rematas et al. 2022; Deng et al. 2022) leveraging depth supervision to help create more accurate 3D geometry of scenes. Panoptic or semantic label synthesis for novel views is also explored in (Zhi et al. 2021; Kundu et al. 2022; Fu et al. 2022). They utilize the density of the volume to render image labels along with the novel view synthesis. Inspired by these works, our method reconstructs the accurate 3D geometry using the NeRF methodology in the driving scene and generates 3D point clouds along with accurate semantic labels for the LiDAR simulation.

NeRF-LiDAR

In this section, we present our NeRF-based LiDAR simulation framework (as shown in Fig. 2). The method consists of three key components: 1) NeRF reconstruction of the driving scenes, 2) realistic LiDAR point clouds generation and 3) point-wise semantic label generation. We formulate the three components into end-to-end deep neural network models for learning LiDAR simulation.

Neural Radiance Fields Neural Radiance Fields learn implicit representation for the scenes and render novel view synthesis through volume rendering. It learns a function $f : (\mathbf{x}, \mathbf{v}) \to (\mathbf{c}, \sigma)$ for mapping coordinates $\mathbf{x}$ and viewing directions $\mathbf{v}$ to color $\mathbf{c}$ and density σ. The volume rendering is based on discrete rays $\mathbf{r} = \mathbf{o} + t\mathbf{d}$ in the space and applying numerical integration along the rays to query color:

$$\hat{\mathbf{C}}(\mathrm{r}) = \sum_{i=1}^{N} T_i(1 - e^{-\sigma_i \delta_i})\mathbf{c}_i,\ T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right), \tag{1}$$

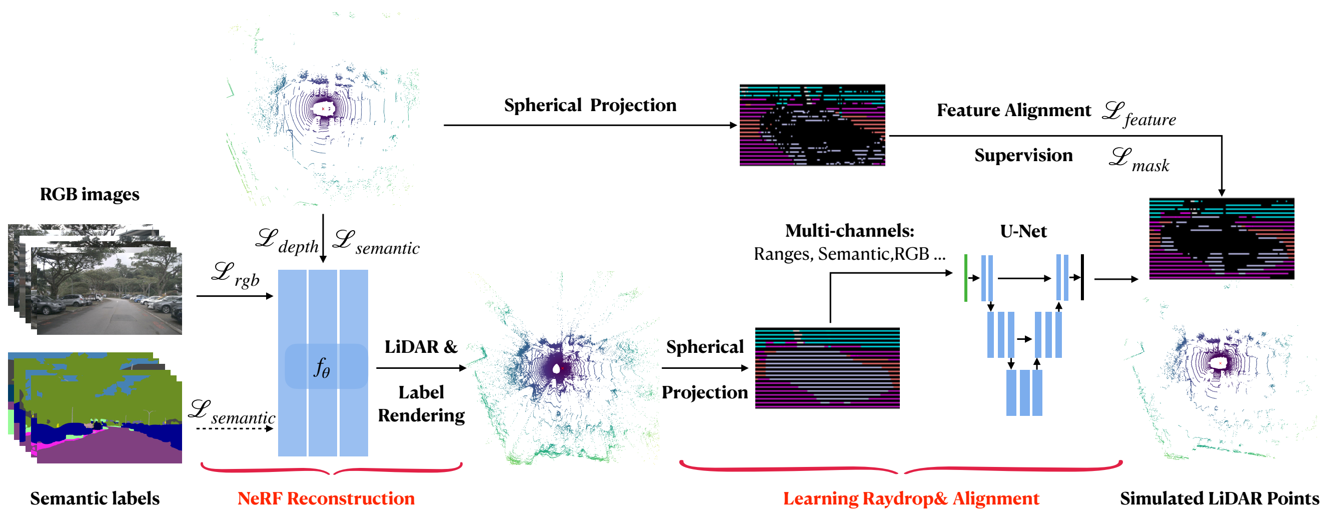

Figure 2: Schematic illustration of NeRF-LiDAR. Image sequences along with the predicted weak semantic labels are used as inputs to reconstruct the implicit NeRF model. LiDAR signals are also used to help create more accurate 3D geometry. Initial coarse point clouds are generated by the NeRF reconstruction through Eq. (5)∼(7). The initial point clouds are projected into 2D equirectangular images. We then utilize a U-Net to learn raydrop and the alignment (detailed in Fig. 3) to make the generated point clouds more realistic.

where $\mathbf{o}$ is the origin of the ray, $\mathbf{d}$ is the direction, T_i is the accumulated transmittance along the ray, and $\mathbf{c}_i$ and σ_i are the corresponding color and density at the sampled point t_i. $\delta_j = t_{j+1} - t_j$ refers to the distance between the adjacent sampled points.

NeRF Reconstruction State-of-the-art LiDAR simulation methods (Manivasagam et al. 2020) rely on dense LiDAR scans for scene reconstruction. To achieve the dense reconstruction of the street, (Fang et al. 2021) uses a special (expensive) LiDAR device to collect dense depth maps. (Manivasagam et al. 2020) scans the street many times to accumulate much denser point clouds. These dense depth maps or point clouds are then used to extract the meshes of the street. Finally, the meshes are used to generate point clouds of different types of LiDAR sensors.

In this paper, we present a new method takes multi-view images and sparse LiDAR signals to reconstruct the street scenes and represent the 3D scenes as an implicit NeRF model. We propose to use the driving-scene data to learn the NeRF reconstruction.

NeRF reconstruction of the unbounded large-scale driving scenes is challenging. This is because most of NeRFs (Mildenhall et al. 2020) are designed for small scene reconstruction with object-centric camera views. However, the driving data are often collected in the unbounded outdoor scenes without object-centric camera views settings (e.g., nuScenes (Caesar et al. 2020)). Moreover, since the ego car moves fast during the data collection, the overlaps between adjacent camera views are too small to be effective for building multi-view geometry. We reconstruct the NeRF representation based on the multi-view images and leverage the LiDAR points to provide extra depth supervision to create more accurate 3D geometries. Besides, the real LiDAR point clouds are used as supervision to learn more realistic simulated LiDAR data.

To reconstruct the driving scenes, we use the unbounded NeRF (Barron et al. 2022) with a modified supervision of:

$$\mathcal{L}_{rgb} = \|\hat{\mathbf{C}}(\mathbf{r}) - \mathbf{C}(\mathbf{r})\|_2, \tag{2}$$

$$\mathcal{L}_{depth} = \|\hat{\mathcal{D}}(\mathbf{r}) - \mathcal{D}(\mathbf{r})\|_1. \tag{3}$$

Here, $\hat{D}$ is the rendered depth by the volume rendering in Eq. (1):

$$\hat{\mathcal{D}}(\mathbf{r}) = \sum_{i=1}^{N} T_i \left(1 - e^{-\sigma_i \delta_i}\right) z_i, \tag{4}$$

where z_i is the depth value at the sampled point t_i on the ray $\mathbf{r}$. Since the original unbounded NeRF (Barron et al. 2022) is extremely slow which takes about one day to train each NeRF block, we adopt hash-encoding NeRF (Barron et al. 2023) to speed up the simulation process.

Point-cloud Generation After learning the implicit NeRF representation of the driving scenes, we set a virtual LIDAR to simulate the real LIDAR sensor. The virtual LIDAR shares the same parameter settings with the real LiDAR sensors. For example, nuScenes (Caesar et al. 2020) uses Velodyne HDL32E LiDAR sensor, of which the spinning speed is 20Hz and the field of view ranges from -30.67 degree to 10.67 degree (consists of 32 channels). We can therefore simulate NeRF-LiDAR rays with the LiDAR center ($\mathbf{o}$) and direction $\mathbf{d}$ accordingly:

$$\mathbf{d} = (\cos\theta\cos\phi,\ \sin\theta\sin\phi,\ \cos\phi)^T, \tag{5}$$

where θ, ϕ represent the azimuth and vertical angle of the ray, determined by the time interval of the lasers and the settings of the LiDAR sensor.

The origin of the rays $\mathbf{o}$ changes according to the defined motion of ego cars.

$$\mathbf{o} = \mathbf{o_0} + \Delta t \cdot \mathbf{v}. \tag{6}$$

Here, $\mathbf{v}$ is the velocity of the ego vehicle and Δt means the time interval from the previous state. Δt is decided by the frame rate of the LiDAR sensors (e.g., 20Hz for nuScenes LiDAR).

Each simulated point $\mathbf{p} = \{x, y, z\}$ can be then calculated by the pre-defined directions $\mathbf{d}$ of rays and the distances $\hat{D}$ from the LiDAR sensor to the real world objects:

$$\mathbf{p} = \mathbf{o} + \hat{D}\mathbf{d}. \tag{7}$$

There are about 20∼40k points in one frame of a standard 32-channel LiDAR point clouds. To simulate the whole point clouds, for each point, we render a ray to compute the exact 3D location.

Label Generation To achieve the point-wise semantic labels of the simulated LiDAR points, we use the 2D semantic labels of the images to learn the 3D label generation.

Semantic NeRF (Zhi et al. 2021) proposes to use the semantic logits that could be rendered through volume rendering (Eq. (1)) like RGB color:

$$\hat{\mathbf{S}}(\mathrm{r}) = \sum_{i=1}^{N} T_i(1 - e^{-\sigma_i \delta_i})\mathbf{s}_i, \tag{8}$$

where T_i, σ_i, δ_i follows the definition of Eq. 1, s_i is the semantic logit of the sampled point.

Here, we first consider the most difficult cases where there is no annotated label from the collected driving data (images and LiDAR points). Given unlabeled real images collected from multiple cameras of the self-driving cars, we train a SegFormer (Xie et al. 2021) model, on the mixture of other datasets including Cityscapes (Cordts et al. 2016), Mapillary (Neuhold et al. 2017), BDD (Yu et al. 2020), IDD (Varma et al. 2019) to compute weak labels that serve as inputs to the NeRF reconstruction model. To achieve better cross-dataset generalization of the SegFormer and avoid conflicts in the label definition, we utilize the learning settings and label merging strategy in the (Lambert et al. 2020).

Considering that the generated weak labels may have many outliers, to reduce the influence of these outliers and generate more accurate 3D point labels, we take full use of multi-view geometric and video spacial temporal consistency in our NeRF reconstruction.

In the NeRF training, we combine the image-label supervision into the reconstruction learning by constructing the semantic radiance fields:

$$\hat{\mathcal{L}}_l = CE(\hat{\mathbf{S}}(\mathbf{r}), \mathbf{S}(\mathbf{r})), \tag{9}$$

where CE is the cross-entropy loss, $\mathbf{r}$ represents pixel-wise camera rays corresponding to each image pixel, $\hat{\mathbf{S}}(\mathbf{r})$ is the rendered labels by the NeRF model (Eq.(8)) and $\mathbf{S}(\mathbf{r})$ is the label predicted by the image segmentation model.

In some other cases, when there is a small number of labeled images or LiDAR frames, we can also leverage the existing ground-truth labels for more robust label generation. For example, in the nuScenes dataset, a small part of the LiDAR frames (about 1/10) was labeled with semantic annotations. We take the sparse 3D point labels along with the weak 2D image labels to learn more accurate semantic radiance fields.

$$\mathcal{L}_l = CE(\hat{\mathbf{S}}(\mathbf{r}_{LiDAR}), \mathbf{S}(\mathbf{r}_{LiDAR})). \tag{10}$$

Here $\mathbf{r}_{LiDAR}$ represents the point-wise rays emitted by the LIDAR sensor. The total loss for learning our NeRF reconstruction can be represented as:

$$\mathcal{L}_{rec} = \mathcal{L}_{depth} + w_{rgb}\mathcal{L}_{rgb} + \hat{w_l}\hat{\mathcal{L}}_l + \mathcal{L}_l, \tag{11}$$

where w_{rgb} and w_l balance the RGB geometry reconstruction, the LiDAR rendering and the semantic label rendering.

Learning Raydrop & Alignment In the real world, the LiDAR sensor cannot receive all beams emitted by itself, influenced by the reflectance ratio of different materials (e.g., glasses), the incidence angle and many other factors (Manivasagam et al. 2020; Fang et al. 2020). Points are usually dropped when the reflected intensity is below the perception threshold. To make generated LIDAR points closer to the real LIDAR points, we learn a raydrop processing on the generated dense points.

NeRF-LiDAR allows us to render depths (3D points) at arbitrary positions and directions. We use the ground-truth LiDAR frames as supervision to learn the raydrop. Given one ground-truth LiDAR frame P, we render the simulated LiDAR frame $\hat{P}$ at the same location accordingly. The ground-truth P and the simulated $\hat{P}$ should have strong point-wise correspondence.

$$P \simeq \hat{P}. \tag{12}$$

We adopt such point-to-point correspondence as the learning target.

Equirectangular Image Projection It's difficult to create a point-to-point correspondence between the two irregular 3D point clouds. To better leverage the point-wise correspondence, we first render all generated points into a 2D equirectangular image (a panorama sparse depth image, as illustrated in Fig. 2). For example, in nuScenes dataset, the resolution of the 32-channel LiDAR equirectangular image is set as 32×1024.

To project the irregular LiDAR points, we adopt the spherical projection (Geiger, Lenz, and Urtasun 2012) to project our points into the equirectangular image grids (as illustrated in Figure 3). Similarly, the real LiDAR frame is also transferred into a 2D 32×1024 equirectangular image. In this way, we can easily create the correspondence in 2D grids.

Point-to-point Alignment To learn the drop probabilities for each 2D grid location in the equirectangular image, we employ a standard U-Net which encodes the depth ranges, semantic labels, RGB textures, and depth variances between neighborhoods into a feature representation. The U-Net outputs a 2D probability map (a binary mask) to represent the raydrop results. We take the corresponding real LiDAR equirectangular image as the learning target.

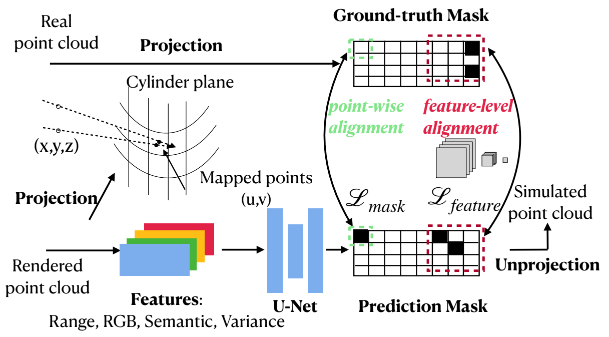

Figure 3: Illustration of learning raydrop and alignment. The initial coarse point clouds are projected into 2D equirect-angular images. We use the projected depth, RGB texture, and depth variances as input to a standard U-Net. The U-Net learns the raydrop mask to improve the initial coarse point clouds through the point-wise alignment (Eq. (13)) and the feature-level alignment (Eq. (14)). Finally, the refined equirectangular images are back-projected to 3D space to achieve the expected LiDAR point clouds.

$$\mathcal{L}_{mask} = CE(\hat{M}, M_{gt}), \tag{13}$$

where $\hat{M}, M_{gt}$ represent predicted and true mask respectively. Extra points/grids are dropped through the learned drop mask. The expected 3D point clouds can be achieved through back-projection of the equirectangular image.

Feature-level Alignment The above point-wise alignment aims at making the generated points more realistic or spatially closer to the real point locations. However, the generated LiDAR data will finally be used to train deep neural network models for 3D perceptions. To make the generated data more effective and able to achieve better accuracy when training deep neural networks (e.g., 3D perception models), we propose the feature-level alignment to further regulate the distribution of the generated point clouds.

$$\mathcal{L}_{feat} = \sum_{k=1}^{n} w_k \|F(\hat{I})_k - F(I_{gt})_k\|_1, \tag{14}$$

where F is a pre-trained and fixed feature extractor (e.g., VGG (Simonyan and Zisserman 2015), Point-cloud segmentation Net (Milioto et al. 2019)). We use n-level pyramidal features to compute the feature distance. $n = 4$ is the number of feature levels, $w_k = 2^{k-n}$ is the weights for kth-level features. The loss L_{feat} measures the feature-level similarity between the real and the simulated point clouds. To enable the back-propagation from the feature loss to the previous raydrop module, we apply the gumble-softmax (Jang, Gu, and Poole 2016) on the ray drop processing.

The whole generation target can be represented as:

$$\mathcal{L}_{gen} = \mathcal{L}_{mask} + w_{feat}\mathcal{L}_{feat}. \tag{15}$$

The feature-level distribution alignment can make the generated data more effective and achieve better accuracy in training segmentation networks. Besides, we find that it also helps to remove extra outliers in the generated point clouds (examples are shown in Fig. 4 and the supplementary).

Experiment

Experimental Settings

Dataset We use the standard nuScenes self-driving dataset for training and evaluation. NuScenes contains about 1000 scenes collected from different cities. Each scene consists of about 1000 images in six views that cover the 360° field of view (captured by the front, right-front, right-back, back, left-back and left-front cameras). 32-channel LiDAR data are also collected at 20 Hz. Human annotations are given in key LiDAR frames (one frame is labeled in every 10 frames).

We use both unlabeled images and LiDAR data for training our NeRF-LiDAR model. The labeled key LiDAR frames are used for evaluations. Limited by computing resources, we take 30 nuScenes sequences from the whole dataset. Each sequence covers a street scene with a length of 100$\sim$200m, and the total length is about 4km. Namely, we reconstruct about 4km of driving scenes for training and evaluation.

Evaluation Settings To avoid conflicts in label definitions, we remap image segmentation labels into five categories (road, vehicles, terrain, vegetation and man-made) in accordance with the nuScenes LiDAR segmentation labels.

In the training set, we use a total of 7000 unlabeled LiDAR frames and 30000 images for training our NeRF-LiDAR model. There are extra 1000 labeled LiDAR frames provided in these nuScenes scenes. We mainly use these labeled data for testing and fine-tuning in the experiments.

To evaluate the quality of the generated point clouds and point-wise labels, we train different LiDAR segmentation models (Cylinder3D (Zhou et al. 2021) and RangeNet++ (Milioto et al. 2019)) on the generated data and compare the segmentation model with those models trained on the real nuScenes LiDAR data(25k iterations).

We use two evaluation sets to evaluate the 3D segmentation results. The first *Test Set 1* consists of 400 labeled real point clouds that is extracted from the 30 reconstructed scenes which are not used for training. This validation set is from the same scenes as the simulation data.

The second *Test Set 2* is the whole nuScenes validation set which consists of $\sim$5700 LiDAR point clouds from other nuScenes scenes (not including the 30 selected scenes). This is used to test the quality and generalization/transfer abilities of the simulation data. We test the trained model in other unseen scenes (*results are available in the supplementary*).

	Test Set 1					
Training Set	road	vege.	terrain	vehicle	manmade	mIoU
CARLA	76.4	47.3	×	33.7	54.4	52.9*
Real 1k	96.2	83.6	83.1	83.0	86.4	86.5
Real 10k	97.0	83.6	84.5	89.3	**87.8**	88.4
Sim20k	93.5	70.4	77.6	79.1	80.7	80.3
Sim20k + real1k	**97.1**	**84.1**	**85.3**	**92.2**	86.9	**89.1**

Table 1: Evaluation an comparisons with the real LiDAR data and CARLA. * mean IoU of four classes.

Raydrop			**Feature loss**		**mIoU**
no raydrop	random	learning	vgg	rangenet	
✓					65.7
	✓				63.5
		✓			66.3
		✓	✓		66.5
		✓		✓	**69.9**

Table 2: Ablations on different settings of ray drop and feature loss. Models are evaluated on validation set *Test Set 2*.

Ablation Study

To demonstrate the effectiveness of our method components, in Table 2, we conduct experiments on different components of our NeRF-LiDAR. We conduct ablation studies on 25 sequences of all data.

Effects of Raydrop We compare three different raydrop settings in Table 2 and Fig. 4. Without any raydrop, the generated LiDAR data report a mIoU of 65.7 after training the 3D segmentation model (Zhou et al. 2021). By adding a random drop strategy, the mIoU is dropped to 63.5. And with our learning-based raydrop, the mIoU is improved to 66.3.

Effects of Feature Loss We also evaluate the effects of the feature loss in Table 2. We use two different feature extractors (VGG (Simonyan and Zisserman 2015) and RangeNet++ (Milioto et al. 2019)) to implement the feature-level alignment. Without feature loss for feature-level alignment, our method reports a mIoU of 66.3. By using the pre-trained VGG Net(Simonyan and Zisserman 2015) for feature alignment, the result is improved to 66.5. By implementing a pre-trained 3D segmentation network as the feature extractor, the results are significantly improved to 69.9.

Label Quality

In Fig. 5, we visualize the generated LiDAR labels and compare them with ground-truth labels in the real data. We can observe that the generated point clouds by our NeRF-LiDAR have strong point-to-point correspondence with the real data and labels. The labels are accurate and close to manual annotations. In the supplementary, we also evaluate the quality of the labels by computing the mIoU by comparing it with

	road	vege.	terrain	vehicle	manmade	mIoU
LiDAR + pseudo Seg	70.6	39.4	30.7	20.9	52.0	42.7
NeRF-LiDAR 10k	**91.6**	**61.1**	**59.2**	**69.7**	**68.0**	**69.9**

Table 3: Comparisons between NeRF-LiDAR data and unlabeled real LiDAR data with pseudo segmentation labels.

the manual labels. NeRF-LiDAR can generate accurate labels with a high mIoU of 80$\sim$95 under different settings.

Pseudo Segmentation of Real LiDAR Scenes. Compared with the pseudo segmentation labels on the real data, our NeRF-LiDAR can generate harder cases which have not been be collected by the dataset. These data significantly improve the diversity of the training dataset and boost the accuracy in training 3D segmentation models (Table 3).

Comparisons with State of the Art

CARLA and Real LiDAR In Table 1, we evaluate the quality of the generated data by training 3D segmentation model (Zhou et al. 2021). Mean IoU is used as evaluation metric.

In Table 1, we use the predicted weak image segmentation labels and 1000 labeled LiDAR frames to train our NeRF-LiDAR. We use the *Test Set 1* which is extracted from the same scenes as the simulation data to evaluate the accuracy of the 3D segmentation models. CARLA simulator (Dosovitskiy et al. 2017) and different real LiDAR sets are taken as baselines. When we train the point cloud segmentation network on the 20k simulation data, it achieves a mIoU of 80.3, which is close to real 1k data and far better than the model trained on the 20k CARLA data. If we use 1k real data for fine-tuning, the mIoU can be further improved to 89.1, which exceeds real 10k data.

Reconstruction-based Simulators Our NeRF-LiDAR possesses apparent advantages over other reconstruction-based LiDAR simulators (Manivasagam et al. 2020; Fang et al. 2020). First, RGB images are used to assist the reconstruction in our NeRF-LiDAR, providing useful multi-view geometry information for reconstruction and label generation. Secondly, no manual annotations on the point clouds are required. We use NeRF representation to learn label generation. Multi-view images provide useful label information and geometry consistency to reduce the outlier labels. Finally, NeRF-LiDAR does not require dense point clouds for reconstruction of the real world. LiDARSim (Manivasagam et al. 2020) uses a 64-channel LiDAR and scans the street many times to achieve dense point clouds for simulation. Augmented LiDAR simulator (Fang et al. 2020) utilizes a special and expensive LiDAR device to achieve the dense point clouds. As a comparison, NeRF-LiDAR relies more on 2D images to achieve 3D geometry accuracy.

We compare our method with LiDARSim (Manivasagam et al. 2020) in Table 4 and Figure 6 . Considering that the official code of LiDARSim is not published, we tried our best to reproduce the procedures, i.e., accumulating the LiDAR points, calculating normals, building meshes and doing ray-casting and ray-dropping. However, the aggregated points are not dense enough to build high-quality meshes and thus

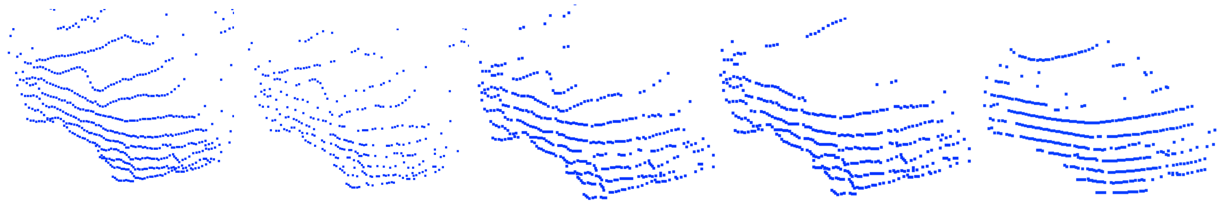

Figure 4: Comparisons of different settings for LiDAR rendering. (a) Point clouds without raydrop, (b) Point clouds after random raydrop. (c) Point clouds after our learning based raydrop but without using the feature-level alignment. (d) The final generated point clouds with both learning based raydrop and the feature-level alignment. (e) the real LiDAR point clouds.

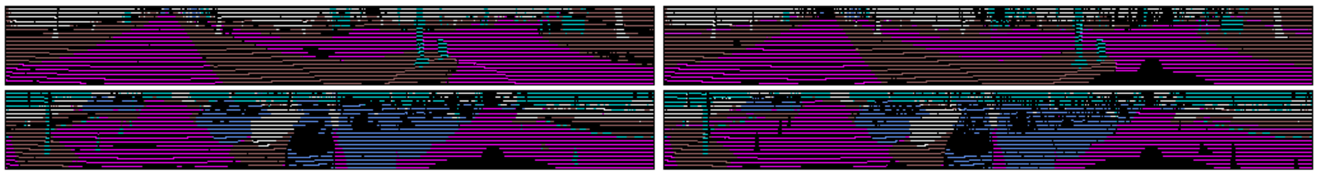
(a) Our Data & Label Rendering
(b) Ground-truth Labels of the Real Data

Figure 5: Comparisons between the data and label generated by the NeRF-LiDAR and the real LiDAR data with human annotations. For better visualization, we project the 3D point cloud as 2D equirectangular image with colorized labels. Our NeRF-LiDAR (a) is shown able to generate accurate labels and realistic point clouds that is almost the same as the real data (b).

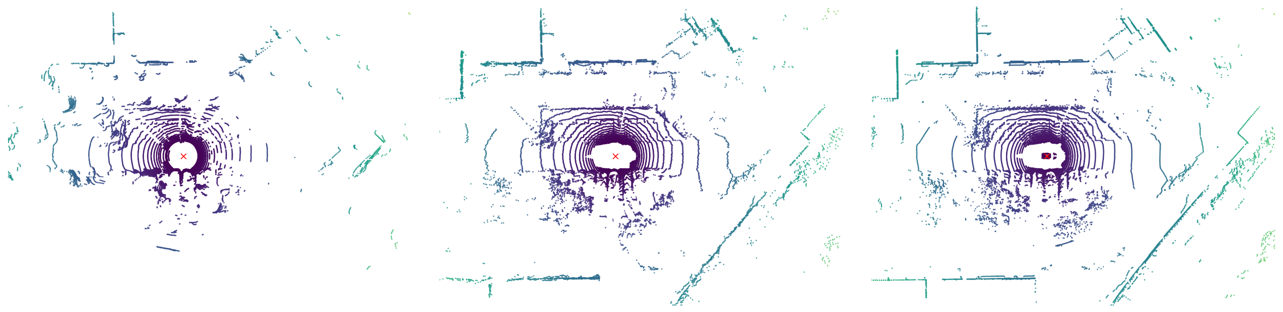

Figure 6: Comparison between our NeRF-LiDAR and LiDARSim (Manivasagam et al. 2020). Sparse aggregated point cloud leads to poor-quality mesh when reconstructing scenes thus the simulated LiDAR points lose reality compared to NeRF-LiDAR.

	road	vege.	terrain	vehicle	manmade	mIoU
LiDARSim	83.1	55.1	39.1	36.7	75.2	57.8
NeRF-LiDAR	**92.5**	**69.9**	**70.1**	**74.7**	**84.8**	**78.4**

Table 4: Comparison with LiDARSim. We reconstruct 25 sequences and generate 10k LiDAR frames to train 3D segmentation models.

produce poor results.

Combining Real and Simulated Data We combine real data with NeRF-LiDAR data generated from ground-truth scenes to see if simulated data can further boost performance for training. As shown in Table 1, simulated data along with 10% real data (real 1k) perform better (Sim10k + real 1k: 89.1) than 100% real data (real 10k: 88.4).

Conclusion

NeRF-LiDAR is proposed to generate realistic LIDAR point clouds via neural implicit representation. Images are utlized to achieve accurate 3D geometry and labels rendering and real data are also adopted as supervision. The effectiveness of NeRF-LiDAR is verified by training 3D segmentation networks. Models trained on the generated LiDAR data can achieve similar mIoU as models trained on real LiDAR data.

Acknowledgments

This work was supported in part by STI2030-Major Projects (Grant No. 2021ZD0200204), National Natural Science Foundation of China (Grant No. 62106050 and 62376060), Natural Science Foundation of Shanghai (Grant No. 22ZR1407500) and USyd-Fudan BISA Flagship Research Program.

References

Achlioptas, P.; Diamanti, O.; Mitliagkas, I.; and Guibas, L. 2018. Learning representations and generative models for 3d point clouds. In *ICML*.

Barron, J. T.; Mildenhall, B.; Tancik, M.; Hedman, P.; Martin-Brualla, R.; and Srinivasan, P. P. 2021. Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields. In *ICCV*.

Barron, J. T.; Mildenhall, B.; Verbin, D.; Srinivasan, P. P.; and Hedman, P. 2022. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In *CVPR*.

Barron, J. T.; Mildenhall, B.; Verbin, D.; Srinivasan, P. P.; and Hedman, P. 2023. Zip-NeRF: Anti-aliased grid-based neural radiance fields. In *ICCV*.

Caesar, H.; Bankiti, V.; Lang, A. H.; Vora, S.; Liong, V. E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; and Beijbom, O. 2020. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*.

Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; and Schiele, B. 2016. The cityscapes dataset for semantic urban scene understanding. In *CVPR*.

Deng, K.; Liu, A.; Zhu, J.-Y.; and Ramanan, D. 2022. Depth-supervised NeRF: Fewer Views and Faster Training for Free. In *CVPR*.

Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; and Koltun, V. 2017. CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*.

Fang, J.; Zhou, D.; Yan, F.; Zhao, T.; Zhang, F.; Ma, Y.; Wang, L.; and Yang, R. 2020. Augmented lidar simulator for autonomous driving. *IEEE Robotics and Automation Letters*.

Fang, J.; Zuo, X.; Zhou, D.; Jin, S.; Wang, S.; and Zhang, L. 2021. LiDAR-Aug: A General Rendering-based Augmentation Framework for 3D Object Detection. In *CVPR*.

Fu, X.; Zhang, S.; Chen, T.; Lu, Y.; Zhu, L.; Zhou, X.; Geiger, A.; and Liao, Y. 2022. Panoptic NeRF: 3D-to-2D Label Transfer for Panoptic Urban Scene Segmentation. In *International Conference on 3D Vision (3DV)*.

Geiger, A.; Lenz, P.; and Urtasun, R. 2012. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *CVPR*.

Gschwandtner, M.; Kwitt, R.; Uhl, A.; and Pree, W. 2011. BlenSor: Blender sensor simulation toolbox. In *International Symposium on Visual Computing*.

Jang, E.; Gu, S.; and Poole, B. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Koenig, N.; and Howard, A. 2004. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *IROS*.

Kundu, A.; Genova, K.; Yin, X.; Fathi, A.; Pantofaru, C.; Guibas, L. J.; Tagliasacchi, A.; Dellaert, F.; and Funkhouser, T. 2022. Panoptic Neural Fields: A Semantic Object-Aware Neural Scene Representation. In *CVPR*.

Lambert, J.; Liu, Z.; Sener, O.; Hays, J.; and Koltun, V. 2020. MSeg: A composite dataset for multi-domain semantic segmentation. In *CVPR*.

Luo, S.; and Hu, W. 2021. Diffusion probabilistic models for 3d point cloud generation. In *CVPR*.

Manivasagam, S.; Wang, S.; Wong, K.; Zeng, W.; Sazanovich, M.; Tan, S.; Yang, B.; Ma, W.-C.; and Urtasun, R. 2020. Lidarsim: Realistic lidar simulation by leveraging the real world. In *CVPR*.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.

Milioto, A.; Vizzo, I.; Behley, J.; and Stachniss, C. 2019. Rangenet++: Fast and accurate lidar semantic segmentation. In *IROS*.

Neuhold, G.; Ollmann, T.; Rota Bulo, S.; and Kontschieder, P. 2017. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*.

Rematas, K.; Liu, A.; Srinivasan, P. P.; Barron, J. T.; Tagliasacchi, A.; Funkhouser, T.; and Ferrari, V. 2022. Urban Radiance Fields. In *CVPR*.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*.

Tancik, M.; Casser, V.; Yan, X.; Pradhan, S.; Mildenhall, B.; Srinivasan, P. P.; Barron, J. T.; and Kretzschmar, H. 2022. Block-nerf: Scalable large scene neural view synthesis. In *CVPR*.

Varma, G.; Subramanian, A.; Namboodiri, A.; Chandraker, M.; and Jawahar, C. 2019. IDD: A dataset for exploring problems of autonomous navigation in unconstrained environments. In *WACV*.

Xie, E.; Wang, W.; Yu, Z.; Anandkumar, A.; Alvarez, J. M.; and Luo, P. 2021. SegFormer: Simple and efficient design for semantic segmentation with transformers. In *NeurIPS*.

Yang, G.; Huang, X.; Hao, Z.; Liu, M.-Y.; Belongie, S.; and Hariharan, B. 2019. Pointflow: 3d point cloud generation with continuous normalizing flows. In *ICCV*.

Yu, F.; Chen, H.; Wang, X.; Xian, W.; Chen, Y.; Liu, F.; Madhavan, V.; and Darrell, T. 2020. BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. In *CVPR*.

Yue, X.; Wu, B.; Seshia, S. A.; Keutzer, K.; and Sangiovanni-Vincentelli, A. L. 2018. A LiDAR Point Cloud Generator: From a Virtual World to Autonomous Driving. In *ICMR*.

Zhang, K.; Riegler, G.; Snavely, N.; and Koltun, V. 2020. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*.

Zhi, S.; Laidlow, T.; Leutenegger, S.; and Davison, A. J. 2021. In-place scene labelling and understanding with implicit scene representation. In *ICCV*.

Zhou, H.; Zhu, X.; Song, X.; Ma, Y.; Wang, Z.; Li, H.; and Lin, D. 2021. Cylinder3d: An effective 3d framework for driving-scene lidar semantic segmentation. In *CVPR*.

Zyrianov, V.; Zhu, X.; and Wang, S. 2022. Learning to Generate Realistic LiDAR Point Clouds. In *ECCV*.