

# NF-Atlas: Multi-Volume Neural Feature Fields for Large Scale LiDAR Mapping

Xuan Yu<sup>1</sup>, Yili Liu<sup>1</sup>, Sitong Mao<sup>2</sup>, Shunbo Zhou<sup>2</sup>, Rong Xiong<sup>1</sup>, Yiyi Liao<sup>1</sup>, Yue Wang<sup>1</sup>

**Abstract**—LiDAR Mapping has been a long-standing problem in robotics. Recent progress in neural implicit representation has brought new opportunities to robotic mapping. In this paper, we propose the multi-volume neural feature fields, called NF-Atlas, which bridge the neural feature volume with pose graph optimization. By regarding the neural feature volume as pose graph nodes and the relative pose between volumes as pose graph edges, the entire neural feature field becomes both *locally rigid* and *globally elastic*. Locally, the neural feature volume employs a sparse feature Octree and a small MLP to encode the signed distance function (SDF) of the submap with an option of semantics. Learning the map using this structure allows for end-to-end solving of maximum a posteriori (MAP) based probabilistic mapping. Globally, the map is built volume by volume independently, avoiding catastrophic forgetting when mapping incrementally. Furthermore, when a loop closure occurs, with the elastic pose graph based representation, only updating the origin of neural volumes is required without remapping. Finally, these functionalities of NF-Atlas are validated. Thanks to the sparsity and the optimization based formulation, NF-Atlas shows competitive performance in terms of accuracy, efficiency and memory usage on both simulation and real-world datasets. The project page is: <https://yuxuan1206.github.io/NFAtlas/>

## I. INTRODUCTION

Mapping is a fundamental task for robotics. In particular, dense mapping is essential to many robotic applications e.g. navigation. For large-scale outdoor scenes, it is challenging to achieve fast and accurate dense mapping while maintaining a low memory cost.

LiDAR has emerged as a popular sensor for mapping due to its ability to provide range measurements. Conventional LiDAR-based methods can achieve fast mapping with a relatively low memory cost, adopting a sparse, discretized volume representation, e.g., Octree [1] or hash table [2]. However, the reconstructed map of conventional methods may be noisy and contain many holes. This is largely due to the fact that the voxels are assumed to be independent of each other [3], [1] to enable efficient sequential fusion. However, the independence assumption yields a crude approximation of the real scene and leads to non-smooth results, thus hampering the fusion accuracy.

Recent years witnessed a great success of implicit neural representations [4], [5], [6] that encode scenes using coordinate-based multi-layer perception (MLP) [5], [7]. In particular, neural radiance fields (NeRF) propose to combine

Xuan Yu, Yili Liu, Rong Xiong and Yue Wang are with the State Key Laboratory of Industrial Control Technology and Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China. Yiyi Liao is with College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. Sitong Mao and Shunbo Zhou are with Huawei Cloud Computing Technologies Co., Ltd., Shenzhen, China.

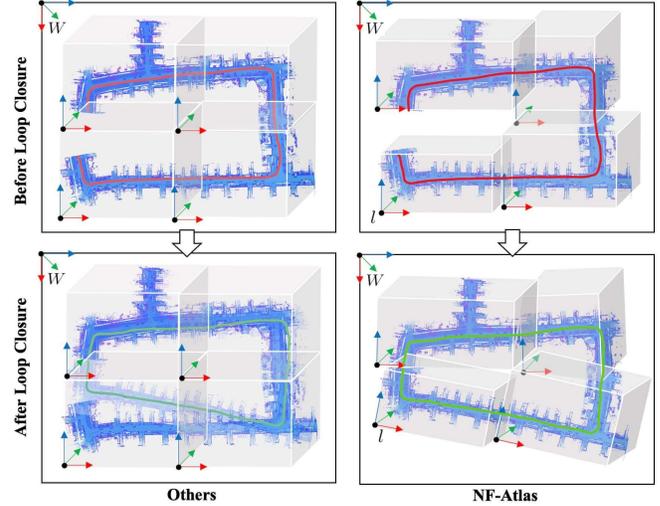


Fig. 1. The map is divided into multiple submaps. Each submap is a neural features volume, which can be fixed to world coordinates  $W$  (left column) and anchor pose coordinates  $l$  (right column). When a loop closure happens, the map in the world coordinates fixed submap calls for remapping due to the trajectory correction, while the anchor pose coordinates fixed submap calls for only a volume transform determined by the trajectory correction.

coordinate-based MLPs with differentiable volume rendering in an end-to-end manner. In this way, NeRF actually formulates mapping as a maximum likelihood optimization given the scene observations. This formulation allows for further adding regularizations, achieving the high-quality view synthesis and the low memory utilization [8], [9], [10], [11], [12]. Some works extend NeRF to integrate range measurements, demonstrating an impressive quality in small-scale environments e.g. rooms [10], [11], [12]. However, when applying to large-scale outdoor mapping, the large MLP and slow rendering make the training difficult [13], [14]. Efforts have been made to improve the training efficiency by representing the map as a 3D feature volume in combined with a small MLP, but such dense representation consumes very high memory when the environment is large [13], [14].

Further challenges arise when the map is required to be built incrementally in some applications. First, the implicit neural representations suffer from catastrophe forgetting. Moreover, if the trajectory is also updated incrementally, both conventional mapping and neural mapping methods call for total re-integration or re-optimization [15], [16]. These difficulties raise a question: *what is the representation suitable for large scale LiDAR mapping?*

In this paper, we propose to represent a large-scale map as multi-volume neural feature fields, named NF-Atlas, bridging the advantages of sparse volume representation and implicit



Fig. 2. The center image shows the semantic reconstruction of the global map using LiDAR and semantic labels on Seq.00 of the KITTI-360. Several example volumes are shown in the map, among which the red and blue ones are highlighted with the surface details in the side images.

neural representations. Our key idea is to represent a large map using multiple submaps connected by a pose graph, where each submap is a sparse, multi-scale neural feature volume encoding the signed distance function (SDF). Therefore, as shown in Fig. 1, the neural feature fields as a whole is *locally rigid* and *globally elastic*. In local, we obtain an Octree using the LiDAR observations and then model the neural feature volume as a multi-scale feature Octree, leading to a lightweight representation. Further, the multi-scale Octree representation allows for modeling details with better accuracy and efficient sampling. On the other hand, we formulate the mapping as maximum a posteriori problem, which can be optimized in an end-to-end manner. The semantic cues can also be easily incorporated in the optimization. In global, our formulation does not suffer from catastrophe forgetting when mapping incrementally, as each submap is modeled independently. Even in the case of loop closure, only the origins of the submaps need to be updated based on the pose graph optimization owing to the elastic inter-submap connection, while the local area’s poses and mapping can be considered invariant. In the experiments on both simulation and real world datasets, we show that NF-Atlas achieve better accuracy and efficiency with a relatively low memory consumption than the comparative learning-based and conventional mapping methods. A semantic reconstruction of the urban environment is shown in Fig. 2. In summary, the contributions involve:

- A sparse neural feature volume formulating LiDAR mapping as a maximum a posteriori problem, which is end-to-end optimized for better quality.
- An atlas organizing multiple neural feature volumes by a pose graph, which forms an incremental elastic feature fields that avoids catastrophe forgetting and remapping.
- Experiments on both simulation and real-world datasets validate the advantages of NF-Atlas. The CUDA implementation of regularizer back-propagation are released.

## II. RELATED WORKS

### A. Learning-free Reconstruction

Map representations calls community’s attention for long years. The SDF stores the closest distance from each point to the surface of the object, which is often used as an implicit representation of the object to represent the surface details. Newcombe et al. [3] popularized with KinectFusion use

the truncated signed distance function (TSDF) for mapping. However, KinectFusion is limited to RGB-D-based indoor reconstruction due to the dense volume. Whelan et al. [17] propose a fused volumetric method for globally consistent surface reconstructions to achieve spatially extended mapping. Subsequently, sparse voxels [18], [19], [20] are applied to the scalable mapping system. Besides, some approaches are also achieved by combining sparse volume octree with occupancy grid map [21], [16]. Suma++ [22] provide surfel-based mapping and accurate odometry, without following a pre-defined grid. In addition, several works [23], [24], [22] integrate semantic information to facilitate the mapping process. When a loop closure occurs, such dense representation calls for remapping along the corrected trajectory. To save the computation, [15] propose to organize multiple dense maps as a pose graph, which simplifies the remapping to coordinate transform.

### B. Learning-based Reconstruction

Neural fields become a competitive representation for reconstruction recently [5], [6], [4]. Sdfdiff [25] and DIST [26] address differentiable sphere tracing to learn SDFs of 3D objects from images. DVR [27] and IDR [28] determine the radiance directly on the surface of an object and provide a differentiable rendering formulation, but requiring foreground mask as supervision. Wang et al. [29] develop a new volume rendering method to train a bias-free neural SDF representation, which, however, takes a long time, failing to reconstruct the featureless surfaces. Along this direction, Sun et al. [9] focus on more efficient sampling strategy that enables accurate surface reconstruction. In order to accelerate the training speed, some methods employ hierarchical representations, say octree [30] and multi-resolution grids [31], [32], [8]. However, these methods are not evaluated on large scale environments due to the memory and efficiency.

### C. Learning-based Incremental Reconstruction

Current research efforts consider incremental mapping of implicit representations as the continual learning about the neural fields. With respect to the implicit reconstruction of indoor scenes using RGB-D sensors, iMAP [33] presents the first implicit SLAM based on neural radiation fields. NICE-SLAM [7] improves upon iMAP by incorporating a pre-trained geometric prior. iSDF [12] employs a neural

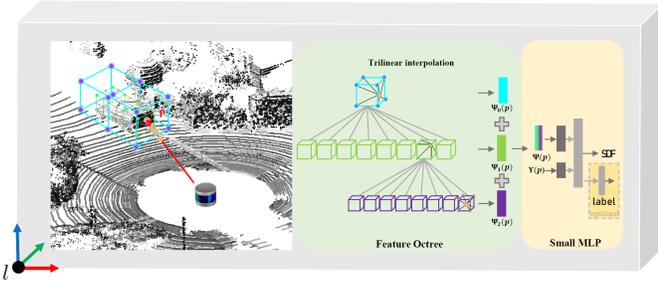


Fig. 3. The architecture of the neural feature volume with an optional semantic branch which is described in detail in Sec.III.

network to regress the input 3D coordinates to the signed distance. Azinović et al. [10] augments the NeRF framework with TSDF to represent the surface instead of the volume. In the context of incremental implicit reconstruction using urban sparse LiDAR point clouds, Shi et al. [14] proposes an SDF-based semantic mapping approach that utilizes a three-layer sampling strategy and panoptic representation to mitigate catastrophic forgetting during incremental reconstruction. These works all replay keyframes from historical buffer and train the network with current observations together. However, the past mapping still degenerates when the learning time is bounded. In addition, these methods do not consider the loop closure explicitly, which may bring significant trajectory correction, causing the remapping.

### III. OPTIMIZATION BASED MAPPING

The NF-Atlas contains several submaps, each of which models the environment rigidly as an implicit function that maps a coordinate to a signed distance value using a neural feature volume. We state the neural feature volume based mapping as an optimization problem, which jointly optimizes volumes parameters and optionally the poses.

#### A. Neural Feature Volume

The architecture of neural feature volume is shown in Fig. 3, which consists of a feature Octree and a small MLP, yielding the SDF of a query 3D point.

**Feature Octree:** The feature Octree maps a query 3D point  $\mathbf{p} \in \mathcal{R}^3$  to a feature vector  $\Psi(\mathbf{p}) \in \mathcal{R}^N$ . First, we employ an Octree based structure  $\Psi$  to sparsely encode the local area into features. Specifically, as shown in [30], each Octree node stores a learnable feature  $\psi_{i,j} \in \mathcal{R}^N$ , where  $i$  indicates the tree level, while  $j$  indicates the index. Given a query point  $\mathbf{p}$ , its feature is acquired by summing the tri-linearly interpolated features in top  $K$  levels as:

$$\Psi(\mathbf{p}) = \sum_{i=0}^{K-1} \text{triInterp}(\psi_{i,j \in \mathcal{N}(\mathbf{p})}) \quad (1)$$

where  $\text{triInterp}$  denotes the tri-linear interpolation,  $\mathcal{N}(\mathbf{p})$  is the set of the eight nearest Octree nodes i.e. corners of a cube containing  $\mathbf{p}$ . The Octree-based sparse volume achieves a balance between reconstruction quality and efficiency through representing 3D shapes in a compressed format which stores multi-level features.

**Small MLP:** We adopt a small MLP to map the feature vector  $\Psi(\mathbf{p})$  and the input coordinate  $\mathbf{p}$  to an SDF value:

$$s = f_{\theta}(\Psi(\mathbf{p}), \gamma(\mathbf{p})) \quad (2)$$

where  $s \in \mathcal{R}$  is the SDF value of  $\mathbf{p}$ ,  $\gamma(\cdot)$  denotes position encoding that is applied to each element of  $\mathbf{p}$ :

$$\gamma(\mathbf{p}) = [\cos(2^0 \pi \mathbf{p}), \sin(2^0 \pi \mathbf{p}), \dots, \cos(2^{L-1} \pi \mathbf{p}), \sin(2^{L-1} \pi \mathbf{p})] \quad (3)$$

Different from the MLP in original NeRF [5],  $f_{\theta}$  is much smaller owing to the octree feature, thus highly efficient while still keeping the quality of the reconstruction.

#### B. Differentiable Range Approximation

Given SDF of a query point, we can find the surface in the field easily. However, at the early stage of the training, the SDF is far from convergence. Therefore, we introduce the differentiable range approximation from the neural feature volume following the differentiable volume rendering in NeuS [29]. Given the origin and direction of a ray, denoted by  $(\mathbf{o}, \mathbf{d})$ , we can represent a spatial point along the ray as  $\mathbf{p} = \mathbf{o} + \rho \mathbf{d}$ . By sampling  $\{\rho_n\}$ , we have a batch of points on the ray  $\{\mathbf{p}_n\}$  arranged in near-to-far order. We can query their corresponding SDFs  $\{s_n\}$  to approximate the range  $r$  along the ray as:

$$r = \sum_{n=1}^N T_n \alpha_n \rho_n \quad (4)$$

where  $T_n = \prod_{m=1}^{n-1} (1 - \alpha_m)$ ,  $\alpha_n$  is the discrete opacity value defined under the S-density function assumption [29] as:

$$\alpha_n = \max\left(\frac{\Phi(s_n) - \Phi(s_{n+1})}{\Phi(s_n)}, 0\right) \quad (5)$$

where  $\Phi(x)$  is Sigmoid function  $\Phi(x) = (1 + e^{-\xi x})^{-1}$  with a temperature coefficient  $\xi$ .

**Sampling:** Thanks to the sparse Octree and the point cloud, we can accurately sample the points by voxel-guided sampling and surface-guided sampling from the near-surface region [9]. By skipping the empty space along the ray, we can omit sampling in the low informative region, improving the sampling efficiency significantly.

#### C. Likelihood Measurement Model

**Range Measurements:** By stitching the neural feature volume and the differentiable range approximation, we follow the classical probabilistic mapping theory [34] to design a range measurement model  $p(r|\mathbf{o}, \mathbf{d}, \theta, \Psi)$

$$p_r(r|\mathbf{o}, \mathbf{d}, \theta, \Psi) = \frac{1}{\sigma_r \sqrt{2\pi}} \exp\left\{-\frac{(\hat{r} - r(\mathbf{o}, \mathbf{d}))^2}{2\sigma_r^2}\right\} \quad (6)$$

where  $\sigma_r^2$  is the variance of the measurement noise,  $r(\mathbf{o}, \mathbf{d})$  is the approximated range along ray  $(\mathbf{o}, \mathbf{d})$  from the neural feature volume,  $\hat{r}$  is the measured range along the ray. In this way, the measurement model reflects the likelihood of the measured range in the real world. The vital advantage of this model is the differentiable pathway to map parameters i.e. neural feature volume, even the pose i.e. ray parameters.

**SDF Measurements:** By formulating as a maximum likelihood problem, we can further add more measurements. In NF-Atlas, We also employ SDF measurement model following [12], [10], [11]. Given a LiDAR point measurement with a range of  $\hat{r}$ , we consider a sample  $\mathbf{p} = \mathbf{o} + \rho \mathbf{d}$  as near-surface if  $|\hat{r} - \rho| \leq \tau$ . We derive the measured SDF of this point as  $b = \hat{r} - \rho$ . Then we define an truncated Laplace distribution for SDF measurement model:

$$p_{near}(s|\mathbf{p}, \theta, \Psi) = \frac{1}{2\lambda} \exp \left\{ -\frac{|b - s(\mathbf{p})|}{\lambda} \right\} \quad (7)$$

where  $\lambda$  is the bandwidth coefficient,  $s(\mathbf{p})$  is the approximated SDF of sample  $\mathbf{p}$  as (2). For the far-surface sample  $\mathbf{p}$  satisfying  $|\hat{r} - \rho| > \tau$ , we define an exponential density [29] as the measurement model:

$$p_{far}(s|\mathbf{p}, \theta, \Psi) = \eta \exp \left\{ -\max(0, e^{-\beta s(\mathbf{p})} - 1, s(\mathbf{p}) - b) \right\} \quad (8)$$

where  $\eta$  is a normalizer for the validness of the density. As a whole, the likelihood of the measured SDF is:

$$p_s(s|\mathbf{p}, \theta, \Psi) = \begin{cases} p_{near}(s|\mathbf{p}, \theta, \Psi) & |b| \leq \tau \\ p_{far}(s|\mathbf{p}, \theta, \Psi) & o.w. \end{cases} \quad (9)$$

**Semantic Measurements:** We further add semantic measurements into our mapping system as [35]. We augment (2) with semantic prediction as:

$$s, p(c) = f_\theta(\Psi(\mathbf{p}), \gamma(\mathbf{p})) \quad (10)$$

where  $p(c)$  is a multinomial distribution of class  $c$  implemented by a softmax branch in parallel with the SDF prediction. We approximate the semantics of the ray as:

$$l(c) = \sum_{n=1}^N T_n \alpha_n p(c_n) \quad (11)$$

Then the likelihood of the measured semantic class  $\hat{c}$  is:

$$p_c(c = \hat{c}|\mathbf{o}, \mathbf{d}, \theta, \Psi) = \text{softmax}(l(c))|_{c=\hat{c}} \quad (12)$$

which is the predicted probability of the measured class.

#### D. Probabilistic Mapping

With measurement models above, we arrive at the data likelihood as a product of (6), (9) and (12), upon which we further add prior as regularization to derive the posterior. As a result, we formulate probabilistic LiDAR mapping as a maximum a posteriori (MAP) problem:

$$\tilde{\theta}, \tilde{\Psi}, \tilde{\mathbf{o}}, \tilde{\mathbf{d}} = \arg \max \prod \underbrace{p_r \cdot p_s \cdot p_c}_{\text{Likelihood}} \cdot \underbrace{p_e \cdot p_h}_{\text{Prior}} \quad (13)$$

where  $p_e$  is a prior on the identity gradient of the SDF,  $p_h$  is a prior on the smoothness of the neighborhood in the real world environment, the product  $\prod$  means multiplication across all samples along all rays. The MAP formulation reserves the differentiable pathway from all terms to map parameters i.e. neural feature volume, and leaves the pose i.e. ray parameters from only likelihood terms, as we apply no prior on the mapping trajectory.

**Prior Terms:** Specifically, we have  $p_e$  as

$$p_e(\theta, \Psi) = \frac{1}{\sigma_e \sqrt{2\pi}} \exp \left\{ -\frac{(1 - \|\nabla_{\mathbf{p}} s(\mathbf{p})\|)^2}{2\sigma_e^2} \right\} \quad (14)$$

where  $\sigma_e^2$  is the variance. For  $p_h$ , we have the smoothness prior as [8], [11]:

$$p_h(\theta, \Psi) = \frac{1}{\sigma_h \sqrt{2\pi}} \exp \left\{ -\frac{\|\nabla_{\mathbf{p}} s(\mathbf{p}) - \nabla_{\mathbf{p}} s(\mathbf{p} + \Delta \mathbf{p})\|^2}{2\sigma_h^2} \right\} \quad (15)$$

where  $\sigma_h^2$  is the variance,  $\Delta \mathbf{p}$  is a small perturbation vector of  $\mathbf{p}$  to evaluate the normal direction consistency.

**Implementation of Back-propagation:** The details of the derivation are shown in Appendix. Note that the back-propagation of the two prior regularizers requires the evaluation of 2nd-order derivatives. For fast computation of the double back-propagation of tri-linear interpolation in Octree, we implement the operation by CUDA based on [11].

## IV. LARGE SCALE MAPPING AS ATLAS

For the large-scale scene, we employ a pose graph to organize the multiple neural feature volumes as nodes that capture the local area. Each edge between the nodes indicates the measured relative pose between the two connected neural feature volumes. In this way, the origins of the volumes can be updated, making the whole map an elastic neural feature field. Like an atlas, we can check the map volume by volume, or combine them as a whole.

### A. Incremental Mapping

When mapping the environment, we build a neural feature volume using a sequence of poses and their measurements following (13). The origin of the  $l$ th neural feature volume is fixed to the starting pose  $\mathbf{T}_l$  of the  $l$ th sequence.

We pick frames in each volume based on the move distance threshold and set the overlap between two neighboring volumes to keep a smooth transition. As the mapping progresses, we freeze the past volumes and incrementally initialize a new volume using the poses spanning a similar distance, as well as their measurements. In this way, the regions covered by the neural feature volumes can be similar, reserving the computation complexity bounded. In addition, as each volume is optimized separately, we avoid the catastrophic forgetting of the neural network.

**Loop Closure without Remapping:** As each neural volume is fixed to the starting pose, when a loop closure occurs, updating the origins of these volumes to match the revised robot trajectory  $\mathbf{T}_l$  is sufficient, and there is no need to adjust their volume parameters. This is the main advantage compared with existing large-scale neural mapping methods that fix the volumes to global area partitions, which calls for remapping when the trajectory is updated.

**On-demand Global Mapping:** If a global consistent map is required, we can extract the map SDF from multiple neural feature volumes according to their most recent origins solved by pose graph optimization. Since the map is frozen after it is built, we only need to query the map. As shown in Fig. 1,

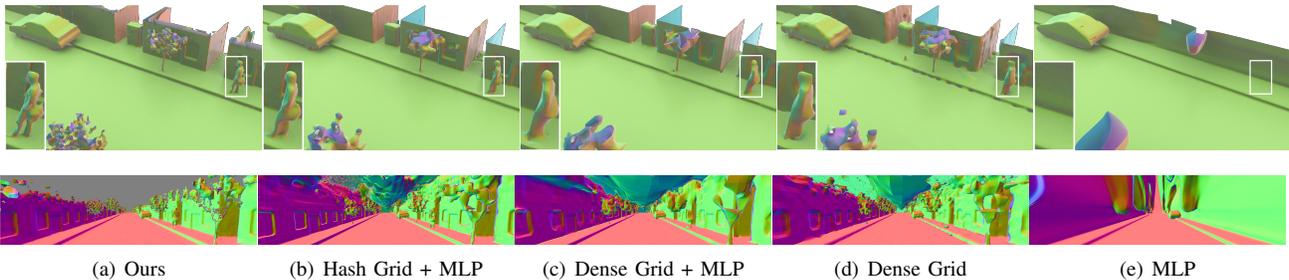


Fig. 4. Case comparison of different model architecture on the Maicity. The first row shows the reconstruction, whose details on pedestrians are highlighted in the white box in the lower left corner. The second row shows the rendered normal images for the specified viewpoint.

given a point  $\mathbf{p}^W$  in world coordinate  $W$ , assume its closest volume is the  $l$ th volume, then its SDF is:

$$s = f_{\theta_l}(\Psi_l(\mathbf{T}_l^{-1}\mathbf{p}^W), \gamma(\mathbf{T}_l^{-1}\mathbf{p}^W)) \quad (16)$$

where  $\theta_l$  and  $\Psi_l$  are the parameter of the  $l$ th volume.

### B. Loop Closure Updating

The loop detection is divided into coarse search and fine registration. We pick the nearest one of the volumes whose timestep is more than a certain step earlier than the current volume for fine registration. Then, we calculate the relative pose between the detected loop closure pair by aligning the two neural feature volumes. Say the two volumes are indexed by  $l$  and  $j$ , they have an overlap since there exists a loop closure. Then we estimate the relative pose by maximizing the likelihood of measurements in volume  $j$  conditioned on the volume  $l$  as:

$$\tilde{\mathbf{o}}_j, \tilde{\mathbf{d}}_j = \arg \max p_r(r_j | \mathbf{o}_j, \mathbf{d}_j, \theta_l, \Psi_l) \quad (17)$$

Note that we do not update the map at the loop closure stage. Since  $\tilde{\mathbf{o}}_j$  and  $\tilde{\mathbf{d}}_j$  are parameterized by pose  $\mathbf{T}_j$ , we arrive at the optimal relative pose, which is added to the pose graph as a new edge, activating the pose graph optimization. Compared with conventional LiDAR based 3D-3D point cloud registration, such inter-volume measurement based alignment utilizes the relationship between range measurements and the 3D SDF to guide the pose estimation.

## V. EXPERIMENTAL RESULTS

We validate the effectiveness of our approach on simulation scenes and real scenes. The impact of additional terms is also verified. We conduct comparison and performance analysis. In addition, we explore the effectiveness of using our atlas to enable global representations.

### A. Setup

**Dataset:** Our study involves experiments on two outdoor urban datasets: Maicity [36] and KITTI-360 [37], and two indoor datasets: Lobby dataset collected by ourselves and Hilti SLAM 2021 [38]. Outdoor datasets were captured using the Velodyne HDL-64 LiDAR. While Maicity is a smaller synthetic dataset, it possesses ground truth mesh and high-fidelity LiDAR measurements, which make it suitable for evaluation tests. In contrast, KITTI-360 is larger in size and has noisy measurements, making it appropriate for both local

TABLE I  
MAPPING QUALITY USING DIFFERENT MODEL ARCHITECTURES

Data	Model	Comp.↓	Acc.↓	C-L1↓	Re.↑	Pre.↑	F1↑
Maicity	MLP	9.26	2.59	5.93	79.90	91.61	85.36
	DG	1.15	1.49	1.32	97.19	97.07	97.13
	DFG+MLP	1.11	1.39	1.25	97.43	97.65	97.54
	HFG+MLP	1.07	1.40	1.23	97.88	97.87	97.88
	<b>Ours</b>	<b>0.95</b>	<b>1.33</b>	<b>1.14</b>	<b>98.87</b>	<b>97.71</b>	<b>98.28</b>
KITTI-360	MLP	13.83	5.40	9.61	60.87	86.34	71.40
	DG	5.85	4.45	5.15	85.33	89.51	87.37
	DFG+MLP	7.72	5.36	6.54	78.23	86.58	82.20
	HFG+MLP	7.39	5.81	6.60	80.12	84.69	82.34
	<b>Ours</b>	<b>4.06</b>	<b>3.39</b>	<b>3.82</b>	<b>93.86</b>	<b>95.23</b>	<b>94.54</b>

TABLE II  
MEMORY AND COMPUTATION TIME USING DIFFERENT MODEL ARCHITECTURE

Data	Model	Memory ↓	Train Time ↓	Render Time ↓
Maicity	MLP	<b>1.32M</b>	207.9min	13.13s
	DG	2781.0M	29.1min	<b>0.28s</b>
	DFG+MLP	1755.4M	51.8min	1.01s
	HFG+MLP	160.5M	45.6min	0.86s
	<b>Ours</b>	64.3M	<b>11.3min</b>	0.30s
KITTI-360	MLP	<b>1.32M</b>	378.6min	15.49s
	DG	2781M	57.8min	0.28s
	DFG+MLP	1755.38M	72.8min	1.14s
	HFG+MLP	164.51M	55.3min	0.92s
	<b>Ours</b>	26.91M	<b>39.0min</b>	<b>0.26s</b>

and global evaluation tests. Besides, KITTI-360 also provides semantic cues and filters out dynamic objects, which enables us to evaluate semantic reconstruction. The setup and results for indoor datasets are shown in Appendix.

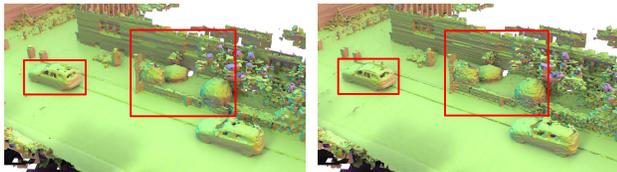
**Metrics:** We conduct a quantitative evaluation of the reconstruction quality of generated maps by comparing the densely sampled point cloud from observed surfaces to the reference point cloud. The reference point cloud for Maicity is obtained from a CAD model, while for the KITTI-360 dataset, we follow the evaluation criterion of other work [20]. We employ completeness, accuracy, Chamfer-L1 distance, recall, precision and F-score. The recall, precision and F-score are computed using a threshold of 5cm for Maicity and 10cm for KITTI-360, respectively. Additionally, we measure the training time and model memory for reconstruction, as well as the time taken for inference.

**Implementation:** We employ a sequence of 80 consecutive poses and their measurements for training a volume. We

TABLE III

MAPPING QUALITY USING MODEL WITH AND WITHOUT THE SMOOTHNESS PRIOR AT DIFFERENT NOISE LEVEL ON MAICITY.

$\sigma_{noise}$	Model	Comp.↓	Acc.↓	C-L1↓	Re.↑	Pre.↑	F1↑
10	w/o $p_h$	1.41	1.88	1.64	<b>98.87</b>	96.01	97.42
	Ours	<b>1.29</b>	<b>1.70</b>	<b>1.50</b>	98.86	<b>96.52</b>	<b>97.67</b>
5	w/o $p_h$	1.01	1.49	1.25	<b>99.38</b>	<b>97.10</b>	<b>98.23</b>
	Ours	<b>1.01</b>	<b>1.46</b>	<b>1.23</b>	99.25	97.09	98.16



(a) Ours

(b) Ours w/o smoothness prior

Fig. 5. Case comparison of our model with and without smoothness prior on KITTI-360. The red boxes highlight the difference in the surface details.

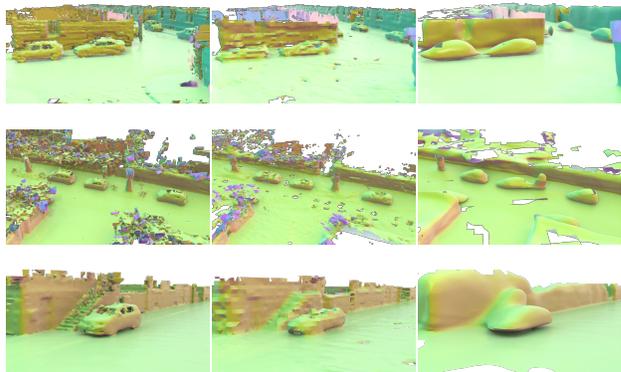
fix poses of the overlapping part and optimize new poses jointly with the volume features and the small MLP. We use top 3 levels features in the Octree where the top level resolution depends on the size of the scene. The SDF branch shares a layer of 128 hidden units with the semantic branch, and the semantic branch is followed by another layer of 64 hidden units. We sample  $1024 \times 80$  rays per batch. On each ray, we have 12 samples using voxel-guided sampling and 12 samples using surface-guided sampling within the range of 25m. Thanks to the CUDA implementation, it takes on average 700ms for each mapping iteration which is  $2 \times$  faster than double back-propagation without CUDA. All experiments run on a single RTX-3090 GPU.

**Baselines:** We adopt the Voxblox [18] and 3D SIREN [14] methods as conventional and learning-based baseline methods. These two methods are shown to have ability to map large-scale urban scenes with LiDAR scans. The source code of Voxblox is accessible. We use the original parameters on KITTI. Regarding 3D SIREN, we have implemented the method in accordance with the original paper [14].

### B. Ablation Study

**Effect of Feature Octree:** We conduct a quantitative evaluation of 5 map representations on Maicity-01 and a 150m segment of Seq.00 of KITTI-360. The map representations that we evaluate include a large MLP [14], Dense SDF grids (DG) [8], a Dense Multi-Level Feature Grids with a small MLP (DFG+MLP) [11], [7], and a Hash Indexed Multi-Level Feature Grids with a small MLP (HFG+MLP) [31]. Notably, DG and DFG+MLP options are set at a resolution of  $0.35m$  due to their high memory consumption. HFG+MLP has the same finest resolution of  $0.088m$  as ours. Since the mapping quality of DFG+MLP and HFG+MLP can be improved by more levels, we set them to 4 and 5 levels respectively, both have more levels than ours.

Upon evaluation of both datasets, our method demonstrates the highest quality as shown in Tab. I. This can be attributed to its superior resolution compared to DG and



(a) Ours

(b) Voxblox

(c) 3D-SIREN

Fig. 6. Case comparison of different methods on the KITTI-360. Each row shows the results of a case area using different methods.

TABLE IV

MAPPING QUALITY USING DIFFERENT METHODS ON KITTI-360

Seq.	Method	Comp.↓	Acc.↓	C-L1↓	Re.↑	Pre.↑	F1↑
00	3D-SIREN	7.61	8.99	8.30	74.69	78.64	76.62
	Voxblox	11.65	4.59	8.12	68.66	90.42	78.05
	Ours	<b>5.07</b>	<b>3.79</b>	<b>4.43</b>	<b>89.60</b>	<b>94.85</b>	<b>92.15</b>
02	3D-SIREN	12.37	8.34	10.35	74.56	69.37	71.87
	Voxblox	8.70	5.21	6.95	76.81	87.28	81.71
	Ours	<b>4.76</b>	<b>3.99</b>	<b>4.37</b>	<b>90.81</b>	<b>94.07</b>	<b>92.41</b>
04	3D-SIREN	7.47	8.71	8.09	74.99	82.78	78.69
	Voxblox	8.83	4.80	6.81	77.71	90.55	83.64
	Ours	<b>4.50</b>	<b>3.57</b>	<b>4.04</b>	<b>92.47</b>	<b>95.33</b>	<b>93.88</b>

DFG+MLP, as well as its less feature sharing in comparison to HFG+MLP. As shown in Tab. II, our method only requires a higher memory than MLP, while the latter exhibits weaker quality. In terms of training efficiency, our method outperforms all others. This is largely due to the adoption of a sparse Octree structure, which effectively reduces the number of features and improves sampling efficiency. For inference efficiency, ours remains competitive with the network-free DG, which is also brought by the Octree guided sampling strategy. In several cases shown in Fig. 4, ours captures more intricate surface details, such as human legs.

**Effect of Smoothness Prior:** The effectiveness of the prior is evaluated on Maicity, as it provides ground truth data. With noiseless data, adding prior knowledge is counter-productive. In order to accurately reflect the influence of the smoothness prior, two levels of noise are manually added to the scans. Tab. III reports that as noise levels increase, the influence of smoothness becomes increasingly significant. This finding is reasonable given the reduction in measurement confidence under high noise conditions. Moreover, Fig. 5 illustrates that the smoothness prior effectively mitigates the noise in the reconstructed surface.

**Effect of Semantic Measurement:** We evaluate the effectiveness of semantic measurements on KITTI-360. The results are slightly surprising that the semantics can marginally improve the mapping quality in addition to purely assigning labels. The C-L1 and F-score are improved from  $5.70cm$  to  $5.58cm$ , and  $87.66\%$  to  $88.80\%$ . This result may be

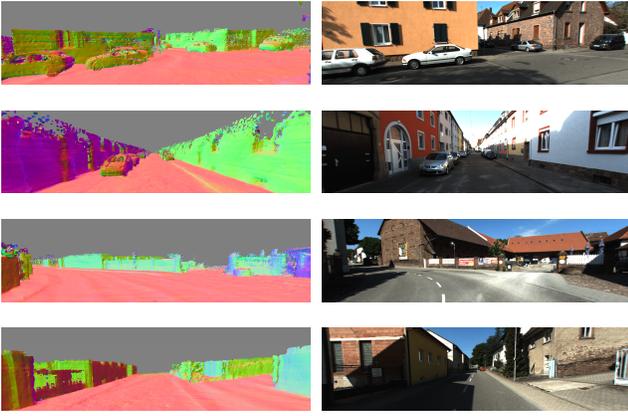


Fig. 7. Cases of rendered normal images (left column) and corresponding real images (right column) using our method on KITTI-360.

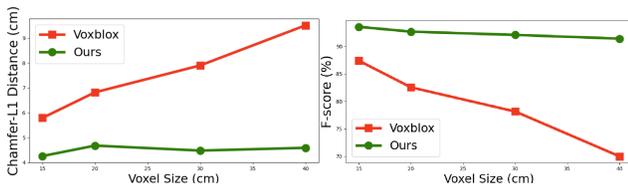


Fig. 8. Mapping quality with respect to voxel size on KITTI-360.

attributed to the advantages of multi-task learning. Two cases of the semantic reconstruction are shown in Fig. 2.

### C. Comparative Study

Our approach is compared against two existing methods, 3D-SIREN [14] and Voxblox [18], on three sequences from the KITTI-360 dataset for testing: scans 6330~6530 of Seq.00, scans 4485~4685 of Seq.02, and scans 5940~6140 of Seq.04. The results presented in Tab. IV demonstrate that our method produces more complete and accurate results than others. Specifically, the volume feature in our approach enhances fitting capacity compared to 3D-SIREN, while the end-to-end MAP optimization outperforms the updating rule of Voxblox. Fig. 6 displays examples where our method preserves details such as car windows and stairs. Fig. 7 illustrates examples of rendered normal images, showing consistent surface structures with the ones in real images.

We further assess the reconstruction quality with different voxel sizes on Seq.02 of KITTI-360. As depicted in Fig. 8, our approach outperforms Voxblox across all voxel size configurations. Furthermore, our method is less prone to degradation of mapping quality as voxel size increases. This phenomenon can be attributed to the employment of neural features, which possess the potential for super-resolution.

### D. Case Study on Incremental Mapping

In incremental mapping, the accumulation of new measurements inevitably limits the proportion of past measurements that can be retained due to constraints in memory and computational resources. This phenomenon leads to the catastrophic forgetting of previously acquired mapping information within the network. However, NF-Atlas allows

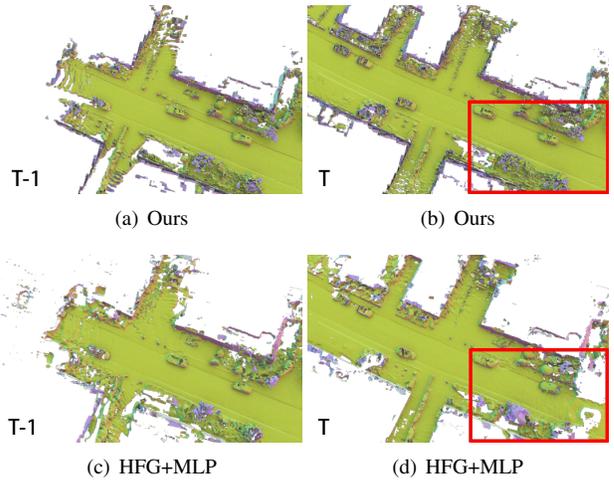


Fig. 9. Case comparison of different methods on the KITTI-360. Each row shows the results of incremental mapping from previous time to current time. The red boxes highlight the change in the previous map caused by the catastrophe forgetting.

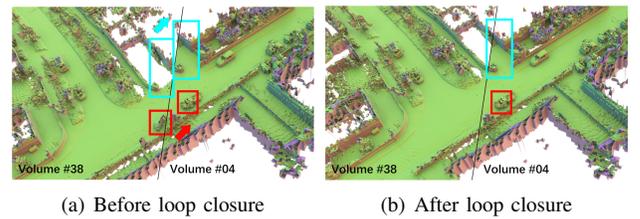


Fig. 10. Case of mapping result when a loop closure occurs using our method on KITTI-360. The red box and the blue box highlight that the objects are aligned after the loop closure. The black line is the boundary between the two volumes stitched in the global map.

each volume to focus solely on local area, thus effectively avoiding catastrophic forgetting. Fig. 9 supports our claim, while single model methods such as HFG+MLP suffer from a noticeable decline in surface detail retention of past maps.

By integrating semantic measurements, the outcome of an incremental semantic mapping over a more than 1km trajectory in KITTI-360 is depicted in Fig. 2, which verifies that our approach is capable of large scale LiDAR mapping.

### E. Case Study on Loop Closure

Upon the occurrence of loop closure, the trajectory updates through pose graph optimization. By representing the map as elastic neural feature fields, our method avoids the remapping along the updated trajectory. In Fig. 10, the results before and after the loop closure are demonstrated. By only updating the origins of submaps, the global map remains consistent, yielding a substantial reduction in computation.

## VI. CONCLUSION

We propose multi-volume neural feature fields, NF-Atlas, for large-scale LiDAR mapping which combines the advantage of the neural implicit representation and the pose graph. In local, we state the mapping as a MAP problem, allowing for end-to-end optimization. The sparsity of feature Octree also improves efficiency and memory usage. In global, we fix the volumes to the pose graph nodes, allowing for only

origin updating when loop closure occurs. The volume-by-volume process enables incremental mapping. On both simulation and real-world datasets, NF-Atlas is shown to be a competitive method. In the future, we set to deal with the dynamic objects in the environment.

#### REFERENCES

- [1] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, pp. 189–206, 2013.
- [2] T. Kühner and J. Kümmerle, "Large-scale volumetric scene reconstruction using lidar," in *2020 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2020, pp. 6261–6267.
- [3] S. Izadi, D. Kim, O. Hilliges, D. Molyneux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison *et al.*, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, 2011, pp. 559–568.
- [4] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 165–174.
- [5] 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," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.
- [6] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4460–4470.
- [7] 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. 12786–12796.
- [8] 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.
- [9] J. Sun, X. Chen, C. Wang, Z. Li, H. Averbuch-Elor, X. Zhou, and N. Snavely, "Neural 3d reconstruction in the wild," in *ACM SIGGRAPH 2022 Conference Proceedings*, 2022, pp. 1–9.
- [10] D. Azinović, R. Martin-Brualla, D. B. Goldman, M. Nießner, and J. Thies, "Neural rgb-d surface reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6290–6301.
- [11] J. Wang, T. Bleja, and L. Agapito, "Go-surf: Neural feature grid optimization for fast, high-fidelity rgb-d surface reconstruction," *arXiv preprint arXiv:2206.14735*, 2022.
- [12] 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.
- [13] 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. 12932–12942.
- [14] Y. Shi, R. Yang, P. Li, Z. Wu, H. Zhao, and G. Zhou, "City-scale incremental neural mapping with three-layer sampling and panoptic representation," *arXiv preprint arXiv:2209.14072*, 2022.
- [15] Y. Pan, X. Xu, X. Ding, S. Huang, Y. Wang, and R. Xiong, "Gem: online globally consistent dense elevation mapping for unstructured terrain," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2020.
- [16] Y. Wang, N. Funk, M. Ramezani, S. Papatheodorou, M. Popović, M. Camurri, S. Leutenegger, and M. Fallon, "Elastic and efficient lidar reconstruction for large-scale exploration tasks," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5035–5041.
- [17] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. J. Leonard, and J. McDonald, "Real-time large-scale dense rgb-d slam with volumetric fusion," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 598–626, 2015.
- [18] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1366–1373.
- [19] Y. Pan, Y. Kompis, L. Bartolomei, R. Mascaro, C. Stachniss, and M. Chli, "Voxfield: Non-projective signed distance fields for online planning and 3d reconstruction," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 5331–5338.
- [20] I. Vizzo, T. Guadagnino, J. Behley, and C. Stachniss, "Vdbfusion: Flexible and efficient tsdf integration of range sensor data," *Sensors*, vol. 22, no. 3, p. 1296, 2022.
- [21] E. Vespa, N. Nikolov, M. Grimm, L. Nardi, P. H. Kelly, and S. Leutenegger, "Efficient octree-based volumetric slam supporting signed-distance and occupancy mapping," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1144–1151, 2018.
- [22] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "Suma++: Efficient lidar-based semantic slam," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4530–4537.
- [23] V. Vineet, O. Miksik, M. Lidegaard, M. Nießner, S. Golodetz, V. A. Prisacariu, O. Kähler, D. W. Murray, S. Izadi, P. Pérez *et al.*, "Incremental dense semantic stereo fusion for large-scale semantic reconstruction," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 75–82.
- [24] L. Schmid, J. Delmerico, J. L. Schönberger, J. Nieto, M. Pollefeys, R. Siegwart, and C. Cadena, "Panoptic multi-tsdfs: a flexible representation for online multi-resolution volumetric mapping and long-term dynamic scene consistency," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 8018–8024.
- [25] Y. Jiang, D. Ji, Z. Han, and M. Zwicker, "Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1251–1261.
- [26] S. Liu, Y. Zhang, S. Peng, B. Shi, M. Pollefeys, and Z. Cui, "Dist: Rendering deep implicit signed distance function with differentiable sphere tracing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2019–2028.
- [27] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [28] L. Yariv, Y. Kasten, D. Moran, M. Galun, M. Atzmon, B. Ronen, and Y. Lipman, "Multiview neural surface reconstruction by disentangling geometry and appearance," *Advances in Neural Information Processing Systems*, vol. 33, pp. 2492–2502, 2020.
- [29] 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.
- [30] 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," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11358–11367.
- [31] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics (ToG)*, vol. 41, no. 4, pp. 1–15, 2022.
- [32] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5459–5469.
- [33] 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.
- [34] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.
- [35] S. Zhi, T. Laidlow, S. Leutenegger, and A. J. Davison, "In-place scene labelling and understanding with implicit scene representation," 2021.
- [36] I. Vizzo, X. Chen, N. Chebroul, J. Behley, and C. Stachniss, "Poisson Surface Reconstruction for LiDAR Odometry and Mapping," 2021. [Online]. Available: <http://www.ipb.uni-bonn.de/pdfs/vizzo2021icra.pdf>
- [37] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," *Pattern Analysis and Machine Intelligence (PAMI)*, 2022.
- [38] M. Helmberger, K. Morin, N. Kumar, D. Wang, Y. Yue, G. Cioffi, and D. Scaramuzza, "The hilti slam challenge dataset," 2021.

## A. Optimization of MAP

By taking the logarithm, the MAP problem:

$$\tilde{\theta}, \tilde{\Psi}, \tilde{\mathbf{o}}, \tilde{\mathbf{d}} = \arg \max \prod p_r \cdot p_s \cdot p_c \cdot p_e \cdot p_h \quad (1)$$

can be converted into a loss function:

$$\begin{aligned} \tilde{\theta}, \tilde{\Psi}, \tilde{\mathbf{o}}, \tilde{\mathbf{d}} &= \arg \max \log(\prod p_r \cdot p_s \cdot p_c \cdot p_e \cdot p_h) \\ &= \arg \max - \sum (\lambda_r L_r + \lambda_s L_s + \lambda_c L_c + \lambda_e L_e + \lambda_h L_h) \\ &= \arg \min \sum (\lambda_r L_r + \lambda_s L_s + \lambda_c L_c + \lambda_e L_e + \lambda_h L_h) \end{aligned} \quad (2)$$

where  $L_r$  is range loss:

$$L_r = \frac{(\hat{r} - r(\mathbf{o}, \mathbf{d}))^2}{2\sigma_r^2} \quad (3)$$

$L_s$  is SDF loss:

$$L_s = \begin{cases} \frac{|b-s(\mathbf{p})|}{\lambda} & |b| \leq \tau \\ \max(0, e^{-\beta s(\mathbf{p})} - 1, s(\mathbf{p}) - b) & o.w. \end{cases} \quad (4)$$

$L_c$  is semantic loss:

$$L_c = \sum_{c=1}^C l_c(c) \log \hat{l}_c(c) \quad (5)$$

where  $\hat{l}_c(c)$  is the multi-class semantic probability at class  $c$  of the ground truth map.  $L_e$  is Eikonal loss:

$$L_e = \frac{(1 - \|\nabla_{\mathbf{p}} s(\mathbf{p})\|)^2}{2\sigma_e^2} \quad (6)$$

$L_h$  is smoothness loss:

$$L_h = \frac{\|\nabla_{\mathbf{p}} s(\mathbf{p}) - \nabla_{\mathbf{p}} s(\mathbf{p} + \Delta \mathbf{p})\|^2}{2\sigma_h^2} \quad (7)$$

That is to convert solving the maximal posterior problem to minimize losses. With this loss function, we use the Adam optimizer based on gradient descent to optimize the iterative solution, which is similar to the training stage of standard differential rendering-based reconstruction methods[29], [9], [11], [8].

The main reason to formulate a probabilistic mapping problem is to compare it with the classic probabilistic mapping in robotics. Note that the ray based likelihood is built upon multiple voxels. The classic way of updating SDF values[18] follows a strong assumption that each voxel in the posterior is independent, which does not follow the Bayesian rule. While in our method, voxels are updated exactly following the Bayesian rule, thus correlated. We consider that this reason mainly explains the more completed and accurate mapping result over the traditional one. In addition, this formulation also allows for more prior and likelihood factors, which may not be easy in the classic updating.

The first-order derivative and inverse second-order derivative of trilinear interpolation of the octree-based grid are implemented on CUDA operators. The details of the derivation

of the gradient for trilinear interpolation of the octree grid are as follows:

**Octree-based Grid:** As described in Section III-A, given a query point  $\mathbf{p}$ , its feature is acquired by summing the trilinearly interpolated features in top  $K$  levels as:

$$\Psi(\mathbf{p}) = \sum_{i=0}^{K-1} triInterp(\psi_{i,j \in \mathcal{N}(\mathbf{p})}) \quad (8)$$

where  $\mathcal{N}(\mathbf{p})$  is the set of the eight nearest Octree nodes i.e. corners  $[(\lfloor x \rfloor, \lfloor y \rfloor, \lfloor z \rfloor), (\lfloor x \rfloor, \lfloor y \rfloor, \lceil z \rceil), \dots, (\lceil x \rceil, \lceil y \rceil, \lceil z \rceil)]^T$  of a cube containing  $\mathbf{p} = (x, y, z)$ . In each octree level,  $\lfloor x \rfloor$  and  $\lceil x \rceil$  are the largest integer less than  $x$  and the smallest integer greater than  $x$ , within the range of the voxel of the current octree hierarchy in which  $x$  is located.

**First-order Derivative:** The trilinear interpolation  $triInterp$  is calculated by first interpolating along the z-axis:

$$\begin{aligned} i_1 &= \Psi[\lfloor x \rfloor, \lfloor y \rfloor, \lfloor z \rfloor] \times (1 - z_d) + \Psi[\lfloor x \rfloor, \lfloor y \rfloor, \lceil z \rceil] \times z_d \\ i_2 &= \Psi[\lfloor x \rfloor, \lceil y \rceil, \lfloor z \rfloor] \times (1 - z_d) + \Psi[\lfloor x \rfloor, \lceil y \rceil, \lceil z \rceil] \times z_d \\ j_1 &= \Psi[\lceil x \rceil, \lfloor y \rfloor, \lfloor z \rfloor] \times (1 - z_d) + \Psi[\lceil x \rceil, \lfloor y \rfloor, \lceil z \rceil] \times z_d \\ j_2 &= \Psi[\lceil x \rceil, \lceil y \rceil, \lfloor z \rfloor] \times (1 - z_d) + \Psi[\lceil x \rceil, \lceil y \rceil, \lceil z \rceil] \times z_d. \end{aligned} \quad (9)$$

where  $x_d = x - \lfloor x \rfloor$ . Then, interpolating along the y-axis, we get:

$$\begin{aligned} w_1 &= i_1(1 - y_d) + i_2 y_d \\ w_2 &= j_1(1 - y_d) + j_2 y_d \end{aligned} \quad (10)$$

Finally, interpolation along the x-axis yields to give the predicted value of the point:

$$\begin{aligned} \Psi(\mathbf{p}) &= w_1(1 - x_d) + w_2 x_d \\ &= \psi_{j \in \mathcal{N}(\mathbf{p})}^T w(\mathbf{p}) \end{aligned} \quad (11)$$

where  $w(\mathbf{p})$  is the interpolation coefficient vector:

$$w(\mathbf{p}) = \begin{bmatrix} (1-x)(1-y)(1-z) \\ x(1-y)(1-z) \\ (1-x)y(1-z) \\ xy(1-z) \\ (1-x)(1-y)z \\ x(1-y)z \\ (1-x)yz \\ xyz \end{bmatrix} \quad (12)$$

The Jacobian of  $\Psi$  w.r.t.  $\mathbf{p}$  is given by:

$$\frac{\partial \Psi}{\partial \mathbf{p}} = \psi_{j \in \mathcal{N}(\mathbf{p})}^T \frac{\partial w}{\partial \mathbf{p}} \quad (13)$$

where  $\frac{\partial w}{\partial \mathbf{p}}$  is the Jacobian of the interpolation coefficient

vector w.r.t.  $\mathbf{p}$ :

$$\frac{\partial w}{\partial \mathbf{p}} = \begin{bmatrix} -(1-y)(1-z) & -(1-x)(1-z) & -(1-x)(1-y) \\ (1-y)(1-z) & -x(1-z) & -x(1-y) \\ -y(1-z) & (1-x)(1-z) & -(1-x)y \\ y(1-z) & x(1-z) & -xy \\ -(1-y)z & -(1-x)z & (1-x)(1-y) \\ (1-y)z & -xz & x(1-y) \\ -yz & (1-x)z & (1-x)y \\ yz & xz & xy \end{bmatrix} \quad (14)$$

**Second-order Derivative:** The second-order Derivative  $\frac{\partial^2 \Psi}{\partial \mathbf{p}^2}$  is given by:

$$\begin{aligned} \frac{\partial^2 \Psi}{\partial \mathbf{p}^2} &= \frac{\partial}{\partial \mathbf{p}} \left( \psi_{j \in \mathcal{N}(\mathbf{p})}^T \frac{\partial w}{\partial \mathbf{p}} \right) \\ &= \psi_{j \in \mathcal{N}(\mathbf{p})}^T \frac{\partial^2 w}{\partial \mathbf{p}^2} \end{aligned} \quad (15)$$

which is a  $3 \times 3$  matrix. We define the result as:

$$\frac{\partial^2 \Psi}{\partial \mathbf{p}^2} = \begin{bmatrix} v_{00} & v_{01} & v_{02} \\ v_{10} & v_{11} & v_{12} \\ v_{20} & v_{21} & v_{22} \end{bmatrix} \quad (15)$$

where

$$\begin{aligned} v_{00} &= v_{11} = v_{22} = 0 \\ v_{01} &= v_{10} = (1-z)(\psi_0 - \psi_1 - \psi_2 + \psi_3) \\ &\quad + z(\psi_4 - \psi_5 - \psi_6 + \psi_7) \\ v_{02} &= v_{20} = (1-y)(\psi_0 - \psi_1 - \psi_4 + \psi_5) \\ &\quad + y(\psi_2 - \psi_3 - \psi_6 + \psi_7) \\ v_{12} &= v_{21} = (1-x)(\psi_0 - \psi_2 - \psi_4 + \psi_6) \\ &\quad + x(\psi_1 - \psi_3 - \psi_5 + \psi_7) \end{aligned} \quad (18)$$

## B. Results on Indoor Datasets

In this section, we provide a quantitative analysis of the reconstructed comparative experiments of two indoor datasets. Indoor data is more cluttered with objects, making it ideal for benchmarking neural fields.

**Dataset:** We additionally do a comparison of the Voxblox reconstruction with our method on two indoor datasets which are captured using the OS0-64 LiDAR. One of them is recorded by ourselves using a mobile robot in the lobby of the university building. The other is the Hilti SLAM dataset 2021[38], recorded at the Hilti office using handheld LiDAR.

**Setting:** Indoor scene experiments involving more cluttered objects require different volume sizes and several parameters for constraints. We use a 5 cm leaf node resolution and a smaller perturbation vector in smoothness prior to better reconstruction performance. The poses are initialized by a LiDAR SLAM method. We follow the evaluation criterion and metrics of KITTI-360 mentioned in the main text. The recall, precision and F-score are computed using a threshold of 5cm for two indoor datasets.

**Quantitative Analysis:** The reconstruction from the sequential point cloud of the lobby dataset and Hilti SLAM 2021 are shown in Fig. 1 and Fig. 2 qualitatively. Quantitative evaluations with comparison methods are reported in Tab. I,

which shows that our method has a better performance on chamfer distance than the Voxblox [18]. It can be seen that our method can reconstruct the details of indoor objects, such as the poles of bulletin boards, tree potted plants, etc., and can also reconstruct the smooth and complete surface of objects.

TABLE I  
MAPPING QUALITY ON INDOOR DATASETS

Dataset	Method	Comp.↓	Acc.↓	C-L1↓	Re.↑	Pre.↑	F1↑
Lobby	Voxblox	4.69	<b>1.14</b>	2.91	78.46	<b>99.89</b>	87.89
	<b>Ours</b>	<b>3.25</b>	1.36	<b>2.31</b>	<b>89.42</b>	99.59	<b>94.24</b>
Hilti	Voxblox	14.94	<b>2.77</b>	8.86	29.17	<b>86.53</b>	43.64
	<b>Ours</b>	<b>4.72</b>	2.92	<b>3.82</b>	<b>65.97</b>	83.59	<b>73.74</b>

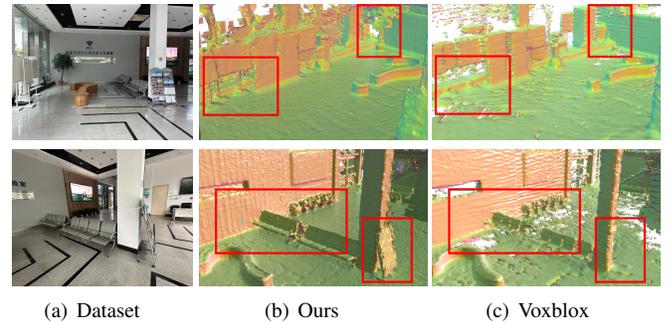


Fig. 1. Case comparison of different methods on the lobby dataset. Each row shows the results of a case area using different methods. Bulletin board pole, plant, magazine rack, and floor are highlighted in the red box.

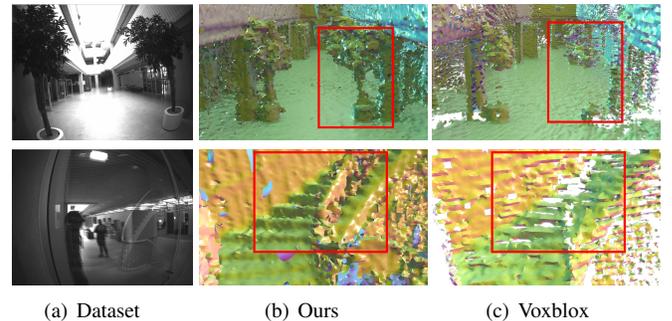


Fig. 2. Case comparison of different methods on the Hilti SLAM 2021 Office. Each row shows the results of a case area using different methods. Plant and stairs are highlighted in the red box.