

OREN: Octree Residual Network for Real-Time Euclidean Signed Distance Mapping

Anonymous For Review

Abstract—Reconstructing signed distance functions (SDFs) from point cloud data benefits many robot autonomy capabilities, including localization, mapping, motion planning, and control. Methods that support online and large-scale SDF reconstruction often rely on discrete volumetric data structures, which affects the continuity and differentiability of the SDF estimates. Neural network methods have demonstrated high-fidelity differentiable SDF reconstruction but they tend to be less efficient, experience catastrophic forgetting and memory limitations in large environments, and are often restricted to truncated SDF. This work proposes OREN, a hybrid method that combines an explicit prior from octree interpolation with an implicit residual from neural network regression. Our method achieves non-truncated (Euclidean) SDF reconstruction with computational and memory efficiency comparable to volumetric methods and differentiability and accuracy comparable to neural network methods. Extensive experiments demonstrate that OREN outperforms the state of the art in terms of accuracy and efficiency, providing a scalable solution for downstream tasks in robotics and computer vision.

I. INTRODUCTION

Accurate and differentiable 3D geometric reconstruction is critical in many robot autonomy and computer vision settings, including localization and mapping [1], [2], rendering and AR/VR [3], [4], autonomous robot navigation [5], [6] and robot manipulation [7], [8]. In robotics, fast updates of the environment model from sensor observations and access to gradient information are important for safe navigation and precise environment interaction, while low memory footprint is important for scalability and onboard operation.

In this work, we focus on signed distance function (SDF) reconstruction. Given a query point, an SDF returns the signed distance to the nearest surface in the environment with sign indicating whether the query is in free (positive) or occupied (negative) space. SDFs have received increasing attention due to their constant-time complexity for distance and collision queries and their ability to capture complex obstacle surfaces implicitly as a zero-level set.

SDF reconstruction methods can roughly be organized into three categories: volumetric methods (e.g., [5], [9]), Gaussian Process (GP) methods (e.g., [10], [11]), and neural network methods (e.g., [12], [13]). We review representative papers from these categories in Sec. II. Volumetric methods utilize advanced data structures, like octrees and hashmaps, and are known for their real-time performance and scalability to large scenes. However, they provide non-differentiable SDF estimates and require increasing storage to achieve higher accuracy. GP methods learn continuous SDF models with uncertainty quantification but often suffer from high computational complexity and poor scalability. Recently, neural network methods have shown great potential to learn

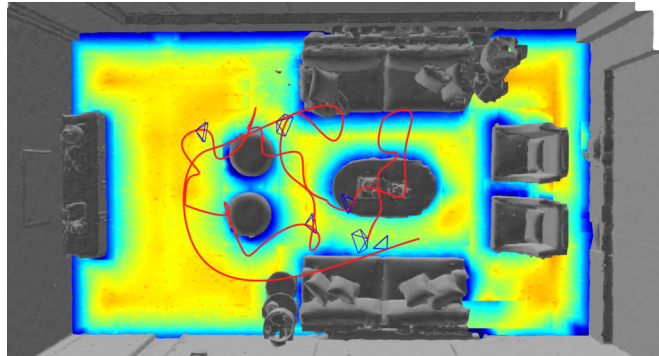


Fig. 1: OREN reconstructs an accurate Euclidean signed distance function online from streaming point cloud data.

compact and accurate SDF representations but are often restricted to truncated SDF and struggle with catastrophic forgetting in large scenes or in online settings.

We propose OREN, a hybrid method that combines the strengths of volumetric methods, in the form of an explicit SDF prior obtained from octree interpolation, and neural network methods, in the form of implicit features decoded into a residual correction of the prior. To construct the prior, we use a semi-sparse octree with SDF and gradient estimates stored at the octant vertices and gradient-augmented interpolation to obtain SDF priors at arbitrary query positions. We augment the prior prediction with a neural network residual, which recovers fine geometric details of the observed surface from implicit features. We train our hybrid explicit-implicit model using three loss functions that supervise both near-surface and distant SDF values and accelerate the convergence to achieve real-time accurate Euclidean SDF reconstruction.

The closest works to our method are H₂-Mapping [14] and Gradient-SDF [15]. Similar to our method, H₂-Mapping applies trilinear interpolation in a sparse octree to obtain an SDF prior and trains a neural network residual. In contrast, H₂-Mapping reconstructs only truncated (near-surface) SDF. Gradient-SDF is voxel-based and also uses SDF gradients but only for improving the quality of surface reconstruction and photometric bundle adjustment, which does not consider free-space SDF estimation. In contrast, our method optimizes the octree parameters to obtain an accurate Euclidean SDF prior, corrects the prior via neural residual predictions and achieves real-time operation.

In summary, our work makes the following contributions.

- We introduce a new gradient-augmented interpolation in a semi-sparse octree to obtain an SDF prior, improving the accuracy, memory, and training speed for subsequent SDF residual learning.
- We formulate a hybrid model that combines the explicit priors with an implicit neural residual, enabling accurate

SDF learning both near to and far from the observed surfaces. We also design loss functions to encourage globally accurate SDF learning and accelerate the training process to achieve real-time performance.

II. RELATED WORK

Various methods have been proposed to learn SDF from point cloud data. They can be categorized into three groups: volumetric methods [5], [9], [16]–[20], GP-based methods [10], [11], [21], and neural network based methods [1], [12]. We review these categories and then discuss the recent trend of using hybrid models for SDF reconstruction.

A. Volumetric SDF Reconstruction

Volumetric methods [5], [19] achieve real-time truncated SDF (TSDF) reconstruction using voxel hashing to efficiently look voxels up for updates and queries. Voxblox [5] builds a TSDF layer by projective distance (distance between the voxel center and the observed surface point) and updates the values via breadth-first search (BFS). Both the projective distance and the BFS path length introduce inaccuracies in the SDF estimates. Subsequent works [18], [19] reduce these errors by using non-projective distance and replacing BFS path length with the distance between the voxel center and the nearest oriented surface point. Gradient-SDF [15] shows that estimating TSDF together with its gradient improves accuracy and can be used for SDF-based camera tracking and bundle adjustment. However, all of these methods rely on discrete SDF representations, which are non-differentiable and have accuracy limited to the grid resolution. In contrast, our method enables real-time learning of continuous and differentiable SDF. We first estimate SDF and its gradient with an explicit discrete prior stored in an octree. Then, we use implicit neural features to predict residuals and correct the prior. The combination of explicit prior and implicit neural features forms a compact differentiable representation of continuous SDF.

B. Gaussian Process SDF Reconstruction

GP methods learn continuous SDF representations, and support gradient computation and uncertainty quantification [10], [11], [21]. GPIS [10] uses GP to estimating the oriented surface points and regressing SDF online. The GP trained to predict SDF learns the surface implicitly as the zero-level set of SDF. However, GPIS does not allow extrapolating SDF prediction to positions away from the surface. Log-GPIS [21] and VDB-GPDF [11] learn unsigned distance function in log space, which generalizes well globally but does not provide sign information. GP methods do not scale well to large environments due to the cubic complexity of the matrix inverse during training. In comparison, our method uses octree interpolation, which has $O(1)$ time complexity, and matrix multiplication, which has a roughly quadratic time complexity $O(n^2)$ for n hidden dimension, to compute the SDF prior and the SDF residual respectively. Our octree structure also has a smaller memory footprint than the gram matrix used by GP.

C. Neural Network SDF Reconstruction

DeepSDF [12] was among the first methods to demonstrate that neural networks can learn compact and continuous implicit SDF representations. This inspired many subsequent neural network methods for SDF reconstruction. iSDF [1] formulates an incremental SDF learning approach by iteratively updating the parameters of an MLP with training samples generated from a key frame set. iSDF also uses Eikonal regularization to encourage the model to satisfy the Eikonal property of SDF. NeuS [13] learns SDF and a neural radiance field (NeRF) [22] simultaneously, allowing the two to improve each other by using SDF-based volume density estimates. Other works, like IGR [23] and NGLoD [24], formulate various neural network designs, loss functions, training procedures to improve the SDF reconstruction accuracy. These works show that neural networks are able to learn SDF near the surface sufficiently accurately for high-fidelity surface reconstruction. However, accuracy far away from the surface is rarely evaluated by neural network approaches. Neural network methods [12], [23]–[25] that learn Euclidean SDF are often object-level and require extensive training data and time to achieve satisfactory accuracy.

D. Hybrid Methods for SDF Reconstruction

Recently, hybrid models that combine explicit geometric structures with implicit neural features show promising results. PIN-SLAM [2] stores neural features in near-surface voxels. Given a query point, its SDF prediction is a weighted sum of k SDF predictions, obtained by feeding k nearest neural features with local positions into a global decoder. H_2 -Mapping [14] presents another model that combines an octree-based SDF prior with a neural residual. LCP-Fusion [26] extends H_2 -Mapping to an SLAM with SDF prior and implicit neural feature as combined feature for localization. However, PIN-SLAM, H_2 -Mapping and LCP-Fusion still learn truncated SDF. HIO-SDF [27] removes truncation by running Voxfield [19] first to generate global SDF priors, which are combined with local SDF approximations to train a neural network. However, the accuracy and speed are limited by the volumetric method. As the observed area grows, the neural network, whose number of parameters is fixed, tends to learn an over-smooth SDF.

In contrast, our method, OREN, builds a semi-sparse octree to store the prior of SDF values and gradients, which is extendable as the scene grows and efficiently represents the SDF of the whole space. With gradient-augmented interpolation in the octree, our method can produce more accurate SDF priors, leaving more capacity for the subsequent network to recover surface geometric details.

III. PROBLEM STATEMENT

Consider a 3D environment with a set of obstacles $\mathcal{O} \subset \mathbb{R}^3$. The SDF $d : \mathbb{R}^3 \rightarrow \mathbb{R}$ of \mathcal{O} is defined as the shortest distance from any point $\mathbf{x} \in \mathbb{R}^3$ to the obstacle surface $\partial\mathcal{O}$, with a sign indicating whether \mathbf{x} is inside or outside of \mathcal{O} :

$$d(\mathbf{x}) = \begin{cases} \min_{\mathbf{y} \in \partial\mathcal{O}} \|\mathbf{x} - \mathbf{y}\|_2, & \mathbf{x} \notin \mathcal{O}, \\ -\min_{\mathbf{y} \in \partial\mathcal{O}} \|\mathbf{x} - \mathbf{y}\|_2, & \mathbf{x} \in \mathcal{O}. \end{cases} \quad (1)$$

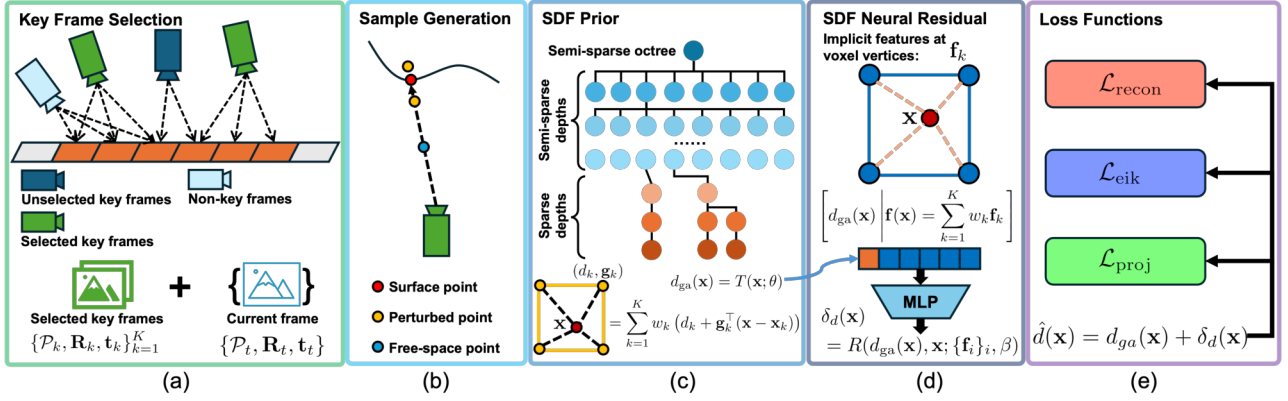


Fig. 2: Method overview: a) We keep key frames with small overlap that maximize surface coverage for training; b) With the selected key frames and the current frame, we generate three types of samples: **surface** points, **perturbed** points around the surface, and **free-space** points; c) To predict SDF, we first obtain an SDF prior $d_{ga}(\mathbf{x})$ with gradient-augmented interpolation in a semi-sparse octree, where each octant vertex has estimated SDF value and gradient; d) We also obtain an implicit neural feature for \mathbf{x} by tri-linear interpolation of implicit features stored at the octant vertices, which is fed together with the prior $d_{ga}(\mathbf{x})$ into an MLP decoder to obtain an SDF residual correction $\delta_d(\mathbf{x})$; e) The SDF prior $d_{ga}(\mathbf{x})$ and the SDF residual $\delta_d(\mathbf{x})$ are combined as the final SDF prediction $\hat{d}(\mathbf{x}) = d_{ga}(\mathbf{x}) + \delta_d(\mathbf{x})$, and the parameters are trained with three loss functions: **reconstruction** loss, **Eikonal** loss and **projection** loss.

The SDF satisfies two key properties: 1) the obstacle surface is encoded as the zero-level set, $d(\mathbf{x}) = 0, \forall \mathbf{x} \in \partial\mathcal{O}$, and 2) the gradient of $d(\mathbf{x})$ is the unit vector pointing away from the nearest surface point and satisfies an Eikonal equation [1]:

$$\nabla d(\mathbf{x}) = \frac{\mathbf{x} - \mathbf{x}_*}{d(\mathbf{x})}, \quad \|\nabla d(\mathbf{x})\|_2 = 1, \text{ a.e.}, \quad (2)$$

where $\mathbf{x}_* \in \arg \min_{\mathbf{y} \in \partial\mathcal{O}} \|\mathbf{x} - \mathbf{y}\|_2$.

Given a stream of point clouds obtained from range sensor measurements (e.g., from LiDAR or depth camera), $\{\mathbf{o}_t, \mathcal{P}_t\}_{t=1}$, where \mathbf{o}_t is the sensor position at time step t and \mathcal{P}_t is the set of observed surface points in the global frame, our objective is to approximate the SDF $d(\mathbf{x})$ of \mathcal{O} as a scalar field, $\hat{d}: \mathbb{R}^3 \rightarrow \mathbb{R}$. We also aim to have \hat{d} capture the SDF gradient accurately.

IV. OCTREE RESIDUAL NETWORK FOR LEARNING SDF

Our method employs a hybrid model to reconstruct SDF. We use a semi-sparse octree, where certain octants with no surface contained are created, to store explicit SDF and SDF gradient estimates in order to compute a coarse SDF prior, described in Sec. IV-A. To recover the geometric details, we also store implicit neural features at the octant vertices and use an MLP decoder to provide corrections to the prior SDF estimates. The implicit neural residual prediction is described in Sec. IV-B. To train our model efficiently, we maintain a set of key frames that cover the observed surface with small overlap between adjacent frames, as described in Sec. IV-C. Then, from the key frames and the latest frame, we generate a dataset containing different types of samples, discussed in Sec. IV-D, and use it to train the model with loss functions proposed in Sec. IV-E. Fig 2 shows an overview.

A. SDF Prior via Gradient-Augmented Octree Interpolation

1) *Semi-Sparse Octree:* The SDF prior for position \mathbf{x} is obtained by interpolation in a semi-sparse octree data structure with N layers. We refer to an octree layer as *dense* when all octants are created as a regular grid; as *semi-sparse* when child octants containing surface points and all

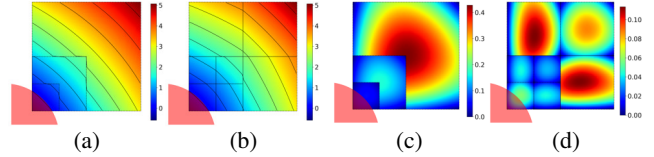


Fig. 3: 2D visualization of SDF interpolation without gradient augmentation using (a) a sparse octree and (b) a semi-sparse with corresponding interpolation error shown in (c) and (d) respectively. The bottom-left red region is an obstacle containing one vertex.

of their siblings (regardless of occupancy) are created simultaneously; and as *sparse* when only child octants containing surface points are created.

We use a semi-sparse octree of resolution r , where the first M layers are semi-sparse and the remaining $N - M$ layers are sparse. This is illustrated in Fig. 2c. In each vertex \mathbf{x}_k of an octant with $k \in \{1, \dots, 8\}$, we store estimates $d_k \in \mathbb{R}$ and $\mathbf{g}_k \in \mathbb{R}^3$ of the SDF $d(\mathbf{x}_k)$ and its gradient $\nabla d(\mathbf{x}_k)$, respectively, which are learnable during training. To maintain memory efficiency, a vertex is shared across neighboring octants from different tree depths. For example, the eight vertices of an octant are also included in the vertices of its eight child octants. This semi-sparse structure is essential to obtain a good SDF prior, especially for query positions away from the surface. The on-demand initialization of all child octants in the first M layers costs extra memory but improves the accuracy significantly.

Fig. 3 shows a 2D example of an SDF prior using trilinear interpolation in a sparse and a semi-sparse octree. In a sparse octree, the SDF interpolation has more discontinuities on the octant boundaries compared to the result in a semi-sparse octree. Given a query point \mathbf{x} , the semi-sparse octree provides the smallest octant containing \mathbf{x} that is not larger than the smallest octant found in the sparse octree. This guarantees that we can find vertices closer to \mathbf{x} for computing the SDF prior because of the creation of sibling octants, which leads to a significantly smaller SDF interpolation error as indicated by Fig. 3c and Fig. 3d.

For any query position where the surface exists in the neighborhood, we can locate an octant no larger than $r \times 2^{N-M}$. For queries distant to the surface, an empty large octant is sufficient for computing an accurate SDF prior using gradient-augmented interpolation, which is discussed next.

2) *Gradient-Augmented Interpolation*: To achieve smaller SDF errors of the prior so that the subsequent neural network can focus on restoring the geometric details, we propose a new gradient-augmented trilinear interpolation method. Given the smallest octant that contains a query position \mathbf{x} , we first obtain an extrapolation result from each vertex \mathbf{x}_k :

$$d_k(\mathbf{x}) = d_k + \mathbf{g}_k^\top (\mathbf{x} - \mathbf{x}_k), \quad k \in \{1, \dots, 8\}. \quad (3)$$

Given the extrapolation results, we compute the gradient-augmented (ga) interpolation:

$$d_{ga}(\mathbf{x}) = \frac{1}{\gamma} \sum_{k=1}^8 w_k d_k(\mathbf{x}), \quad \gamma = \sum_{k=1}^8 w_k, \quad (4)$$

where $w_k = 1/|\text{diag}(\mathbf{x}_i - \mathbf{x}_k)|$ is the interpolation weight. It is a standard formulation but expressed in vector form for brevity. In contrast, the regular trilinear (tl) interpolation is:

$$d_{tl}(\mathbf{x}) = \frac{1}{\gamma} \sum_{k=1}^8 w_k d_k, \quad \gamma = \sum_{k=1}^8 w_k. \quad (5)$$

Empirically, gradient-augmented interpolation generates more accurate SDF priors. Fig. 4 shows two 2D examples. Each row shows the ground truth SDF, interpolation results of using w/ and w/o gradient augmentation, the corresponding errors, and the Hessian spectral norm of SDF. As shown in Fig. 4d and 4e, gradient-augmented interpolation has smaller errors, especially when more obstacles are in the scene. The gradient-augmented interpolation requires extra memory and computation for the gradient \mathbf{g}_k and the extrapolation, respectively. However, theoretically, interpolation with gradient augmentation causes a smaller error upper bound.

Proposition 1. Consider an octant $\mathcal{V} \subset \mathbb{R}^3$ of size L . Assume that the SDF $d(\mathbf{x})$ is twice differentiable and the spectral norm of its Hessian is bounded:

$$M := \sup_{\mathbf{x} \in \mathcal{V}} \|\nabla^2 d(\mathbf{x})\|_2 < \infty. \quad (6)$$

Assume that each vertex \mathbf{x}_k has ground-truth SDF value $d_k = d(\mathbf{x}_k)$ and gradient $\mathbf{g}_k = \nabla d(\mathbf{x}_k)$. Then, given arbitrary $\mathbf{x} \in \mathcal{V}$, the errors of gradient-augmented interpolation in (4) and trilinear interpolation in (5) satisfy:

$$\begin{aligned} e_{ga}(\mathbf{x}) &= |d_{ga}(\mathbf{x}) - d(\mathbf{x})| \leq \bar{e}_{ga} = \frac{3ML^2}{8}, \\ e_{tl}(\mathbf{x}) &= |d_{tl}(\mathbf{x}) - d(\mathbf{x})| \leq \bar{e}_{tl} = \frac{\sqrt{3}L}{2}. \end{aligned} \quad (7)$$

Proof. For each octant vertex \mathbf{x}_k , by Taylor's theorem: $d(\mathbf{x}) = d_k + \mathbf{g}_k^\top (\mathbf{x} - \mathbf{x}_k) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_k)^\top \nabla^2 d(\boldsymbol{\xi}_k)(\mathbf{x} - \mathbf{x}_k)$, for some $\boldsymbol{\xi}_k$ on the line segment joining \mathbf{x} and \mathbf{x}_k . Using (4) and (6), the gradient-augmented interpolation error satisfies:

$$\begin{aligned} e_{ga}(\mathbf{x}) &= \frac{1}{2\gamma} \left| \sum_{k=1}^8 w_k (\mathbf{x} - \mathbf{x}_k)^\top \nabla^2 d(\boldsymbol{\xi}_k) (\mathbf{x} - \mathbf{x}_k) \right| \\ &\leq \frac{M}{2\gamma} \sum_{k=1}^8 w_k \|\mathbf{x} - \mathbf{x}_k\|_2^2 \leq \frac{3ML^2}{8} = \bar{e}_{ga}, \end{aligned} \quad (8)$$

where equality holds when \mathbf{x} is the octant center so that $\frac{1}{\gamma} \sum_{k=1}^8 w_k \|\mathbf{x} - \mathbf{x}_k\|_2^2 = 3L^2/4$. Similarly, for the error of the regular trilinear interpolation, we have:

$$d(\mathbf{x}) = d_k + \nabla d(\boldsymbol{\zeta}_k)^\top (\mathbf{x} - \mathbf{x}_k) \quad (9)$$

for some $\boldsymbol{\zeta}_k$ on the line segment joining \mathbf{x} and \mathbf{x}_k . Then, since $\|\nabla d(\boldsymbol{\zeta}_k)\| = 1$ and using (5), the trilinear interpolation error satisfies:

$$\begin{aligned} e_{tl}(\mathbf{x}) &= \left| \frac{1}{\gamma} \sum_{k=1}^K w_k \nabla d(\boldsymbol{\xi}_k)^\top (\mathbf{x} - \mathbf{x}_k) \right| \\ &\leq \frac{1}{\gamma} \sum_{k=1}^K w_k \|\mathbf{x} - \mathbf{x}_k\|_2 \leq \frac{\sqrt{3}L}{2} = \bar{e}_{tl}. \quad \square \end{aligned} \quad (10)$$

When an octant does not contain positions where the gradient is not well defined (e.g., the medial axes where the closest point projection is not unique), (6) holds. For positions without well-defined gradients, although the Hessian norm blows up mathematically, gradient-augmented interpolation has $e_{ga}(\mathbf{x})$ significantly lower than \bar{e}_{ga} based on empirical observation. The second row of Fig. 4d and 4f shows an example of such cases, that the Hessian spectral norm is large on the medial axes, but gradient-augmented interpolation has smaller errors. Since we are looking for an upper bound on the error, we ignore such cases.

As shown in Fig. 4, the gradient-augmented interpolation has smaller errors than without gradient augmentation. Empirically, as shown in Fig. 4f, $M \ll 1$ so that $\bar{e}_{ga}/\bar{e}_{tl} = \sqrt{3}ML/4 < 1$. Especially, when the octant is surrounded by multiple obstacles, the SDF prior values stored at the vertices are smaller than the ground truth SDF values inside the octant. This makes SDF priors obtained from interpolation without gradient augmentation no larger than the vertex SDF values, leading to significantly larger errors.

Hence, the prior network $T(\mathbf{x}; \theta)$ of our method is a semi-sparse octree where each vertex has an estimate of SDF and gradient, i.e., $\theta = \{d_k, \mathbf{g}_k\}_{k=1}^K$, which are learnable parameters optimized together with the residual network. In the experiments, we maintain a semi-sparse octree for each scene with $N = 8$, $M = 5$ and $r = 10$ cm.

B. SDF Residual via Neural Feature Decoding

The accuracy of the SDF prior is limited to the octree resolution, causing lack of geometric details. To achieve high fidelity, OREN learns a residual correction to the SDF values via a neural network $R(d_{ga}(\mathbf{x}), \mathbf{x}; \{\mathbf{f}_i\}_i, \beta)$, is a composition of octree feature interpolation $\mathbf{f}(x) = \sum_k w_k \mathbf{f}_k$ with implicit neural features $\mathbf{f}_i \in \mathbb{R}^F$ stored at octree vertices and an MLP decoder $D(d, \mathbf{f}(x); \beta)$.

To enable continual learning as the sensor moves in the environment, we assign each octant vertex an implicit feature \mathbf{f}_i . Octree expansion extends the number of implicit neural features automatically by allocating more \mathbf{f}_i to the near-surface regions in smaller octants. The features \mathbf{f}_i are initialized as zeros and are optimized together with the network weights β .

As shown in Fig. 2d, for each query point \mathbf{x} , we locate the leaf octant containing \mathbf{x} and compute the interpolated

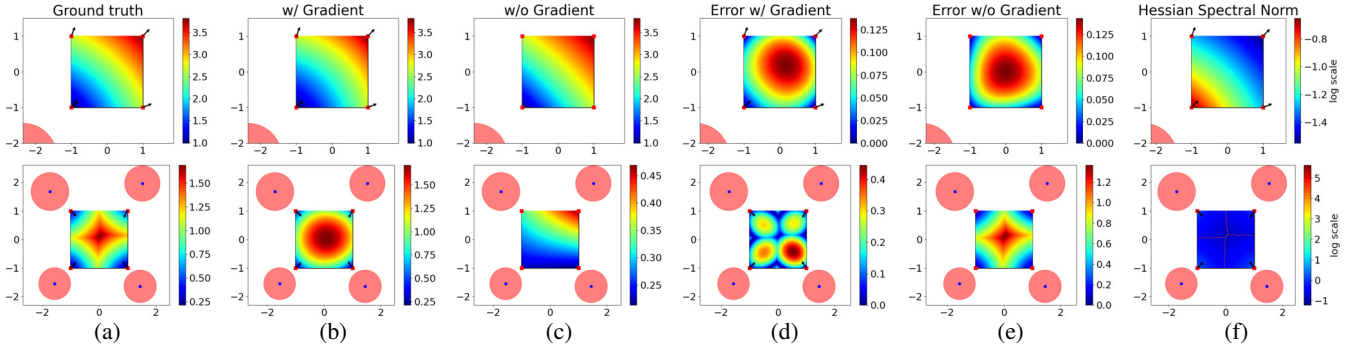


Fig. 4: 2D visualization of interpolation with and without gradient augmentation for one (red region, top row) and four obstacles (red regions, bottom row). Gradient-augmented interpolation produces a better SDF prior in (b) with smaller error in (d). Empirically, positions where the SDF gradient is not well defined (large Hessian spectral norm), as shown in (f), have small interpolation error with gradient augmentation as shown in (d).

neural feature $\mathbf{f}(\mathbf{x}) = \sum_{k=1}^8 w_k \mathbf{f}_k$ with \mathbf{f}_k assigned to the octant vertices, where w_k is the same weight used in (4). The feature $\mathbf{f}(\mathbf{x})$ and the prior $d_{ga}(\mathbf{x})$ are fed into a decoder to predict the SDF residual $\delta_d(\mathbf{x}) = D(d_{ga}(\mathbf{x}), \mathbf{f}(\mathbf{x}); \beta)$. In the evaluation, we have $F = 3$ and the MLP has two 32-dim hidden layers with LeakyReLU activation.

C. Key Frame Selection

To learn SDF in real time, it is important to keep a compact and representative set of sensor frames for training. We adopt the key-frame selection strategy of H_2 -Mapping [14]. As shown in Fig. 2a, we insert a new sensor frame when the newly observed area compared with the last key frame is large enough, i.e., $\frac{|V_c \cap V_l|}{|V_c \cup V_l|} > c_{\min}$, where V_c is the set of surface octants observed by the current frame, V_l is the set of surface octants observed by the last inserted key frame, and $c_{\min} \in [0, 1]$ is a threshold. This strategy makes sure that the frames cover the observed surface with little overlap between adjacent key frames.

Over time, the number of key frames grows. It is essential to select only W key frames to maintain real-time operation. We incrementally select the frame that observes the most octants, mask out these octants, and repeat until W frames are collected. If all octants are masked out, we reset the mask except for the octants masked out by the last selected key frame and continue the selection. This strategy ensures that the selected key frames maximally cover the scene. In the evaluation, we set $W = 8$.

D. Dataset Generation

During online training, it is important to generate a high-quality dataset consisting of a small number of representative samples. At time step t , suppose the set of selected key frame time steps is $\mathcal{T} = \{k, 1 \leq k \leq t\}$, $|\mathcal{T}| \leq W$. For each frame $\mathcal{P}_{i \in \mathcal{T} \cup \{t\}}$, we randomly choose $\lfloor N/|\mathcal{T} \cup \{t\}| \rfloor$ rays $\{\mathbf{o}_j = \mathbf{o}_i, \mathbf{q}_j \in \mathcal{P}_{i \in \mathcal{T} \cup \{t\}}\}_{j=1}^N$ to generate samples. As shown in Fig. 2b, we generate three types of points for training:

1) *Free-space Points*: To learn the SDF in free space, we sample free-space points \mathcal{P}_F by drawing n_F points $\{\mathbf{x}_n\}_{n=1}^{n_F}$ along each ray j : $\mathbf{x}_n = \mathbf{o}_j + \lambda(\mathbf{q}_j - \mathbf{o}_j)$, where λ is drawn from the uniform distribution $\mathcal{U}(\delta, 1 - \delta)$ with margin $\delta > 0$;

2) *Surface Samples and Perturbed Points*: To provide supervision for the surface reconstruction, we generate \mathcal{P}_S

by collecting the surface point \mathbf{q}_j of each ray j and the perturbed points \mathcal{P}_P by sampling n_P points $\{\mathbf{x}_n\}_{n=1}^{n_P}$ along each ray j , i.e., $\mathbf{x}_n = \mathbf{q}_j + \alpha(\mathbf{q}_j - \mathbf{o}_j) / \|\mathbf{q}_j - \mathbf{o}_j\|_2$, where α has uniform distribution on $(-3\sigma, \sigma) \cup (\sigma, 3\sigma)$.

3) *Ground Truth SDF Computation*: For surface points \mathcal{P}_S , we have ground truth SDF $d(\mathbf{x}) = 0, \mathbf{x} \in \mathcal{P}_S$. For perturbed points and free-space points, we approximate the ground truth as $\hat{d}(\mathbf{x}) = \text{sign}(\mathbf{x}) \min_{\mathbf{y} \in \mathcal{P}_S} \|\mathbf{x} - \mathbf{y}\|_2$.

In our experiments, we have $W = 8$, $N = 20480$, $\delta = 0.05$, $\sigma = 0.06$, $n_F = 1$, and $n_P = 2$.

E. Loss Functions

As shown in Fig. 2e, the SDF prior $d_{ga}(\mathbf{x}) = T(\mathbf{x}; \theta)$ and the neural residual $\delta_d(\mathbf{x}) = R(d_{ga}(\mathbf{x}), \mathbf{x}; \{\mathbf{f}_i\}_i, \beta)$ are combined together to obtain a final SDF prediction:

$$\hat{d}(\mathbf{x}) = d_{ga}(\mathbf{x}) + \delta_d(\mathbf{x}). \quad (11)$$

It is important to design appropriate loss functions to train the octree and neural network parameters θ, β .

1) *Reconstruction Loss*: We apply an L1 loss over the surface points \mathcal{P}_S and the perturbation points \mathcal{P}_P to capture the surface geometry, which is critical for accurate 3D reconstruction:

$$\mathcal{L}_{\text{recon}} = \underbrace{\frac{w_{\text{recon}}^S}{|\mathcal{P}_S|} \sum_{\mathbf{x} \in \mathcal{P}_S} |\hat{d}(\mathbf{x})|}_{\mathcal{L}_{\text{surface}}} + \underbrace{\frac{w_{\text{recon}}^P}{|\mathcal{P}_P|} \sum_{\mathbf{x} \in \mathcal{P}_P} (c_j^{\text{lo}} + c_j^{\text{up}})}_{\mathcal{L}_{\text{perturbation}}}, \quad (12)$$

$$c_j^{\text{lo}} = \max(e^{\eta(\underline{b}_j - \hat{d}(\mathbf{x}))} - 1, 0), c_j^{\text{up}} = \max(\hat{d}(\mathbf{x}) - \bar{b}_j, 0),$$

where w_{recon}^S and w_{recon}^P are the corresponding weights and \underline{b}_j and \bar{b}_j are lower and upper bounds, respectively. We set $\eta = 10$. For perturbed points with $\alpha > 0$, $\underline{b}_j = \hat{d}(\mathbf{x})$, $\bar{b}_j = -\sigma$, while for $\alpha < 0$, $\underline{b}_j = \sigma$, $\bar{b}_j = \hat{d}(\mathbf{x})$.

2) *Eikonal Loss*: To enforce the Eikonal property in (2), we apply another L1 loss for the gradient norm:

$$\mathcal{L}_{\text{eik}} = \sum_{k \in \{S, P, F\}} \frac{w_{\text{eik}}^k}{|\mathcal{P}_k|} \sum_{\mathbf{x} \in \mathcal{P}_k} \left| \|\hat{\mathbf{g}}(\mathbf{x})\|_2 - 1 \right|, \quad (13)$$

over three different kinds of samples $\mathcal{P}_S, \mathcal{P}_P$ and \mathcal{P}_F , where w_{eik}^k is the weight for the corresponding sample set \mathcal{P}_k . Here, $\hat{\mathbf{g}}(\mathbf{x})$ is obtained from numerical differentiation instead of the auto gradient graph because the numerical

differentiation involves more positions $\mathbf{x} \pm \epsilon \mathbf{e}_i$ for each dimension i , which helps the network converge and shows better training stability when the SDF gradient is not well defined at certain positions.

3) *Projection Loss*: Although the Eikonal loss \mathcal{L}_{eik} enforces the gradient magnitude, the supervision for the gradient direction and SDF in the distant space is still missing. Hence, we propose a projection loss for free-space points \mathcal{P}_F that are collected along rays:

$$\mathcal{L}_{\text{proj}} = \frac{w_{\text{proj}}}{|\mathcal{P}_F|} \sum_{\mathbf{x} \in \mathcal{P}_F} \left| \hat{d}(\mathbf{x}) - \tilde{d}(\mathbf{x}) \right|. \quad (14)$$

Although the above loss has a form similar to (12), we call it *projection loss* because $\tilde{d}(\mathbf{x})$ is actually a loose upper bound for the ground truth SDF $d(\mathbf{x})$. The purpose of this loss is not to make the model predict $\tilde{d}(\mathbf{x})$ exactly at \mathbf{x} but to provide the implicit supervision of the gradient direction so that it speeds up the convergence of other loss functions.

In our experiments, we set $w_{\text{recon}}^S = 1000$, $w_{\text{recon}}^P = 200$, $w_{\text{eik}}^S = w_{\text{eik}}^F = 10$, $w_{\text{eik}}^P = 3$, and $w_{\text{proj}} = 100$.

V. EVALUATION

In this section, we compare OREN with four baselines: Voxblox [5], H₂-Mapping [14], PIN-SLAM [2], and HIO-SDF [27]. We first examine mesh and SDF reconstructions as a qualitative comparison, then quantitatively evaluate the methods using different metrics. In addition, we perform an ablation study to evaluate the contribution of each component in our method. We use the Replica dataset [28], which provides eight synthesized scenes with one trajectory per scene to compare OREN with the baselines. We also test all methods on real-data from the Newer College dataset [29]. For all methods and datasets, we use the ground truth poses. For PIN-SLAM, we disable its localization module.

A. Qualitative Results

1) *Mesh Reconstruction*: Fig. 5 shows mesh reconstruction results on the Replica room 0 scene. H₂-Mapping, PIN-SLAM, and OREN generate complete and high-quality meshes. H₂-Mapping tends to produce smoother surfaces as it only allocates octree voxels near the surface. The completeness of our reconstructions is better than H₂-Mapping. Compared with PIN-SLAM [2], which only predicts SDF values in regions close to the surface, our meshes exhibit smoother geometry with less noise. Compared to HIO-SDF [27], which also estimates continuous and differentiable SDFs, our reconstructions demonstrate substantially higher fidelity. We also present the reconstructed meshes of the Newer College dataset by OREN and PIN-SLAM in Fig. 6, which shows that our method also outperforms the baselines on large scale real dataset.

2) *SDF Reconstruction*: We visualize z-plane slices of the SDF predictions in Fig. 5. H₂-Mapping and PIN-SLAM estimate truncated SDF only. Although HIO-SDF can predict SDF values over the entire space, it struggles to precisely encode the surface as a zero-level set, which is crucial for robotic tasks where accurate perception of obstacle boundaries is required. In contrast, OREN faithfully reconstructs

the surface position and provides reliable SDF estimates in free space. The predictions of OREN outside the scene boundaries are less accurate due to the lack of sensor observations. But this has little impact on applications where robots operate within the observed workspace.

B. Quantitative Results

We compute two sets of metrics: mesh metrics to evaluate the surface reconstruction quality and SDF metrics to evaluate the overall SDF predictions.

1) *Mesh Metrics*: We uniformly sample two point clouds, $\mathcal{P}_{\text{g.t.}}$ and $\mathcal{P}_{\text{recon}}$, of 200k points each from the ground-truth mesh and from the reconstructed mesh, and report *Chamfer-L1 distance*, *F1 score*, *precision*, *recall*, *completion*, *completion ratio*, and *accuracy* [2], [14].

As shown in Table I, OREN outperforms the baselines in recall, completion, and completion ratio. Our method is mostly the second best in the other mesh metrics. Since H₂-Mapping and PIN-SLAM are specially optimized for surface reconstruction, they perform slightly better in metrics like F1 score. However, their mesh has more holes, which can also improve metrics like precision. Voxblox performs mostly worse than other methods. Since HIO-SDF relies on a volumetric method like Voxfield [19] to generate the SDF dataset, it also performs worse than OREN.

2) *SDF Metrics*: We evaluate the SDF predictions of all methods using a regular grid of resolution 1.25 cm (20 cm for Newer College). For each scene, the ground-truth SDF d of each grid center is computed from the ground-truth mesh (point cloud for Newer College). In Table II, we report the *mean absolute error* (MAE) of SDF, the *angular MAE* of the SDF gradient, and the *SDF valid ratio*, defined as the proportion of test positions where a method can predict SDF. To examine the prediction quality in different regions, we categorize the grid points with $-0.1 \leq d \leq 0.2$ m as near surface, and the other points as far from the surface. OREN marginally performs better than the baselines in all SDF metrics. HIO-SDF fails to train the network stably when its volumetric method does not provide good results. Since H₂-Mapping and PIN-SLAM are able to predict SDF only near the surface, their SDF valid ratios are extremely low, while HIO-SDF and OREN both cover the whole scene.

3) *Runtime Metrics*: We also measure the timing of each method with Intel 14900K and NVIDIA RTX 3090. As shown in Table III, OREN runs at 13.96 fps, which is the fastest. Voxblox is known for its speed in learning TSDF. But it is actually very slow for ESDF due to the integration. As for memory consumption, since different methods have the memory allocation distributed on CPU and GPU in different ways, we do not compare the memory usage. However, OREN can run on NVIDIA Jetson Orin with 16 GB RAM at 7 fps, which shows that OREN is very efficient.

C. Ablation Study

To investigate the contribution of the semi-sparse octree and gradient-augmented interpolation, we train three variants, without the neural residual network, using a regular sparse

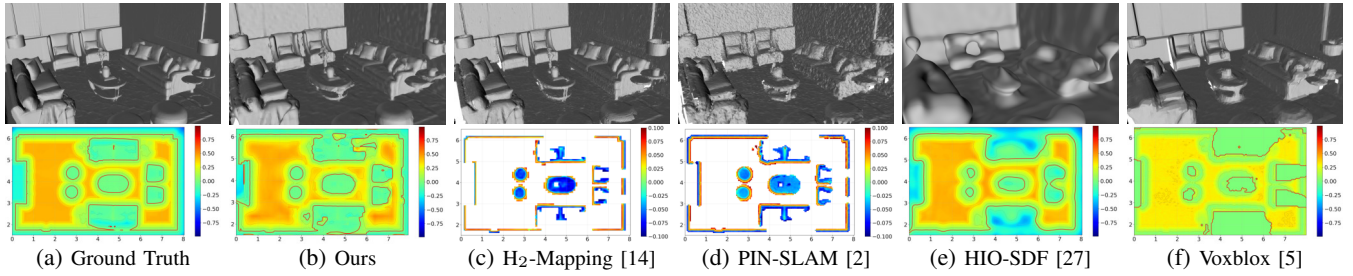


Fig. 5: Qualitative comparison of mesh reconstruction (top row) and z-plane slice of SDF reconstruction (bottom row) on Replica room 0 [28]. OREN reconstructs a mesh with the highest completion ratio and accurate SDF both near and far from the surface. H₂-Mapping and PIN-SLAM only learn truncated SDF. HIO-SDF learns an over smooth result. Voxblox significantly under-estimates the SDF.

TABLE I: Mesh reconstruction metrics on the Replica dataset [28] and the Newer College dataset [29]. The best and second best results are bold and underlined, respectively. For the Replica dataset, $\delta = 5\text{cm}$. For the Newer College dataset, $\delta = 20\text{cm}$.

Metric	Method	room 0	room 1	room 2	office 0	office 1	office 2	office 3	office 4	Newer College
Completion [cm] ↓	OREN	2.55	1.71	1.72	1.52	1.50	2.41	2.83	2.76	10.66
	H ₂ -Mapping	3.05	2.51	2.14	1.64	1.98	3.06	3.09	3.35	21.94
	PIN-SLAM	3.46	2.90	2.89	1.96	2.58	3.54	3.34	3.77	16.71
	HIO-SDF	4.50	3.63	3.51	3.26	3.87	4.09	4.50	4.17	72.86
	Voxblox	3.69	2.16	2.31	1.71	2.07	3.52	4.37	4.38	21.30
Completion Ratio [$< \delta$] % ↑	OREN	93.32	96.06	95.66	97.13	96.51	92.44	89.52	90.67	94.20
	H ₂ -Mapping	90.79	91.87	92.87	94.75	92.11	89.19	88.45	87.88	61.58
	PIN-SLAM	88.84	90.30	89.71	92.96	89.34	86.72	86.64	86.36	72.83
	HIO-SDF	78.41	83.04	84.68	84.83	79.82	79.20	78.65	80.30	10.05
	Voxblox	87.09	91.26	90.41	91.92	89.85	84.33	80.21	83.06	60.31
Recall [$< \delta$] % ↑	OREN	92.87	95.84	95.48	96.81	96.21	92.26	88.49	90.51	93.99
	H ₂ -Mapping	91.53	92.46	93.32	95.00	92.65	90.21	89.56	89.14	57.96
	PIN-SLAM	89.83	91.03	90.55	93.29	90.22	88.31	87.87	87.82	70.40
	HIO-SDF	79.93	84.00	85.53	85.75	81.44	80.31	80.52	81.74	4.72
	Voxblox	88.17	91.81	90.79	92.18	90.62	85.67	82.21	84.94	56.64
Precision [$< \delta$] % ↑	OREN	87.04	90.77	91.70	86.90	88.69	90.14	80.59	89.00	90.69
	H ₂ -Mapping	99.57	99.78	99.59	99.65	99.44	99.66	99.08	99.52	52.97
	PIN-SLAM	98.57	98.39	98.59	97.95	98.40	98.06	96.83	98.39	<u>64.63</u>
	HIO-SDF	85.86	89.04	90.54	91.24	88.55	84.83	88.24	88.15	4.46
	Voxblox	96.18	97.94	94.56	95.31	98.13	93.69	91.44	95.55	51.84
F1 Score [$< \delta$] % ↑	OREN	89.86	93.24	93.55	91.59	92.29	91.19	84.36	89.75	92.31
	H ₂ -Mapping	95.38	95.98	96.35	97.27	95.92	94.70	94.08	94.05	55.35
	PIN-SLAM	93.00	94.57	94.40	95.57	94.13	92.93	92.13	92.81	67.39
	HIO-SDF	82.83	86.45	87.96	88.41	84.85	82.51	84.20	84.82	4.59
	Voxblox	92.00	94.77	92.64	93.72	94.23	89.50	86.58	89.94	54.14
Chamfer-L1 Distance [cm] ↓	OREN	2.58	1.96	1.85	2.07	1.85	2.35	3.26	2.60	9.36
	H ₂ -Mapping	2.29	1.86	1.73	1.83	1.49	2.24	2.39	2.46	28.4
	PIN-SLAM	2.56	2.13	2.19	1.70	1.99	2.57	2.60	2.73	21.64
	HIO-SDF	3.79	3.24	3.14	2.93	3.33	3.79	3.90	3.60	422.29
	Voxblox	2.72	1.65	1.90	1.47	1.44	2.60	3.26	3.02	23.37
Accuracy [cm] ↓	OREN	2.60	2.21	1.98	2.62	2.20	2.29	3.69	2.44	8.07
	H ₂ -Mapping	1.54	1.21	1.33	1.22	1.01	1.43	1.69	1.58	34.86
	PIN-SLAM	1.66	1.37	1.49	1.45	1.31	1.61	1.87	1.70	26.58
	HIO-SDF	3.09	2.85	2.78	2.62	2.79	1.49	3.31	3.03	771.72
	Voxblox	1.74	1.14	1.50	<u>1.22</u>	0.81	1.68	2.16	1.67	<u>25.44</u>

TABLE II: SDF reconstruction metrics on the Replica dataset [28] and the Newer College dataset [29].

Metric	Region	Method	room 0	room 1	room 2	office 0	office 1	office 2	office 3	office 4	Newer College	
SDF MAE [cm] ↓	All	OREN	2.15	2.13	2.26	1.93	2.16	2.32	2.65	2.36	56.00	
		HIO-SDF	3.27	2.79	39.98	2.60	2.72	3.34	3.40	3.39	301.56	
		Voxblox	<u>3.13</u>	2.54	2.73	3.02	2.73	3.47	3.74	3.08	62.92	
	Near	OREN	1.67	1.26	1.82	1.30	1.43	1.84	2.15	2.11	16.96	
		HIO-SDF	3.45	2.90	28.32	2.57	3.00	3.36	3.52	3.62	58.34	
		H ₂ -Mapping	6.13	6.01	5.54	5.88	5.75	5.81	5.99	6.19	9.60	
		PIN-SLAM	4.65	8.10	8.06	8.06	8.15	8.11	8.13	8.07	22.64	
		Voxblox	<u>2.78</u>	<u>2.15</u>	<u>2.23</u>	<u>2.20</u>	<u>2.17</u>	<u>2.85</u>	<u>3.41</u>	<u>2.96</u>	19.63	
	Far	OREN	2.39	2.66	2.49	2.33	2.70	2.58	2.98	2.49	59.62	
		HIO-SDF	3.12	2.69	50.22	2.62	2.39	3.33	3.30	3.20	324.12	
		Voxblox	3.37	2.85	3.08	3.71	3.27	3.92	4.00	3.16	67.32	
	Grad. MAE [rad] ↓	All	OREN	0.153	0.157	0.160	0.160	0.174	0.162	0.158	0.154	0.184
			HIO-SDF	0.924	1.315	1.534	1.272	1.101	1.340	1.326	1.256	1.529
			Voxblox	0.230	0.195	0.222	0.251	0.190	0.271	0.279	0.232	0.777
		Near	OREN	0.120	0.116	0.117	0.128	0.129	0.117	0.130	0.117	0.156
H ₂ -Mapping			1.076	1.095	1.043	1.095	1.081	1.085	1.081	1.096	1.379	
PIN-SLAM			0.870	1.566	1.571	1.569	1.530	1.578	1.550	1.572	1.533	
HIO-SDF			0.975	1.301	1.560	1.281	1.115	1.324	1.331	1.257	1.565	
Voxblox			<u>0.282</u>	<u>0.183</u>	0.204	0.259	0.179	0.301	0.336	0.299	1.435	
Far		OREN	0.165	0.177	0.176	0.175	0.199	0.180	0.171	0.167	0.198	
		HIO-SDF	0.884	1.328	1.512	1.262	1.084	1.356	1.325	1.255	1.526	
		Voxblox	0.208	<u>0.201</u>	0.230	0.246	0.198	0.257	0.256	0.205	<u>0.720</u>	
SDF Valid Ratio [%] ↑		H ₂ -Mapping	<u>16.05</u>	16.62	18.15	17.46	20.30	16.84	17.05	15.43	28.22	
		PIN-SLAM	25.01	<u>6.47</u>	<u>5.82</u>	<u>7.86</u>	8.63	<u>7.09</u>	<u>7.19</u>	<u>5.67</u>	47.19	

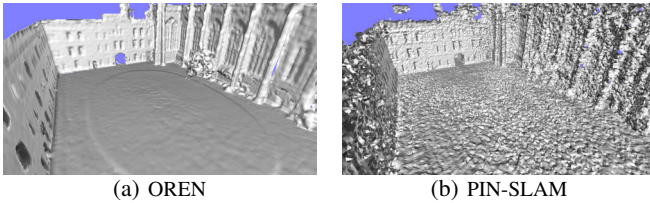


Fig. 6: Comparison of mesh reconstruction using OREN versus PIN-SLAM [2] on the Newer College dataset [29].

TABLE III: Runtime comparison on Replica room 0 [28].

	OREN	H ₂ -Mapping	PIN-SLAM	HIO-SDF	Voxblox
FPS	13.96	<u>12.36</u>	8.43	1.99	0.87

TABLE IV: Ablation study results on Replica room 1 [28].

Metric		OREN	Prior Only	Sparse Octree	w/o Grad. Aug.	w/o Proj. Loss
F1 Score %↑		93.33	<u>93.07</u>	93.01	92.09	89.22
Precision %↑		90.98	<u>90.48</u>	90.45	88.61	84.04
Recall %↑		95.79	<u>95.82</u>	95.71	95.86	95.07
Completion Ratio %↑		96.00	<u>96.05</u>	95.95	96.17	95.65
Completion [cm]↓		1.705	1.709	1.717	1.709	1.775
Accuracy [cm]↓		2.201	2.234	<u>2.228</u>	2.441	3.045
Chamfer Distance ↓		1.953	1.971	1.972	2.075	2.410
SDF All		1.862	2.185	<u>2.090</u>	2.208	18.688
MAE Near [cm]↓		1.203	1.320	<u>1.278</u>	1.612	2.303
MAE Far [cm]↓		2.468	2.981	2.837	<u>2.757</u>	33.756
Grad. All		0.1527	<u>0.1529</u>	0.1626	0.1661	0.1713
MAE Near [rad]↓		<u>0.1166</u>	0.1107	0.1229	0.1298	0.1328
MAE Far [rad]↓		0.1770	<u>0.1823</u>	0.1895	0.1890	0.2519

octree and interpolation without gradient augmentation. The neural network residual improves both the mesh reconstruction quality and the SDF prediction accuracy, as shown in Table IV. When trained with a sparse octree instead, our method performs worse due to more discontinuities caused by the sparsity. Without gradient augmentation, the SDF priors become worse, leading to worse performance. We also train our model without the projection loss, which shows significantly worse metrics because the projection loss provides important guidance about the SDF scale and gradient direction.

VI. CONCLUSION

This paper developed OREN, an online hybrid method for globally accurate SDFs estimation from streaming point cloud data at large scales. Our method combines an explicit octree prior with an implicit neural residual. Our evaluation shows that this combination achieves more accurate and efficient SDF reconstruction than than state-of-the-art methods. Future work will focus on utilizing OREN in robot localization, navigation, and manipulation.

REFERENCES

- [1] 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,” in *Robotics: Science and Systems (RSS)*, 2022.
- [2] Y. Pan, X. Zhong, L. Wiesmann, T. Posewsky, J. Behley, and C. Stachniss, “PIN-SLAM: LiDAR SLAM Using a Point-Based Implicit Neural Representation for Achieving Global Map Consistency,” *IEEE Transactions on Robotics*, 2024.
- [3] G. Chou, Y. Bahat, and F. Heide, “Diffusion-SDF: Conditional Generative Modeling of Signed Distance Functions,” in *ICCV*, 2023.
- [4] Y. Wang, Q. Han, M. Habermann, K. Daniilidis, C. Theobalt, and L. Liu, “NeuS2: Fast Learning of Neural Implicit Surfaces for Multi-view Reconstruction,” in *ICCV*, 2023.
- [5] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, “Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-board MAV planning,” in *IROS*, 2017.

- [6] K. Long, Y. Yi, Z. Dai, S. Herbert, J. Cortés, and N. Atanasov, “Sensor-based Distributionally Robust Control for Safe Robot Navigation in Dynamic Environments,” *The International Journal of Robotics Research*, 2025.
- [7] P. Liu, K. Zhang, D. Tateo, S. Jauhari, J. Peters, and G. Chaltatzaki, “Regularized Deep Signed Distance Fields for Reactive Motion Generation,” in *IROS*, 2022.
- [8] Y. Li, X. Chi, A. Razmjoo, and S. Calinon, “Configuration Space Distance Fields for Manipulation Planning,” 2024.
- [9] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, “KinectFusion: Real-time Dense Surface Mapping and Tracking,” in *IEEE International Symposium on Mixed and Augmented Reality*, 2011.
- [10] B. Lee, C. Zhang, Z. Huang, and D. D. Lee, “Online Continuous Mapping using Gaussian Process Implicit Surfaces,” in *ICRA*, 2019.
- [11] L. Wu, C. Le Gentil, and T. Vidal-Calleja, “VDB-GPDF: Online Gaussian Process Distance Field with VDB Structure,” *IEEE Robotics and Automation Letters*, 2025.
- [12] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation,” in *CVPR*, 2019.
- [13] 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,” in *Conference on Neural Information Processing Systems*, 2021.
- [14] C. Jiang, H. Zhang, P. Liu, Z. Yu, H. Cheng, B. Zhou, and S. Shen, “H2-Mapping: Real-time Dense Mapping Using Hierarchical Hybrid Representation,” *IEEE Robotics and Automation Letters*, 2023.
- [15] C. Sommer, L. Sang, D. Schubert, and D. Cremers, “Gradient-SDF: A Semi-Implicit Surface Representation for 3D Reconstruction,” in *IEEE/CVF International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [16] B. Curless and M. Levoy, “A Volumetric Method for Building Complex Models from Range Images,” in *SIGGRAPH*, 1996.
- [17] O. Kähler, V. Adrian Prisacariu, C. Yuheng Ren, X. Sun, P. Torr, and D. Murray, “Very High Frame Rate Volumetric Integration of Depth Images on Mobile Devices,” *IEEE Transactions on Visualization and Computer Graphics*, 2015.
- [18] L. Han, F. Gao, B. Zhou, and S. Shen, “FIESTA: Fast Incremental Euclidean Distance Fields for Online Motion Planning of Aerial Robots,” in *IROS*, 2019.
- [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 *IROS*, 2022.
- [20] A. Millane, H. Oleynikova, E. Wirbel, R. Steiner, V. Ramasamy, D. Tingdahl, and R. Siegwart, “nvblox: GPU-Accelerated Incremental Signed Distance Field Mapping,” in *ICRA*, 2024.
- [21] L. Wu, K. M. B. Lee, L. Liu, and T. Vidal-Calleja, “Faithful Euclidean Distance Field From Log-Gaussian Process Implicit Surfaces,” *IEEE Robotics and Automation Letters*, 2021.
- [22] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis,” in *ECCV*, 2020.
- [23] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, “Implicit Geometric Regularization for Learning Shapes,” in *International Conference on Machine Learning*, 2020.
- [24] 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 *CVPR*, 2021.
- [25] Z. Wang, C. Wang, T. Yoshino, S. Tao, Z. Fu, and T.-M. Li, “HotSpot: Signed Distance Function Optimization with an Asymptotically Sufficient Condition,” in *CVPR*, 2025.
- [26] J. Wang, Y. Deng, Y. Yang, and Y. Yue, “LCP-Fusion: A Neural Implicit SLAM with Enhanced Local Constraints and Computable Prior,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.
- [27] V. Vasilopoulos, S. Garg, J. Huh, B. Lee, and V. Isler, “HIO-SDF: Hierarchical Incremental Online Signed Distance Fields,” in *ICRA*, 2024.
- [28] J. Straub, T. Whelan *et al.*, “The Replica Dataset: A Digital Replica of Indoor Spaces,” *arXiv preprint arXiv:1906.05797*, 2019.
- [29] L. Zhang, M. Camurri, D. Wisth, and M. Fallon, “Multi-Camera LiDAR Inertial Extension to the Newer College Dataset,” 2021.