SNH-SLAM: Implicit Dense SLAM based on Scalable Neural-Hash Representation

Xiong Li, Zhenyu Wen, *Senior Member IEEE*, Zhanshuo Dong, Haoran Duan, Tianrun Chen, Zhen Hong

***Abstract*—We present SNH-SLAM, a novel expandable dense neural simultaneous localization and mapping (SLAM) method that constructs a neural field in real-time based on run-time observation. To reach this challenging goal without any scene prior, we utilize instant depth supervision to drive the extension of planar convex hulls, where a single hash table maintains multi-level feature units embedded in the planar convex hulls. This design facilitates high-fidelity, hole-free, and low-memory map reconstruction while adding only a tiny time burden to the training process. Our approach performs mapping by minimizing both RGBD-based re-rendering loss and Truncated Signed Distance Field (TSDF) loss. In addition, for camera tracking, our optimization strategy allows SNH-SLAM to converge faster on the pose estimation and maintain robustness. We evaluate our method on common benchmarks and compare it with existing dense neural RGB-D SLAM methods. The evaluation results show the competitiveness of the SNH-SLAM in tracking accuracy, reconstruction quality, memory usage, and frame processing speed. Project page: https://xiaoshumiao123.github.io.**

***Index Terms*—planar convex hulls, hash table, hole-free, RGBD-SLAM.**

I. Introduction

DENSE visual simultaneous localization and mapping (SLAM) aims to estimate the motion of the camera and construct a high-resolution 3D map in real-time from a continuous stream of visual data. As a core technology for applications such as robotics, autonomous driving, and augmented reality, it has remained a focal point of research in 3D computer vision.

In recent years, the emergence of NERF [1] has provided new insights for dense visual SLAM. NERF can learn the neural implicit representation of a scene and render scenes from novel viewpoints. Thus, researchers have explored using neural implicit representations as the underlying scene representation for dense SLAM and developed NeRF-based SLAM systems [2]–[7]. Compared to traditional methods, such approaches exhibit significant advantages in mapping performance and efficiency. Introducing neural implicit functions allows for describing three-dimensional space with fewer parameters. More significantly, these methods compensate for the shortcomings of traditional dense SLAM to some extent by performing smooth interpolation of unobserved regions within the scene, ultimately achieving non-holey, high-precision reconstruction.

While NERF-based dense SLAM methods produce moderately robust and complete dense maps, they rely on scene boundary priors to pre-allocate feature grids of limited capacity, thereby imposing fundamental limitations for open-world deployment. To solve this issue, Vox-Fusion [8] employed an octree-based incremental approach to create sparse voxel grids, integrating neural implicit embeddings to achieve more detailed reconstructions without boundary priors. However, the voxel grids are only distributed near the surface points with valid depth measurements and do not cover the occluded area. This allocation method, commonly observed in traditional SLAM systems [9]–[12], results in the loss of hole-filling capability. Similarly, while Point-SLAM [13] dynamically generates surface neural point clouds for more flexible memory utilization, it still fails to bridge this gap. Additionally, for the NERF-based SLAM methods, the parameters are optimized by repeating the process of sampling sparse rays and minimizing the loss of the rays. Unlike traditional dense SLAM methods, neural scene representations struggle with a stark trade-off between real-time performance and accuracy in run-time pose estimation. Particularly, when boundary priors are canceled and neural map units are dynamically expanded based on estimated poses, it may introduce spatial drift that exacerbates this phenomenon.

In this work, we aspire to apply neural SLAM to unknown environments devoid of boundary priors, while preserving the advantages that neural scene representations offer in completing unobservable regions. To achieve this, we introduce SNH-SLAM, a hash-based neural RGBD SLAM solution that dynamically expands neural scene representation units and enables hole-filling. Our key idea is to leverage real-time depth sensing to supervise the expansion of unstructured planes managed by a hash table, allowing for scalability and low-memory compatibility. Unlike methods that encode only surfaces [8], [13], we extend the encoding scope to the unobservable areas around the surface using planar convex hulls, enabling the reconstruction of more comprehensive maps while maintaining minimal memory consumption. For points within the domain governed by the planes, SNH-SLAM utilizes hash lookup to swiftly access corresponding neighboring features across different planes and obtains their feature embedding through interpolation. After being decoded by Multi-layer Perceptrons (MLPs), the feature embeddings produce predicted color and geometry. Through a combination of selective ray sampling and reduction of discrepancies between predicted

Xiong Li, Zhenyu Wen, Zhanshuo Dong, and Zhen Hong are with the Institute of Cyberspace Security and College of Information Engineering, Zhejiang University of Technology, China. (E-mail: lx.3958@gmail.com, wenluke427@gmail.com, dzhanshuo7@gmail.com, zhong1983@zjut.edu.cn)
Haoran Duan is with the Department of Computer Science, Durham University, UK. (E-mail: haoran.duan@ieee.org)
Tianrun Chen is with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China, and also with KOKONI, Moxin (Huzhou) Technology Company, Ltd., Huzhou 313200, China (E-mail: tianrun.chen@zju.edu.cn).
Corresponding author: Zhenyu Wen

and actual observations, tracking and mapping are alternately performed. For the tracking, previous [2], [3], [5]–[8] adhere to the NeRF paradigm, performing a full rerendering pipeline to achieve rigid frame-to-model alignment for enhanced robustness, which imposes a consistent computational burden and leads to inefficiency. To improve the system's efficiency, we introduce customized inter-frame constraints as a substitute for rerendering supervision. This approach enables us to sample only near the surface along the rays and decode the geometry selectively, thereby maintaining robust camera tracking with reduced computational demand. Our method's performance is rigorously assessed using various indoor RGBD datasets, and the experimental results demonstrate its superiority over existing state-of-the-art (SOTA) systems. In summary, our contributions encompass:

- We introduce SNH-SLAM, a novel dense RGBD SLAM system that leverages a single hash table for the efficient expansion and access of multi-level scene feature units, permitting real-time, high-quality reconstruction without any scene prior knowledge.
- We integrate planar convex hulls for the dynamic allocation of feature units, which effectively enhances the performance of neural scene representations in filling holes while maintaining low memory consumption.
- We abandon re-rendering in favor of jointly supervising camera tracking using surface geometry constraints and inter-frame appearance constraints. This strategy significantly improves the overall efficiency of the system while maintaining robust tracking.
- We conduct comprehensive evaluations across multiple datasets, confirming the competitive performance of our method in various aspects, including tracking, mapping, and efficiency.

II. Related Work

Dense Visual SLAM. DTAM [14], an early representative of dense SLAM, utilizes full-pixel photometric consistency constraints to track camera poses and reconstruct detailed environments. However, its application has often been confined to small-scale spaces due to significant storage and computational costs. Concurrently, the advent of commodity RGB-D cameras such as the Microsoft Kinect rapidly propelled depth-measurement-based dense SLAM to the forefront. KinectFusion [15] employs the Truncated Signed Distance Function (TSDF) to represent 3D models and utilizes iterative closest point (ICP) for frame-to-model tracking. It was the pioneering system to achieve real-time dense reconstruction in indoor environments using GPU acceleration. Subsequently, researchers made various improvements to KinectFusion, including optimizing memory usage [16], [17], implementing loop closure [18], adapting to dynamic environments [19], [20], and enhancing tracking performance [21]. In recent years, inspired by deep learning, some studies [22]–[28] extend real-time dense reconstruction to low-cost sensors such as monocular and binocular, and further improve the robustness of dense SLAM. Despite these methods showing encouraging results, they still follow the traditional scene representations and do not have predictive power for unobserved regions.

Neural Implicit SLAM. Recently, NERF [1] has exhibited remarkable potential in tasks involving pose estimation [29]–[32] and offline 3D reconstruction [33]–[36]. Encouraged by these advancements, researchers have extended its application to SLAM. iMAP [2] introduced MLP as the underlying scene representation for dense SLAM, which constructs NERF in real-time through RGB-D sequences and implements camera tracking. Leveraging the inherent continuity of MLP, it achieves smooth and plausible fill-ins for occluded regions. However, constrained by the limited capacity of a single MLP, imap shows severe forgetting in larger scenes. To address this issue, NICE-SLAM [3] combines MLPs with explicit feature grids, further enhancing scene representation. Continuing this hybrid explicit-implicit representation paradigm, subsequent efforts [4]–[7] primarily focus on optimizing memory efficiency, reconstruction performance, and real-time capabilities. It's noteworthy that these methods pre-define finite feature grids based on given scene boundaries, which restricts their adaptability to unknown scenes.

The most relevant work to our method is VoxFusion [8]. It implements dynamic allocation of sparse feature voxels based on an octree structure and utilizes neural radiance fields for photometrically accurate reconstruction. In contrast to methods like NICE-SLAM [8], Vox-Fusion [8] supports dynamic expansion and marks the first achievement of neural dense SLAM without boundary priors. However, the allocation of voxels strictly follows the spatial distribution of the point cloud mapped by depth, causing the radiation field to lose the predictive ability of those areas with unknown depth (i.e., occluded areas). Similarly, Point-SLAM [13] anchors neural features in dynamically created point clouds close to the surface, and is subject to the same limitation. In order to preserve the predictive performance and be scalable, we propose a neural hash representation in this work. This representation utilizes instantaneous depth to supervise the boundary and performs real-time expansion of multi-level latent embeddings through a single hash table. This dynamic representation allows us to maintain robust camera tracking in unknown scenes while recovering dense and hole-free scene geometry.

III. Method

An overview of SNH-SLAM is shown in Fig 1. SNH-SLAM takes a continuous RGB-D sequence with known camera intrinsic parameters as the input. The system aims to concurrently maintain two processes: the *mapping process* is responsible for dynamically creating and optimizing a neural map, i.e. the proposed scalable neural hash representation. Meanwhile, the *tracking process* estimates the 6-DOF camera pose of the current input frame based on the predicted map. The tracking and mapping are performed alternatively in the proposed system.

Once the system is started, the first frame is used to initialize the scene representation. After that, each frame achieves the pose estimation by minimizing the loss on the sparsely sampled rays. At regular intervals, defined by a set number of frames, we execute a mapping process to refresh the scene's representation. In our method, the scene is represented through

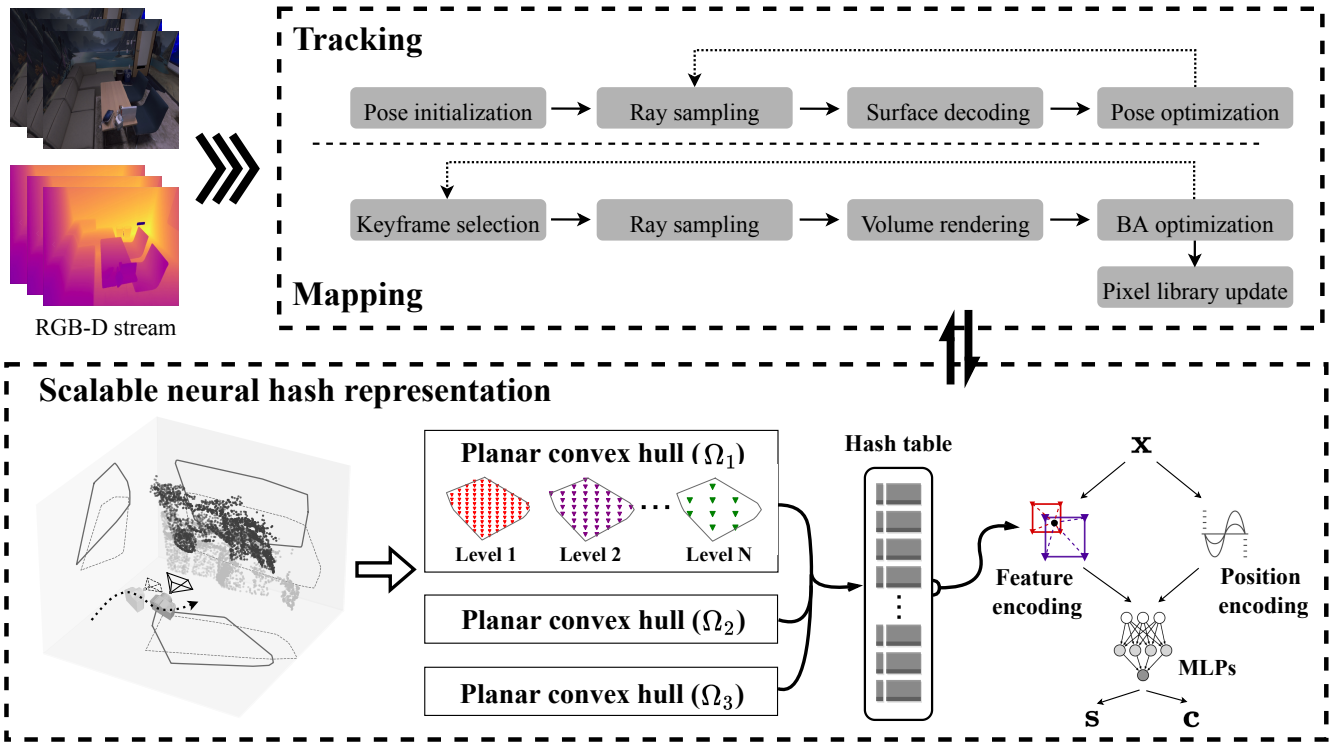


Fig. 1. **System Overview.** 1) Scalable neural hash representation: for each mapping frame, the multi-scale grid features are partitioned based on planar convex hulls and inserted into a hash table. While decoding the RGB and TSDF values for a given spatial coordinate using MLPs, the input feature vector is obtained by joint feature coding and position coding. 2) Tacking: keep the map fixed and sample rays only on the current frame to optimize the camera pose. 3) Mapping: Sample rays from selected global keyframes and jointly optimize the scene representation along with the poses of these frames through bundle adjustment.

latent embeddings, controlled by a hash table and MLPs. During each mapping phase, we initially expand the latent embeddings to ensure they encompass newly explored areas. Following this, we conduct a joint optimization of the complete scene representation and the positions of chosen keyframes using global bundle adjustment (BA).

A. Scalable Neural Hash Representation

We first introduce our neural scene representation that anchors the latent embeddings in multi-level unstructured planes and manages them by a single hash map. By encoding latent embedding and MLP decoding, any spatial coordinate $\mathbf{x}$ within the jurisdiction of the current planes can be mapped to truncated signed distance (TSDF) $\mathbf{s}$ and color $\mathbf{c}$.

Plane-based Perception. For single-frame input $\{\mathbf{I_t}, \mathbf{D_t}\}$, we define the viewing frustum between the camera's optical center and the plane at the maximum depth currently captured as the current perception space. We anticipate that the scene representation will encompass the full range of perceptual space, thus allowing it to address any location within that space, even those areas that are obscured. A simple approach is to fill the entire perception space with latent embeddings in voxel form. However, it causes cubic memory growth resulting in limited applications. Inspired by [6], [37], we register the perceptual space to planes to achieve dimensionality reduction for memory savings. It is noteworthy that the straightforward mapping from 3D to 2D often blurs and amplifies boundaries, rendering it inadequate in accurately guiding the creation of potential embeddings on the planes, thus leading to memory redundancy. To address this issue, we introduce planar convex hulls to appropriately define the editable bounds of potential embeddings. Specifically, we first recover the point cloud from the input frame, then merge it with the camera's optical center o_t and project them onto three orthogonal planes.

$$V_\varsigma = \mathcal{P}_{(\varsigma)}\left(\left[\Pi^{-1}\left(\mathbf{D}_t, \mathbf{T}_t\right); o_t\right]\right), \ \varsigma \in \{1,2,3\} \tag{1}$$

where Π^{-1} denotes the computational function that obtains the 3D coordinates from the given depth $\mathbf{D}_t$ and estimated pose $\mathbf{T}_t$, and $\mathcal{P}_{(\varsigma)}$ denotes the projection operation from 3D points to the three planes. V_ς are the projected points on the planes. Based on these projected points, we employ the classic Quickhull algorithm [38] to create the planar convex hulls as formulated below.

$$\Omega_\varsigma = \mu \cdot ConvexHull\left(V_\varsigma\right) \tag{2}$$

where Ω_ς denotes the planar convex hulls. μ (>1) is a scaling factor, which is used to scale up the convex hull proportionally.

Hierarchical Embeddings. To estimate more detailed scene appearance and geometry, we partition multi-level regular grids $\left\{X_\varsigma^l\right\}_{l=1}^{L} \subset \Omega_\varsigma$ within the convex hulls and construct latent embeddings $\left\{\varphi_\varsigma^l\right\}_{l=1}^{L}$ to combine features at different frequencies. To set the size of different level grids, we imitate the multi-resolution rule in [39] as illustrated below.

$$z_l := \left\lfloor z_{min} b^l \right\rfloor, \ b := exp\left(\frac{ln\ z_{max} - ln\ z_{min}}{L-1}\right) \tag{3}$$

z_{min} and z_{max} correspond to the minimum and maximum size of the grids, respectively.

To address the frequent updates and lookups of latent embeddings during the tracking and mapping processes, we consider hash table [40] as a suitable data structure. This is because the hash table supports fast retrieval and insertion operations. Furthermore, our method only utilizes a single hash table avoiding overheads caused by frequent switching between hash tables. To do this, we use the grids X_ς^l with tagged bits $\eta := l \cdot \varsigma$ and their associated latent embeddings φ_ς^l as the keys and values of the hash table. The tagged bits η make the grids unique to avoid key collisions. Before each formal mapping, we create latent embeddings of the newly explored region and insert them into the hash table.

Scene Codec. For a certain point $\mathbf{x}$ in space, we obtain its feature vector by querying the latent embeddings in the hash table:

$$\Upsilon(\mathbf{x}) = \left[\sum_{\varsigma=1}^{3} \varphi_\varsigma^1(\mathbf{x}); \sum_{\varsigma=1}^{3} \varphi_\varsigma^2(\mathbf{x}); ...; \sum_{\varsigma=1}^{3} \varphi_\varsigma^L(\mathbf{x})\right] \quad (4)$$

where $\varphi_\varsigma^l(\mathbf{x})$ denotes that the latent embedding φ_ς^l is bilinearly interpolated at the planar projection point of $\mathbf{x}$.

After that, TSDF $\mathbf{s}$ and color $\mathbf{c}$ are decoded by a mini-MLP that contains only two hidden layers of size 32:

$$(\mathbf{s}, \rho) = f_g(\gamma(\mathbf{x}), \Upsilon(\mathbf{x})), \ \mathbf{c} = f_c(\gamma(\mathbf{x}), \rho) \quad (5)$$

where ρ denotes a geometric feature vector. $\gamma(\cdot)$ is a positional encoding function recommend in [41].

B. Rendering

To utilize the input RGB-D images as directly supervised signals for optimizing the neural map, we render depth and color by integrating the prediction of TSDF $\mathbf{s}$ and color $\mathbf{c}$ at sample points on rays. Given the camera's intrinsic parameters, rays can be determined by a camera pose and pixel coordinates. For each selected ray, we uniformly sample N_{base} samples in the range of $[0.01D, 1.2D]$ and N_{sur} near-surface samples in the range of $[D - T_r, D + T_r]$, where T_r is the truncation distance. After obtaining the prediction $\{(s_i, c_i) \mid i \in \{1, 2, ..., N\}\}$ of the $N := N_{base} + N_{sur}$ samples, we perform the rendering following the method proposed in [42] as illustrated below.

$$\omega_i = \sigma\left(\frac{s_i}{T_r}\right)\sigma\left(-\frac{s_i}{T_r}\right) \quad (6)$$

$$\hat{C} = \frac{1}{\sum_{i=1}^{N} w_i}\sum_{i=1}^{N} w_i c_i, \quad \hat{D} = \frac{1}{\sum_{i=1}^{N} w_i}\sum_{i=1}^{N} w_i d_i \quad (7)$$

w_i denotes the weights of the sample points, and d_i denotes the depths of the sample points under their corresponding camera poses.

C. Mapping

In order to optimize the scene representation mentioned in section III-A, we perform the mapping operation to each q frame. During the mapping process, we sample rays from both the current frame and selected keyframes and jointly optimize the scene representation and the poses of these frames using global bundle adjustment.

Keyframe Policy. In our approach, we follow Co-SLAM [5] to maintain a keyframe pixel library, which adds a portion of the mapped frame to the pixels at the end of each mapping. For each mapping, Co-SLAM randomly samples a fixed number of rays from the library and the current frame for global optimization to prevent forgetting. However, as the system continues to run, the amount of samples allocated to the current frame gradually decreases causing a drastic degradation of the local mapping capability. To alleviate this problem, we utilize a scalable sliding window and select historical keyframes and local keyframes at a specific ratio to perform ray sampling. Specifically, the length sliding window W is extended as the number of keyframes increases during the running process, and is always maintained at about 10% of the total number of keyframes. The W selected frames consist of 20% neighboring keyframes of the current frame and 80% random selected frames from earlier keyframes. The amount of ray sampling per time is constant at K and is equally distributed to all selected frames.

BA optimization. We perform BA optimization based on rerendering supervision and TSDF supervision of spatial points. The re-rendering loss defines the difference between the sensor readings and our rendered results as illustrated in equation 8,

$$\mathcal{L}_c = \frac{1}{|K|}\sum_{k\in K}\left(\hat{C}_k - C_k\right)^2, \mathcal{L}_d = \frac{1}{|K|}\sum_{k\in K}\left(\hat{D}_k - D_k\right)^2 \quad (8)$$

where C_k and D_k denote the rendered color and depth of the k-th ray, respectively. $\hat{C}_k$, $\hat{D}_k$ are the corresponding sensor readings. Referring to [42], the TSDF loss is used to accurately define the surface of objects. For the points P_k^{fs} located along the sampled ray k and within the free space between the camera center and the surface truncation boundary ($D_k - d_p > T_r$), we expect their TSDF values to be one. Meanwhile, for those near-surface points P_k^{sur} within the truncation region ($|D_k - d_p| \leq T_r$), their TSDF values should be close to the approximation of the ground-truth TSDF value, i.e. $\frac{D_k - d_p}{T_r}$. Ultimately, free space loss and surface loss are defined as:

$$\mathcal{L}_{fs} = \frac{1}{|K|}\sum_{k\in K}\frac{1}{\left|P_k^{fs}\right|}\sum_{p\in P_k^{fs}}(s_p - 1)^2 \quad (9)$$

$$\mathcal{L}_{sur} = \frac{1}{|K|}\sum_{k\in K}\frac{1}{|P_k^{sur}|}\sum_{p\in P_k^{sur}}(d_p + T_r \cdot s_p - D_k)^2 \quad (10)$$

where s_p denotes the TSDF prediction at point p. We combine the above losses to optimize the latent embeddings, the decoders, as well as the camera pose $\{\mathbf{T}_i\}$ of the K selected keyframes:

$$\underset{\{\varphi_\varsigma^l\},\{f_g,f_c\},\{\mathbf{T}_i\}}{\arg\min} (\lambda_c L_c + \lambda_d L_d + \lambda_{fs} L_{fs} + \lambda_{sur} L_{sur}) \quad (11)$$

where $\{\lambda_c, \lambda_d, \lambda_{fs}, \lambda_{sur}\}$ are the weighting coefficients.

D. Tracking

In parallel to the mapping, we sample Q rays each time on the current frame t and then optimize its camera pose $\mathbf{T}_t \in \mathbb{SE}(3)$ to achieve camera tracking. The initial pose is determined by a constant motion model:

$$\mathbf{T}_t = \mathbf{T}_{t-1}\mathbf{T}_{t-2}^{-1}\mathbf{T}_{t-1} \quad (12)$$

Given that tracking requires execution on every frame, its computational efficiency significantly affects the overall system speed. In our approach, we seek to enhance the computation efficiency from the following two aspects: (1) **Reducing Sampling.** Undoubtedly, reducing the number of sampled points along the ray is a direct and effective way to lessen the computational load. From our experience, the influence of surface constraints significantly outweighs that of the free space during the camera pose optimization process. Hence, along each ray, we uniformly sample $\tau \cdot N_{sur}$ points only within the truncation region, where τ is a constant coefficient for adjusting sample density. (2) **Abandoning Re-rendering.** For pose optimization based on neural scene representation, decoding geometry and appearance of sampled points to support re-rendering supervision is relatively time-consuming yet crucial. Particularly, re-rendering color loss serves as a regularization term for TSDF loss, effectively mitigating the influence of degenerate surfaces (e.g., plain walls). To this end, we introduce inter-frame constraints as a replacement, which accelerates computation while preserving robust tracking. Specifically, for the current frame t, we designate the most recently added keyframe as its reference frame and determine the projected pixel $\hat{q}$ on the reference frame corresponding to the sampled pixel $q \in Q$:

$$\hat{q} = \Pi \left(\Pi^{-1} \left(\mathbf{D}_q, \mathbf{T}_t \right), \mathbf{T}_{kf} \right) \tag{13}$$

where Π denotes the computational function for mapping 3D coordinates to pixel coordinates, and $\mathbf{T}_{kf}$ denotes the estimated pose of the reference frame. We optimize the initial pose by combining surface TSDF loss and inter-frame color loss, as illustrated below.

$$\underset{\mathbf{T}_t}{\arg\min} \quad \left(L_{sur} + \frac{\lambda_{ct}}{|Q|} \sum_{q \in Q} |C_q - C_{\hat{q}}| \right) \tag{14}$$

where λ_{ct} is a weighting coefficient.

IV. Experiments

A. Experimental Setup

Datasets. We evaluate SNH-SLAM on three commonly used 3D benchmarks that cover both synthetic scenes and real-world scans. (1) The Replica dataset [43] provides 18 synthesized highly realistic 3D indoor scenes. Additionally, we use the RGB-D sequences collected by Sucar et al. [2] for the replica dataset. (2) The ScanNet dataset [44] comprises multiple RGB-D sequences captured in the real world, as well as ground truth pose obtained via BundleFusion [45]. (3) The TUM-RGBD dataset [46] contains indoor RGB-D sequences recorded by a Microsoft Kinect sensor. Their ground truth trajectory was obtained from a high-accuracy motion-capture system.

Baselines. We compare our method to existing state-of-the-art NeRF-based dense visual SLAM methods, including NICE-SLAM [3], Vox-Fusion [8] and Co-SLAM [5]. NICE-SLAM and Co-SLAM require scene boundary information as prior, while Vox-Fusion and our method perform incremental mapping without any prior. We reproduce the results of Vox-Fusion using their official implementation and report the results as Vox-Fusion*.

Metrics. Following previous work [3], [5], we use both 2D and 3D metrics to evaluate the reconstruction quality, including *Depth L1* [cm], *Accuracy* [cm], *Completion* [cm], and *Completion Ratio* [<5 cm %]. Following Co-SLAM [5], we conduct mesh culling. This process eliminates regions not visible within any camera's field of view and also removes noisy points that, although within the camera's field of view, are situated outside the target scene. In addition, camera tracking accuracy is evaluated by ATE RMSE [46]. Without special instructions, we report the average results of 5 runs.

Implementation Details. Our experiments are conducted on a Ubuntu server equipped with an NVIDIA RTX 3090 GPU. Across all experiments, the iteration number of initialization is fixed to 1500, the window size W of initialization is set to 10, and mapping is set to be performed every 5 frames. We partition the grid into $L = 6$ levels ranging from $z_{min} = 2cm$ to $z_{max} = 32cm$, with the latent embedding dimensions set to 3 for all levels. During the tracking process, the number of randomly selected rays per iteration was set to $Q = 1024$. For mapping, $K = 2048$ rays were used. Note that for the ray sampling, we disregard rays that do not have valid depth measurements. Under the default setting (for Replica [43]), we sample along each ray with $N_{base} = 36$ regular points and $N_{sur} = 12$ near-surface points within the truncation distance $T_r = 6cm$. The coefficient τ is set to 1. The weights during mapping are set to $\lambda_c = 5$, $\lambda_d = 0.1$, $\lambda_{fs} = 10$, and $\lambda_{sur} = 1000$. During tracking, the weight is set to $\lambda_{ct} = 0.001$. For additional detailed settings regarding all datasets, please refer to the supplementary materials.

B. Experimental Results

Evaluation of Tracking. To demonstrate that our framework can handle real-world scenes, we evaluate 9 real-world sequences, 6 from the ScanNet dataset [44] and 3 from the TUM-RGBD dataset [46]. As shown in Table I, our method significantly outperforms Vox-Fusion and even shows competitiveness with closed-bound methods like NICE-SLAM and Co-SLAM. In addition, we report the quantitative analysis of the tracking results on eight synthetic scenes of the Replica dataset [43] in Table II. Despite the significant improvement in tracking performance for all methods, our overall performance demonstrates an advantage(see Avg column).

Evaluation of Mapping. Table III shows the quantitative results of the reconstruction of our method on the Replica dataset [43]. Our method outperforms the baseline approaches on almost all metrics. Qualitatively, Fig 2 shows that our neural hash representation produces more complete scene geometry than Vox-Fusion, and is comparable to the closed-bound methods. Fig 3 further illustrates the reconstruction comparison for the real-world sequences. We find that Vox-Fusion even leads to severely distorted reconstructions, like the carpet in the first scene and the dining table in the third scene. In contrast, our method is able to reconstruct all scene sequences incrementally and accurately. We attribute this to our tracking method and global BA strategy, which lead to more stable pose tracking without large drift even if subject to motion blur or noisy depth detection.

TABLE I

TRACKING RESULTS ON SCANNET [44] AND TUM-RGBD [46]. ATE RMSE [CM] (↓) IS USED AS THE EVALUATION METRIC. FOR NICE-SLAM [3] AND CO-SLAM [5], WE TAKE THE RESULTS FROM THE RESPECTIVE ORIGINAL PAPERS. VOX-FUSION* INDICATES OUR REPRODUCED RESULTS OF VOX-FUSION [8]. OUR APPROACH OUTPERFORMS ALL BASELINES IN TRACKING FOR THE MAJORITY OF SCENES.

	closed-bound	open-bound	ScanNet							TUM-RGBD			
			0000	0059	0106	0169	0181	0207	Avg.	fr1/desk	fr2/xyz	fr3/office	Avg.
NICE-SLAM	√		8.64	12.25	8.09	10.28	12.93	**5.59**	9.63	**2.7**	1.8	3.0	2.5
Co-SLAM	√		7.18	12.29	9.57	**6.62**	13.43	7.13	9.37	**2.7**	1.9	2.6	2.4
Vox-Fusion*		√	15.99	9.16	**8.02**	9.85	16.30	7.73	11.18	N/A	N/A	N/A	N/A
Ours		√	**6.94**	**8.71**	8.42	7.35	**10.44**	9.18	**8.51**	2.8	**1.7**	**2.4**	**2.3**

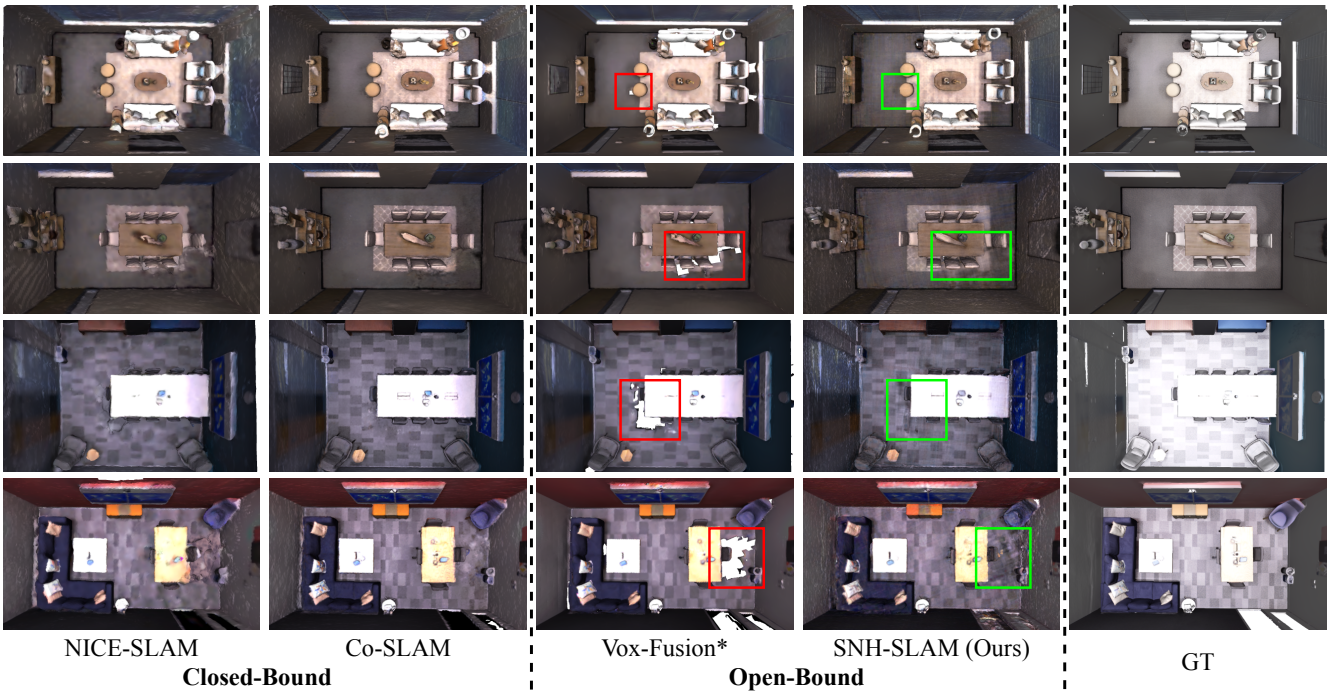


Fig. 2. **Qualitative Reconstruction Results on Replica [43].** All NERF-based SLAM methods except Vox-Fusion restore relatively complete rooms closer to GT. We mark Vox-Fusion missing regions with red boxes, and green boxes correspond to improvements achieved by our method.

TABLE II

TRACKING RESULTS ON REPLICA [43]. ATE RMSE [CM] (↓) IS USED AS THE EVALUATION METRIC. OUR METHOD ROBUSTLY HANDLES EACH SYNTHETIC SEQUENCE, AND ON AVERAGE, ITS TRACKING PERFORMANCE CONSISTENTLY REMAINS OPTIMAL.

	rm-0	rm-1	rm-2	off-0	off-1	off-2	off-3	off-4	Avg.
NICE-SLAM	1.88	1.78	1.90	0.96	0.87	1.70	3.18	2.97	1.91
Co-SLAM	0.62	0.82	1.17	0.54	0.58	2.06	1.56	**0.66**	1.00
Vox-Fusion*	0.58	1.13	0.80	17.8	0.90	**0.88**	**0.70**	0.86	2.96
Ours	**0.54**	**0.49**	**0.78**	**0.44**	**0.56**	1.49	0.98	0.71	**0.75**

TABLE III

QUANTIFIED RECONSTRUCTION RESULTS ON REPLICA [43] (AVERAGE OVER 8 SCENES). THE RESULTS OF NICE-SLAM [3] ARE REPORTED IN CO-SLAM [5]. OUR METHOD OUTPERFORMS ALL OTHER APPROACHES, REGARDLESS OF WHETHER THEY PRE-ALLOCATE FEATURE GRIDS BASED ON BOUNDARY PRIORS.

	Closed-Bound		Open-Bound	
	NICE-SLAM	Co-SLAM	Vox-Fusion*	Ours
Depth L1 ↓	1.90	1.51	3.19	**1.23**
Acc. ↓	2.37	**2.10**	3.26	2.18
Comp. ↓	2.64	2.08	2.64	**1.87**
Comp. Ratio ↑	91.13	93.44	90.86	**95.71**

C. Runtime and Memory Analysis

Besides the evaluation of scene reconstruction and camera tracking, we further analyze the efficiency of the proposed pipeline. We perform runtime and memory tests on Replica [43] and ScanNet [44] datasets to do this. The results are reported in Table IV. The FPS of our method achieves 3 and 12 times the open-bound baseline (i.e. VoxFusion) on Replica [43] and ScanNet [44]. Compared with the closed-bound methods, our method requires extra time on the map expansion before each mapping. The execution time of our method is still less than that of NICE-SLAM and only more than that of Co-SLAM. In terms of memory usage, it is apparent that SNH-SLAM significantly outperforms the closed-bound baselines that allocate feature embeddings over the entire space. Vox-Fusion achieves the best performance by creating feature embeddings only at the surface, but it results in maps with holes. In contrast, our

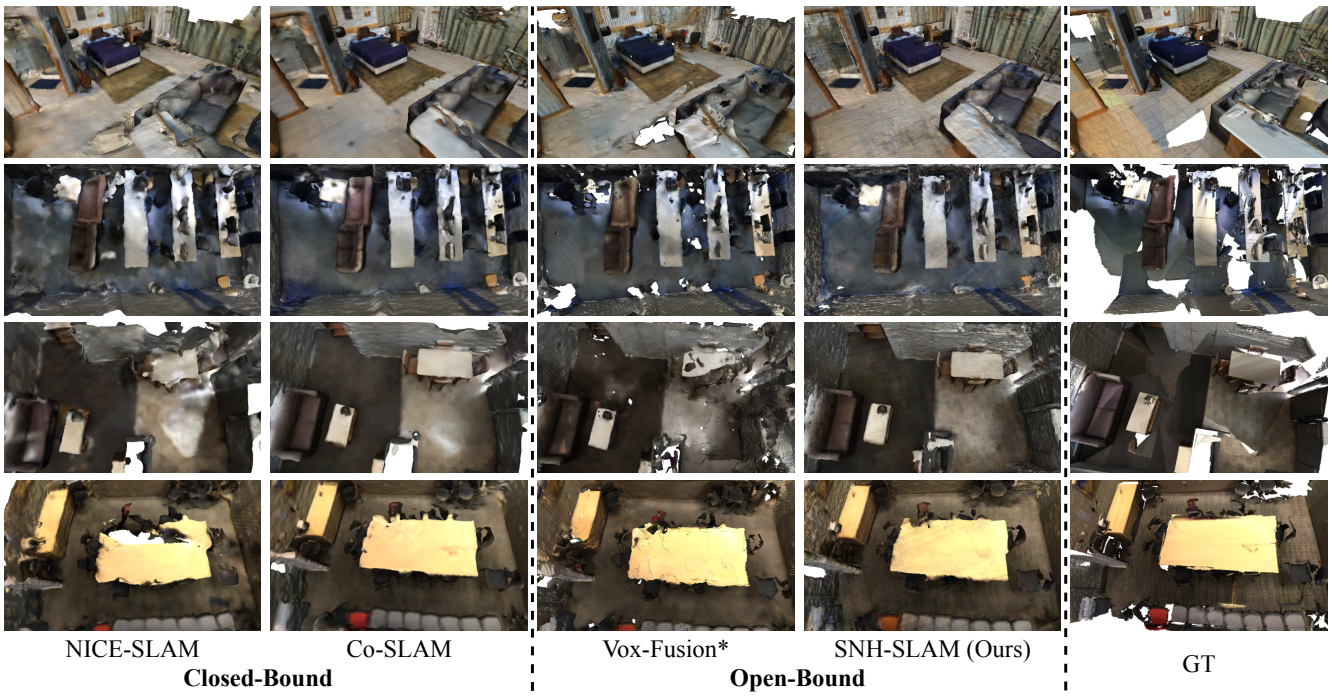


Fig. 3. **Qualitative Reconstruction Results on ScanNet [44].** Our approach enables the recovery of scene geometry with greater detail and global consistency from noisy real-world depth measurements. Furthermore, the holes within the scene are filled through our neural hash representation.

TABLE IV
RUNTIME AND MEMORY USAGE ON REPLICA [43] AND SCANNET [44]. IN PARENTHESIS WE REPORT THE NUMBER OF ITERATIONS. NOTE THAT VOX-FUSION* PERFORMS MAPPING ON EVERY FRAME TO MAINTAIN THE GLOBAL CONSISTENCY OF THE MAP. IN CONTRAST, SIMILAR TO CLOSED-BOUND METHODS, WE PERFORM MAPPING EVERY 5 FRAMES, ENABLING OUR FRAME PROCESSING SPEED TO SIGNIFICANTLY EXCEED THAT OF VOX-FUSION*.

		Closed-Bound		Open-Boud	
		NICE-SLAM	Co-SLAM	Vox-Fusion*	Ours
Replica	Track. [ms] ↓	145(10)	**64(10)**	552(30)	139(15)
	Map. [ms] ↓	4146(60)	**102(10)**	454(10)	524(15)
	FPS ↑	0.95	**10.15**	0.85	2.82
	Mem. [MB] ↓	48.75	5.43	**1.17**	2.23
ScanNet	Track. [ms] ↓	855(50)	**74(10)**	805(30)	97(8)
	Map. [ms] ↓	4570(60)	**222(10)**	691(15)	607(15)
	FPS ↑	0.47	**7.29**	0.31	3.73
	Mem. [MB] ↓	36.53	8.46	**1.09**	2.14

approach achieves more accurate reconstructions and supports hole-filling with only a minor increase in memory consumption.

D. Ablation Study

Tracking scheme. To validate the advantage of the proposed novel tracking scheme in terms of accuracy and efficiency, we compared it with the standard scheme recommended by Co-SLAM [5] and Vox-Fusion [8]. Their tracking follows the rule closely aligned with mapping, concurrently considering TSDF loss and RGB-D re-rendering loss. Additionally, we investigate the tracking performance of our scheme with only surface TSDF constraint. All results are shown in Table V. It is observed that the introduction of inter-frame appearance constraint significantly improves the average ATE RMSE on the Replica [43] dataset. Furthermore, compared to the standard tracking scheme, our scheme achieves an approximate 16% increase in FPS and even holds an advantage in tracking precision.

TABLE V
PERFORMANCE COMPARISON OF DIFFERENT TRACKING SCHEMES ON REPLICA [43]. WE REPORT AVERAGES ACROSS 8 SCENES. CONSIDERING BOTH ACCURACY AND SPEED, OUR METHOD EXHIBITS THE BEST PERFORMANCE.

	Std.	w/o Inter-frame const.	Ours
ATE RMSE ↓	0.77	0.91	**0.75**
FPS ↑	2.43	**3.03**	**2.82**

Bundle Adjustment Policy. We test the performance of our method using different BA strategies and report the results in Table VI. (1) LBA: Referring to NICE-SLAM, select 10 local keyframes, and sample an equal amount of rays for each frame. (2) GBA-Vox: Referring to Vox-Fusion, randomly select 10 frames from all the keyframes and sample an equal amount of rays for each frame. (3) GBA-Co: Reference Co-SLAM to perform random sampling from rays of all keyframes. GBA-Co emphasizes globally consistent mapping, the decay of local mapping power during system running causes a degradation in overall reconstruction performance, which also leads to poorer camera tracking. Meanwhile, both LBA and GBA-Vox suffer from the negative effects of insufficient global constraint. In contrast, our strategy consistently maintains a balance between local and global mapping. Under an equal amount of ray sampling, our strategy exhibits optimal performance.

TABLE VI
ABLATION OF BA STRATEGIES ON THE REPLICA DATASET [43]. WE REPORT AVERAGES ACROSS 8 SCENES. TO ENSURE FAIRNESS, THE NUMBER OF RAYS SAMPLED PER ITERATION IS FIXED AND CONSISTENT ACROSS ALL STRATEGIES (I.E. 2048). OUR PROPOSED STRATEGY PROMOTES THE BEST PERFORMANCE IN BOTH CAMERA TRACKING AND MAP RECONSTRUCTION.

	LBA	GBA-Vox	GBA-Co	Ours
ATE RMSE ↓	0.91	0.83	0.98	**0.75**
Depth L1 ↓	1.35	1.39	1.58	**1.23**
Acc. ↓	2.59	2.33	2.42	**2.18**
Comp. ↓	2.13	2.05	2.34	**1.87**
Comp. Ratio ↑	92.73	93.55	92.14	**95.71**

V. CONCLUSION

We introduce SNH-SLAM, an innovative dense SLAM technique leveraging neural scene representation. This system dynamically constructs a neural map from real-time observations, making it highly effective for scenes without prior data. Our method utilizes instantaneous deep weak supervision to foster the development of planar feature units. In conjunction with implicit decoders, this strategy results in highly precise and comprehensive map reconstructions. To enhance system efficiency, SNH-SLAM employs a singular hash table for all feature unit insertion and lookup operations. Additionally, our approach focuses on rapid camera tracking without relying on re-rendering supervision and incorporates an inter-frame constraint to bolster robustness. Compared to existing open-bound neural SLAM methods, SNH-SLAM shows marked improvements in filling gaps and processing frames more rapidly. Moreover, extensive experimental evaluations reveal that our method outperforms the latest closed-bound solutions in terms of tracking accuracy and reconstruction completeness.

Limitations. Our SLAM approach presents some areas for refinement. Firstly, it currently lacks an explicit loop closure feature, which is useful for correcting cumulative errors over time. Additionally, compared to other neural SLAM methods our method achieves significant runtime improvements, but there is a challenge for use in systems with limited computational capacity, such as embedded devices or mobile robots. This could potentially affect the system's real-time responsiveness, necessitating a careful balance between precision and operational efficiency. To adapt neural SLAM methods for real-time applications in limited computational devices, we will leave it as future work.

REFERENCES

[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[2] 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.

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

[4] H. Li, X. Gu, W. Yuan, L. Yang, Z. Dong, and P. Tan, "Dense rgb slam with neural implicit maps," *arXiv preprint arXiv:2301.08930*, 2023.

[5] H. Wang, J. Wang, and L. Agapito, "Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13 293–13 302.

[6] M. M. Johari, C. Carta, and F. Fleuret, "Eslam: Efficient dense slam system based on hybrid representation of signed distance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 408–17 419.

[7] Y. Zhang, F. Tosi, S. Mattoccia, and M. Poggi, "Go-slam: Global optimization for consistent 3d instant reconstruction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3727–3737.

[8] X. Yang, H. Li, H. Zhai, Y. Ming, Y. Liu, and G. Zhang, "Vox-fusion: Dense tracking and mapping with voxel-based neural implicit representation," in *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2022, pp. 499–507.

[9] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. Davison, and S. Leutenegger, "Mid-fusion: Octree-based object-level multi-instance dynamic slam," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5231–5237.

[10] E. Vespa, N. Funk, P. H. Kelly, and S. Leutenegger, "Adaptive-resolution octree-based volumetric slam," in *2019 International Conference on 3D Vision (3DV)*. IEEE, 2019, pp. 654–662.

[11] 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.

[12] N. Fairfield, G. Kantor, and D. Wettergreen, "Real-time slam with octree evidence grids for exploration in underwater tunnels," *Journal of Field Robotics*, vol. 24, no. 1-2, pp. 03–21, 2007.

[13] E. Sandström, Y. Li, L. Van Gool, and M. R. Oswald, "Point-slam: Dense neural point cloud-based slam," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 18 433–18 444.

[14] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *2011 international conference on computer vision*. IEEE, 2011, pp. 2320–2327.

[15] 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 *2011 10th IEEE international symposium on mixed and augmented reality*. Ieee, 2011, pp. 127–136.

[16] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3d reconstruction at scale using voxel hashing," *ACM Transactions on Graphics (ToG)*, vol. 32, no. 6, pp. 1–11, 2013.

[17] M. Zeng, F. Zhao, J. Zheng, and X. Liu, "A memory-efficient kinectfusion using octree," in *Computational Visual Media: First International Conference, CVM 2012, Beijing, China, November 8-10, 2012. Proceedings*. Springer, 2012, pp. 234–241.

[18] V. A. Prisacariu, O. Kähler, S. Golodetz, M. Sapienza, T. Cavallari, P. H. Torr, and D. W. Murray, "Infinitam v3: A framework for large-scale 3d reconstruction with loop closure," *arXiv preprint arXiv:1708.00783*, 2017.

[19] R. A. Newcombe, D. Fox, and S. M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 343–352.

[20] X. Yang, Z. Yuan, D. Zhu, C. Chi, K. Li, and C. Liao, "Robust and efficient rgb-d slam in dynamic environments," *IEEE Transactions on Multimedia*, vol. 23, pp. 4208–4219, 2020.

[21] Z. Yuan, K. Cheng, J. Tang, and X. Yang, "Rgb-d dso: Direct sparse odometry with rgb-d cameras for indoor scenes," *IEEE Transactions on Multimedia*, vol. 24, pp. 4092–4101, 2021.

[22] Z. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *Advances in neural information processing systems*, vol. 34, pp. 16 558–16 569, 2021.

[23] K. Tateno, F. Tombari, I. Laina, and N. Navab, "Cnn-slam: Real-time dense monocular slam with learned depth prediction," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 6243–6252.

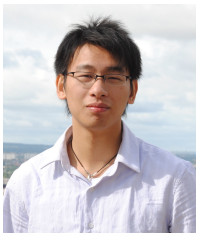
[24] L. Koestler, N. Yang, N. Zeller, and D. Cremers, "Tandem: Tracking and dense mapping in real-time using deep multi-view stereo," in *Conference on Robot Learning*. PMLR, 2022, pp. 34–45.

[25] J. Czarnowski, T. Laidlow, R. Clark, and A. J. Davison, "Deepfactors: Real-time probabilistic dense monocular slam," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 721–728, 2020.

[26] J. Wu, R. Ji, Q. Wang, S. Zhang, X. Sun, Y. Wang, M. Xu, and F. Huang, "Fast monocular depth estimation via side prediction aggregation with

continuous spatial refinement," *IEEE Transactions on Multimedia*, vol. 25, pp. 1204–1216, 2022.
[27] Y. Guo, Y. Wang, L. Wang, Z. Wang, and C. Cheng, "Cvcnet: Learning cost volume compression for efficient stereo matching," *IEEE Transactions on Multimedia*, 2022.
[28] X. Song, H. Hu, L. Liang, W. Shi, G. Xie, X. Lu, and X. Hei, "Unsupervised monocular estimation of depth and visual odometry uusing attention and depth-pose consistency loss," *IEEE Transactions on Multimedia*, 2023.
[29] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, "Barf: Bundle-adjusting neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5741–5751.
[30] Z. Wang, S. Wu, W. Xie, M. Chen, and V. A. Prisacariu, "Nerf–: Neural radiance fields without known camera parameters," *arXiv preprint arXiv:2102.07064*, 2021.
[31] Y. Chen, X. Chen, X. Wang, Q. Zhang, Y. Guo, Y. Shan, and F. Wang, "Local-to-global registration for bundle-adjusting neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 8264–8273.
[32] Y. Jeong, S. Ahn, C. Choy, A. Anandkumar, M. Cho, and J. Park, "Self-calibrating neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5846–5854.
[33] M. Oechsle, S. Peng, and A. Geiger, "Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5589–5599.
[34] 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.
[35] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman, "Volume rendering of neural implicit surfaces," *Advances in Neural Information Processing Systems*, vol. 34, pp. 4805–4815, 2021.
[36] X. Meng, W. Chen, and B. Yang, "Neat: Learning neural implicit surfaces with arbitrary topologies from multi-view images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 248–258.
[37] S. Fridovich-Keil, G. Meanti, F. R. Warburg, B. Recht, and A. Kanazawa, "K-planes: Explicit radiance fields in space, time, and appearance," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 12 479–12 488.
[38] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, pp. 469–483, 1996.
[39] 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.
[40] J. E. Hopcroft, J. D. Ullman, and A. V. Aho, *Data structures and algorithms*. Addison-wesley Boston, MA, USA:, 1983, vol. 175.
[41] T. Müller, B. McWilliams, F. Rousselle, M. Gross, and J. Novák, "Neural importance sampling," *ACM Transactions on Graphics (ToG)*, vol. 38, no. 5, pp. 1–19, 2019.
[42] 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.
[43] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma *et al.*, "The replica dataset: A digital replica of indoor spaces," *arXiv preprint arXiv:1906.05797*, 2019.
[44] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.
[45] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, p. 1, 2017.
[46] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 573–580.

Xiong Li received the B.S. degree from Jiangxi University of Science and Technology, Ganzhou, China, in 2020. He is currently pursuing the Ph.D. degree at the College of Information Engineering, Zhejiang University of Technology, in Hangzhou, China. His current research interests include 3D reconstruction and 3D scene understanding.



Zhenyu Wen (Member, IEEE) is currently a Tenure-Tracked Professor with the Institute of Cyberspace Security, and College of Information Engineering, Zhejiang University of Technology. His current research interests include IoT, crowd sources, AI system, and cloud computing. For his contributions to the area of scalable data management for the Internet of Things, he was awarded the IEEE TCSC Award for Excellence in Scalable Computing (Early Career Researchers) in 2020.

Zhanshuo Dong received the B.S. degree from Changzhou Institute of Technology, Changzhou, China, in 2022. He is currently working toward the Ph.D. degree at the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China. His current research interests include large model applications and unmanned system security.

Haoran Duan (Graduate Student Member, IEEE) received a Distinction M.S. degree in Data Science from Newcastle University UK in 2019. He was also a research student at OpenLab, Newcastle University. He obtained his PhD degree from Durham University, UK, and now he is a postdoc research associate at Networked and Ubiquitous Systems Engineering Group, School of Computing, Newcastle University, working on deep learning applications. His current research interests focus on the applications/theories of deep learning.

Tianrun Chen (Graduate Student Member, IEEE) received the bachelor's degree from the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China, where he is currently working toward the Ph.D. degree with the College of Computer Science and Technology. He is also the founder and technical Director of Moxin (Huzhou) Technology Co., Ltd., Huzhou, China. His research interests include computer vision and its enabling applications.

Zhen Hong received the B.S. degree from Zhejiang University of Technology (China) and University of Tasmania (Australia) in 2006, respectively, and the Ph.D. degree from the Zhejiang University of Technology (ZJUT) in Jan. 2012. He is a full professor with the Institute of Cyberspace Security, and College of Information Engineering, Zhejiang University of Technology, China. Before joining to ZJUT, he was an associate professor with the Faculty of Mechanical Engineering & Automation, Zhejiang Sci-Tech University, China. He has visited at the Sensorweb Lab, Department of Computer Science, Georgia State University in 2011. He also has been at CAP Research Group, School of Electrical & Computer Engineering, Georgia Institute of Technology as a research scholar in 2016 to 2018. His research interests include Internet of things, wireless sensor networks, cyberspace security, and data analytics. He received the first Zhejiang Provincial Young Scientists Title in 2013 and the Zhejiang Provincial New Century 151 Talent Project in 2014. He is a member of IEEE, and senior member of CCF and CAA, and serves on the Youth Committee of Chinese Association of Automation and Blockchain Committee and CCF YOCSEF, respectively.