

# ESLAM: Efficient Dense SLAM System Based on Hybrid Representation of Signed Distance Fields

Mohammad Mahdi Johari  
Idiap Research Institute, EPFL  
mohammad.johari@idiap.ch

Camilla Carta  
ams OSRAM  
camilla.cart@ams-osram.com

François Fleuret  
University of Geneva, EPFL  
francois.fleuret@unige.ch



Figure 1. Our pre-train-free ESLAM model reconstructs scene details more accurately than existing works: iMAP\* [59] and NICE-SLAM [87], while it runs up to  $\times 10$  faster (see Sec. 4.2 for runtime analysis). The ground truth image is rendered with ReplicaViewer [57].

## Abstract

We present ESLAM, an efficient implicit neural representation method for Simultaneous Localization and Mapping (SLAM). ESLAM reads RGB-D frames with unknown camera poses in a sequential manner and incrementally reconstructs the scene representation while estimating the current camera position in the scene. We incorporate the latest advances in Neural Radiance Fields (NeRF) into a SLAM system, resulting in an efficient and accurate dense visual SLAM method. Our scene representation consists of multi-scale axis-aligned perpendicular feature planes and shallow decoders that, for each point in the continuous space, decode the interpolated features into Truncated Signed Distance Field (TSDF) and RGB values. Our extensive experiments on three standard datasets, Replica, ScanNet, and TUM RGB-D show that ESLAM improves the accuracy of 3D reconstruction and camera localization of state-of-the-art dense visual SLAM methods by more than 50%, while it runs up to  $\times 10$  faster and does not require any pre-training. Project page: <https://www.idiap.ch/paper/eslam>

## 1. Introduction

Dense visual Simultaneous Localization and Mapping (SLAM) is a fundamental challenge in 3D computer vision with several applications such as autonomous driving, robotics, and virtual/augmented reality. It is defined as constructing a 3D map of an unknown environment while simultaneously approximating the camera pose.

While traditional SLAM systems [16, 41, 45, 55, 76, 77]

mostly focus on localization accuracy, recent learning-based dense visual SLAM methods [2, 11, 25, 35, 60, 64, 65, 67, 81, 86] provide meaningful global 3D maps and show reasonable but limited reconstruction accuracy.

Following the advent of Neural Radiance Fields (NeRF) [37] and the demonstration of their capacity to reason about the geometry of a large-scale scene [8, 13, 20, 22, 26, 75, 78] and reconstruct 3D surfaces [1, 29, 47, 48, 62, 71, 72, 82, 85], novel NeRF-based dense SLAM methods have been developed. In particular, iMAP [59] and NICE-SLAM [87] utilize neural implicit networks to achieve a consistent geometry representation.

IMAP [59] represents the geometry with a single huge MLP, similar to NeRF [37], and optimizes the camera poses during the rendering process. NICE-SLAM [87] improves iMAP by storing the representation locally on voxel grids to prevent the forgetting problem. Despite promising reconstruction quality, these methods are computationally demanding for real-time applications, and their ability to capture geometry details is limited. In addition, NICE-SLAM [87] uses frozen pre-trained MLPs, which limits its generalizability to novel scenes. We take NICE-SLAM [87] as a baseline and provide the following contributions:

- We leverage implicit Truncated Signed Distance Field (TSDF) [1] to represent geometry, which converges noticeably faster than the common rendering-based representations like volume density [59] or occupancy [87] and results in higher quality reconstruction.
- Instead of storing features on voxel grids, we propose employing multi-scale axis-aligned feature planes [6]

which leads to reducing the memory footprint growth rate w.r.t. scene side-length from cubic to quadratic.

- We benchmark our method on three challenging datasets, Replica [57], ScanNet [12], and TUM RGB-D [58], to demonstrate the performance of our method in comparison to existing ones and provide an extensive ablation study to validate our design choices.

Thanks to the inherent smoothness of representing the scene with feature planes, our method produces higher-quality smooth surfaces without employing explicit smoothness loss functions like [70].

Concurrent with our work, the followings also propose Radiance Fields-based SLAM systems: iDF-SLAM [38] also uses TSDF, but it is substantially slower and less accurate than NICE-SLAM [87]. Orbeez-SLAM [10] operates in real-time at the cost of poor 3D reconstruction. Compromising accuracy and quality, MeSLAM [27] introduces a memory-efficient SLAM. MonoNeuralFusion [88] proposes an incremental 3D reconstruction model, assuming that ground truth camera postures are available. Lastly, NeRF-SLAM [54] presents a monocular SLAM system with hierarchical volumetric Neural Radiance Fields optimized using an uncertainty-based depth loss.

## 2. Related Work

**Dense Visual SLAM.** The ubiquity of cameras has made visual SLAM a field of major interest in the last decades. Traditional visual SLAM employs pixel-wise optimization of geometric and/or photometric constraints from image information. Depending on the information source, visual SLAM divides into three main categories: visual-only [15–17, 41, 45, 66], visual-inertial [3, 28, 39, 42, 53, 69] and RGB-D [5, 14, 23, 44] SLAM. Visual-only SLAM uses single or multi-camera setups but presents higher technical challenges compared to others. Visual-inertial information can improve accuracy, but complexifies the system and requires an extra calibration step. The advent of the Kinect brought popularity to RGB-D setups with improved performance but had drawbacks such as larger memory and power requirements, and limitation to indoor settings. Recently, learning-based approaches [2, 11, 30, 31, 67] have made great advances in the field, improving both accuracy and robustness compared to traditional methods.

**Neural Implicit 3D Reconstruction.** Neural Radiance Fields (NeRF) have impacted 3D Computer Vision applications, such as novel view synthesis [34, 36, 37, 68], surface reconstruction [46, 49, 71, 72, 82, 83, 85], dynamic scene representation [19, 50–52], and camera pose estimation [21, 32, 73, 79, 84]. The exploitation of neural implicit representations for 3D reconstruction at real-world scale is studied in [1, 4, 9, 29, 43, 63, 70, 74, 80]. The most related works

to ours are iMAP [59] and NICE-SLAM [87]. IMAP [59] presents a NeRF-style dense SLAM system. NICE-SLAM [87] extends iMAP [59] by modeling the scene with voxel grid features and decoding them into occupancies using pre-trained MLPs. However, the generalizability of NICE-SLAM [87] to novel scenes is limited because of the frozen pre-trained MLPs. Another issue is the cubic memory growth rate of their model, which results in using low-resolution voxel grids and losing fine geometry details. In contrast, we employ compact plane-based features [6] which are directly decoded to TSDF, improving both efficiency and accuracy of localization and reconstruction.

## 3. Method

The overview of our method is shown in Fig. 2. Given a set of sequential RGB-D frames  $\{I_i, D_i\}_{i=1}^M$ , our model predicts camera poses  $\{R_i | t_i\}_{i=1}^M$  and an implicit TSDF  $\phi_g$  representation that can be used in marching cubes algorithm [33] to extract 3D meshes. We expect TSDF to denote the distance to the closest surface with a positive sign in the free space and a negative sign inside the surfaces. We employ normalized TSDF, such that it is zero on the surfaces and has a magnitude of one at the truncation distance  $T$ , which is a hyper-parameter. Sec. 3.1 describes how we represent a scene with axis-aligned feature planes. Sec. 3.2 walks through the rendering process, which converts raw representations into pixel depths and colors. Sec. 3.3 introduces our loss functions. And Sec. 3.4 provides the details of localization and reconstruction in our SLAM system.

### 3.1. Axis-Aligned Feature Planes

Although voxel grid-based NeRF architectures [18, 61, 70] exhibit rapid convergence, they struggle with cubical memory growing. Different solutions have been proposed to mitigate the memory growth issue [6, 7, 40]. Inspired by [6], we employ a tri-plane architecture (see Fig. 2), in which we store and optimize features on perpendicular axis-aligned planes. Mimicking the trend in voxel-based methods [7, 40, 61], we propose using feature planes at two scales, *i.e.* coarse and fine. Coarse-level representation allows efficient reconstruction of free space with fewer sample points and optimization iterations. Moreover, we suggest employing separate feature planes for representing geometry and appearance, which mitigates the forgetting problem for geometry reconstruction since appearance fluctuates more frequently in a scene than geometry.

Specifically, we use three coarse feature planes  $\{F_{g-xy}^c, F_{g-xz}^c, F_{g-yz}^c\}$  and three fine ones  $\{F_{g-xy}^f, F_{g-xz}^f, F_{g-yz}^f\}$  for representing the geometry. Similarly, three coarse  $\{F_{a-xy}^c, F_{a-xz}^c, F_{a-yz}^c\}$  and three fine  $\{F_{a-xy}^f, F_{a-xz}^f, F_{a-yz}^f\}$  planes are used for representing appearance of a scene. This architecture prevents model size from growing cubically with the

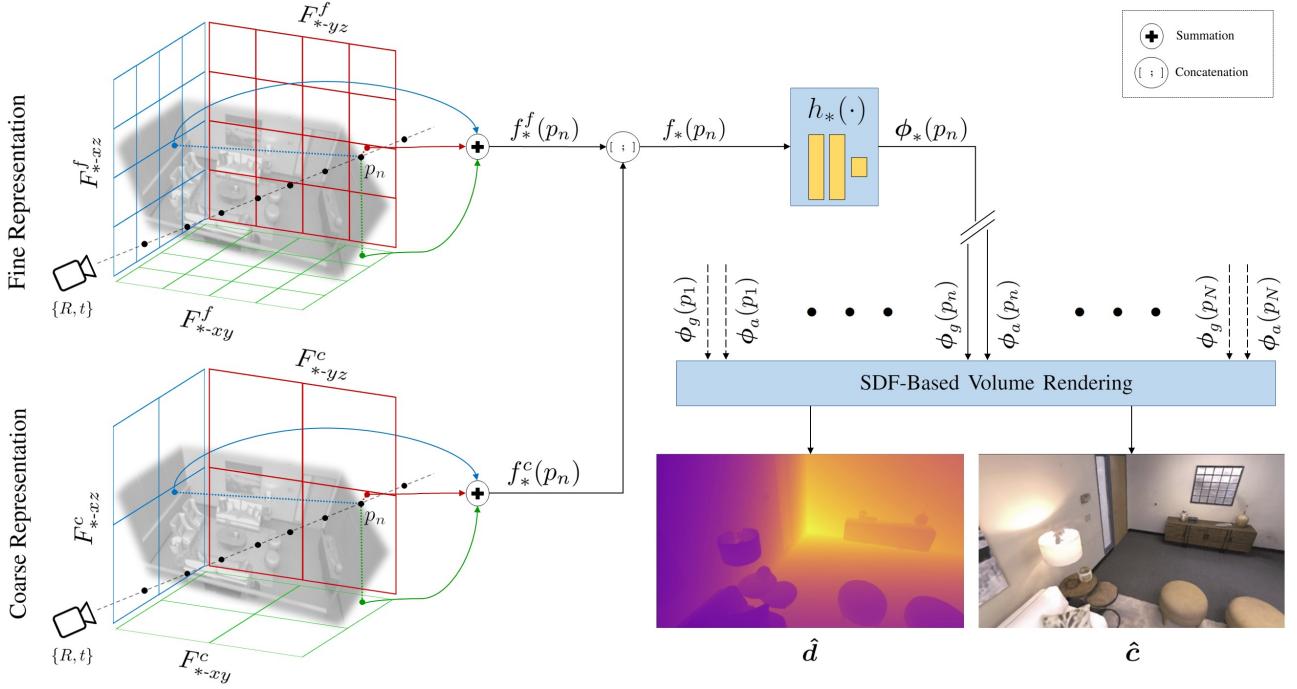


Figure 2. The overview of ESLAM. Given the symmetry of our processes for geometry and appearance, we exhibit both processes on the same pipeline for simplicity. The symbol  $*$  represents both geometry  $g$  and appearance  $a$ , e.g.,  $f_*(p_n)$  can be either  $f_g(p_n)$  or  $f_a(p_n)$ . At an estimated camera pose  $\{R, t\}$ , we cast a ray for each pixel and sample  $N$  points  $\{p_n\}_{n=1}^N$  along it (Sec. 3.2). Each point  $p_n$  is projected onto the coarse and fine feature planes and bilinearly interpolates the four nearest neighbor features on each plane (Sec. 3.1). The interpolated features at each level are added together, and the results from both levels are concatenated together to form the inputs  $\{f_g(p_n), f_a(p_n)\}$  of the decoders  $\{h_g, h_a\}$  (Sec. 3.1). The geometry decoder  $h_g$  estimates TSDF  $\phi_g(p_n)$  based on  $f_g(p_n)$ , and the appearance decoder  $h_a$  estimates the raw color  $\phi_a(p_n)$  based on  $f_a(p_n)$  for each point  $p_n$  (Sec. 3.1). Once TSDFs and raw colors of all points on a ray are generated, our SDF-based rendering process estimates the depth  $\hat{d}$  and the color  $\hat{c}$  for each pixel (Sec. 3.2).

scene side-length as is the case for voxel-based models.

To reason about the geometry of a point  $p$  in the continuous space, we first project it onto all the geometry planes. The geometry feature  $f_g(p)$  for point  $p$  is then formed by 1) bilinearly interpolating the four nearest neighbors on each feature plane, 2) summing the interpolated coarse features and the fine ones respectively into the coarse output  $f_g^c(p)$  and fine output  $f_g^f(p)$ , and 3) concatenating the outputs together. Formally:

$$\begin{aligned} f_g^c(p) &= F_{g-xy}^c(p) + F_{g-xz}^c(p) + F_{g-yz}^c(p) \\ f_g^f(p) &= F_{g-xy}^f(p) + F_{g-xz}^f(p) + F_{g-yz}^f(p) \\ f_g(p) &= [f_g^c(p); f_g^f(p)] \end{aligned} \quad (1)$$

The appearance feature  $f_a(p)$  is obtained similarly:

$$\begin{aligned} f_a^c(p) &= F_{a-xy}^c(p) + F_{a-xz}^c(p) + F_{a-yz}^c(p) \\ f_a^f(p) &= F_{a-xy}^f(p) + F_{a-xz}^f(p) + F_{a-yz}^f(p) \\ f_a(p) &= [f_a^c(p); f_a^f(p)] \end{aligned} \quad (2)$$

These features are decoded into TSDF  $\phi_g(p)$  and raw color  $\phi_a(p)$  values via shallow two-layer MLPs  $\{h_g, h_a\}$ :

$$\phi_g(p) = h_g(f_g(p)) \quad \text{and} \quad \phi_a(p) = h_a(f_a(p)) \quad (3)$$

These raw TSDF and color outputs can be utilized for depth/color rendering as well as mesh extraction.

### 3.2. SDF-Based Volume Rendering

When processing input frame  $i$ , emulating the ray casting in NeRF [37], we select random pixels and calculate their corresponding rays using the current estimate of the camera pose  $\{R_i | t_i\}$ . For rendering the depths and colors of the rays, we first sample  $N_{strat}$  samples on each ray by stratified sampling and then sample additional  $N_{imp}$  points near surfaces. For pixels with ground truth depths, the  $N_{imp}$  additional points are sampled uniformly inside the truncation distance  $T$  w.r.t. the depth measurement, whereas for other pixels,  $N_{imp}$  points are sampled with the importance sampling technique [34, 37, 59, 70] based on the weights computed for the stratified samples.

For all  $N = N_{strat} + N_{imp}$  points on a ray  $\{p_n\}_{n=1}^N$ , we query TSDF  $\phi_g(p_n)$  and raw color  $\phi_a(p_n)$  from our networks and use the SDF-Based rendering approach in

StyleSDF [47] to convert SDF values to volume densities:

$$\sigma(p_n) = \beta \cdot \text{Sigmoid}(-\beta \cdot \phi_g(p_n)) \quad (4)$$

where  $\beta$  is a learnable parameter that controls the sharpness of the surface boundary. Negative values of SDF push Sigmoid toward one, resulting in volume density inside the surface. The volume density then is used for rendering the color and depth of each ray:

$$w_n = \exp \left( - \sum_{k=1}^{n-1} \sigma(p_k) \right) (1 - \exp(-\sigma(p_n)))$$

$$\hat{c} = \sum_{n=1}^N w_n \phi_a(p_n) \quad \text{and} \quad \hat{d} = \sum_{n=1}^N w_n z_n \quad (5)$$

where  $z_n$  is the depth of point  $p_n$  w.r.t. the camera pose.

### 3.3. Loss Functions

One advantage of TSDF over other representations, such as occupancy, is that it allows us to use per-point losses, along with rendering ones. These losses account for the rapid convergence of our model. Following the practice in [1], assuming a batch of rays  $R$  with ground truth depths are selected, we define the free space loss as:

$$\mathcal{L}_{fs} = \frac{1}{|R|} \sum_{r \in R} \frac{1}{|P_r^{fs}|} \sum_{p \in P_r^{fs}} (\phi_g(p) - 1)^2 \quad (6)$$

where  $P_r^{fs}$  is a set of points on the ray  $r$  that lie between the camera center and the truncation region of the surface measured by the depth sensor. This loss function encourages TSDF  $\phi_g$  to have a value of one in the free space.

For sample points close to the surface and within the truncation region, we use the signed distance objective, which leverages the depth sensor measurement to approximate the signed distance field:

$$\mathcal{L}_T(P_r^T) =$$

$$\frac{1}{|R|} \sum_{r \in R} \frac{1}{|P_r^T|} \sum_{p \in P_r^T} (z(p) + \phi_g(p) \cdot T - D(r))^2 \quad (7)$$

where  $z(p)$  is the planar depth of point  $p$  w.r.t. camera,  $T$  is the truncation distance,  $D(r)$  is the ray depth measured by the sensor, and  $P_r^T$  is a set of points on the ray  $r$  that lie in the truncation region, i.e.  $|z(p) - D(r)| < T$ . We apply the same loss to all points in the truncation region, but we differentiate the importance of points that are closer to the surface in the middle of the truncation region  $P_r^{T-m}$  from those that are at the tail of the truncation region  $P_r^{T-t}$ . Formally, we define  $P_r^{T-m}$  as a set of points that  $|z(p) - D(r)| < 0.4T$ , and define  $P_r^{T-t} = P_r^T - P_r^{T-m}$ , then:

$$\mathcal{L}_{T-m} = \mathcal{L}_T(P_r^{T-m}) \quad \text{and} \quad \mathcal{L}_{T-t} = \mathcal{L}_T(P_r^{T-t}) \quad (8)$$

This enables us to decrease the importance of  $\mathcal{L}_{T-t}$  in mapping, which leads to having a smaller effective truncation distance, reducing artifacts in occluded areas, and reconstructing with higher accuracy while leveraging the entire truncation distance in camera tracking.

In addition to these two per-point loss functions, we also employ reconstruction losses. For pixels with ground truth depths, we impose consistency between the rendered depth and the depth measured by the sensor:

$$\mathcal{L}_d = \frac{1}{|R|} \sum_{r \in R} (\hat{d}(r) - D(r))^2 \quad (9)$$

Similarly, we impose consistency between the pixel colors and rendered colors:

$$\mathcal{L}_c = \frac{1}{|R|} \sum_{r \in R} (\hat{c}(r) - I(r))^2 \quad (10)$$

where  $I(r)$  is the pixel color of ray  $r$ .

The global loss function of our method is defined as:

$$\mathcal{L} = \lambda_{fs} \mathcal{L}_{fs} + \lambda_{T-m} \mathcal{L}_{T-m} + \lambda_{T-t} \mathcal{L}_{T-t} + \lambda_d \mathcal{L}_d + \lambda_c \mathcal{L}_c \quad (11)$$

where  $\{\lambda_{fs}, \lambda_{T-m}, \lambda_{T-t}, \lambda_d, \lambda_c\}$  are the weighting coefficients. Note that  $\mathcal{L}_c$  is defined on all rays in a training batch, while other losses are only imposed on rays with ground truth measured depths. The global objective is the same for both mapping and tracking in our method, but the weighting coefficients are different.

### 3.4. Mapping and Tracking

**Mapping.** Our scene representation, *i.e.* the feature planes and MLP decoders, are randomly initialized at the beginning. With the first input frame  $\{I_0, D_0\}$ , we fix the camera pose and optimize the feature planes and MLP decoders to best represent the first frame. For subsequent inputs, we update the scene representation iteratively every  $k$  frames, and add the latest frame to the global keyframe list, following the practice in iMAP [59] and NICE-SLAM [87]. For mapping, we first choose  $|R|$  pixels randomly from  $W$  frames, which include the current frame, the previous two keyframes, and  $W - 3$  frames randomly selected from the keyframe list. Then, we jointly optimize the feature planes, MLP decoders, and camera poses of the  $W$  selected frames using the loss functions introduced in Sec. 3.3. Unlike NICE-SLAM [87], our method does not require a staged-optimization policy, and we simply optimize all scene parameters and camera poses simultaneously.

**Tracking.** The localization process of our method is initiated for each input frame. The current estimate of the camera parameters, represented by translation vectors and quaternion rotations [56]  $\{R|t\}$ , are optimized solely based

Method	Reconstruction (cm)				Localization (cm)	
	Depth L1↓	Acc.↓	Comp.↓	Comp. Ratio (%)↑	ATE Mean↓	ATE RMSE↓
iMAP* [59]	$8.23 \pm 0.88$	$7.16 \pm 0.26$	$5.83 \pm 0.27$	$67.17 \pm 2.70$	$2.59 \pm 0.58$	$3.42 \pm 0.87$
NICE-SLAM [87]	$3.29 \pm 0.33$	$1.66 \pm 0.07$	$1.63 \pm 0.05$	$96.74 \pm 0.36$	$1.56 \pm 0.29$	$2.05 \pm 0.45$
ESLAM (ours)	<b><math>1.18 \pm 0.05</math></b>	<b><math>0.97 \pm 0.02</math></b>	<b><math>1.05 \pm 0.01</math></b>	<b><math>98.60 \pm 0.07</math></b>	<b><math>0.52 \pm 0.03</math></b>	<b><math>0.63 \pm 0.05</math></b>

Table 1. Quantitative comparison of our proposed ESLAM with existing NeRF-based dense visual SLAM models on the Replica dataset [57] for both reconstruction and localization accuracy. The results are the average and standard deviation of five runs on eight scenes of the Replica dataset [57]. Our method outperforms previous works by a high margin and has lower variances, indicating it is also more stable from run to run. The evaluation metrics for reconstruction are L1 loss (cm) between rendered and ground truth depth maps of 1000 random camera poses, reconstruction accuracy (cm), reconstruction completion (cm), and completion ratio (%). The evaluation metrics for localization are mean and RMSE of ATE (cm) [58]. For the details of the evaluations for each scene, refer to the supplementary. It should also be noted that our method runs up to  $\times 10$  faster on this dataset (see Sec. 4.2 for runtime analysis).

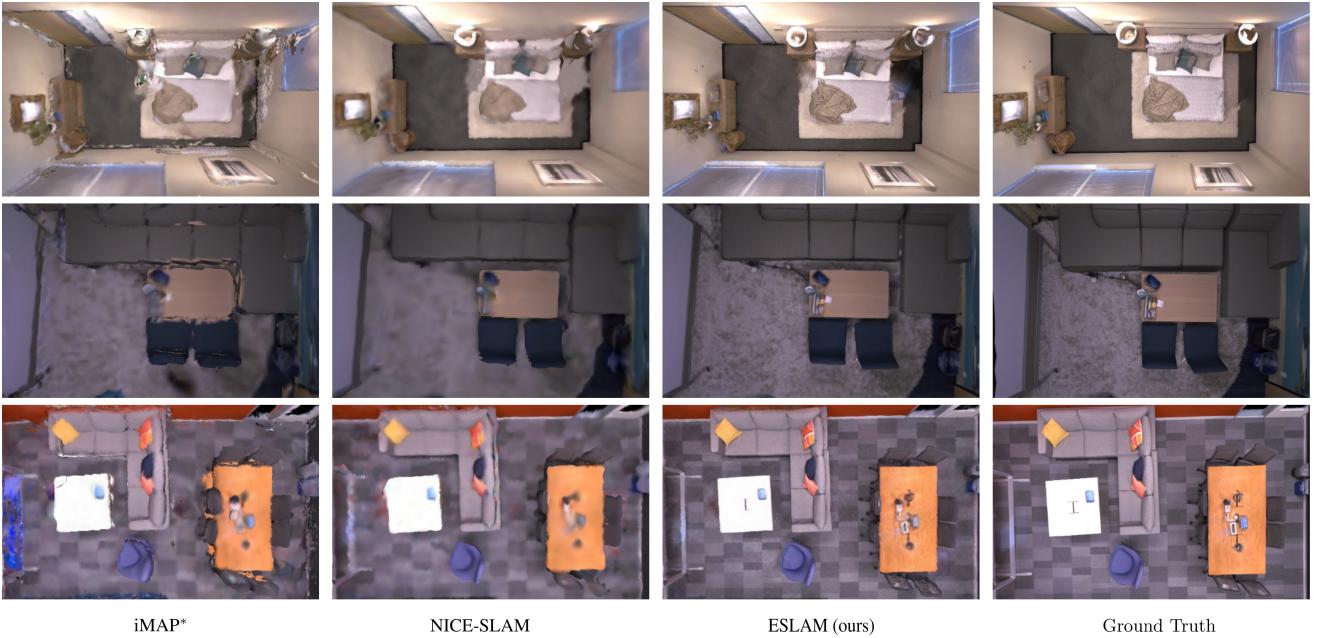


Figure 3. Qualitative comparison of our proposed ESLAM method’s geometry reconstruction with existing NeRF-based dense visual SLAM models, iMAP\* [59] and NICE-SLAM [87], on the Replica dataset [57]. Our method produces more accurate detailed geometry as well as higher-quality textures. The ground truth images are rendered with the ReplicaViewer software [57]. It should also be noted that our method runs up to  $\times 10$  faster on this dataset (see Sec. 4.2 for runtime analysis). For further qualitative analysis on this dataset, as well as videos demonstrating the localization and reconstruction process, refer to the supplementary.

on our global loss function (see Sec. 3.3) with the gradient-based Adam optimizer [24]. No second-order optimizers or manifold operations are employed for camera tracking in our method. We exclude rays with no ground truth depths and outlier pixels from each optimization step. A pixel is considered an outlier if the difference between its measured depth and rendered depth is ten times greater than the batch’s median rendered depth error.

## 4. Experiments

In this section, we validate that our method outperforms existing implicit representation-based methods in both localization and reconstruction accuracy on three standard benchmarks while running up to  $\times 10$  faster.

**Baselines.** We compare our method to two existing state-of-the-art NeRF-based dense visual SLAM methods: iMAP [59] and NICE-SLAM [87]. Because iMAP is not open source, we use the iMAP\* model in our experiment, which is the reimplementation of iMAP in [87].

**Datasets.** We evaluate our method on three standard 3D benchmarks: Replica [57], ScanNet [12], and TUM RGB-D [58] datasets. We select the same scenes for evaluation as NICE-SLAM [87].

**Metrics.** We borrow our evaluation metrics from NICE-SLAM [87]. For evaluating scene geometry, we use both 2D and 3D metrics. For the 2D metric, we render depth maps from 1000 random camera poses in each scene and calculate

Method	ATE	Sc. 0000	Sc. 0059	Sc. 0106	Sc. 0169	Sc. 0181	Sc. 0207	Ave.
iMAP* [59]	Mean	$34.2 \pm 12.8$	$13.0 \pm 2.4$	$12.9 \pm 1.7$	$33.6 \pm 15.3$	$20.8 \pm 3.8$	$18.6 \pm 6.0$	$22.2 \pm 7.0$
	RMSE	$42.7 \pm 16.6$	$17.8 \pm 7.4$	$15.0 \pm 1.7$	$39.1 \pm 18.2$	$24.7 \pm 5.8$	$20.1 \pm 6.8$	$26.6 \pm 9.4$
NICE-SLAM [87]	Mean	$9.9 \pm 0.4$	$11.9 \pm 1.8$	$7.0 \pm 0.2$	$9.2 \pm 1.0$	$12.2 \pm 0.3$	$5.5 \pm 0.3$	$9.3 \pm 0.7$
	RMSE	$12.0 \pm 0.5$	$14.0 \pm 1.8$	$7.9 \pm 0.2$	$10.9 \pm 1.1$	$13.4 \pm 0.3$	$6.2 \pm 0.4$	$10.7 \pm 0.7$
ESLAM (ours)	Mean	<b><math>6.5 \pm 0.1</math></b>	<b><math>6.4 \pm 0.4</math></b>	<b><math>6.7 \pm 0.1</math></b>	<b><math>5.9 \pm 0.1</math></b>	<b><math>8.3 \pm 0.2</math></b>	<b><math>5.4 \pm 0.1</math></b>	<b><math>6.5 \pm 0.2</math></b>
	RMSE	<b><math>7.3 \pm 0.2</math></b>	<b><math>8.5 \pm 0.5</math></b>	<b><math>7.5 \pm 0.1</math></b>	<b><math>6.5 \pm 0.1</math></b>	<b><math>9.0 \pm 0.2</math></b>	<b><math>5.7 \pm 0.1</math></b>	<b><math>7.4 \pm 0.2</math></b>

Table 2. Quantitative comparison of our proposed ESLAM method’s localization accuracy with existing NeRF-based dense visual SLAM models on the ScanNet dataset [12]. The results are the average and standard deviation of five runs on each scene of ScanNet [12]. Our method outperforms previous works and has lower variances, indicating it is also more stable from run to run. The evaluation metrics for localization are mean and RMSE of ATE (cm) [58]. It should also be noted that our method runs up to  $\times 6$  faster on this dataset (see Sec. 4.2 for runtime analysis).

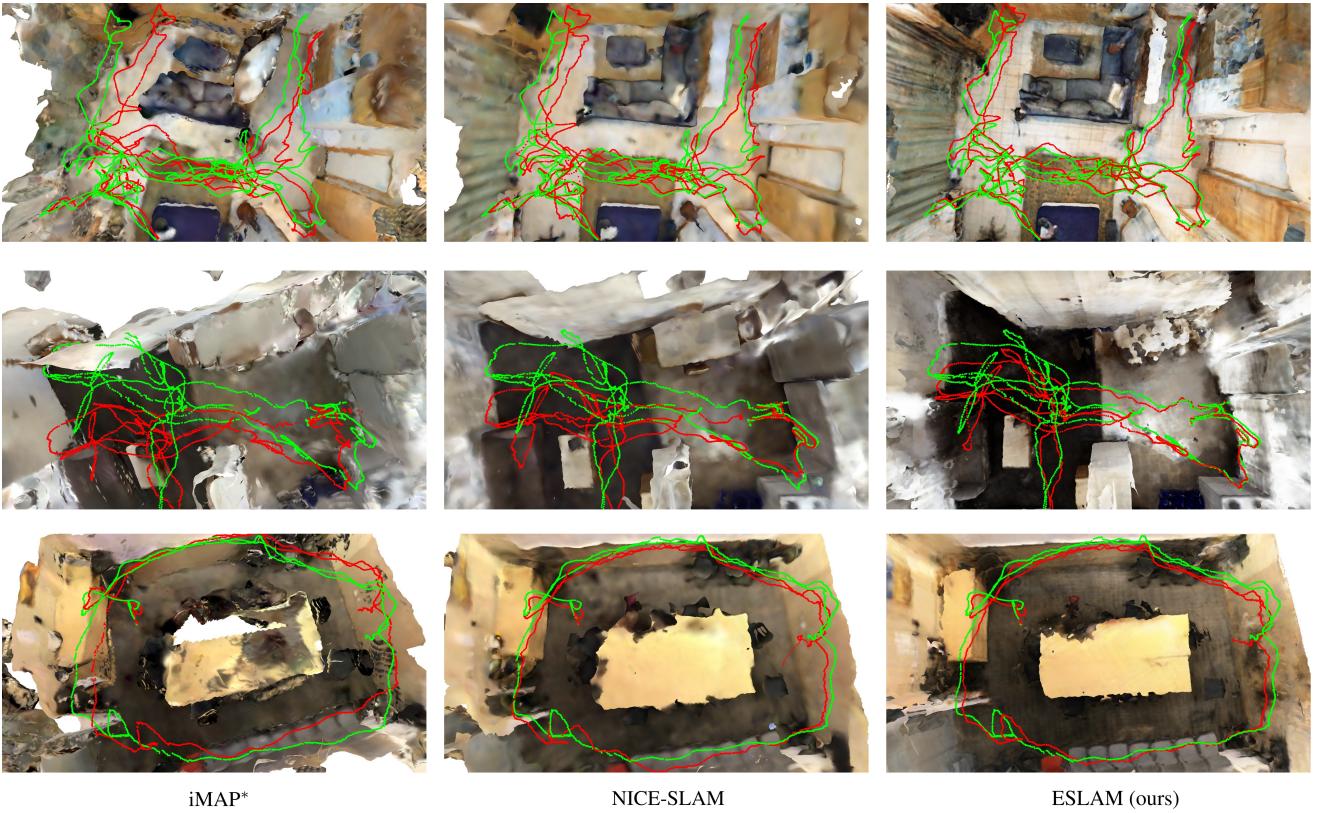


Figure 4. Qualitative comparison of our proposed ESLAM method’s localization accuracy with existing NeRF-based dense visual SLAM models, iMAP\* [59] and NICE-SLAM [87], on the ScanNet dataset [12]. The ground truth camera trajectory is shown in green, and the estimated trajectory is shown in red. Our method predicts more accurate camera trajectories and does not suffer from drifting issues. It should also be noted that our method runs up to  $\times 6$  faster on this dataset (see Sec. 4.2 for runtime analysis).

the L1 difference between depths from ground truth meshes and the reconstructed ones. For the 3D metrics, we consider reconstruction accuracy [cm], reconstruction completion [cm], and completion ratio [ $< 5$  cm %]. For evaluating these metrics, we build a TSDF volume for a scene with a resolution of 1 cm and use the marching cubes algorithm [33] to obtain scene meshes. Before evaluating the 3D metrics for our method and for the baselines, we perform mesh culling as recommended in [1, 70]. For this purpose, we remove faces from a mesh that are not inside any

camera frustum or are occluded in all RGB-D frames. For evaluating camera localization, we use ATE [58].

**Implementation Details.** The truncation distance  $T$  is set to 6 cm in our method. We employ coarse feature planes with a resolution of 24 cm for both geometry and appearance. For fine feature planes, we use a resolution of 6 cm for geometry and 3 cm for appearance. All feature planes have 32 channels, resulting in a 64-channel concatenated feature input for the decoders. The decoders are two-layer MLPs

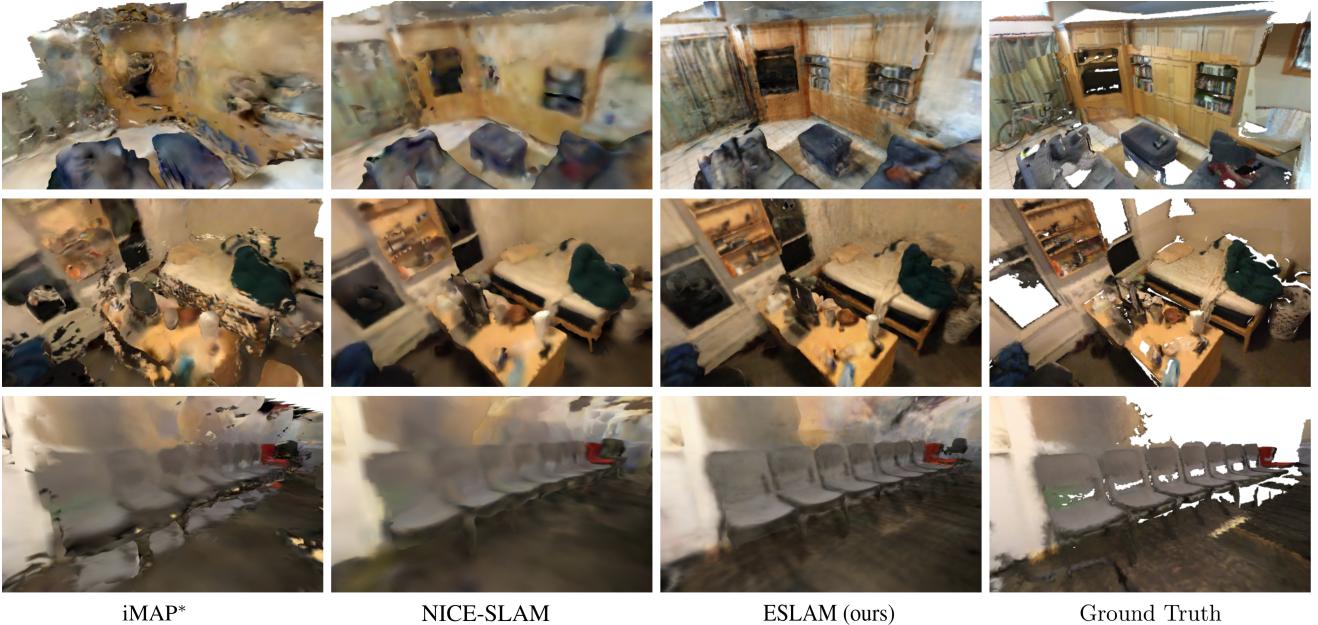


Figure 5. Qualitative comparison of our proposed ESLAM method’s geometry reconstruction with existing NeRF-based dense visual SLAM models, iMAP\* [59] and NICE-SLAM [87], on the ScanNet dataset [12]. Our method produces more accurate detailed geometry as well as higher-quality textures. The appearance of white backgrounds in ground truth meshes is due to the fact that the ground truth meshes of the ScanNet dataset [12] are incomplete. It should also be noted that our method runs up to  $\times 6$  faster on this dataset (see Sec. 4.2 for runtime analysis).

with 32 channels in the hidden layer. For the Replica [57] dataset, we sample  $N_{strat} = 32$  points for stratified sampling and  $N_{imp} = 8$  points for importance sampling on each ray. And for the ScanNet [12] and TUM RGB-D [58] datasets, we set  $N_{strat} = 48$  and  $N_{imp} = 8$ .

We use different set of loss coefficients for mapping and tracking. During mapping we set  $\lambda_{fs} = 5$ ,  $\lambda_{T-m} = 200$ ,  $\lambda_{T-t} = 10$ ,  $\lambda_d = 0.1$ , and  $\lambda_c = 5$ . And during tracking, we set  $\lambda_{fs} = 10$ ,  $\lambda_{T-m} = 200$ ,  $\lambda_{T-t} = 50$ ,  $\lambda_d = 1$ , and  $\lambda_c = 5$ . These coefficients are obtained by performing grid search in our experiments. For further details of our implementation, refer to the supplementary.

#### 4.1. Experimental Results

**Evaluation on Replica [57].** We provide the quantitative analysis of our experimental results on eight scenes of the Replica dataset [57] in Tab. 1. The numbers represent the average and standard deviation of the metrics for five independent runs. As shown in Tab. 1, our method outperforms the baselines for both reconstruction and localization accuracy. Our method also has lower variances, indicating that it is more stable and more robust than existing methods.

Qualitative analysis on the Replica dataset [57] is provided in Fig. 3. The results show that our method reconstructs the details of the scenes more accurately and produces fewer artifacts. Although it is not the focus of this paper, our method also produces higher-quality colors for the reconstructed meshes.

**Evaluation on ScanNet [12].** We also benchmark ours and existing methods on multiple large scenes from ScanNet [12] to evaluate and compare their scalability. For evaluating camera localization, we conduct five independent experiments on each scene and report the average and standard deviation of the mean and RMSE of ATE [58] in Tab. 2. As demonstrated in the table, our method’s localization is more accurate than existing methods. Our method is also considerably more stable from run to run as it has much lower standard deviations. We provide qualitative analysis of camera localization, along with geometry reconstruction, in Fig. 4. The results show that our method does not suffer from any large drifting and is more robust than existing methods.

Since the ground truth meshes of the ScanNet dataset [12] are incomplete, we only provide qualitative analysis for geometry reconstruction, similar to previous works. The qualitative comparison in Fig. 5 validates that our model reconstructs more precise geometry and detailed textures compared to existing approaches.

**Evaluation on TUM RGB-D [58].** To further contrast the robustness of our method with the existing ones, we conduct an evaluation study on the real-world TUM RGB-D dataset [58]. Since there are no ground truth meshes for the scenes in this dataset, we only present the localization results in Tab. 3 and the qualitative analysis of rendered meshes in Fig. 6.

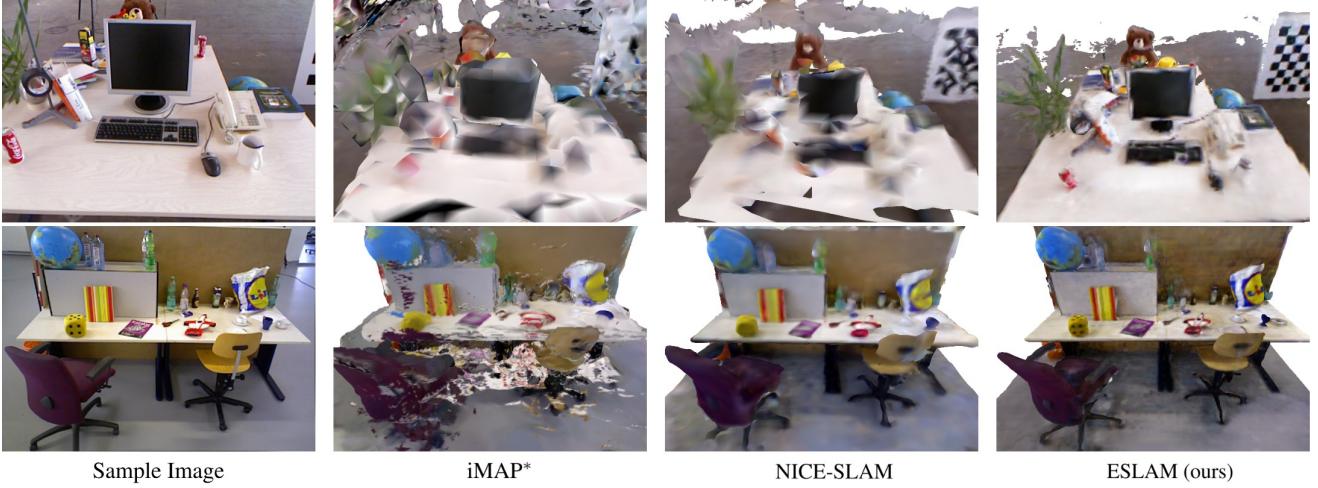


Figure 6. Qualitative comparison of our proposed ESLAM method’s geometry reconstruction with existing NeRF-based dense visual SLAM models, iMAP\* [59] and NICE-SLAM [87], on the TUM RGB-D dataset [58]. Our method produces more accurate detailed geometry as well as higher-quality textures. Since there are no ground truth meshes for this dataset, we depict a sample input image.

	fr1/desk	fr2/xyz	fr3/office
iMAP* [59]	4.90	2.05	5.80
NICE-SLAM [87]	2.85	2.39	3.02
ESLAM (ours)	<b>2.47</b>	<b>1.11</b>	<b>2.42</b>

Table 3. Quantitative comparison of our proposed ESLAM method’s localization accuracy with existing NeRF-based dense visual SLAM models on the TUM RGB-D dataset [58]. The evaluation metric is ATE RMSE $\downarrow$  (cm) [58].

	Method	Speed FPT (s)	Memory	
			# Param.	Grow. R.
Replica	iMAP* [59]	5.20	<b>0.22 M</b>	?
	NICE-SLAM [87]	2.10	12.18 M	$O(L^3)$
	ESLAM (ours)	<b>0.18</b>	6.79 M	$O(L^2)$
ScanNet	iMAP* [59]	5.20	<b>0.22 M</b>	?
	NICE-SLAM [87]	3.35	22.04 M	$O(L^3)$
	ESLAM (ours)	<b>0.55</b>	17.63 M	$O(L^2)$

Table 4. Runtime analysis of our method in comparison with existing ones in terms of average frame processing time (FPT), number of parameters, and model size growth rate w.r.t. scene side-length  $L$ . All methods are benchmarked with an NVIDIA GeForce RTX 3090 GPU on room0 of Replica [57] and scene0000 of ScanNet [12]. Our method is significantly faster and does not grow cubically in size w.r.t. scene side-length  $L$ . Note that iMAP [59] represents a whole scene in a single MLP, hence its small number of parameters. Accordingly, the scalability and growth rate of iMAP [59] w.r.t. the scene side-length  $L$  are also unclear.

## 4.2. Runtime Analysis

We evaluate the speed and size of our method in comparison with existing approaches in Tab. 4. We report the average frame processing time (FPT), the number of parameters of the model, and memory growth rate w.r.t. scene side-length for the scenes room0 of Replica [57] and

scene0000 of ScanNet [12] datasets. All methods are benchmarked with an NVIDIA GeForce RTX 3090 GPU. The results indicate that our method is significantly faster than previous works on both datasets. Furthermore, in contrast to NICE-SLAM [87], our model size is smaller and does not grow cubically with the scene side-length.

## 5. Conclusion

We presented ESLAM, a dense visual SLAM approach that leverages the latest advances in the Neural Radiance Fields study to improve both the speed and accuracy of neural implicit-based SLAM systems. We proposed replacing the voxel grid representation with axis-aligned feature planes to prevent the model size from growing cubically with respect to the scene side-length. We also demonstrated that modeling the scene geometry with Truncated Signed Distance Field (TSDF) leads to efficient and high-quality surface reconstruction. We verified through extensive experiments that our approach outperforms existing methods significantly in both reconstruction and localization accuracy while running up to one order of magnitude faster.

ESLAM accepts and deals with the forgetting problem in exchange for memory preservation. Due to the structure of our feature plane representation, updating features to adapt to new geometry may affect previously reconstructed geometries. To address this issue, we keep track of previous keyframes and allocate a large portion of computation resources to retain and remember previously reconstructed regions. Although ESLAM is substantially faster than competing approaches, handling the forgetting problem more efficiently could further reduce frame processing time.

## Acknowledgement

This research was supported by ams OSRAM.

## References

- [1] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6290–6301, 2022. [1](#) [2](#) [4](#) [6](#)
- [2] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison. Codeslam—learning a compact, optimisable representation for dense visual slam. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2560–2568, 2018. [1](#) [2](#)
- [3] Michael Bloesch, Sammy Omari, Marco Hutter, and Roland Siegwart. Robust visual inertial odometry using a direct ekf-based approach. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 298–304. IEEE, 2015. [2](#)
- [4] Aljaz Bozic, Pablo Palafox, Justus Thies, Angela Dai, and Matthias Nießner. Transformerfusion: Monocular rgb scene reconstruction using transformers. *Advances in Neural Information Processing Systems*, 34:1403–1414, 2021. [2](#)
- [5] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021. [2](#)
- [6] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133, 2022. [1](#) [2](#)
- [7] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022. [2](#)
- [8] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021. [1](#)
- [9] Jaesung Choe, Sunghoon Im, Francois Rameau, Minjun Kang, and In So Kweon. Volumefusion: Deep depth fusion for 3d scene reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16086–16095, 2021. [2](#)
- [10] Chi-Ming Chung, Yang-Che Tseng, Ya-Ching Hsu, Xiang-Qian Shi, Yun-Hung Hua, Jia-Fong Yeh, Wen-Chin Chen, Yi-Ting Chen, and Winston H Hsu. Orbeez-slam: A real-time monocular visual slam with orb features and nerf-realized mapping. *arXiv preprint arXiv:2209.13274*, 2022. [2](#)
- [11] Jan Czarnowski, Tristan Laidlow, Ronald Clark, and Andrew J Davison. Deepfactors: Real-time probabilistic dense monocular slam. *IEEE Robotics and Automation Letters*, 5(2):721–728, 2020. [1](#) [2](#)
- [12] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. [2](#) [5](#) [6](#) [7](#) [8](#)
- [13] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891, 2022. [1](#)
- [14] Felix Endres, Jürgen Hess, Jürgen Sturm, Daniel Cremers, and Wolfram Burgard. 3-d mapping with an rgb-d camera. *IEEE transactions on robotics*, 30(1):177–187, 2013. [2](#)
- [15] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017. [2](#)
- [16] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014. [1](#) [2](#)
- [17] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 15–22. IEEE, 2014. [2](#)
- [18] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. [2](#)
- [19] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. [2](#)
- [20] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5885–5894, 2021. [1](#)
- [21] Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-calibrating neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5846–5854, 2021. [2](#)
- [22] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18365–18375, 2022. [1](#)
- [23] Christian Kerl, Jürgen Sturm, and Daniel Cremers. Dense visual slam for rgb-d cameras. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2100–2106. IEEE, 2013. [2](#)
- [24] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014. [5](#)
- [25] Lukas Koestler, Nan Yang, Niclas Zeller, and Daniel Cremers. Tandem: Tracking and dense mapping in real-time using deep multi-view stereo. In *Conference on Robot Learning*, pages 34–45. PMLR, 2022. [1](#)
- [26] Adam R Kosiorek, Heiko Strathmann, Daniel Zoran, Pol Moreno, Rosalia Schneider, Sona Mokrá, and

- Danilo Jimenez Rezende. Nerf-vae: A geometry aware 3d scene generative model. In *International Conference on Machine Learning*, pages 5742–5752. PMLR, 2021. 1
- [27] Evgenii Kruzhkov, Alena Savinykh, Pavel Karpyshев, Mikhail Kurenkov, Evgeny Yudin, Andrei Potapov, and Dzmitry Tsetserukou. Meslam: Memory efficient slam based on neural fields. *arXiv preprint arXiv:2209.09357*, 2022. 2
- [28] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual-inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015. 2
- [29] Kejie Li, Yansong Tang, Victor Adrian Prisacariu, and Philip HS Torr. Bnv-fusion: Dense 3d reconstruction using bi-level neural volume fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6166–6175, 2022. 1, 2
- [30] Ruihao Li, Sen Wang, and Dongbing Gu. Deepslam: A robust monocular slam system with unsupervised deep learning. *IEEE Transactions on Industrial Electronics*, 68(4):3577–3587, 2020. 2
- [31] Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 7286–7291. IEEE, 2018. 2
- [32] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5741–5751, 2021. 2
- [33] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 2, 6
- [34] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 2, 3
- [35] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *2017 IEEE International Conference on Robotics and automation (ICRA)*, pages 4628–4635. IEEE, 2017. 1
- [36] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P Srinivasan, and Jonathan T Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16190–16199, 2022. 2
- [37] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I*, pages 405–421, 2020. 1, 2, 3
- [38] Yuhang Ming, Weicai Ye, and Andrew Calway. idf-slam: End-to-end rgb-d slam with neural implicit mapping and deep feature tracking. *arXiv preprint arXiv:2209.07919*, 2022. 2
- [39] Anastasios I Mourikis and Stergios I Roumeliotis. A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proceedings 2007 IEEE international conference on robotics and automation*, pages 3565–3572. IEEE, 2007. 2
- [40] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*, 2022. 2
- [41] Raúl Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017. 1, 2
- [42] Raúl Mur-Artal and Juan D Tardós. Visual-inertial monocular slam with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803, 2017. 2
- [43] Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-to-end 3d scene reconstruction from posed images. In *European conference on computer vision*, pages 414–431. Springer, 2020. 2
- [44] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee, 2011. 2
- [45] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE, 2011. 1, 2
- [46] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5589–5599, 2021. 2
- [47] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. Stylesdf: High-resolution 3d-consistent image and geometry generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13503–13513, 2022. 1, 4
- [48] Joseph Ortiz, Alexander Clegg, Jing Dong, Edgar Sucar, David Novotny, Michael Zollhoefer, and Mustafa Mukadam. isdf: Real-time neural signed distance fields for robot perception. *arXiv preprint arXiv:2204.02296*, 2022. 1
- [49] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deep sdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 2
- [50] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 2

- [51] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021. 2
- [52] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2
- [53] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018. 2
- [54] Antoni Rosinol, John J Leonard, and Luca Carlone. Nerf-slam: Real-time dense monocular slam with neural radiance fields. *arXiv preprint arXiv:2210.13641*, 2022. 2
- [55] Thomas Schops, Torsten Sattler, and Marc Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 134–144, 2019. 1
- [56] Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, 1985. 4
- [57] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 1, 2, 5, 7, 8
- [58] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 573–580. IEEE, 2012. 2, 5, 6, 7, 8
- [59] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021. 1, 2, 3, 4, 5, 6, 7, 8
- [60] Edgar Sucar, Kentaro Wada, and Andrew Davison. Nodeslam: Neural object descriptors for multi-view shape reconstruction. In *2020 International Conference on 3D Vision (3DV)*, pages 949–958. IEEE, 2020. 1
- [61] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 2
- [62] Jiaming Sun, Xi Chen, Qianqian Wang, Zhengqi Li, Hadar Averbuch-Elor, Xiaowei Zhou, and Noah Snavely. Neural 3d reconstruction in the wild. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–9, 2022. 1
- [63] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. Neuralrecon: Real-time coherent 3d reconstruc-
- tion from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15598–15607, 2021. 2
- [64] Niko Sünderhauf, Trung T Pham, Yasir Latif, Michael Milford, and Ian Reid. Meaningful maps with object-oriented semantic mapping. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5079–5085. IEEE, 2017. 1
- [65] Chengzhou Tang and Ping Tan. Ba-net: Dense bundle adjustment networks. In *International Conference on Learning Representations*, 2018. 1
- [66] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2
- [67] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in Neural Information Processing Systems*, 34:16558–16569, 2021. 1, 2
- [68] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022. 2
- [69] Lukas Von Stumberg, Vladislav Usenko, and Daniel Cremers. Direct sparse visual-inertial odometry using dynamic marginalization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2510–2517. IEEE, 2018. 2
- [70] Jingwen Wang, Tymoteusz Bleja, and Lourdes Agapito. Go-surf: Neural feature grid optimization for fast, high-fidelity rgb-d surface reconstruction. *arXiv preprint arXiv:2206.14735*, 2022. 2, 3, 6
- [71] Jiepeng Wang, Peng Wang, Xiaoxiao Long, Christian Theobalt, Taku Komura, Lingjie Liu, and Wenping Wang. Neuris: Neural reconstruction of indoor scenes using normal priors. *arXiv preprint arXiv:2206.13597*, 2022. 1, 2
- [72] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 1, 2
- [73] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf–: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021. 2
- [74] Silvan Weder, Johannes L Schonberger, Marc Pollefeys, and Martin R Oswald. Neuralfusion: Online depth fusion in latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3162–3172, 2021. 2
- [75] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5610–5619, 2021. 1
- [76] Thomas Whelan, Michael Kaess, Maurice Fallon, Hordur Johannsson, John Leonard, and John McDonald. Kintinuous:

- Spatially extended kinectfusion. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, 2012. 1
- [77] Thomas Whelan, Stefan Leutenegger, Renato Salas-Moreno, Ben Glocker, and Andrew Davison. Elasticfusion: Dense slam without a pose graph. In *Robotics: Science and Systems (RSS)*. Robotics: Science and Systems, 2015. 1
- [78] Xiuchao Wu, Jiamin Xu, Zihan Zhu, Hujun Bao, Qixing Huang, James Tompkin, and Weiwei Xu. Scalable neural indoor scene rendering. *ACM Transactions on Graphics (TOG)*, 41(4):1–16, 2022. 1
- [79] Yitong Xia, Hao Tang, Radu Timofte, and Luc Van Gool. Sinerf: Sinusoidal neural radiance fields for joint pose estimation and scene reconstruction. *arXiv preprint arXiv:2210.04553*, 2022. 2
- [80] Zike Yan, Yuxin Tian, Xuesong Shi, Ping Guo, Peng Wang, and Hongbin Zha. Continual neural mapping: Learning an implicit scene representation from sequential observations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15782–15792, 2021. 2
- [81] Xingrui Yang, Yuhang Ming, Zhaopeng Cui, and Andrew Calway. Fd-slam: 3-d reconstruction using features and dense matching. In *2022 International Conference on Robotics and Automation (ICRA)*, page 8040–8046, 2022. 1
- [82] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34:4805–4815, 2021. 1, 2
- [83] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33:2492–2502, 2020. 2
- [84] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inferf: Inverting neural radiance fields for pose estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1323–1330. IEEE, 2021. 2
- [85] Jason Zhang, Gengshan Yang, Shubham Tulsiani, and Deva Ramanan. Ners: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. *Advances in Neural Information Processing Systems*, 34:29835–29847, 2021. 1, 2
- [86] Shuaifeng Zhi, Michael Bloesch, Stefan Leutenegger, and Andrew J Davison. Scenecode: Monocular dense semantic reconstruction using learned encoded scene representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11776–11785, 2019. 1
- [87] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022. 1, 2, 4, 5, 6, 7, 8
- [88] Zi-Xin Zou, Shi-Sheng Huang, Yan-Pei Cao, Tai-Jiang Mu, Ying Shan, and Hongbo Fu. Mononeuralfusion: Online monocular neural 3d reconstruction with geometric priors. *arXiv preprint arXiv:2209.15153*, 2022. 2

# Supplementary Materials for ESLAM: Efficient Dense SLAM System Based on Hybrid Representation of Signed Distance Fields

Mohammad Mahdi Johari  
Idiap Research Institute, EPFL  
mohammad.johari@idiap.ch

Camilla Carta  
ams OSRAM  
camilla.cart@ams-osram.com

François Fleuret  
University of Geneva, EPFL  
francois.fleuret@unige.ch

## 1. Further Implementation Details

This section provides additional implementation details of our method. For the sake of completeness, we also reiterate the points mentioned in the main article.

The truncation distance  $T$  is set to 6 cm in our method. We employ coarse feature planes with a resolution of 24 cm for both geometry and appearance. For fine feature planes, we use a resolution of 6 cm for geometry and 3 cm for appearance. All feature planes have 32 channels, resulting in a 64-channel concatenated feature input for the decoders. The decoders are two-layer MLPs with 32 channels in the hidden layer. ReLU activation function is used for the hidden layer, and Tanh and Sigmoid are respectively used for the output layers of TSDF and raw colors.

We use different set of loss coefficients for mapping and tracking. During mapping we set  $\lambda_{fs} = 5$ ,  $\lambda_{T-m} = 200$ ,  $\lambda_{T-t} = 10$ ,  $\lambda_d = 0.1$ , and  $\lambda_c = 5$ . And during tracking, we set  $\lambda_{fs} = 10$ ,  $\lambda_{T-m} = 200$ ,  $\lambda_{T-t} = 50$ ,  $\lambda_d = 1$ , and  $\lambda_c = 5$ . These coefficients are obtained by performing grid search in our experiments.

For the scenes from Replica [5], we sample  $N_{strat} = 32$  points for stratified sampling and  $N_{imp} = 8$  points for importance sampling on each ray. We perform 15 optimization iterations for mapping and randomly select 4000 rays for each iteration. For camera tracking, 2000 rays are chosen at random and 8 optimization iterations are performed. Since ScanNet's [1] scenes are at a larger scale and more challenging, we set  $N_{strat} = 48$  and  $N_{imp} = 8$ . Also, we perform 30 optimization iterations for both mapping and tracking in ScanNet's [1] scenes. For the scenes in TUM RGB-D dataset [6], we similarly set  $N_{strat} = 48$  and  $N_{imp} = 8$ . For this dataset, We perform 60 optimization iterations for mapping and 200 optimization iterations for tracking, and we randomly sample 5000 rays for each iteration.

We initiate the mapping process every 4 input frames and use a window of  $W = 20$  keyframes for jointly optimizing the feature planes, MLP decoders, and camera poses of the selected keyframes. We use Adam [2] for optimizing

all learnable parameters of our method and set the learning rates according to a simple grid search in our experiments. We use a learning rate of 0.001 for the MLP decoders and a learning rate of 0.005 for the feature planes. We always use a learning rate of 0.001 for the camera poses, *i.e.* rotation and translation  $\{R, t\}$ , of the selected keyframes during the joint optimization of the mapping step. During the tracking step in the Replica's [5] scenes, we use a learning rate of 0.001 for camera rotation and translation. For camera tracking in the scenes of ScanNet [1], we use a learning rate of 0.0005 for camera translation and a learning rate of 0.0025 for camera rotation. Lastly, For camera tracking in the scenes of TUM RGB-D [6], we use a learning rate of 0.01 for camera translation and a learning rate of 0.002 for camera rotation. We model the camera rotation parameters with quaternions [4].

Once all input frames are processed, and for evaluation purposes, we build a TSDF volume for each scene and use the marching cubes algorithm [3] to obtain 3D meshes. We do not employ any post-processing for our representation or extracted meshes except that for quantitative evaluation, we cull faces from a mesh that are not inside any camera frustum or are occluded in all RGB-D frames. To ensure fairness, we do the same mesh culling before evaluating the previous approaches.

## 2. Ablation Study

In this section, we conduct various experiments to show the robustness of our method in different experimental settings and to validate our architecture design choices.

**Robustness to Depth Quality.** In this experiment, we evaluate how robust the methods are to the quality of input depths. Accordingly, we downsample input depths of `room0` of the replica dataset [5] to  $\frac{1}{8}$  of the original resolution. The results in Tab. 1 reveal that our method's reconstruction and localization are less sensitive to the resolution of input depth maps.

	Method	ATE↓	Acc.↓	Comp.↓
$\frac{1}{1}$ D	NICE-SLAM [8]	1.69	1.71	1.69
	ESLAM (ours)	<b>0.71</b>	<b>1.07</b>	<b>1.12</b>
$\frac{1}{8}$ D	NICE-SLAM [8]	2.01	2.18	1.98
	ESLAM (ours)	<b>0.72</b>	<b>1.16</b>	<b>1.23</b>

Table 1. Robustness to depth resolution comparison of our method with NICE-SLAM [8] in terms of ATE RMSE (cm), reconstruction accuracy (cm), and reconstruction completion (cm) on `room0` of the Replica [5] dataset. Our method’s accuracy is less affected when input depth is downsampled by a factor of  $\frac{1}{8}$ .

Method	ATE↓	Acc.↓	Comp.↓
NICE-SLAM [8]	1.69	1.71	1.69
NICE-SLAM w/ Our Key. Policy	1.65	1.68	1.66

Table 2. Analysis of the impact of our keyframe updating policy on NICE-SLAM [8] (Sec. 3.4 in the main paper). The experiment is conducted on `room0` of Replica [5], and the metrics are ATE RMSE (cm), reconstruction accuracy (cm), and reconstruction completion (cm). NICE-SLAM [8] only slightly benefits from our updating policy.

**Keyframe Policy.** Whenever we perform a mapping step for an input frame, we always include that frame in our global keyframe list (see Sec. 3.4 in the main paper). NICE-SLAM [8], on the other hand, only updates its keyframe list once per 10 mapping steps. To make sure that our evaluations are fair, we also run NICE-SLAM [8] with our own keyframe updating policy on `room0` of the Replica [5] dataset. The results in Tab. 2 show that NICE-SLAM [8] only slightly benefits from this updating policy.

**Our Design Choices.** We conduct multiple experiments in Tab. 3 to defend our design choices in ESLAM. These experiments are conducted on the scenes in the Replica [5] and ScanNet [1] datasets, and the details of the experimental settings are as follows. (a) We use shared feature planes for geometry and appearance (see Sec. 3.1 and Fig. 2 in the main paper). (b) We only employ coarse feature planes (see Sec. 3.1 and Fig. 2 in the main paper). (c) We only employ fine feature planes (see Sec. 3.1 and Fig. 2 in the main paper). (d) We add the interpolated coarse  $f_*^c(p_n)$  and fine  $f_*^f(p_n)$  features instead of concatenating them (see Sec. 3.1 and Fig. 2 in the main paper). (e) We discard importance sampling and use stratified sampling for all  $N$  points on a ray (see Sec. 3.2 in the main paper). (f) We only exploit depth inputs and discard color rendering and RGB inputs (see Sec. 3.2 in the main paper). (g) We do not consider separate loss functions for the points that are at the tail of the truncation region  $P_r^{T-t}$  and for the points that are in the middle  $P_r^{T-m}$  (see Sec. 3.3 in the main paper). (h) We do not jointly optimize camera poses during the mapping step (see Sec. 3.4 in the main paper). (i) We evaluate our full model. Note that due to the incompleteness of ScanNet’s [1] ground

truth meshes, we only evaluate localization accuracy on this dataset.

### 3. Additional Qualitative Analysis

This section provides additional qualitative analysis to contrast the capability of our method to preserve scene details in comparison to previous NeRF-based dense visual SLAM methods, iMAP\* [7] and NICE-SLAM [8]. We provide this analysis on the Replica dataset [5] in Fig. 1 with both textured and untextured meshes. The results demonstrate that our method produces more accurate meshes with fewer artifacts.

### 4. Per-Scene Breakdown of the Results

In this section, we breakdown the quantitative analysis of Tab. 1 in the main paper into a per-scene analysis. Tab. 4 shows the per-scene quantitative evaluation of our method in comparison with iMAP\* [7] and NICE-SLAM [8] on the Replica dataset [5]. As it is shown in Tab. 4, our method outperforms previous approaches in all scenes of Replica [5]. Also, lower variances in our experiments are an indication that our method is more stable from run to run.

### 5. Effect of Frame Processing Time

In this section, we investigate the trade-off between frame processing time and our method’s reconstruction and localization accuracy. In this study, we increase the number of optimization iterations during the mapping and tracking. By default, our ESLAM method performs  $Iter_m = 15$  optimization iterations during mapping and  $Iter_t = 8$  optimization iterations during tracking for the scenes of the Replica dataset [5]. We define ESLAM x2 as our method when we double the number of optimization iterations, *i.e.*  $Iter_m = 30$  and  $Iter_t = 16$ . And similarly, we define ESLAM x10 as our method with  $Iter_m = 150$  and  $Iter_t = 80$ .

Tab. 5 provides a quantitative analysis of ESLAM x2 and ESLAM x10, as well as a comparison with our default ESLAM method and existing approaches. The results show that at the cost of increased frame processing time, our method yields more accurate scene reconstruction and camera trajectory. It should be noted that even ESLAM x10 runs faster than the existing state-of-the-art method, NICE-SLAM [8].

Fig. 2 provides a qualitative analysis of ESLAM x10 compared to our default ESLAM method. In this analysis, we render the scenes with untextured meshes to contrast the quality of geometry reconstruction. While the quality difference is subtle, Fig. 2 indicates that increasing the number of optimization iterations results in more accurate geometry reconstruction and smoother surfaces.

Experiment	ScanNet [1]	Replica [5]		
		ATE $\downarrow$	ATE $\downarrow$	Accuracy $\downarrow$
a. Using shared feature planes for geometry and appearance.	7.49	0.65	0.99	1.08
b. Using only the coarse planes and discarding the fine ones.	7.53	0.97	1.12	1.29
c. Using only the fine planes and discarding the coarse ones.	8.27	0.72	1.00	1.09
d. Replacing the concatenation with a summation.	7.55	0.64	0.98	1.07
e. No importance sampling.	7.44	0.67	1.08	1.14
f. No color rendering.	8.31	0.68	1.03	1.08
g. One loss function for the whole truncation region.	8.28	0.71	1.01	1.10
h. No camera pose optimization during mapping.	11.27	4.85	2.23	2.21
i. Full ESLAM method.	<b>7.38</b>	<b>0.63</b>	<b>0.97</b>	<b>1.05</b>

Table 3. Ablation study of our design choices on the ScanNet [1] and Replica [5] datasets. The metrics are ATE RMSE (cm), reconstruction accuracy (cm), and reconstruction completion (cm). For the details of this study, see Sec. 2.

	Methods	Reconstruction (cm)				Localization (cm)	
		Depth L1 $\downarrow$	Acc. $\downarrow$	Comp. $\downarrow$	Comp. Ratio (%) $\uparrow$	ATE Mean $\downarrow$	ATE RMSE $\downarrow$
room0	iMAP* [7]	6.56 $\pm$ 0.39	5.89 $\pm$ 0.19	6.07 $\pm$ 0.22	66.55 $\pm$ 1.58	3.12 $\pm$ 0.84	5.23 $\pm$ 1.41
	NICE-SLAM [8]	2.77 $\pm$ 0.13	1.71 $\pm$ 0.03	1.69 $\pm$ 0.03	97.61 $\pm$ 0.09	1.43 $\pm$ 0.09	1.69 $\pm$ 0.17
	ESLAM (Ours)	<b>0.97 <math>\pm</math> 0.04</b>	<b>1.07 <math>\pm</math> 0.01</b>	<b>1.12 <math>\pm</math> 0.01</b>	<b>99.06 <math>\pm</math> 0.05</b>	<b>0.61 <math>\pm</math> 0.06</b>	<b>0.71 <math>\pm</math> 0.13</b>
room1	iMAP* [7]	5.97 $\pm$ 1.14	5.71 $\pm$ 0.31	5.57 $\pm$ 0.40	66.04 $\pm$ 3.45	2.54 $\pm$ 0.37	3.09 $\pm$ 0.48
	NICE-SLAM [8]	2.52 $\pm$ 0.11	1.36 $\pm$ 0.03	1.34 $\pm$ 0.04	98.60 $\pm$ 0.14	1.70 $\pm$ 0.29	2.13 $\pm$ 0.24
	ESLAM (Ours)	<b>1.07 <math>\pm</math> 0.07</b>	<b>0.85 <math>\pm</math> 0.01</b>	<b>0.88 <math>\pm</math> 0.01</b>	<b>99.64 <math>\pm</math> 0.06</b>	<b>0.56 <math>\pm</math> 0.02</b>	<b>0.70 <math>\pm</math> 0.02</b>
room2	iMAP* [7]	7.82 $\pm$ 0.94	6.34 $\pm$ 0.32	5.47 $\pm$ 0.27	69.87 $\pm$ 4.15	2.31 $\pm$ 0.20	2.58 $\pm$ 0.19
	NICE-SLAM [8]	3.54 $\pm$ 0.35	1.75 $\pm$ 0.06	1.71 $\pm$ 0.03	96.52 $\pm$ 0.26	1.41 $\pm$ 0.24	1.87 $\pm$ 0.39
	ESLAM (Ours)	<b>1.28 <math>\pm</math> 0.07</b>	<b>0.93 <math>\pm</math> 0.01</b>	<b>1.05 <math>\pm</math> 0.01</b>	<b>98.84 <math>\pm</math> 0.06</b>	<b>0.43 <math>\pm</math> 0.01</b>	<b>0.52 <math>\pm</math> 0.01</b>
office0	iMAP* [7]	7.57 $\pm$ 0.70	7.44 $\pm$ 0.26	5.13 $\pm$ 0.37	70.97 $\pm$ 3.52	1.69 $\pm$ 1.06	2.40 $\pm$ 1.05
	NICE-SLAM [8]	2.17 $\pm$ 0.14	1.43 $\pm$ 0.06	1.56 $\pm$ 0.05	96.30 $\pm$ 0.33	1.12 $\pm$ 0.22	1.26 $\pm$ 0.24
	ESLAM (Ours)	<b>0.86 <math>\pm</math> 0.02</b>	<b>0.85 <math>\pm</math> 0.01</b>	<b>0.96 <math>\pm</math> 0.01</b>	<b>98.34 <math>\pm</math> 0.05</b>	<b>0.42 <math>\pm</math> 0.03</b>	<b>0.57 <math>\pm</math> 0.04</b>
office1	iMAP* [7]	8.91 $\pm$ 0.65	10.34 $\pm$ 0.15	5.58 $\pm$ 0.24	72.08 $\pm$ 3.21	1.03 $\pm$ 0.17	1.17 $\pm$ 0.25
	NICE-SLAM [8]	2.41 $\pm$ 0.11	1.16 $\pm$ 0.07	1.15 $\pm$ 0.03	98.04 $\pm$ 0.19	0.74 $\pm$ 0.19	0.84 $\pm$ 0.17
	ESLAM (Ours)	<b>1.26 <math>\pm</math> 0.02</b>	<b>0.83 <math>\pm</math> 0.06</b>	<b>0.81 <math>\pm</math> 0.01</b>	<b>98.85 <math>\pm</math> 0.08</b>	<b>0.46 <math>\pm</math> 0.05</b>	<b>0.55 <math>\pm</math> 0.04</b>
office2	iMAP* [7]	11.04 $\pm$ 0.69	9.15 $\pm$ 0.39	6.27 $\pm$ 0.37	62.24 $\pm$ 2.62	3.99 $\pm$ 0.98	5.67 $\pm$ 1.82
	NICE-SLAM [8]	4.96 $\pm$ 0.58	1.83 $\pm$ 0.07	1.72 $\pm$ 0.03	96.96 $\pm$ 0.25	1.42 $\pm$ 0.10	1.71 $\pm$ 0.14
	ESLAM (Ours)	<b>1.71 <math>\pm</math> 0.07</b>	<b>1.02 <math>\pm</math> 0.01</b>	<b>1.09 <math>\pm</math> 0.01</b>	<b>98.60 <math>\pm</math> 0.12</b>	<b>0.47 <math>\pm</math> 0.03</b>	<b>0.58 <math>\pm</math> 0.09</b>
office3	iMAP* [7]	10.12 $\pm$ 1.31	7.14 $\pm$ 0.27	6.02 $\pm$ 0.20	66.07 $\pm$ 1.65	4.05 $\pm$ 0.93	5.08 $\pm$ 1.37
	NICE-SLAM [8]	4.91 $\pm$ 0.70	2.24 $\pm$ 0.17	2.17 $\pm$ 0.05	93.08 $\pm$ 0.40	2.31 $\pm$ 0.51	3.98 $\pm$ 1.79
	ESLAM (Ours)	<b>1.43 <math>\pm</math> 0.05</b>	<b>1.21 <math>\pm</math> 0.01</b>	<b>1.42 <math>\pm</math> 0.01</b>	<b>96.80 <math>\pm</math> 0.03</b>	<b>0.61 <math>\pm</math> 0.03</b>	<b>0.72 <math>\pm</math> 0.02</b>
office4	iMAP* [7]	7.85 $\pm$ 1.32	5.32 $\pm$ 0.18	6.51 $\pm$ 0.20	63.63 $\pm$ 1.39	1.93 $\pm$ 0.21	2.23 $\pm$ 0.35
	NICE-SLAM [8]	3.81 $\pm$ 0.74	2.09 $\pm$ 0.16	2.03 $\pm$ 0.17	95.00 $\pm$ 1.31	2.22 $\pm$ 0.68	2.82 $\pm$ 0.71
	ESLAM (Ours)	<b>1.06 <math>\pm</math> 0.08</b>	<b>1.15 <math>\pm</math> 0.02</b>	<b>1.27 <math>\pm</math> 0.01</b>	<b>97.65 <math>\pm</math> 0.14</b>	<b>0.52 <math>\pm</math> 0.02</b>	<b>0.63 <math>\pm</math> 0.03</b>
Average	iMAP* [7]	8.23 $\pm$ 0.88	7.16 $\pm$ 0.26	5.83 $\pm$ 0.27	67.17 $\pm$ 2.70	2.59 $\pm$ 0.58	3.42 $\pm$ 0.87
	NICE-SLAM [8]	3.29 $\pm$ 0.33	1.66 $\pm$ 0.07	1.63 $\pm$ 0.05	96.74 $\pm$ 0.36	1.56 $\pm$ 0.29	2.05 $\pm$ 0.45
	ESLAM (Ours)	<b>1.18 <math>\pm</math> 0.05</b>	<b>0.97 <math>\pm</math> 0.02</b>	<b>1.05 <math>\pm</math> 0.01</b>	<b>98.60 <math>\pm</math> 0.07</b>	<b>0.52 <math>\pm</math> 0.03</b>	<b>0.63 <math>\pm</math> 0.05</b>

Table 4. Per-scene quantitative comparison of our proposed ESLAM with existing NeRF-based dense visual SLAM models on the Replica dataset [5] for both reconstruction and localization accuracy. The results are the average and standard deviation of five independent runs on each scene of the Replica dataset [5]. Our method outperforms previous works by a high margin and has lower variances, indicating it is also more stable from run to run. The evaluation metrics for reconstruction are L1 loss (cm) between rendered and ground truth depth maps of 1000 random camera poses, reconstruction accuracy (cm), reconstruction completion (cm), and completion ratio (%). The evaluation metrics for localization are mean and RMSE of ATE (cm) [6]. It should also be noted that our method runs up to  $\times 10$  faster on this dataset (see Sec. 4.2 in the main paper for runtime analysis).

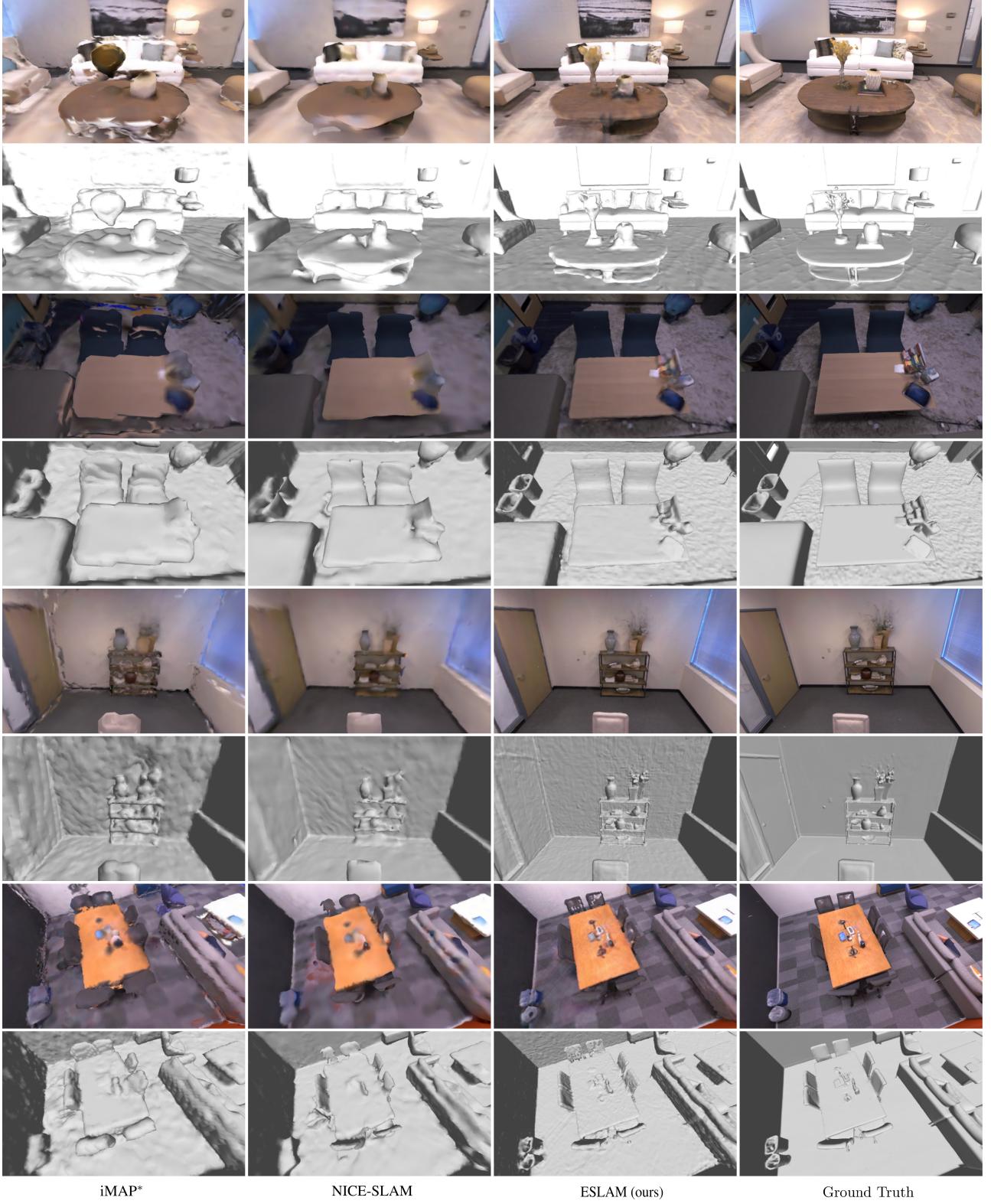


Figure 1. Qualitative comparison of our method’s scene reconstruction with iMAP\* [7] and NICE-SLAM [8] on Replica [5]. Our method produces more accurate detailed geometry as well as higher-quality textures. The scenes are rendered with both textured and untextured meshes and the ground truth textured images are rendered with the ReplicaViewer software [5]. It should also be noted that our method runs up to  $\times 10$  faster on this dataset (see Sec. 4.2 in the main paper for runtime analysis).

Method	Optimization Iterations	Acc. (cm)↓	Comp. (cm)↓	ATE (cm)↓	FPT (s)↓
iMAP* [7]	-	7.16	5.83	3.42	5.20
NICE-SLAM [8]	-	1.66	1.63	2.05	2.10
ESLAM (ours)	$Iter_m = 15, Iter_t = 8$	0.97	1.05	0.63	<b>0.18</b>
ESLAM x2 (ours)	$Iter_m = 30, Iter_t = 16$	0.95	1.03	0.42	0.35
ESLAM x10 (ours)	$Iter_m = 150, Iter_t = 80$	<b>0.92</b>	<b>1.01</b>	<b>0.31</b>	1.72

Table 5. Quantitative analysis of the effect of the number of optimization iterations during mapping and tracking on our method’s reconstruction and localization accuracy.  $Iter_m$  stands for the number of optimization iterations during mapping, and  $Iter_t$  denotes the number of optimization iterations during tracking. The evaluation metrics are reconstruction accuracy (cm), reconstruction completion (cm), and ATE RMSE (cm) [6]. Average Frame Processing Time (FTP) is also shown to highlight the trade-off between the accuracy and throughput of our method. For reference, we reiterate the performance of the existing approaches, iMAP\* [7] and NICE-SLAM [8]. It should be noted that even ESLAM x10 runs faster than the existing state-of-the-art method, NICE-SLAM [8]. Refer to Sec. 5 for the details of this experiment, and see Fig. 2 for the qualitative analysis.

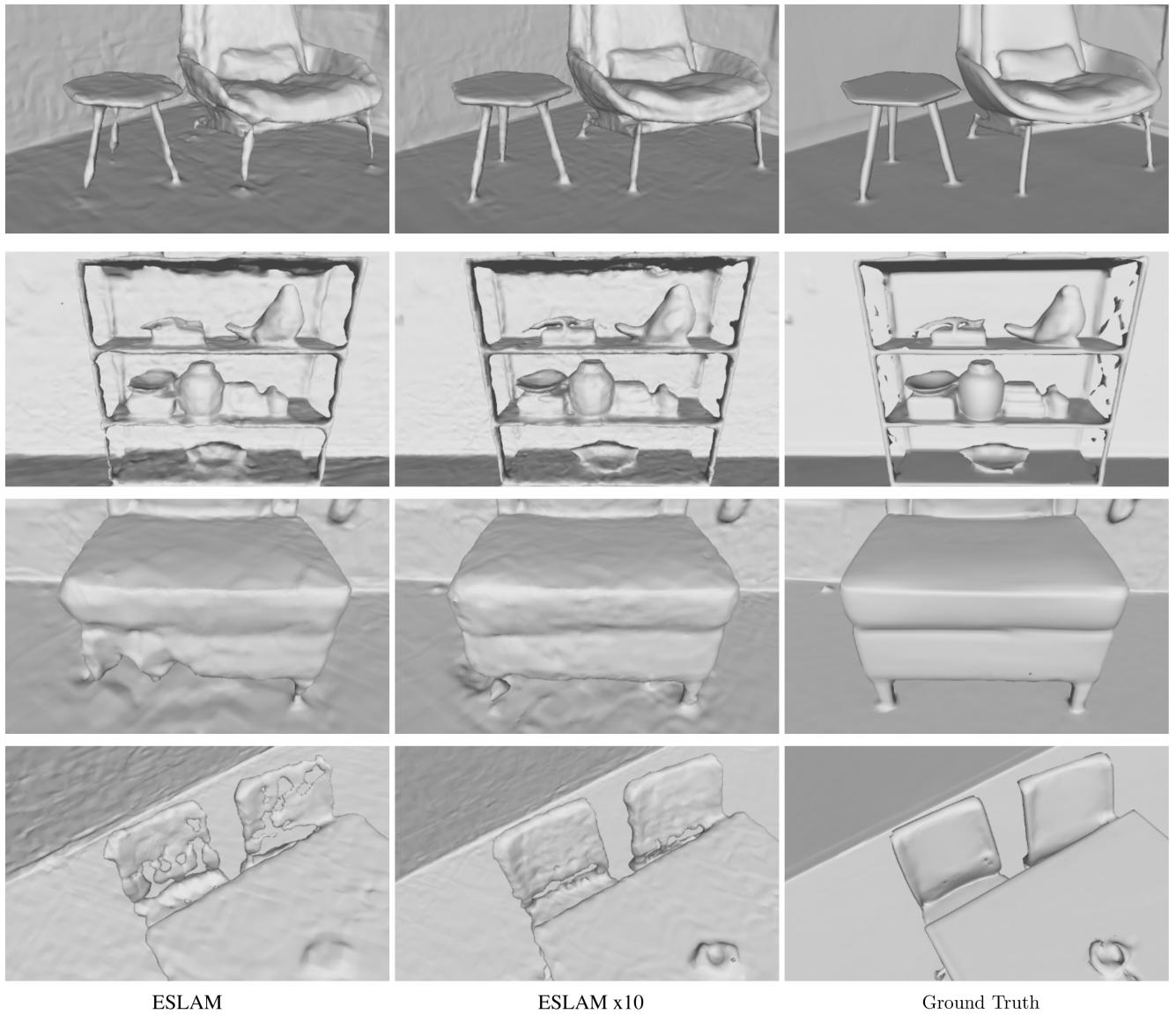


Figure 2. Qualitative analysis of the effect of the number of optimization iterations during mapping and tracking on our method’s reconstruction quality. ESLAM x10 is our method when we multiply the number of optimization iterations by 10. Refer to Sec. 5 for the details of this experiment, and see Tab. 5 for the quantitative analysis.

## References

- [1] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. [1](#), [2](#), [3](#)
- [2] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014. [1](#)
- [3] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. [1](#)
- [4] Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, 1985. [1](#)
- [5] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. [1](#), [2](#), [3](#), [4](#)
- [6] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 573–580. IEEE, 2012. [1](#), [3](#), [5](#)
- [7] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021. [2](#), [3](#), [4](#), [5](#)
- [8] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022. [2](#), [3](#), [4](#), [5](#)