# IBD-SLAM: Learning Image-Based Depth Fusion for Generalizable SLAM

Minghao Yin[1] Shangzhe Wu[2] Kai Han[1†]
[1]The University of Hong Kong [2]University of Oxford

## Abstract

*In this paper, we address the challenging problem of visual SLAM with neural scene representations. Recently, neural scene representations have shown promise for SLAM to produce dense 3D scene reconstruction with high quality. However, existing methods require scene-specific optimization, leading to time-consuming mapping processes for each individual scene. To overcome this limitation, we propose IBD-SLAM, an **I**mage-**B**ased **D**epth fusion framework for generalizable **SLAM**. In particular, we adopt a Neural Radiance Field (NeRF) for scene representation. Inspired by multi-view image-based rendering, instead of learning a fixed-grid scene representation, we propose to learn an image-based depth fusion model that fuses depth maps of multiple reference views into a $xyz$-map representation. Once trained, this model can be applied to new, uncalibrated monocular RGBD videos of unseen scenes, without the need for retraining, and reconstructs full 3D scenes efficiently with a light-weight pose optimization procedure. We thoroughly evaluate IBD-SLAM on public visual SLAM benchmarks, outperforming the previous state-of-the-art while being 10× faster in the mapping stage. Project page: https://visual-ai.github.io/ibd-slam.*

## 1. Introduction

Simultaneous Localization and Mapping (SLAM) refers to the task of creating a map of an unknown environment while simultaneously determining the location of the camera or robot within that environment. Visual SLAM specifically utilizes visual information (such as images from a camera) to perform this task. This has been a long-standing problem in computer vision, and much of the recent effort focuses on improving the accuracy, robustness, and efficiency. Popular approaches include feature-based methods [16, 24], direct methods [11, 28], and, more recently, learning-based methods [10, 44]. In particular, neural fields have emerged as a powerful representation for producing high-quality dense 3D scene reconstructions, such as NICE-SLAM [53], but such methods often require a lengthy training process and typically only optimize over a single scene at a time.

In this paper, our goal is to design a learning-base SLAM framework that can efficiently generalize to arbitrary scenes unseen during training. In particular, we draw inspiration from a recent work, IBRNet [43], which renders novel views of a 3D scene through a learned image-based fusion model that generalizes to arbitrary test scenes. We borrow this powerful idea and propose a learning-based generalizable SLAM framework, dubbed *IBD-SLAM*. Unlike IBRNet, which learns to fuse RGB images, we propose to learn an image-based *depth* fusion model that directly fuses depth maps from multiple reference frames using a shared canonical $xyz$-map representation. Crucially, this depth fusion model is not specific to one particular scene; it is trained on a large collection of monocular videos, allowing for zero-shot generalization in novel scenes without retraining.

Specifically, the framework interweaves mapping (*i.e.*, scene recovery) and tracking (*i.e.*, camera pose estimation) steps, similar to typical visual SLAM systems. During each mapping step, given a current frame, we retrieve a set of temporal neighboring frames and convert their correspondence depth maps into $xyz$-maps, given the current estimates of their relative poses. We then task the model to predict the $xyz$-map of the current frame from these reference $xyz$-maps of the neighboring frames, through image-based rendering. This eliminates the need for scene-dependent features like the fixed feature grid in NICE-SLAM, allowing the model to generalize to unseen scenes at test time. During each tracking step, we establish the 2D geometric correspondences, and optimize the camera poses by minimizing the distances of matches of the 3D coordinates in the $xyz$-maps.

After pre-training, our model can be applied to new uncalibrated monocular RGBD videos of unseen scenes at test time, without the need to retrain the model to get any scene-dependent representations, and only needs to run a fast optimization process to obtain the poses, resulting in a strong generalization capability and a heavily reduced computation overhead.

We thoroughly evaluate our framework on public visual SLAM benchmarks, including Replica [32], ScanNet [6],

†Corresponding author.

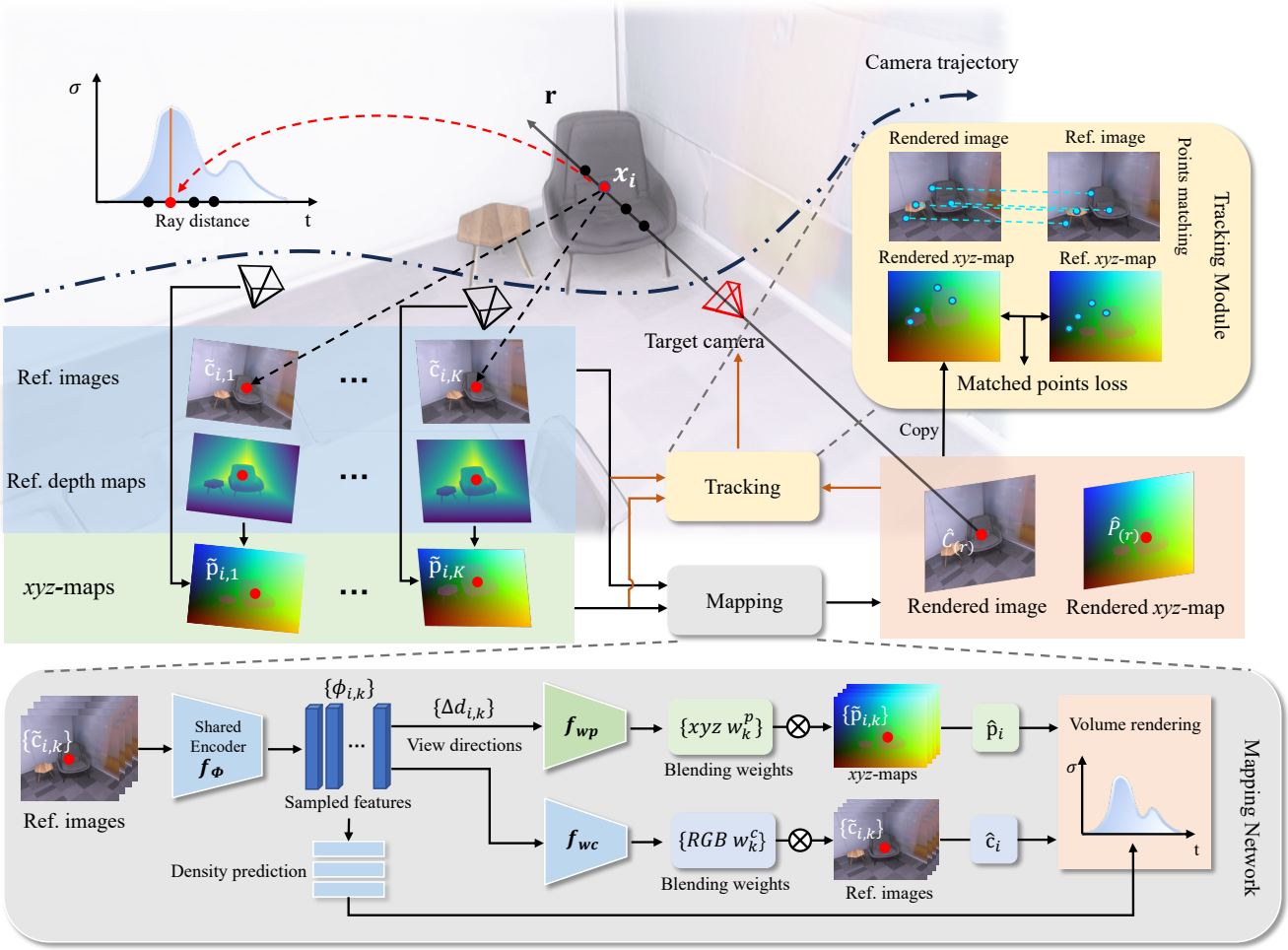


Figure 1. **Mapping:** To render the target view, we select a set of $K$ reference views. We then extract the reference image feature $\Phi_k$ by utilizing a shared encoder and derive reference views' *xyz*-maps based on the depths and reference cameras. For each sampled point $x_i$ on the ray $\mathbf{r}$, we project it onto all $K$ reference views, which allows us to sample the feature vector $\phi_{i,k}$ from the image feature $\Phi_k$, as well as sample the corresponding color $\tilde{c}_{i,k}$ and *xyz* $\tilde{P}_{i,k}$ from the reference views. We then aggregate $\tilde{c}_{i,k}$ and $\tilde{P}_{i,k}$ by fusing the feature vectors $\phi_{i,k}$ and the relative viewing directions $\Delta\mathbf{d}_{i,k}$ to predict the blending weights $w_k^c$ for color and $w_k^p$ for *xyz* values. The final color and *xyz* values are obtained as a weighted sum of their correspondences in reference views. The $\sigma$ value in the rendering process is predicted based on the feature $\phi_{i,k}$ and viewing direction. **Tracking:** After rendering the target color and *xyz*-maps, we utilize RGB images to establish geometric correspondences between the target novel view and the reference views. By minimizing the distances between the matched *xyz* points, we can optimize the target camera pose.

and TUM [34], and show that our approach performs on par with other state-of-the-art methods that require per-scene model training in terms of both mapping and tracking. Notably, as a result of the generalizable formulation, our method runs 10× faster than the previous state-of-the-art during the mapping stage, while maintaining similar efficiency for tracking.

## 2. Related Work

**Dense SLAM.** Classic SLAM methods SLAM stands for Simultaneous Localization and Mapping, which refers to the task of building a map of an unknown environment while at the same time localizing the robot's pose within that map. Classic SLAM, also known as sparse SLAM, is a traditional approach that uses feature-based methods to detect and track features in the environment, such as landmarks or edges [10, 16, 24, 25]. Dense SLAM, on the other hand, is a newer approach that uses direct methods to estimate the scene geometry and the robot's pose, without relying on feature detection and tracking. Several works use view-centric representations, including DTAM [28], DeepV2D [38], BA-Net [37], and Droid-SLAM [39], as well as other related works such as Demon [41], DeepTAM [51], NodeSLAM [36], DeepFactors [5], and SceneCode [50]. Meanwhile, world-centric maps anchor the geome-

try in uniform world coordinates, often using voxel grids or surfels [30, 46], with occupancy and Truncated Signed Distance Field (TSDF) [4] being common methods for storing geometry information. For RGBD SLAM systems, depth fusion is a crucial step in 3D reconstruction and has been the focus of many research efforts in computer vision and robotics. KinectFusion [27] was one of the pioneering works in depth fusion. It uses a volumetric approach to represent the scene as a 3D grid of voxels and fuses depth measurements into this grid as the camera moves through the environment, while DynamicFusion [26] is an extension of KinectFusion that allows for real-time reconstruction of dynamic scenes. More recently, VolumeFusion [3] focuses on fusing high-resolution depth maps by TSDF volume.

**SLAM with Neural Fields.** Recently neural implicit representation methods have shown great potential for 3D modeling, rendering, and reconstruction [2, 17, 19, 20, 22]. SLAM with implicit Neural Fields is also a promising direction, which utilizes neural networks and fields to represent and process the sensor data for 3D mapping and localization. iMAP [35] is one of the pioneering SLAM systems that adopt a neural field representation for tracking and mapping. However, iMAP relies on a single multi-layer perceptron to represent the scene, deeming it less effective for large or complex structures. NICE-SLAM [53] addresses this limitation by introducing multi-resolution feature grids to better capture geometry information, resulting in improved performance for larger scenes and more detailed structures. NICER-SLAM [52] builds upon the multi-resolution feature grids introduced in NICE-SLAM, but it utilizes Signed Distance Function (SDF) representations instead of occupancy representations. The system takes a collaborative approach by training different aspects of the map, which allows for more robust and flexible scene representation. NICER-SLAM demonstrates encouraging results using only RGB information. Besides, Co-SLAM [42] employs multi-resolution hash-grids to represent local features, improving reconstruction efficiency and accuracy. ESLAM [14] leverages TSDF for implicit representation and employs axis-aligned feature planes to store features, resulting in improved mapping and tracking performance.

**Generalizable Novel View Synthesis.** Generalizable Novel View Synthesis refers to the task of synthesizing novel views of a scene from a limited set of input views, while also being able to generalize to new and unseen scenes at test time without retraining. As a representative work of this flavor, General Radience Field (GRF) [40] uses an implicit neural function that models 3D scenes as a general radiance field and utilizes multi-view geometry to obtain internal representations that are multi-view consistent, enabling it to represent and render complex 3D scenes using only 2D observations. Another work IBRNet [43] also uses a neural network architecture that synthesizes new views of a scene by combining image features from reference views to predict density and radiance. Other notable works tackling generalizable novel view synthesis include MVSNeRF [1] and PixelNeRF [48]. In this work, we draw inspiration from IBRNet to devise a generalizable depth fusion model, which will be detailed next.

## 3. Method

Given a collection of calibrated monocular RGB videos, denoted as $\mathcal{D} = \{S_1, \cdots, S_N\}$, where each video consists of a sequence of images $S_i = \{I_k\}_{k=1}^{K}$, serving as pre-training data, our goal is to learn a generalizable function. This function should have the capability, during testing, to directly infer the 3D geometry and generate novel views from a new monocular RGBD video sequence $S'$ of an unseen scene, requiring only a fast optimization of the camera poses.

To achieve this goal, we introduce IBD-SLAM, a novel framework for generalizable SLAM. Our approach leverages the power of NeRF [22] for scene representation. Drawing inspiration from multi-view image-based rendering in IBRNet [43], we introduce a learning-based depth fusion technique that derives $xyz$-maps from inferred depth maps. Once trained, our model can be applied to new, uncalibrated monocular RGBD videos of unseen scenes without retraining. Only optimizing pose parameters is required for efficient processing of new scenes.

In the following, we first give a brief background overview of NeRF and IBRNet in Sec. 3.1, followed by the details of our proposed IBD-SLAM framework, including mapping based on $xyz$-maps in Sec. 3.2, tracking based on $xyz$-maps in Sec. 3.3, and training objectives in Sec. 3.4.

### 3.1. Neural Scene Representation

**Neural Radiance Fields.** NeRF [22] has recently emerged as a powerful 3D representation for novel view synthesis, owing to its remarkable capability in modeling the complex appearance of 3D scenes, which leads to photo-realistic novel rendering results. It represents a 3D scene as a continuous function $f$, parametrized by an MLP, which takes in coordinates of a point location $\mathbf{x}$ and a viewing direction $\mathbf{d}$ and predicts its color and density: $(\mathbf{c}, \sigma) = f(\mathbf{x}, \mathbf{d})$. In order to render an image from a given camera pose, we cast a ray from the camera location through each pixel in the image plane and into the 3D scene. Let $\mathbf{o}$ denote the camera location and $\mathbf{d}$ the ray direction, we can represent any point on the ray as $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, where $t$ is the distance from the camera. We then sample a set of points $\{\mathbf{x}_i\}_{i=1}^{N}$ along the ray, and query the MLP to obtain the colors $\mathbf{c}_i$ and densities $\sigma_i$. Using the volume rendering equation [22], the color of

the ray $\mathbf{r}$ is given by:

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

where $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$ is the accumulated transmittance along the ray up to a distance $t$, and $\delta_i = t_{i+1} - t_i$ is the distance between consecutive samples. Given a set of posed multi-view images as training data, the RGB rendering loss [22] is then employed to train the model, which aims to minimize the color error between the rendered RGB color and the ground-truth RGB color:

$$\mathcal{L}_{\text{rgb}} = \frac{1}{|\Gamma|} \sum_{\mathbf{r} \in \Gamma} ||\hat{C}(\mathbf{r}) - C(\mathbf{r})||_2, \tag{2}$$

where $\hat{C}$ denotes the volume rendered RGB color, $C$ denotes the ground truth color information, and $\Gamma$ represents all sampled rays.

**Multi-view Image-Based Rendering.** One key limitation of NeRF is the time-consuming per-scene optimization process on each individual scene. To alleviate this, IBRNet [43] learns a function $f_{\text{IBR}}$ that directly predicts the color and density of a point in space by simply aggregating image features sampled from multi-view reference images. Similarly to the original NeRF, to render a pixel in a target image, IBRNet samples a set of points $\{\mathbf{x}_i\}_{i=1}^{N}$ along the camera ray $\mathbf{r}$. Instead of optimizing a scene-specific NeRF to obtain the color and density of each point $\mathbf{x}_i$, IBRNet proposes to directly predict them from the reference images. Specifically, given a set of neighboring reference views $\mathcal{I} = \{I_k\}_{k=1}^{K}$, IBRNet first extracts their feature maps $\Phi_k$ using an image encoder network: $\Phi_k = f_\Phi(I_k)$. Let $\pi_k$ denote the camera projection of view $k$. The 3D point $\mathbf{x}_i$ is projected into each of these reference views and the feature maps are sampled at the projected pixel location $\mathbf{u}_{i,k} = \pi_k(\mathbf{x}_i)$ to obtain a feature vector from each reference image: $\phi_{i,k} = \Phi_k(\mathbf{u}_{i,k})$. By aggregating the features from all reference views, the densities $\{\sigma_i\}$ of all the points sampled along the ray can be predicted through a ray transformer. To predict the color $\mathbf{c}_i$, an MLP takes in the reference features $\{\phi_{i,k}\}$ and relative viewing directions $\{\Delta\mathbf{d}_{i,k}\}$ w.r.t. the target view and predicts a set of weights $\{w_k^{\text{c}}\}$, which are used to blend the reference pixels $\tilde{\mathbf{c}}_{i,k} = I_k(\mathbf{u}_{i,k})$ to obtain the color of the target point:

$$\hat{\mathbf{c}}_i = \sum_k w_k^{\text{c}} \tilde{\mathbf{c}}_{i,k}. \tag{3}$$

Finally, IBRNet aggregates the colors $\{\hat{\mathbf{c}}_i\}$ of all sample points $\{\mathbf{x}_i\}$ along ray $\mathbf{r}$ to obtain the target pixel color $\hat{C}(\mathbf{r})$ using Eq. (1).

The main advantage of IBRNet comes from the generalization capability of the learned function $f_{\text{IBR}}$, which infers the color and density of a 3D point $\mathbf{x}$ directly from multi-view image features: $(\hat{C}, \sigma) = f_{\text{IBR}}(\mathbf{x}; \{I_k\}, \{\Phi_k\}, \{\Delta\mathbf{d}_k\})$. After training, given multi-view images of a new test scene at inference time, IBRNet can directly infer novel views by querying feature extractor $f_\Phi$ and $f_{\text{IBR}}$ in a feed-forward pass, without the need for any further optimization. In this work, we draw inspiration from this multi-view image-based rendering pipeline to develop our IBD-SLAM framework for efficient visual SLAM without the need for per-scene model optimization.

### 3.2. Mapping Based on $xyz$-maps

For mapping, taking inspiration from [43], one straightforward idea is to directly apply IBRNet on the image sequence and obtain the 3D scene by running a surface reconstruction method on the density field queried from the model. However, due to the lack of scene geometry during rendering, this method leads to unsatisfactory results.

**$xyz$-maps for Depth Fusion.** A more plausible idea is to treat the depth maps $D_k \in \mathbb{R}^{1\times H\times W}$ as analogous to images $I_k \in \mathbb{R}^{3\times H\times W}$, where each pixel stores the *3D location* of the surface point in the form of a distance from the camera, instead of its RGB color. With this analogy, we can render novel-view depth images using the same approach as rendering color images in IBRNet. This allows us to obtain surface geometry through image-based depth fusion. Consequently, we can learn another function, denoted as $f_{\text{IBD}}$, which directly infers the 3D surface location by predicting a set of blending weights and fusing depth values from multiple views. Similar to $f_{\text{IBR}}$, this function is also generalizable to test scenes. However, depth maps, usually represented in view-dependent camera coordinates, inherently suffer from the characteristic of multi-view inconsistency. This poses a significant challenge for IBRNet in seamlessly synthesizing a novel view image from reference inputs. As shown in Fig. 6, the depth map reconstruction result exhibits geometry claws. To address this issue, we propose to use $xyz$-maps, which are represented in world coordinates. By doing so, we enhance the robustness of the multi-view fusion process. In addition, we also include an $f_{\text{IBR}}$ module following [43] that operates on the color images to learn to render the scene appearance. Furthermore, $xyz$-maps offer promising potential for camera localization, as will be discussed in Sec. 3.3. Specifically, we first construct a set of $xyz$-maps $P_k \in \mathbb{R}^{3\times H\times W}$ from the depth maps by back-projecting the pixels to 3D, to transform the view-dependent depth values into the $xyz$ coordinate values in the canonical world coordinate system. Given a depth value $d_{uv} = D(u, v)$ at a pixel location $(u, v)$, we map it to a 3D point $P_{uv}$ by inverting the projection function: $d_{uv} p_{uv} = K[R|t]P_{uv}$, where $p_{uv} = (u, v, 1)$, and $K$ and $[R|t]$ are the intrinsic and extrinsic parameters of the camera associated with the depth map. Note that the cam-

era extrinsics are not available as input and are obtained by a pose estimation step to be introduced in Sec. 3.3.

**Feed-forward Mapping.** Given reference view $xyz$-maps, we can then render the $xyz$-map of a *novel* view using the volume rendering equation. For each point $\mathbf{x}_i$ on a visual ray $\mathbf{r}$, we project it to each neighboring view $k$, and sample the image feature $\phi_{i,k}$ and the $xyz$ value $\tilde{\mathbf{p}}_{i,k} = P_k(\mathbf{u}_{i,k})$ at the projected pixel locations $\mathbf{u}_{i,k} = \pi_k(\mathbf{x}_i)$. Crucially, note that the sampled $xyz$ value $\tilde{\mathbf{p}}_{i,k}$ is not necessarily the same as sample point location $\mathbf{x}_i$, as $\tilde{\mathbf{p}}_{i,k}$ describes the *surface point* observed from pixel $\mathbf{u}_{i,k}$, whereas $\mathbf{x}_i$ is an arbitrary point in the 3D space and may or may not lie on the surface. Subsequently, we input all the features $\phi_{i,k}$ into an MLP $f_{wp}$ to predict another set of blending weights $w_k^p$. The predicted $xyz$ value for $\mathbf{x}_i$ is then obtained by blending the sampled $xyz$ values from neighboring views with these weights:

$$\hat{\mathbf{p}}_i = \sum_k w_k^p \tilde{\mathbf{p}}_{i,k}. \tag{4}$$

All the predicted $xyz$ values $\hat{\mathbf{p}}_i$ along the target visual ray $\mathbf{r}$ are then aggregated via the same volume rendering equation as in Eq. (1), recycling the densities $\sigma_i$ predicted from $f_{\text{IBR}}$:

$$\hat{P}(\mathbf{r}) = \sum_{i=1}^{N} T_i \left(1 - \exp(-\sigma_i \delta_i)\right) \hat{\mathbf{p}}_i, \tag{5}$$

which represents the reconstructed surface location observed through $\mathbf{r}$. We denote this image-based depth fusion network as $f_{\text{IBD}}$, which essentially predicts an estimated surface location for each 3D point $\mathbf{x}$ by fusing the depth values from reference views: $\hat{P} = f_{\text{IBD}}(\mathbf{x}; \{P_k\}, \{\Phi_k\})$. Unlike other methods [52, 53] based on the implicit representation requiring per-scene iterative learning to obtain the feature grid representing the specific scene, once trained, our method can generalize to new scenes at test time, without the need for model retraining or fine-tuning.

### 3.3. Tracking Based on $xyz$-maps

For camera pose tracking, we adopt a method leveraging the pixel correspondences and $xyz$-maps. Specifically, we first detect key points using SuperPoint [7] and then use matching techniques such as SuperGlue [29] to establish correspondences between the reference frames and the novel frame. To optimize the camera pose of the novel view, we minimize the following loss:

$$\mathcal{L}_t = \sum_{k=1}^{K} \sum_{\mathbf{r} \in \Omega} ||\hat{P}(\mathbf{r}) - P_k(\mathbf{r}_k)||^2, \tag{6}$$

where $\Omega$ denotes all selected rays in the novel view. The matched rays in the $k$-th reference view are denoted as $\mathbf{r}_k$. $\hat{P}$ is the predicted $xyz$-map by $f_{\text{IBD}}$ under the current (target) camera pose, and $P_k(\mathbf{r}_k)$ represents the $xyz$ values of the matched points in the $k$-th view. Let the rotation and translation of the current pose w.r.t. the world coordination system be $(R_c, t_c)$. Together with the poses $\{(R_k, t_k)\}$ for the reference views, we can obtain $(R_c, t_c)$ by $\text{argmin}_{(R_c,t_c)} \mathcal{L}_t$. Note that during the inference process, the poses for the reference views are estimated progressively. Following common practice, only the pose of the initial view is provided, and as the process continues, the poses for the remaining views are gradually estimated. As the SLAM process unfolds, the "current" views progressively transition into "reference" views.

After successfully tracking the camera pose, we introduce novel views by incorporating random perturbations into camera translations and rotations. The novel view camera rotation is updated as $R_n := R_c + \delta R$, and the novel view camera translation as $t_n := t_c + \delta t$. Utilizing the novel camera pose $(R_n, t_n)$, we can infer the novel $xyz$-map $\hat{P}_n$ by the pre-trained $f_{\text{IBD}}$. The rendering of novel-view $xyz$-maps enhances geometric details and addresses potential gaps between the provided depth images. As a result, our model demonstrates strong capability in generating high-quality scene reconstructions.

### 3.4. Training Objectives

Here we introduce the training objectives employed in our framework. To train our model, we adopt the RGB rendering loss [22] $\mathcal{L}_{\text{rgb}}$ defined in Eq. (2) to minimize the color error between the rendered RGB color and the ground-truth RGB color. We also include the depth consistency loss [49], which has been shown to be effective in improving reconstruction details. The loss is defined as:

$$\mathcal{L}_{\text{depth}} = \sum_{\mathbf{r} \in \Gamma} ||(w\hat{D}(\mathbf{r}) + q) - \bar{D}(\mathbf{r})||_2, \tag{7}$$

where $w$ and $q$ are scale and shift aligning the rendered depth map and the depth estimated map by the monocular predictor [9]. $w$ and $q$ can be solved with a least-squares criterion. The rendered depth value is obtained by:

$$\hat{D}(\mathbf{r}) = \sum_{i=1}^{N} T_i \left(1 - \exp(-\sigma_i \delta_i)\right) d_i. \tag{8}$$

Similarly, the normal regularization loss [49] is also used to improve local geometry, by minimizing the difference between the normal $\bar{N}$ calculated by the monocular predictor [9] and the rendered normal $\hat{N}$ in a similar way as in Eq. (8). The loss is written as:

$$\mathcal{L}_{\text{normal}} = \sum_{\mathbf{r} \in \Gamma} \|\hat{N}(\mathbf{r}) - \bar{N}(\mathbf{r})\|_1 + \|1 - \hat{N}(\mathbf{r})^\top \bar{N}(\mathbf{r})\|_1. \tag{9}$$

After obtaining the rendered depth maps of $K$ reference views, they can be converted to $xyz$-maps $\{P_k\}$. The $xyz$-map $\hat{P}$ of the current view can then be predicted by $f_{\text{IBD}}$. We introduce the $xyz$-loss as follows:

$$\mathcal{L}_{\text{xyz}} = \sum_{\mathbf{r}\in\Gamma} ||\pi_c(\hat{P}(\mathbf{r})) - \hat{D}(\mathbf{r})||_2 + ||(w\cdot\pi_c(\hat{P}(\mathbf{r})) + q) - \bar{D}(\mathbf{r})||_2, \tag{10}$$

where $\pi_c$ denotes the projection matrix from 3D world coordinates location to current camera coordinates depth value, and $\bar{D}$ denotes the monocular depth estimation.

Moreover, we leverage two additional regularization terms following [52] to further enforce the geometric consistency, namely an RGB warping loss and an optical flow loss. For the RGB warping loss, given a sampled ray $\mathbf{r}_i$ passing through pixel $\mathbf{u}_i$ in frame $i$, we initially compute its depth value by neural rendering, followed by the process of back-projecting $\mathbf{u}_i$ to world coordinates to obtain its 3D coordinates. We project the 3D coordinates onto another reference frame $j$ at pixel $\mathbf{u}_{i\to j}$. The corresponding ray passing through $\mathbf{u}_{i\to j}$ is represented as $\mathbf{r}_{i\to j}$. The reference frame set of frame $i$ is denoted as $\mathcal{K}_i$. The warping loss is thus formulated as follows:

$$\mathcal{L}_{\text{warp}} = \sum_{\mathbf{r}_i\in\Gamma} \sum_{j\in\mathcal{K}_i} \left\| C\left(\mathbf{r}_i\right) - C\left(\mathbf{r}_{i\to j}\right) \right\|_1. \tag{11}$$

For the optical flow loss, we compute the optimal flow with GMFlow [47], denoted by `Flow`. The loss is defined as:

$$\mathcal{L}_{\text{flow}} = \sum_{\mathbf{r}_i\in\Gamma} \sum_{j\in\mathcal{K}_i} \left\| \left(\mathbf{u}_i - \mathbf{u}_j\right) - \texttt{Flow}\left(\mathbf{u}_{i\to j}\right) \right\|_1, \tag{12}$$

where $\mathbf{u}_i$ is the pixel location corresponding to $\mathbf{r}_i$ and $\mathbf{u}_i$ is the pixel location corresponding to $\mathbf{r}_{i\to j}$.

Finally, the overall loss is written as:

$$\mathcal{L} = \mathcal{L}_{\text{rgb}} + \lambda_d\mathcal{L}_{\text{depth}} + \lambda_n\mathcal{L}_{\text{normal}} + \lambda_x\mathcal{L}_{\text{xyz}} + \lambda_w\mathcal{L}_{\text{wrap}} + \lambda_f\mathcal{L}_{\text{flow}},$$

where the weights $\lambda_d$, $\lambda_x$, $\lambda_n$, $\lambda_f$ and $\lambda_w$ are set to 0.1, 0.5, 0.04, 0.002 and 0.2 respectively in our experiments.

# 4. Experiments

We evaluate our IBD-SLAM on various datasets and compare it with the previous state-of-the-art methods.

## 4.1. Experimental Details

**Training Datasets.** We pre-train our model using both synthetic and real-world datasets. For the synthetic datasets, we use 4 object 3D scans from DeepVoxels [31], with each scene consisting of 497 views, as well as 1,024 object scanning results from the Google Scanned Objects [8], with 250 views for each scene. For the real-world datasets, we include 100 scenes from the Spaces Dataset [12], with 100 scanned RGB images per scene, and 24 scenes from the Local Light Field Fusion Dataset [21], with around 20-30 views per scene. In addition, we use data collected by IBRNet [43], which includes 3D scanning data from 67 scenes.

| | Depth L1↓ | Acc.↓ | Comp.↓ | Comp. Ratio↑ | PSNR↑ | SSIM↑ |
|---|---|---|---|---|---|---|
| iMAP [35] | 4.39 | 4.77 | 5.02 | 75.5 | 18.26 | 0.750 |
| DI-Fusion [13] | 19.21 | 16.33 | 9.19 | 78.1 | - | - |
| Orb-SLAM2 [25] | 3.35 | 3.36 | 3.60 | 86.3 | - | - |
| NICE-SLAM [53] | 2.49 | 2.42 | 2.65 | 90.3 | 24.7 | 0.844 |
| ESLAM [14] | **1.29** | 2.34 | 2.14 | **94.7** | 25.8 | 0.869 |
| Co-SLAM [42] | 1.60 | 2.21 | 2.36 | 92.7 | 27.9 | 0.882 |
| Ours | 1.53 | **1.83** | **2.02** | 93.8 | **28.5** | **0.893** |

Table 1. **Reconstruction and novel view synthesis results on Replica Datasets [32].**

**Evaluation Datasets.** To evaluate our model, we utilize multiple datasets. Specifically, we select 8 scenes from the Replica Dataset [32], each containing 2,000 frames of RGBD input and corresponding ground-truth camera poses. Additionally, we employ the TUM-RGBD Dataset [33] and the ScanNet Dataset [6] for further evaluation of our model.

**Implementation Details.** In our implementation, we use a server with an Intel(R) Xeon(R) Silver 4314 CPU @ 2.40GHz and an NVIDIA RTX 3090 GPU. For ray sampling, we employ 64 points for coarse sampling and another 64 points for importance sampling. The size of ray batch of reference RGBD image aggregation is set to be 2,000. During the mapping process, we utilize the 10 temporal nearby reference frames to predict the target frame. For the Replica Dataset [32], we sample 200 pixels for tracking. For the ScanNet Dataset [6], we sample 1,000 pixels for tracking. Additionally, we sample 2,000 pixels for tracking in the TUM-RGBD Dataset [33]. We apply the Poisson surface reconstruction [15] to reconstruct the surface mesh from the rendered point cloud. We synthesize a novel frame after tracking every 10 frames.

## 4.2. Main Comparison

**Baselines.** We compare our IBD-SLAM with state-of-the-art learning based SLAM methods using sparse RGB-D inputs, including DI-Fusion [26], iMAP [35], NICE-SLAM [53], ESLAM [14], Co-SLAM [42]. Among them, iMAP and NICE-SLAM employ the plain NeRF-based scene representations. Our method also adopts the plain NeRF-based scene representations. Other methods employ different variants of neural scene representations. DI-Fusion utilizes PLIVox [13] for scene representation, ESLAM [14] employs axis-aligned planes feature representation and TSDF-based volume rendering [6], Co-SLAM [42] adopts a hash-based grid [23] for feature encoding and TSDF for scene representation, differing from the plain NeRF-based methods.

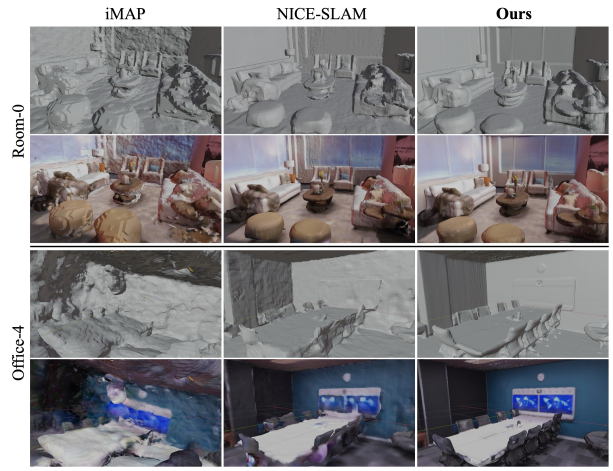


Figure 2. **Reconstruction results of geometries and colors on Replica Dataset [32].** Results are on room-0 and office-4.

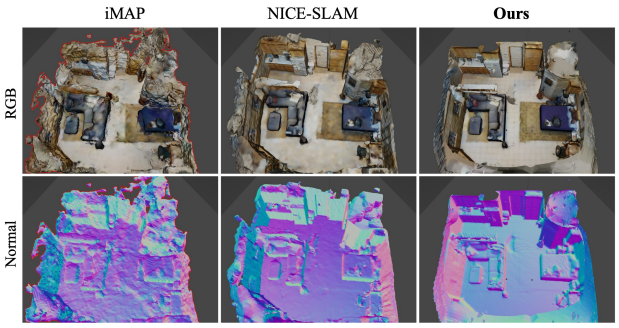


Figure 3. **3D reconstruction results on ScanNet Dataset [6].** Results are on $scene0000\_00$.

**Metrics.** To evaluate the quality of our reconstructions, we employ multiple metrics. For 2D reconstruction, we calculate the average L1 depth error between the projected depth from the rendered $xyz$-map and the ground-truth depth. Regarding 3D reconstruction, we measure the reconstruction quality of the 3D point cloud. We adopt Accuracy $[cm]$, Completion $[cm]$, and Completion Ratio $[< 5cm\%]$ as our evaluation metrics. For tracking, we use ATE RMSE $[cm]$ [34] to measure the performance. Moreover, we use PSNR and SSIM [45] as metrics to assess the quality of the novel view synthesis.

**Reconstruction Results.** We assess the reconstruction and novel view synthesis performance of our model using the Replica Dataset [32], as detailed in Tab. 1. Our method achieves superior or on-par results among other state-of-the-art methods. Figures 2 and 3 showcase the qualitative reconstruction results with methods using the same type of plain NeRF-based representations, demonstrating the model's efficacy in handling intricate geometries. More qualitative comparisons with other methods using different representations can be found in Appendix Figs. 8 and 9.

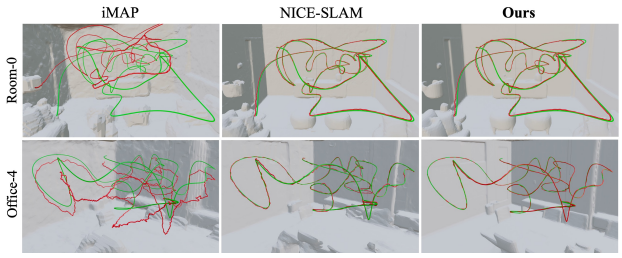


Figure 4. **Tracking results comparison.** The results are compared on the novel scenes from Replica Dataset [32]. The ground-truth camera trajectory is shown in green; the predicted trajectory is shown in red.

| | fr1/desk | fr2/xyz | fr3/office | Avg. |
|---|---|---|---|---|
| iMAP[35] | 7.2 | 2.1 | 9.0 | 6.1 |
| DI-Fusion[13] | 4.4 | 2.1 | 15.6 | 7.4 |
| NICE-SLAM[53] | 2.7 | 1.8 | 3.0 | 2.5 |
| ESLAM[14] | 2.5 | **1.1** | 2.4 | **2.0** |
| Co-SLAM[42] | 2.4 | 1.7 | 2.4 | 2.2 |
| Ours | **1.7** | 1.6 | **2.6** | **2.0** |

Table 2. **Camera tracking results on TUM-RGBD [33].** ATE RMSE [cm] is used as the tracking evaluation metric.

| | 0000 | 0059 | 0106 | 0169 | Avg. |
|---|---|---|---|---|---|
| iMAP[35] | 55.95 | 32.06 | 17.50 | 70.51 | 44.00 |
| DI-Fusion[13] | 66.99 | 128.00 | 18.50 | 75.80 | 72.32 |
| NICE-SLAM[53] | 8.64 | 12.25 | 8.09 | 10.28 | 9.89 |
| ESLAM[14] | 7.27 | **9.02** | 7.53 | 6.50 | 7.58 |
| Co-SLAM[42] | 7.13 | 11.14 | 9.36 | **5.90** | 8.38 |
| Ours | **6.69** | 9.07 | **7.17** | 6.34 | **7.32** |

Table 3. **Camera tracking results on ScanNet [6].** ATE RMSE [cm] is used as the tracking evaluation metric.

| | Track ↓ [ms x it.] | Map ↓ [ms x it.] | #param ↓ |
|---|---|---|---|
| iMAP [35] | 16.8x6 | 44.8x10 | 0.26M |
| NICE-SLAM [53] | 7.8x10 | 82.5x60 | 17.4M |
| ESLAM [14] | **6.9x8** | 18.4x15 | 9.29M |
| Co-SLAM [42] | 5.8x10 | 9.8x10 | 0.26M |
| Ours | 5.4x15 | **12.3x1** | **0.08M** |

Table 4. **Runtime and model size comparison.**

**Tracking Results.** We evaluate our method on two popular SLAM datasets: TUM-RGBD [33] and ScanNet [6]. In Tab. 2 and Tab. 3, we summarize the tracking RMSE results on TUM-RGBD Dataset [33] and ScanNet Dataset in Fig. 11, where our method achieves best performance compared to other methods. The qualitative comparison for camera tracking trajectories is shown in Fig. 4.

**Runtime.** In Tab. 4, we report the running time consumption on Replica Dataset [32] and the number of parameters. Other methods require per-scene model training, leading to


Figure 5. **Reconstruction results w/ and w/o novel views.** A comparison of reconstruction normal maps is conducted both with and without the inclusion of novel views, as outlined in Section 3.4. The sub-images highlighted in red and black correspond to specific regions in the left reconstruction results. It is observed that the incorporation of novel views enhances prediction capabilities, leading to improved reconstruction outcomes.

more time cost on the mapping process. In contrast, our model necessitates only a single step of feed-forward inference for mapping. Compared to other methods, our proposed system excels in the speed of the mapping stage. Additionally, our method features a modest parameter number of 0.08M, significantly smaller compared to other models.

### 4.3. Analysis on Design Choices

**Model Design.** In Tab. 5a, we present the results for two design choices in our study. Firstly, we explore the inclusion of novel views during reconstruction, and qualitative comparisons can be found in Fig. 5. As can be seen, the novel views are effective in recovering the scene details. Secondly, we investigate the use of a single shared network for both RGB and *xyz* data. Our default configuration incorporates novel views and employs two separate networks, as alternative configurations deteriorate performance.

**Comparison under Controlled Conditions.** Our method uses Poisson surface reconstruction [15] to generate 3D mesh from the point cloud obtained from rendered $xyz$-maps, while the mesh of iMAP [35]/NICE-SLAM [53] is generated through marching cube [18]. Here, we also show the Poisson surface reconstruction [15] results for iMAP/NICE-SLAM (refer to Tab. 5b). Our method still performs the best among them. Additionally, we also experiment by excluding $\mathcal{L}_{\text{normal}}$, $\mathcal{L}_{\text{flow}}$, $\mathcal{L}_{\text{warp}}$ from the loss function. IBD-SLAM still performs better.

**Training Loss.** We ablate the effectiveness of different terms in the training loss in Tab. 5c. It can be seen that the RGB, depth, and *xyz* loss terms affect the performance most, indicating the effectiveness of our idea on leveraging *xyz* maps. Meanwhile, introducing the regularization terms on geometry consistency also helps.

Please refer to the Appendix and the project website for additional results, comparisons and analysis.

| | Depth L1 ↓ | Acc. ↓ | Comp. ↓ | Comp. Ratio ↑ |
|---|---|---|---|---|
| w/o novel | 1.80 | 2.66 | 2.91 | 91.2 |
| Shared-net | 2.35 | 3.02 | 3.89 | 87.7 |
| **Ours** | **1.53** | **1.83** | **2.02** | **93.8** |

(a) Ablation study of model design

| | Depth L1 ↓ | Acc. ↓ | Comp. ↓ | Comp. Ratio ↑ |
|---|---|---|---|---|
| iMAP | 4.39 | 4.77 | 5.02 | 75.5 |
| iMAP‡ | 5.17 | 6.19 | 6.87 | 61.4 |
| NICE-SLAM | 2.49 | 2.42 | 2.62 | 90.3 |
| NICE-SLAM‡ | 3.13 | 3.05 | 3.16 | 87.2 |
| Ours† | 1.72 | 2.05 | 2.30 | 92.2 |
| **Ours** | **1.53** | **1.83** | **2.02** | **93.8** |

(b) Effects of mesh generation methods

| | Depth L1 ↓ | Acc. ↓ | Comp. ↓ | Comp. Ratio ↑ |
|---|---|---|---|---|
| w/o $\mathcal{L}_{xyz}$ | 3.02 | 2.81 | 3.23 | 88.2 |
| w/o $\mathcal{L}_{depth}$ | 2.09 | 2.23 | 2.45 | 92.0 |
| w/o $\mathcal{L}_{rgb}$ | 2.17 | 2.35 | 2.64 | 91.6 |
| w/o $\mathcal{L}_{normal}$ | 1.65 | 1.98 | 2.20 | 92.7 |
| w/o $\mathcal{L}_{wrap}$ | 1.62 | 1.92 | 2.13 | 93.3 |
| w/o $\mathcal{L}_{flow}$ | 1.57 | 1.86 | 2.20 | 93.1 |
| **Ours** | **1.53** | **1.83** | **2.02** | **93.8** |

(c) Ablation study of pretraining loss functions

Table 5. **Ablation study results on Replica Dataset [32].** The ablation analysis is conducted on a single selected Replica scene (`room-2`). In Tab. 5a, the first row depicts results obtained without including novel views in the reconstruction process. The second row showcases outcomes utilizing a shared network for RGB and *xyz* data. In Tab. 5b, we report Poisson surface reconstruction [15] results of point clouds generated by iMAP/NICE-SLAM, denoted as iMAP‡ and NICE-SLAM‡. Ours† denotes the reconstruction results without $\mathcal{L}_{\text{normal}}$, $\mathcal{L}_{\text{flow}}$, $\mathcal{L}_{\text{warp}}$. In Tab. 5c, we compare the results by excluding each of the loss terms.

## 5. Conclusions

We have proposed a new method, IBD-SLAM, for visual SLAM that overcomes the limitations of existing methods in terms of generalization and efficiency. By adopting a NeRF for scene representation, we propose to learn a generalizable image-based depth fusion network, which allows the model to be applied to new scenes without retraining. Unlike existing methods which optimize on a per-scene basis, IBD-SLAM is trained on a collection of RGBD videos and directly generalize to novel scenes, thus significantly more efficient. IBD-SLAM outperforms the previous state-of-the-art methods across several public SLAM benchmarks while being 10× faster.

**Acknowledgements.** This work is supported by Hong Kong Research Grant Council - Early Career Scheme (Grant No. 27208022), National Natural Science Foundation of China (Grant No. 62306251), and HKU Seed Fund for Basic Research.

## References


[1] 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. 3

[2] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 3

[3] Jaesung Choe, Sunghoon Im, Francois Rameau, Minjun Kang, and In So Kweon. Volumefusion: Deep depth fusion for 3d scene reconstruction. In *ICCV*, 2021. 3

[4] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996. 3

[5] 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. 2

[6] 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, 6, 7, 3

[7] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. 5, 1

[8] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022. 6

[9] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10786–10796, 2021. 5

[10] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017. 1, 2

[11] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13*, pages 834–849. Springer, 2014. 1

[12] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2367–2376, 2019. 6

[13] Jiahui Huang, Shi-Sheng Huang, Haoxuan Song, and Shi-Min Hu. Di-fusion: Online implicit 3d reconstruction with deep priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8932–8941, 2021. 6, 7, 2

[14] Mohammad Mahdi Johari, Camilla Carta, and François 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*, pages 17408–17419, 2023. 3, 6, 7, 4, 5

[15] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, page 0, 2006. 6, 8, 1

[16] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007. 1, 2

[17] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2019–2028, 2020. 3

[18] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*, pages 347–353. 1998. 8

[19] 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. 3

[20] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 3

[21] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 6

[22] 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. *Communications of the ACM*, 65(1):99–106, 2021. 3, 4, 5

[23] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 6

[24] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 1, 2

[25] Raul 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. 2, 6

[26] Richard A Newcombe, Dieter Fox, and Steven 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*, pages 343–352, 2015. 3, 6

[27] 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. 3

[28] 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

[29] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 5, 1

[30] 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. 3

[31] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019. 6

[32] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 1, 6, 7, 8, 2, 3, 4, 5

[33] Jürgen Sturm, Wolfram Burgard, and Daniel Cremers. Evaluating egomotion and structure-from-motion approaches using the tum rgb-d benchmark. In *Proc. of the Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RJS International Conference on Intelligent Robot Systems (IROS)*, volume 13, 2012. 6, 7, 1

[34] 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, 7

[35] 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. 3, 6, 7, 8, 2

[36] 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. 2

[37] Chengzhou Tang and Ping Tan. Ba-net: Dense bundle adjustment network. *arXiv preprint arXiv:1806.04807*, 2018. 2

[38] Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. *arXiv preprint arXiv:1812.04605*, 2018. 2

[39] 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. 2

[40] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15182–15192, 2021. 3

[41] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5038–5047, 2017. 2

[42] Hengyi Wang, Jingwen Wang, and Lourdes 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*, pages 13293–13302, 2023. 3, 6, 7, 4, 5

[43] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021. 1, 3, 4, 6

[44] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2043–2050. IEEE, 2017. 1

[45] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 7

[46] Thomas Whelan, Stefan Leutenegger, Renato Salas-Moreno, Ben Glocker, and Andrew Davison. Elasticfusion: Dense slam without a pose graph. Robotics: Science and Systems, 2015. 3

[47] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8121–8130, 2022. 6

[48] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 3

[49] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in neural information processing systems*, 35:25018–25032, 2022. 5

[50] 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. 2

[51] Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. Deeptam: Deep tracking and mapping. In *Proceedings of*

*the European conference on computer vision (ECCV)*, pages 822–838, 2018. 2

[52] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. *arXiv preprint arXiv:2302.03594*, 2023. 3, 5, 6

[53] 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 *CVPR*, 2022. 1, 3, 5, 6, 7, 8, 2