2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

DNS-SLAM: Dense Neural Semantic-Informed SLAM

Kunyi Li[1], Michael Niemeyer[2], Nassir Navab[1,3] and Federico Tombari[1,2]

***Abstract*— In recent years, coordinate-based neural implicit representations have shown promising results for the task of Simultaneous Localization and Mapping (SLAM). While achieving impressive performance on small synthetic scenes, these methods often suffer from losing details, especially for complex real-world scenes. In this work, we introduce DNS SLAM, a novel neural RGB-D semantic SLAM approach featuring a hybrid representation. Relying only on 2D semantic priors, we propose the first semantic neural SLAM method that trains class-wise scene representations while providing stable camera tracking at the same time. Our method integrates multi-view geometry constraints with image-based feature extraction to improve appearance details and to output color, occupancy, and semantic class information, enabling many downstream applications. To further enable fast tracking, we introduce a lightweight coarse scene representation which is trained in a self-supervised manner in latent space. Our experimental results achieve state-of-the-art performance on both synthetic data and real-world data tracking while maintaining a commendable operational speed on off-the-shelf hardware. Further, our method outputs class-wise decomposed reconstructions with better texture, capturing appearance and geometric details.**

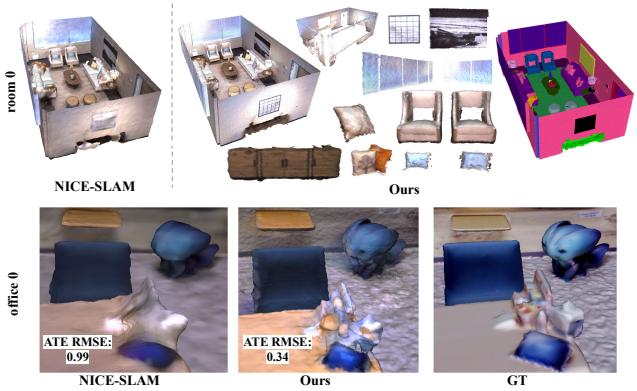

Fig. 1: **Class-Wise Reconstruction using DNS SLAM.** In contrast to previous neural SLAM systems, our method leads to semantically-decomposed 3D reconstructions of the scene (top) required for many downstream applications. Further, by using image-based features, we improve camera tracking and our reconstructions contain more geometric and appearance details (bottom).

I. INTRODUCTION

Dense Visual Simultaneous Localization and Mapping (SLAM) is a fundamental problem in the field of computer vision and plays a crucial role in various applications such as autonomous driving, indoor robotics, mixed reality, and more. Its primary goal is to create a 3D map of an unknown environment while simultaneously estimating camera poses. Traditional SLAM systems [1], [2] rely on multi-view geometry and focus primarily on camera pose accuracy but are prone to trajectory drifting. Recent learning-based dense visual SLAM methods [3], [4], [5], [6], [7], [8], [9] have aimed instead to generate meaningful global 3D maps, albeit with limited reconstruction accuracy and demanding extensive training for accurate pose estimation.

Inspired by the advancements in neural field research, neural implicit SLAM methods like iMAP [4] and NICE-SLAM [5] have recently emerged. These methods estimate poses from a scene representation, prioritizing world-space geometry over image-space geometry, showing promising results on synthetic indoor datasets. They leverage the inherent smoothness and coherence encoded in Multilayer Perceptron (MLP) weights, making them suitable for sequential tracking and mapping tasks. However, these methods tend to lose details in the reconstruction, which causes additional tracking errors. To address this challenge, Co-SLAM [7] uses parametric embeddings, while Point-SLAM [8] combines anchor points with neural representations to enhance reconstruction details. However, we find that their remarkable performance heavily relies on accurate pixel-perfect depth supervision. But in the real-world the depth is usually noisy, and we observe that these methods degrade drastically in these scenarios.

For robotics and interactive real-world vision applications, a semantic model that represents scene classes separately is required. vMAP [10] excels at accurate and complete scene reconstruction by modeling objects separately, but focuses only on reconstruction and does not perform localization.

To bridge the gap and fully leverage multi-view geometry and semantic information, we introduce Dense Neural Semantic-Informed (DNS) SLAM. Our method utilizes class-wise scene representations, which can build constraints between each class and the camera poses. Further, 2D features from reference images provide multi-view geometry constraints for better camera pose estimation. In summary, our main **contributions** include:

- Leveraging 2D semantic priors, we investigate a semantic-informed multi-class scene representation, yielding an efficient, comprehensive, and semantically decomposed geometry representation.
- Utilizing multi-view geometry, we extract image features by back-projecting points into reference frames, establishing constraints on relative camera poses and enhancing the appearance details.
- To speed up tracking, we introduce a lightweight

[1]Technical University of Munich <first>.<last>@tum.de
[2]Google Zurich mniemeyer@google.com
[3]Johns Hopkins University

coarse scene representation which is trained with a novel self-supervision strategy, utilizing the multi-class representation as pseudo ground-truth.
- To achieve accurate and detailed reconstructions, we approximate occupancy probabilities with Gaussian distributions as pseudo ground-truth for additional geometry supervision.

Compared with state-of-the-art (SOTA) methods, our method achieves better performance on camera pose estimation on both synthetic and real-world datasets, improving ATE RMSE by over 10% on average.

II. RELATED WORK

Scene Representations. Since the introduction of neural fields in the context of 3D reconstruction, they have been applied to many 3D computer vision tasks, including novel view synthesis [11], semantic scene reconstruction [12], and camera pose estimation [13]. Approaches such as Neural Radiance Fields (NeRF) [14] have achieved remarkable scene reconstruction results through differentiable rendering. However, it's worth noting that coordinate-based encoding methods [5], [15] come with the drawback of extended training times. To mitigate this, recent approaches employ parametric encoding techniques, expediting the training process. This enables neural implicit methods for real-time SLAM applications. While some works focus on improving the efficiency, Pixel-NeRF [16] preserves the spatial alignment between images and 3D representation, which is also an important concept in SLAM, by extracting 2D features from reference images and operating in view-space. Semantic-NeRF [12] provides precise geometry and semantic outcomes even with minimal semantic guidance. This demonstrates that with solely 2D semantic supervision, neural implicit methods can generate compelling 3D semantic scene reconstruction. For real-world SLAM applications, a semantic model is undoubtedly more meaningful. Our work partly builds upon the foundations of both Pixel-NeRF and Semantic-NeRF, but takes a significant step forward by simultaneously optimizing the camera poses while they assuming pose data is given.

Dense Visual SLAM. Modern SLAM methods follow the overall architecture introduced by ORB SLAM [1], decomposing the task into mapping and tracking. Based on the working space of map reconstruction, SLAM methods can be generally divided into two categories: image view-centric space and world view-centric space. The former one aims at maintaining multi-view geometry among keyframes in the dense setting, and the later one anchors the 3D geometry representation in a uniform world coordinate. DTAM [17] is an early example of image view-centric methods. Droid SLAM [3] uses optical flow to define geometrical residuals. World view-centric methods store global map in surfels [18], octrees [19] or grids like voxel grid [15] and hash grid [20]. KinectFusion [2] performs a frame-to-model camera tracking strategy and updates the scene geometry via Truncated Signed Distance Function (TSDF) fusion. RoutedFusion [21] outputs the TSDF update of the volumetric grid. In our method, the benefits from both image and world view are adopted, using a hybrid scene representation.

Neural Implicit SLAM. More recently, there has been a popularity in neural implicit representations for dense visual SLAM. iMAP [4], as the first neural implicit SLAM method, employs an MLP-based representation to conduct joint tracking and mapping, but the reconstruction and camera pose estimation result are far worse than traditional methods. To address computational overhead and scalability, NICE-SLAM [5] introduces multi-level feature grids. Nevertheless, the feature grids' local update approach limits its hole-filling capabilities. On the other hand, Co-SLAM [7] attains quasi-real time performance by combining coordinate and sparse parametric encodings for scene representation, employing dense global bundle adjustment using rays sampled from all keyframes. E-SLAM [6] suggests the use of multi-scale axis-aligned feature planes to prevent the model size from growing cubically concerning the scene's size. Point-SLAM [8] introduces a neural point cloud that iteratively grows in a data-driven manner during scene exploration for both mapping and tracking. ADFP [9] proposes an attentive depth fusion prior and uses either currently learned geometry or the one from depth fusion in volume rendering.

III. METHOD

A. Scene Representation

Notation: Let $\xi_t^{c2w} = [R_t^{c2w}|\mathbf{t}_t^{c2w}] \in SE(3)$ be the camera-to-world transform of frame t, where $R_t^{c2w} \in SO(3)$ and $\mathbf{t}_i^{c2w} \in \mathbb{R}^3$ are the rotation and translation, respectively. Denote camera intrinsic matrix as $K \in \mathbb{R}^{3\times3}$. For the rest of the manuscript we set $\xi = \xi^{c2w}$ to avoid cluttered notation. Camera projection and back-projection operator is denoted as π^{-1} and π , respectively, where π maps a 3D point $\mathbf{x}^c$ in the camera coordinate system into the image coordinate system and π^{-1} maps a pixel $\mathbf{u}$ in image coordinate system with depth $d(\mathbf{u})$ into the camera coordinate system:

$$\pi(\mathbf{x}^c) \cong K\mathbf{x}^c, \pi^{-1}(\mathbf{u}, d(\mathbf{u})) \cong d(\mathbf{u})K^{-1}\mathbf{u} \tag{1}$$

Note that we drop explicit homogeneous notation for clarity.

Neural Rendering: The input of our SLAM system is a RGB-D and semantic label stream $\{I_t\}_{t=1}^T, \{D_t\}_{t=1}^T, \{S_t\}_{t=1}^T$ with known intrinsics K. Our goal is to estimate camera poses $\{\hat{\xi}_t\}_{t=1}^T$ and train a scene representation θ. Each pixel in image space $\mathbf{u} \in \mathbb{R}^2$ determines a ray in world space with origin $\mathbf{o} = \hat{\mathbf{t}}_t$ and ray direction $\mathbf{d} = \hat{R}_t K^{-1}\mathbf{u}$. For rendering, we randomly sample M points $\mathbf{x}_i = \mathbf{o} + d_i\mathbf{d}, i \in \{1, ..., M\}$ along the ray where d_i denotes depth value of sampled point $\mathbf{x}_i$. We use One-blob [22] encoding $\gamma(\mathbf{x}_i)$ similar to [7] and a multi-resolution hash-based feature grid $\mathcal{V}_\alpha = \{\mathcal{V}_\alpha^l\}_{l=1}^L$ [23] as geometry representation. Our method first maps $\mathbf{x}_i$ into occupancy value $\hat{o}_i$ and latent vector $\mathbf{h}_i$ with geometry representation:

$$f_\psi(\gamma(\mathbf{x}_i), \mathcal{V}_\alpha(\mathbf{x}_i)) \mapsto (\mathbf{h}_i, \hat{o}_i). \tag{2}$$

We use multiple geometry representations f_ψ^k for each class in the scene instead of single one, which will be explained

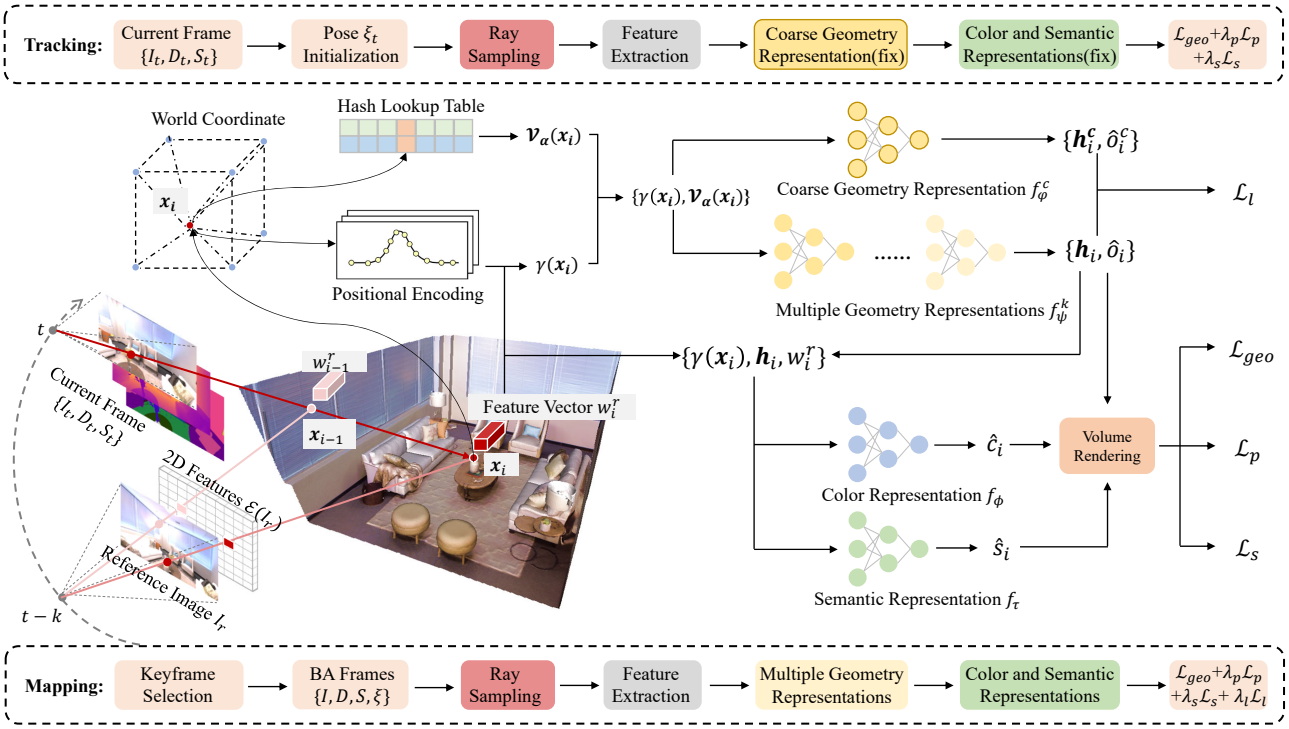

Fig. 2: **Overview.** 1) Scene Representation: For each point along the ray, we first query a 3D feature from our hash-based feature grid and a 2D feature from reference image(s). Next, we use multiple geometry together with color and semantic representations to render occupancy, color, semantic logits to 2D. 2) Tracking: We optimize per-frame camera poses by utilizing a coarse geometry representation which in turn is trained in a self-supervised manner using our fine decomposed representation. 3) Mapping: In global and local bundle adjustment, we jointly optimize the scene representation and camera poses.

in Sec. III-B. Then, the color and semantic representations predict color value $\hat{\mathbf{c}}_i$ and semantic logit values $\hat{\mathbf{s}}_i$ as:

$$f_{\phi,\tau}(\gamma(\mathbf{x}_i), \mathbf{h}_i) \mapsto (\hat{\mathbf{c}}_i, \hat{\mathbf{s}}_i). \tag{3}$$

We set all learnable parameters of our scene representation as $\theta = \{\alpha, \psi, \phi, \tau\}$. We render depth $\hat{d}$, color $\hat{\mathbf{c}}$ and semantic logit value $\hat{\mathbf{s}}$ by integrating the predicted values along the sample rays [14]:

$$\hat{\mathbf{g}} = \hat{\mathbf{g}}(\mathbf{u}; \hat{\xi}_t, \theta) = \sum_{i=1}^{M} w_i \hat{\mathbf{g}}_i, \tag{4}$$

where $\hat{\mathbf{g}} \in \{\hat{\mathbf{c}}, \hat{d}, \hat{\mathbf{s}}\}$. The weight w_i represents the discretized probability that the ray terminates at point $\mathbf{x}_i$:

$$w_i = \hat{o}_j \prod_{j=1}^{i-1} (1 - \hat{o}_j). \tag{5}$$

Like traditional SLAM methods [1], our method can be split into two processes: tracking and mapping. Tracking process first estimates camera pose of each frame while keeping the scene representations fixed. Mapping process jointly optimizes both the scene representations and camera poses. More details will be explained in the following sections. Fig. 2 shows an overview of our method.

B. Multi-Class Consistency

Instead of using a single shared MLP similar to previous neural SLAM approaches, decomposing the prediction into smaller class-specific MLPs can lead to a significant performance boost and speedup [10]. While obtaining 3D semantic priors is difficult in most cases, 2D semantic segmentation [24] is a well-studied task. As a result, in our method, we choose to utilize 2D semantic maps as they are often directly provided in datasets or alternatively can also be obtained from off-the-shelf methods. While we focus on class-level semantic maps in this work, other decompositions such as instance-level maps could be further explored in the future.

Multi-Class Rendering: For each pixel $\mathbf{u}$ in the image, first the 2D semantic map is queried to obtain its corresponding class id $k = \mathbf{s}(\mathbf{u})$, where $\mathbf{s}(\mathbf{u})$ denotes ground-truth semantic label at pixel $\mathbf{u}$. In this case, scene representation Eq. 2 can be further rewritten in class-wise as:

$$f_\psi^k(\gamma(\mathbf{x}_i), \mathcal{V}_\alpha(\mathbf{x}_i)) \mapsto (\mathbf{h}_i, \hat{o}_i), \tag{6}$$

and depth value of pixel $\mathbf{u}$ can be rendered as:

$$\hat{d} = \hat{d}(\mathbf{u}; \hat{\xi}, \psi^k) = \sum_{i=1}^{M} w_i \hat{o}_i, \tag{7}$$

where $k = 1, ..., K$, denotes different class. Therefore, the

geometry loss function can be expressed as:

$$\min \quad \mathcal{L}_{geo} = \sum_{\mathbf{u}\in|I|} \left\| d(\mathbf{u}) - \hat{d}(\mathbf{u}; \hat{\xi}, \theta) \right\|_1 \tag{8}$$

where $d(\mathbf{u})$ denotes ground-truth depth value at pixel $\mathbf{u}$, and $|I|$ represents a set of pixels sampled from image I. We use the L_1 norm as our geometry loss $\mathcal{L}_{geo}$.

C. Multi-View Consistency

Previous neural implicit SLAM methods directly predict RGB colors from latent representations and they tend to lose details. As shown in image-based rendering approaches such as Pixel-NeRF [16], spatial image features aligned to each pixel as an input allows the model to learn not only more texture information, but also scene priors. Therefore, we use features from 2D images as a conditional input in our model.

Image-Based Feature Pooling: Specifically, for every target frame I, we choose a reference frame I_r and extract 2D features $\mathcal{E}(I_r)$ using off-the-shelf method (ResNet-18 [25] in our case). Next, we cast rays for the target frame I to get sets of 3D points $\{\mathbf{x}_i\}$, and each 3D point $\mathbf{x}_i = \hat{\xi}_t \pi^{-1}(\mathbf{u}, d_i(\mathbf{u}))$ is back-projected to the reference frame coordinate to retrieve the corresponding image feature vectors $\mathbf{w}_i^r$:

$$\mathbf{w}_i^r = \mathcal{E}(I_r(\pi((\hat{\xi}_r)^{-1}\hat{\xi}_t\pi^{-1}(\mathbf{u}, d_i(\mathbf{u})))))). \tag{9}$$

Instead of directly passing these features to our scene representation, we encode each feature with the corresponding reference view $(\mathbf{o}_r, \mathbf{d}_r)$ to obtain intermediate feature vectors:

$$f_\sigma(\gamma(\mathbf{o}_r, \mathbf{d}_r), \mathbf{w}_i^r) \mapsto \bar{\mathbf{w}}_i^r, \tag{10}$$

where σ is the learnable parameters. In multi-reference frames case, feature vectors from each frame are then aggregated with an average pooling operator as the final image feature vector $\mathbf{w}_i = (\sum_{r=1}^{N} \bar{\mathbf{w}}_i^r)/N$. With image feature as conditional input, the color and semantic representation Eq. 3 can be rewirtten as:

$$f_{\phi,\tau}(\gamma(\mathbf{x}_i), \mathbf{h}_i, \mathbf{w}_i) \mapsto (\hat{\mathbf{c}}_i, \hat{\mathbf{s}}_i), \tag{11}$$

Color value $\hat{\mathbf{c}}(\mathbf{u}; \hat{\xi}_t, \theta)$ and semantic logits value $\hat{\mathbf{s}}(\mathbf{u}; \hat{\xi}_t, \theta)$ of pixel $\mathbf{u}$ are rendered as in Eq. 4. The photometric and semantic loss are defined as:

$$\mathcal{L}_p = \sum_{\mathbf{u}\in|I|} \left\| \mathbf{c}(\mathbf{u}) - \hat{\mathbf{c}}(\mathbf{u}; \hat{\xi}_t, \theta) \right\|_2, \tag{12}$$

$$\mathcal{L}_s = -\sum_{\mathbf{u}\in|I|} \sum_{k=1} (\mathbf{s}^k(\mathbf{u}) \log(\hat{\mathbf{s}}^k(\mathbf{u}; \hat{\xi}_t, \theta))), \tag{13}$$

where $\mathbf{c}(\mathbf{u}), \mathbf{s}(\mathbf{u})$ are ground-truth color and semantic logits. For the photometric loss $\mathcal{L}_p$, we use the L_2 norm and for the semantic loss $\mathcal{L}_s$, we use multi-class cross-entropy loss.

D. Self-Supervised Coarse Scene Representation

SLAM requires real-time performance, but our multi-class scene representations can become hardware-intense on smaller-compute devices. To tackle this, we exploit the fact that while a single MLP is not ideal to represent an entire scene due to catastrophic forgetting, it can however be used to represent a small neighborhood of the scene in a lightweight manner and is hence an ideal candidate for tracking. Thus, we introduce a lightweight coarse scene representation using Eq. 2:

$$f_\varphi^c(\gamma(\mathbf{x}_i), \mathcal{V}_\alpha(\mathbf{x}_i)) \mapsto (\mathbf{h}_i^c, \hat{o}_i^c). \tag{14}$$

Instead of training this coarse scene representation separately, which is time consuming, latent multi-class scene representations are used to supervise it during mapping:

$$\mathcal{L}_l = \sum_{i=1} \|\mathbf{h}_i - \mathbf{h}_i^c\|_2, \tag{15}$$

where $\mathbf{h}_i$ is latent vectors from Eq. 6. L_2 norm is applied on this self-supervised latent loss $\mathcal{L}_l$.

E. Occupancy Probability Approximation

To achieve accurate and detailed reconstructions, we propose an occupancy probability approximation loss inspired by Co-SLAM [7]. Specifically, for samples within the truncation region, i.e., points where $|d_i - d(\mathbf{u})| \leq tr$, we assume the occupancy probability follows a Gaussian distribution with zero mean value at the observed distance:

$$o_i = f(|d_i - d(\mathbf{u})|) = \frac{1}{2} e^{-\frac{1}{2}\left(\frac{|d_i - d(\mathbf{u})|}{\sigma}\right)^2}, \tag{16}$$

where σ is a hyper-parameter. We use this approximation as pseudo ground-truth occupancy probability to supervise outputs of geometry representations:

$$\mathcal{L}_o = \sum_{\mathbf{u}\in|I|} \sum_{i\in\{|d_i - d(\mathbf{u})|\leq tr\}} \|o_i - \hat{o}_i\|_2. \tag{17}$$

For samples before the surface $(d_i - d(\mathbf{u})) < -tr$, we apply a free-space loss [26] which forces the occupancy to be zero:

$$\mathcal{L}_{fs} = \sum_{\mathbf{u}\in|I|} \sum_{i\in\{(d_i - d(\mathbf{u}))<-tr\}} \|\hat{o}_i\|_2. \tag{18}$$

F. Mapping

Our model consisting of a feature grid $\mathcal{V}_\alpha$ and scene representations $\theta = \{\alpha, \psi^k, \phi, \tau, \varphi\}$ are randomly initialized at the beginning. During the whole process, 2D feature encoder $\mathcal{E}$ is fixed. For the first input frame, camera pose is fixed and only the hash grid and scene representations are optimized. For subsequent frames, scene representations and camera poses are optimized jointly and iteratively every k frames.

Keyframe Selection and BA: We follow [5] to choose the keyframes and frames for bundle adjustment (BA). For each mapping step, bundle adjustment is applied with current frame, the latest keyframe, and $W-2$ selected keyframes. Instead of only carrying local BA, which means selecting $W-2$ keyframes those have overlaps with current frame like in Nice-SLAM [5], we also randomly choose $W-2$ keyframes from the whole keyframes for global BA. These two selection strategies are applied iteratively.

Reference Frame Selection. We denote these selected BA frames as target frames. For each target frame, two reference frames are chosen by following strategy:

- for current frame, the two latest keyframes are chosen,
- for latest keyframe, two previous keyframes are chosen,

- for other keyframe, one previous keyframe and one later keyframe are chosen.

Ray Sampling: Fisrt, R pixels are randomly sampled from W target frames. 60% of them are randomly sampled among the image plane, 40% of them are randomly sampled among each class. Then, as described in Sec. III-A, points are sample along the rays. We use depth guided sampling such that M_s points are sampled near the surface and M_a points are sampled in free space.

New Class Initialization: If we encounter a previously unseen class, a new scene representation will be created and initialized. Then, it will be trained separately for 100 iterations before it joins the multi-class scene representations. This allows the system dynamically adding new class, which is more meaningful in practice.

Mapping Loss: Mapping and bundle adjustment are performed via minimizing the loss functions with respect to the learnable parameters θ and camera poses ξ. The final loss $\mathcal{L}_M$ is

$$\begin{aligned}&\mathcal{L}_M(\theta,\xi)=\\&\mathcal{L}_{geo}+\lambda_p\mathcal{L}_p+\lambda_s\mathcal{L}_s+\lambda_l\mathcal{L}_l+\lambda_o\mathcal{L}_o+\lambda_{fs}\mathcal{L}_{fs}\end{aligned}\tag{19}$$

where $\lambda_p, \lambda_s, \lambda_l, \lambda_o, \lambda_{fs}$ are respective loss weighting factors.

G. Tracking

For tracking, the hash-based feature grid and the scene representations are fixed, and only the camera pose are updated. Current camera pose is initialized by constant speed assumption like in Nice-SLAM [5]. The latest optimized frame is chosen as the reference frame for extracting features and R_t pixels are randomly sampled among the whole image. We use the same ray sampling strategy as in mapping. Occupancy values $\hat{o}_i^c$ and latent vectors $\mathbf{h}_i^c$ are obtained by Eq. 14 and latent vectors are passed to Eq. 11 to obtain the color $\hat{\mathbf{c}}_i$ and semantic values $\hat{\mathbf{s}}_i$.

Tracking Loss: A modified version of our geometry loss function is used in tracking, where the geometry term is weighted by the standard deviation of the depth prediction $\hat{d}_{var}$:

$$\mathcal{L}_{geo}^{track}=\sum_{\mathbf{u}\in|I|}\frac{\left\|d(\mathbf{u})-\hat{d}(\mathbf{u};\hat{\xi}_t,\theta)\right\|_1}{\sqrt{\hat{d}_{var}(\mathbf{u};\hat{\xi}_t,\theta)}},\tag{20}$$

where $\hat{d}_{var}(\mathbf{u};\hat{\xi}_t,\theta)=\sum_{i=1}^{M}w_i(\hat{d}(\mathbf{u};\hat{\xi}_t,\theta)-\hat{d}_i)^2$.

Given an initial guess of current camera pose and under the constant speed assumption [5], camera poses are iteratively updated by minimizing the following loss $\mathcal{L}_T$:

$$\mathcal{L}_T(\xi)=\mathcal{L}_{geo}^{track}+\lambda_p\mathcal{L}_p+\lambda_s\mathcal{L}_s,\tag{21}$$

where ξ is the camera poses and λ_p, λ_s are respective loss weighting factors.

IV. Experiments

A. Experimental Setup

Datasets We evaluate our method on both synthetic and real-world datasets with semantic maps. Following other neural implicit SLAM methods, for the reconstruction quality, we evaluate quantitatively on 8 synthetic scenes from Replica [27] and qualitatively on 6 scenes from ScanNet [28]. As for camera pose accuracy, we evaluate quantitatively on both Replica and ScanNet. The ground-truth poses of Replica are from simulation, while ground-truth poses of ScanNet are obtained with BundleFusion [29].

Metrics We evaluate the reconstruction quality using $DepthL1(cm), Accuracy(cm), Completion(cm)$ and $CompletionRatio(\%)$ with a threshold of 5cm. For evaluation of camera tracking, we adopt $ATE\ RMSE(cm)$. To evaluate our semantic reconstruction, we report $mIoU$.

Baselines We compare against the state-of-the-art methods NICE-SLAM [5], Co-SLAM [7], ESLAM [6], Point-SLAM [8] and ADFP [9] as our main baselines for reconstruction quality and camera tracking. For semantic evaluation, we compare against the very recent NIDS-SLAM [30] as all other methods do not predict semantic information.

Implementation Details We perform single GPU training (NVIDIA 2080ti), and for experiments with default settings, we use $R_t = 500$ pixels with 30 iterations for tracking and $R = 2000$ pixels for mapping with 100 iterations on Replica, and 200 iterations on ScanNet. For other settings, we follow Co-SLAM [7]. We use two-layer MLPs with 32 hidden units for each scene representation. We use a learning rate of 0.005 and 0.001 for all learnable parameters on Replica and ScanNet, respectively. For camera poses, we use a learning rate of 0.001 in tracking and 0.0005 in mapping. We use the first layer of ResNet [25] output as our 2D features. We set σ for occupancy approximation as 0.05.

B. Tracking Evaluation

We show quantitative results on both Replica [27] and ScanNet [28] in Table I. Our method demonstrates state-of-the-art tracking performance on the Replica dataset, outperforming other methods over 10% on average in terms of ATE RMSE. It is important to highlight that while both our method and Point-SLAM [8] achieve impressive tracking results on Replica, our method proves to be more robust and effective, particularly in complex real-world scenarios like those encountered in the ScanNet dataset. We attribute this is to the fact that Point-SLAM strongly relies on accurate depth supervision, while our approximated occupancy probability formulation provides reliable supervision and our multi-view consistency constraints can effectively address depth map ambiguities even if the depth input is not accurate. Fig. 3 shows a qualitative comparison on ScanNet. Compared with NICE-SLAM [5] and ESLAM [6], our method suffers from less trajectory drifting and, especially for those scenes with more texture, our method can improve ATE RMSE significantly. This highlights the adaptability and robustness of our approach, especially in challenging scenarios, making it a promising solution for practical applications.

C. Reconstruction Evaluation

Qualitative and quantitative reconstruction results on Replica [27] are shown in Fig. 4 and Table II, respectively. We find that SDF-based Co-SLAM [7] leads to good but

	ScanNet [28]							Replica [27]								
	0000	0059	0106	0169	0181	0207	Avg.	room_0	room_1	room_2	office_0	office_1	office_2	office_3	office_4	Avg.
NICE-SLAM [5]	8.64	12.25	8.09	10.28	12.93	5.59	9.63	1.69	2.04	1.55	0.99	1.39	3.97	3.08	1.95	2.08
Co-SLAM [7]	7.18	12.29	9.57	6.62	13.43	7.13	9.37	0.65	1.13	1.43	0.55	0.50	**0.46**	1.40	0.77	0.86
ESLAM [6]	7.3	8.5	7.5	**6.5**	**9.0**	5.7	7.42	0.76	0.71	0.56	0.53	0.49	0.58	0.74	0.64	0.62
Point-SLAM [8]	10.24	7.81	8.65	22.16	14.77	9.54	13.97	0.61	**0.41**	**0.37**	0.38	0.48	0.54	0.72	0.63	0.52
ADFP [9]	-	10.50	**7.48**	9.31	-	5.67	8.24	1.39	1.55	2.60	1.09	1.23	1.61	3.61	1.42	1.81
Ours	**5.42**	**5.20**	9.11	7.70	10.12	**4.91**	**7.07**	**0.49**	0.46	0.38	**0.34**	**0.35**	**0.39**	**0.62**	**0.60**	**0.45**

TABLE I: **Quantitative Tracking Comparison.** We show ATE RMSE↓ (cm) results of an average of 3 runs on ScanNet [28] and Replica [27]. We find that our method leads to state-of-the-art performance, improving tracking results on both synthetic and real-world data.

Fig. 3: **Qualitative Comparison on ScanNet.** 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 pose drifting.

	Method	Depth L1↓	Acc.↓	Comp.↓	Comp. Ratio↑
SDF-Based	Co-SLAM	1.58	2.10	2.08	92.99
	ESLAM	**1.18**	**0.97**	**1.05**	**98.60**
Occupancy-Based	NICE-SLAM	3.53	2.85	3.00	89.33
	ADFP	1.81	2.59	2.28	93.38
	Ours	3.16	2.76	2.74	91.73

TABLE II: **Quantitative Reconstruction Comparison on Replica.**

less detailed reconstructions and both Co-SLAM and NICE-SLAM [5] do not lead to satisfactory texture reconstruction. In contrast, our method is able to reconstruct fine geometric structures as well as appearance details (see e.g. the flowers on the table of "room_1" or the side table of "office_4" in Fig. 4). While we improve also quantitatively over NICE-SLAM, we find that the reconstruction metrics favor the less detailed reconstructions from the SDF-based methods such as Co-SLAM, mainly due to the fact that errors in unobserved regions (such as under a table) dominate the evaluation metrics. We plan to investigate appropriate regularization techniques for unobserved regions, e.g. based on diffusion priors, for our image-based approach in the future.

D. Semantics Evaluation

Table III shows the quantitative evaluation of our method in comparison to another neural semantic SLAM method NIDS-SLAM [30]. We follow NIDS-SLAM and report the

Scene	room_0	room_1	room_2	office_0	Avg.
NIDS-SLAM	82.45	84.08	76.99	**85.94**	82.37
Ours	**88.32**	**84.90**	**81.20**	84.66	**84.77**

TABLE III: **Semantic Segmentation Comparison on Replica.**

Per frame	NICE-SLAM	Co-SLAM	Point-SLAM	**Ours**
Tracking	1.32 s	0.12 s	0.85 s	0.36 s
Mapping	10.92 s	0.33 s	9.85 s	7.58 s

TABLE IV: **Runtime Comparison on Replica.**

mIoU on four scenes. Our method achieves better results despite NIDS-SLAM using ORB-SLAM [1] as its tracking processor, and we are the first neural SLAM method to achieve simultaneous localization, reconstruction and segmentation all at once. Fig. 5 shows class-wise reconstruction results on Replica dataset. Our method can represent and reconstruct each class separately leading to decomposed representations, which can be used as a 3D prior for further downstream tasks.

E. Runtime Analysis

We report runtimes on Replica's room_0 scene in Table IV, using a NVIDIA 2080ti GPU. The tracking and mapping time are reported per frame. We observe that our approach effectively balances performance and runtime considerations, enabling high-fidelity as well as fast SLAM.

F. Ablation Study

In Table V we report quantitative results for our method where we ablate various components which we discuss in greater detail in the following. We test on scene room_0 from Replica and scene 0059 from ScanNet.

New Class Initialization. We find that if a new class representation is not initialized appropriately before it is added to the system, it leads to inadequate RGB predictions not only for the new class but also for existing classes due to a bleeding effect. It is important to note that this in turn

Method	w/o multi-class	w/o 2D feature	w/o init.	w/o occ. approx.	full model
Replica	1.16	0.71	0.53	0.86	0.49
ScanNet	7.52	6.22	174.4	5.77	5.20

TABLE V: **Quantitative Ablation Study (ATE RMSE).**

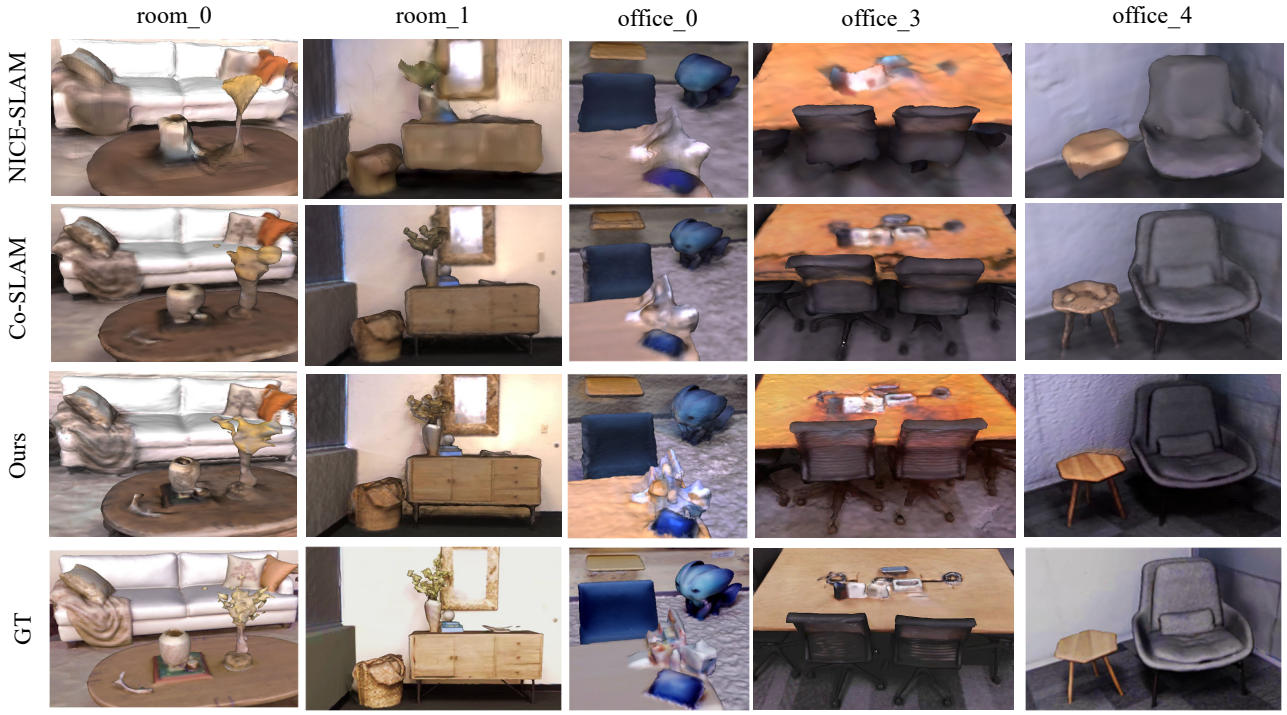

Fig. 4: **Qualitative Reconstruction on Replica.** Our method achieves reconstructions with more geometric and appearance details.

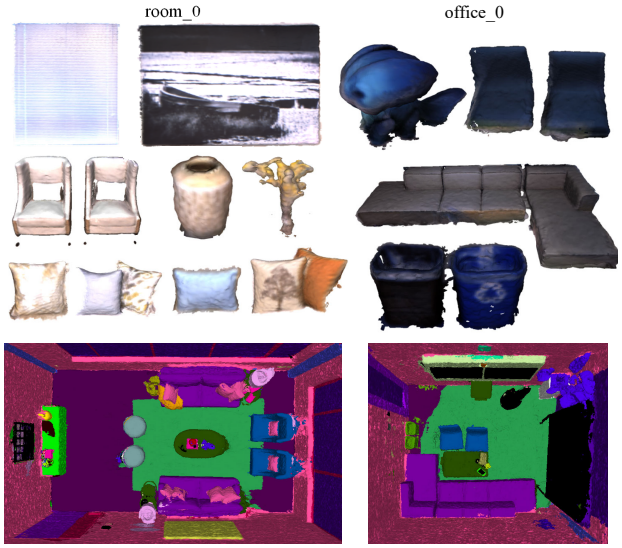

Fig. 5: **Decomposed Reconstruction on Replica.** We show the 3D semantic mesh (bottom) and its decomposition with RGB colors (top) of two scenes from Replica.

also leads to an increased camera pose error and sometimes causing further tracking failures. This is in particular visible on the real-world dataset ScanNet.

Multi-Class Scene Representations. We ablate our multi-class scene representations by using only a single representation during the whole process. Table V shows that our full mode leads to higher accuracy considering ATE RMSE than only using a single geometry representation. The difference is larger on ScanNet where the input data is noisy and the size of the scene is larger compared to Replica.

Conditional Image Feature Input. Image features serve as a conditional input for our system, enforcing a multi-view geometry constraint and enhancing the texture prediction. Table V shows a decline in ATE RMSE on both synthetic and real-world datasets when removing the conditional input. Fig. 6 shows a visual comparison of the reconstructed meshes. We find that without 2D image features, the reconstruction become smoother and contain less details.

Occupancy Probability Approximation. Finally, We investigate the effectiveness of our occupancy probability approximation and show a qualitative comparison in Fig. 6. The output mesh tends to be noisier and artifacts appear. In contrast, the meshes of our full model contains fine details. Table V further shows that our full model also leads to better tracking performance.

V. CONCLUSIONS

We present DNS SLAM, the first method that uses a neural representation for simultaneously localization, reconstruction and segmentation. We propose to use a semantically decomposed scene representation in combination with conditional 2D image features, enforcing stronger geometric constraints and hence enabling better tracking and more detailed appearance prediction while being more robust to noisy depth inputs. Our occupancy probability approximation serves as a strong supervision signal, enabling accurate and detailed mesh reconstruction. Extensive experiments show that our

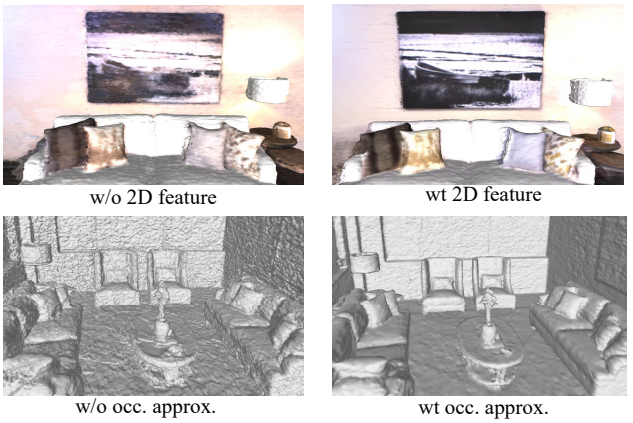

Fig. 6: **Ablation Study.** We show that 2D image feature can provide conditional information for reconstruction (top). And occupancy probability approximation can help to smooth the reconstruction (bottom).

method significantly improves the tracking accuracy compared to state-of-the-art baselines while also providing class-wise decompositions.

Limitations. Our method relies on RGB-D and semantic maps as input. While being more robust to noisy depth inputs than previous works, we plan to investigate RGB only input in the future.

REFERENCES

[1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[2] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, *et al.*, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, 2011, pp. 559–568.

[3] Z. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *NeurIPS*, vol. 34, pp. 16 558–16 569, 2021.

[4] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "imap: Implicit mapping and positioning in real-time," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6229–6238.

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

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

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

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

[9] P. Hu and Z. Han, "Learning neural implicit through volume rendering with attentive depth fusion priors," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[10] X. Kong, S. Liu, M. Taher, and A. J. Davison, "vmap: Vectorised object mapping for neural field slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 952–961.

[11] B. Mildenhall, P. Hedman, R. Martin-Brualla, P. P. Srinivasan, and J. 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*, 2022, pp. 16 190–16 199.

[12] S. Zhi, T. Laidlow, S. Leutenegger, and A. J. Davison, "In-place scene labelling and understanding with implicit scene representation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 838–15 847.

[13] W. Bian, Z. Wang, K. Li, J.-W. Bian, and V. A. Prisacariu, "Nope-nerf: Optimising neural radiance field with no pose prior," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4160–4169.

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

[15] S. Weder, J. L. Schonberger, M. Pollefeys, and M. R. Oswald, "Neuralfusion: Online depth fusion in latent space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3162–3172.

[16] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "pixelnerf: Neural radiance fields from one or few images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4578–4587.

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

[18] T. Schops, T. Sattler, and M. Pollefeys, "Bad slam: Bundle adjusted direct rgb-d slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 134–144.

[19] E. Vespa, N. Nikolov, M. Grimm, L. Nardi, P. H. Kelly, and S. Leutenegger, "Efficient octree-based volumetric slam supporting signed-distance and occupancy mapping," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1144–1151, 2018.

[20] O. Kähler, V. Prisacariu, J. Valentin, and D. Murray, "Hierarchical voxel block hashing for efficient integration of depth images," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 192–197, 2015.

[21] S. Weder, J. Schonberger, M. Pollefeys, and M. R. Oswald, "Routedfusion: Learning real-time depth map fusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4887–4897.

[22] T. Müller, B. McWilliams, F. Rousselle, M. Gross, and J. Novák, "Neural importance sampling," *ACM Transactions on Graphics (ToG)*, vol. 38, no. 5, pp. 1–19, 2019.

[23] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics (ToG)*, vol. 41, no. 4, pp. 1–15, 2022.

[24] X. Zhou, R. Girdhar, A. Joulin, P. Krähenbühl, and I. Misra, "Detecting twenty-thousand classes using image-level supervision," in *European Conference on Computer Vision*. Springer, 2022, pp. 350–368.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[26] M. Niemeyer, L. M. Mescheder, M. Oechsle, and A. Geiger, "Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision," in *CVPR*, 2020.

[27] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, *et al.*, "The replica dataset: A digital replica of indoor spaces," *arXiv preprint arXiv:1906.05797*, 2019.

[28] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *CVPR*, 2017.

[29] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, p. 1, 2017.

[30] Y. Haghighi, S. Kumar, J. P. Thiran, and L. Van Gool, "Neural implicit dense semantic slam," *arXiv preprint arXiv:2304.14560*, 2023.