Benchmarking Implicit Neural Representation and Geometric Rendering in Real-Time RGB-D SLAM

Supplementary Material

1. Implementation Details

We conducted the benchmark leaderboard using a 2.60GHz Intel Xeon Platinum 8358P CPU and an A800-SXM4-80GB GPU. In contrast, the influence of network structure (refer to Table 1 in the main paper) and mapping time of Explicit Hybrid Encoding was evaluated on a 2.10GHz Intel Xeon Gold 5218R CPU and an NVIDIA GeForce RTX 3090. Our experiments were carried out within the unified SLAM framework, utilizing synthetic datasets for consistency and control. The system initialization was performed with the first posed RGB-D frame, followed by an optimization over 500 iterations. For subsequent tracking and mapping, we conducted optimizations over 20 iterations, sampling 1024 and 2048 pixels, respectively. A keyframe was defined as every 5th input frame. Here, 5% of the pixels were randomly sampled and stored for global bundle adjustment purposes. For each sampled pixel, we selected 60 coordinates along its ray. Within the truncated region ($T = 10$cm), 12 coordinates were uniformly sampled, and an additional 48 were drawn uniformly from the free space for pixels with depth information. In cases without depth information, the 60 points were uniformly sampled. All hybrid representations (*i.e.* $\mathcal{F}! =$ MLP) utilized coarse and fine feature spatial splittings with resolutions of 24cm and 2cm, respectively. The feature dimensions for each resolution level were set to 2 and concatenated into a 4-dimensional vector for processing by color and geometry decoders. These decoders are comprised of small, shallow MLPs with 2 layers and 32 hidden channels. We configured the color loss as $\lambda_p = 5$, the rendered depth loss as $\lambda_g = 1$, and the SDF losses as $\lambda_t = 200$, $\lambda_c = 50$, and $\lambda_{fs} = 10$, respectively.

Cover figure (Fig. 1) was produced using configurations that differ slightly from our standard setup. For this figure, the initialization phase was extended to 1000 iterations. We documented the highest *PSNR* value obtained in the last three iterations for both our method and NICE-SLAM. For mapping, we increased the number of sampled pixels to 4080, while for tracking purposes, we used 2048 pixels. The iterations for mapping and tracking were set to 30 and 20, respectively. Notably, the figure relies on SDF (density) functions with 8-dimensional features, which is $\times 4$ less than the configuration used in [12] and does not require pre-trained models.

The differences in feature dimensions and SDF transformation functions utilized during initialization are further detailed in Fig. 2. The reconstructed accuracy (*Acc.*) for Fig. 1 is reported as 1.53cm, with *ATE* of 0.73cm. This represents a significant improvement, approximately 60% better, compared to the 3.87cm (*Acc.*) and 1.95cm (*ATE*) achieved by NICE-SLAM. The *ATE* record for NICE-SLAM can be found in Table 1 of [11]. A comparative demonstration of the dense grid representation with our baseline configuration, used for the main leaderboard, can be found in Fig. 5.

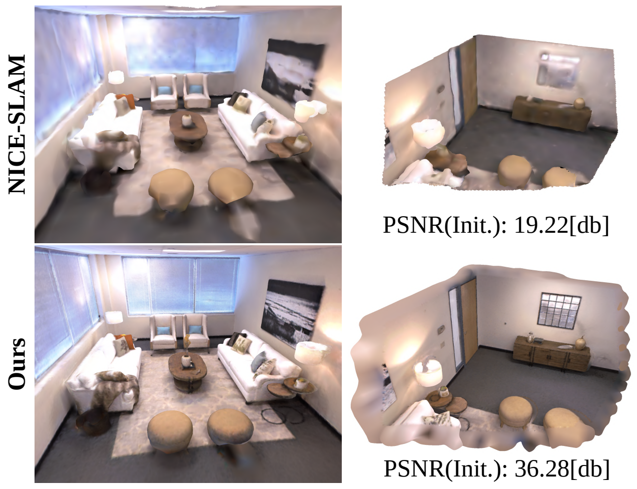

Figure 1. Comparison of the hierarchical dense grid as implemented in our benchmark with the previous state-of-the-art (SOTA) [12]. Right column: Meshes trained on the initial RGB-D frame for 1000 iterations.

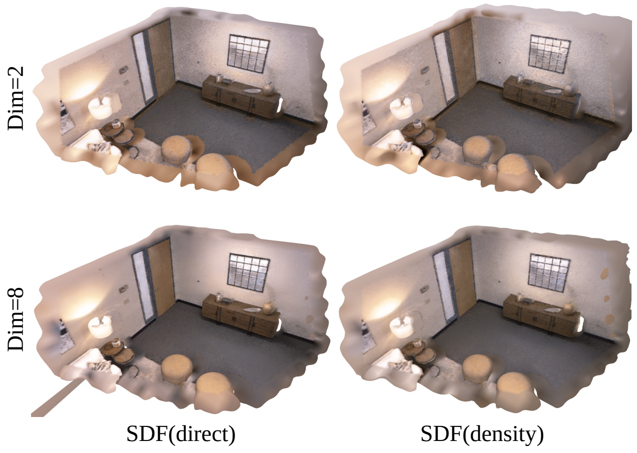

Figure 2. Dense grid initialization visualized at the first posed frame over 500 iterations for different rendering functions, featuring 2 and 8-dimensional settings.

Training specification. Firstly, we specify the photometric and geometric losses in accordance with Eq. (4) in the main paper:

$$\mathcal{L}_p = \frac{1}{\mathcal{M}} \sum_x (e_x^p)^2, \mathcal{L}_g = \frac{1}{\mathcal{M}_{sub}} \sum_x (e_x^g)^2 \quad (1)$$

In this context, $\mathcal{M}$ denotes the global list of sampling pixels (*e.g.*, key frames), and $\mathcal{M}_{\text{sub}}$ represents the subset of those pixels with non-zero depth values. To estimate the SDF s_i for each sample point p_i, a geometric constraint is applied:

$$e_i^{sdf} = d_i + \tilde{s}_i \cdot T - d_x \quad (2)$$

Hence, for points p_i located within the truncated region $i \in \mathcal{P}^T$, defined by the condition $|d_x - d_i| < T$, the SDF loss is formulated as follows:

$$\mathcal{L}_{sdf} = \frac{1}{\mathcal{M}_{sub}} \sum_x \frac{1}{\mathcal{P}^T} \sum_i (e_i^{sdf})^2 \quad (3)$$

Moreover, by defining the area between the camera center and the truncation boundary T as free space, we apply constraints to points in this region, denoted as $i \in \mathcal{P}^{\text{fs}}$:

$$e_x^{fs} = \tilde{s}_x - 1 \quad (4)$$

This leads to the establishment of the free space loss:

$$\mathcal{L}_{fs} = \mathcal{L}_{fs} = \frac{1}{\mathcal{M}_{sub}} \sum_x \frac{1}{\mathcal{P}^{fs}} \sum_i (e_i^{fs})^2 \quad (5)$$

Following the approach described in [4], we differentiate the importance of samples within the truncation region by applying distinct weighting factors: λ_c for samples close to the surface and λ_t for those near the truncation boundary. The overall objective function is as follows:

$$\mathcal{L} = \lambda_p \mathcal{L}_p + \lambda_g \mathcal{L}_g + \lambda_t \mathcal{L}_{sdf}^t + \lambda_c \mathcal{L}_{sdf}^c + \lambda_{fs} \mathcal{L}_{fs} \quad (6)$$

2. Performance Trade-off

Benchmark Considerations. Our benchmark study strictly controls feature dimensions and spatial resolution. However, optimal performance in such controlled comparisons may not always align with real-world application priorities. For example, a minor compromise in accuracy (a 1mm decrease in trajectory estimation accuracy) can lead to substantial gains in efficiency, such as an 87.93% reduction in memory usage, as shown in Fig. 3. Hence, in practical scenarios, *a hybrid approach combining tri-plane and hash encodings* might be a preferable alternative to the tri-plane and dense grid combination.

Comparison with Existing Work. We compared our hybrid designs with a SLAM system that employs similar joint encoding strategies, *i.e.* Co-SLAM [8]. The dense grid-based hybrid encoding exhibits superior performance, albeit at the cost of significantly higher memory consumption Fig. 3. Interestingly, the Hybrid (Hash+Tri) configuration, while not excelling in inference capabilities or memory efficiency in Tab. 1, emerges as the most practical option for real-world applications. As illustrated in Fig. 4, our hybrid encoding notably reduces artifacts compared to Co-SLAM.

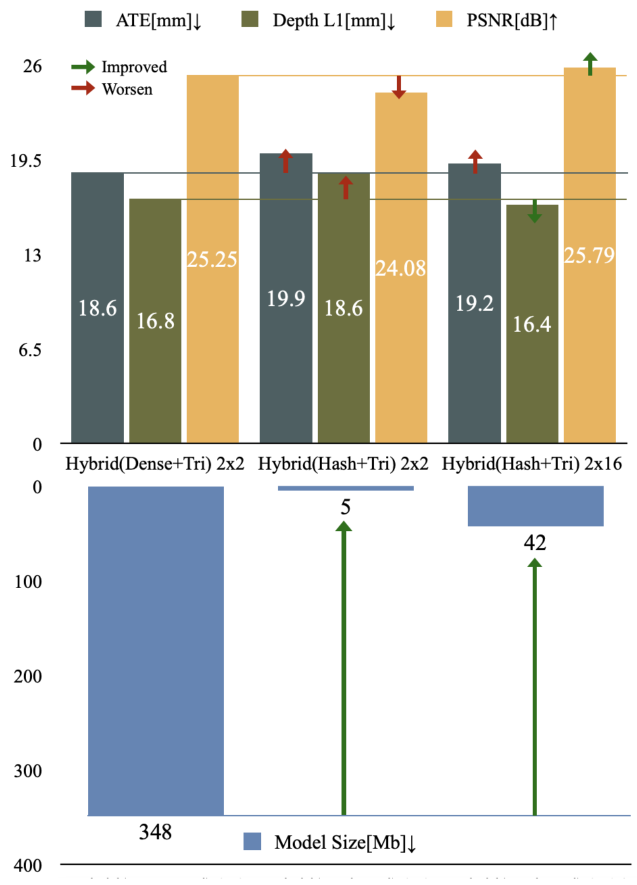

Figure 3. Memory-performance trade-offs: Using dense grid-based hybrid encoding as a baseline, hash-based alternatives demonstrate significant memory savings. This efficiency is maintained even with an ×8 increase in resolution levels for hybrid encoding of hash grid and tri-plane.

SLAM	ATE↓ [cm]	Depth L1↓ [cm]	PSNR↑ [dB]	Model Size↓ [MB]
Hybrid(Dense+Tri) 2 × 8	**1.73**	**1.55**	**26.46**	1364.28
Hybrid(Hash+Tri) 16 × 2	1.92	1.64	25.79	42.45
Co-SLAM* 16 × 2	1.83	1.64	25.41	**7.25**

Table 1. Grid-based Hybrid Representations vs. Co-SLAM ([8]). The asterisk denotes Co-SLAM's performance as measured by implementing its open-source code.

This is achieved with a minimal trade-off in trajectory accuracy (only 1mm lower than Co-SLAM) for the Hash+Tri combination. Regarding inferring performance, our Hybrid (Dense+Tri) encoding achieves the best *ATE*, *Depth L1*, and *PSNR* scores, as detailed in Tab. 1.

Pose Estimation & Scene Completion. Previous research has established that greater completeness in unobserved re-

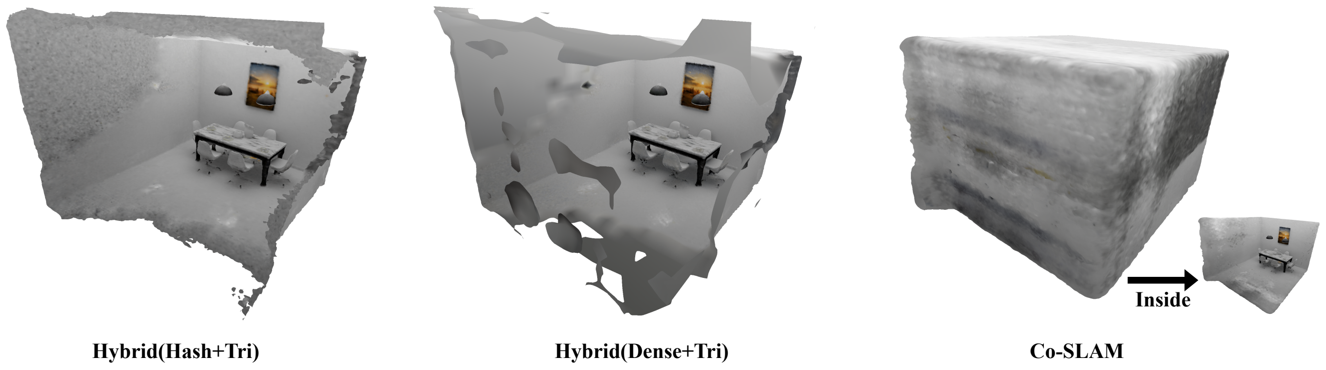

Figure 4. Visualization of *'breakfast room'* sequence of NeuralRGBD dataset [1].

SLAM	Acc.↓ [cm]	Comp.↓[cm]	Comp.% ↑ [%]
Hybrid(Dense+Tri) 2×2	2.40	4.64	83.48
Hybrid(Dense+Tri) 2×8	2.41	4.64	83.81
Hybrid(Hash+Tri) 16×2	2.48	4.67	83.83
Co-SLAM* 16×2	**2.23**	**4.52**	**84.27**

Table 2. Comparison of Final Reconstruction Metrics. Evaluation based on the culling method from [1].

Strategy	DepthL1↓	PSNR↑	TrackTime↓	MapTime↓
Sampling variant	3.41↑	24.59↓	274↑	752↑
Loss variant	85.5 ↓	22.47 ↓	290 ↑	776 ↑
Keyframe variant	4.82↑	26.00 ↑	279 ↑	757 ↑
Original	4.83	25.58	300	785

Table 3. ↑ & ↓ indicating improvement & worsen compared to the original F(dense)+G(density). Variants are on importance sampling, free space loss, and global keyframe sampling, respectively.

gions often enhances overall trajectory estimation [8, 12]. Nonetheless, deploying the entire coarse level of the dense grid is computationally demanding [12], and alternative one-blob encoding strategies can produce excessive artifacts [8], potentially posing risks or inconveniences in applications like robotic navigation. Our hybrid encoding, which combines a tri-plane and hash grid, successfully addresses these challenges. Ultimately, trajectory accuracy seems to be more dependent on precise scene perception, as indicated by superior *PSNR* and *Depth L1* metrics in Tab. 1, than on completeness Tab. 2.

About sampling & loss & keyframe settings. Although settings like loss & keyframe are critical and might pose different levels of impact on $\mathcal{F}$ & $\mathcal{G}$, these are not the focus of our work. The reasons are two-fold: **1)** Given the intention of benchmarking, we must avoid an explosion of choices that overshadow the key insights within limited pages. **2)** Discussion of these choices in the current benchmark might be less insightful. We suggest establishing an additional novel benchmark based on real-world challenges, e.g. large-scale and multi-agents scenarios [3, 7], where settings like loss & keyframe selection are discussed within more meaningful contexts, rather than just facilitating numerical improvement in current benchmark scenarios. Nevertheless, we provide a toy example of setting alternatives from other SOTA baselines on '*room0*' sequence in Tab. 3.

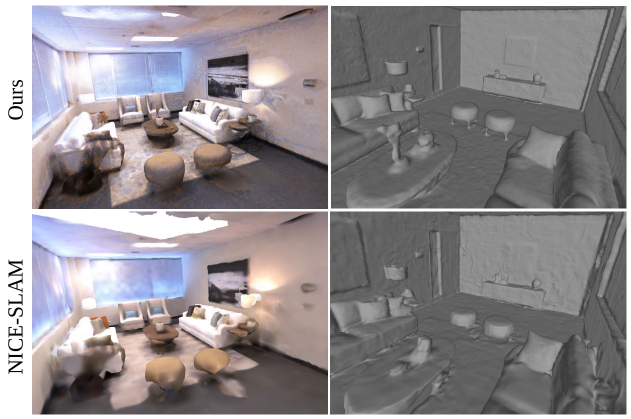

Figure 5. Dense Grid (density) compared to NICE-SLAM under benchmark configurations (Dimensions×Resolutions are 2×2 for Ours vs 32×3 for NICE-SLAM).

3. Runtime

We report the runtime of hybrid encoding (for SLAM) and explicit hybrid encoding (for mapping), evaluated on NeuralRGBD [1] and Scannet Dataset [2] in Tab. 4 and Tab. 5, respectively. For uniformly recorded map updating times, explicit hybrid encoding achieves speeds approximately ×3 faster than NICE-SLAM. Visual demonstration is available in Fig. 6.

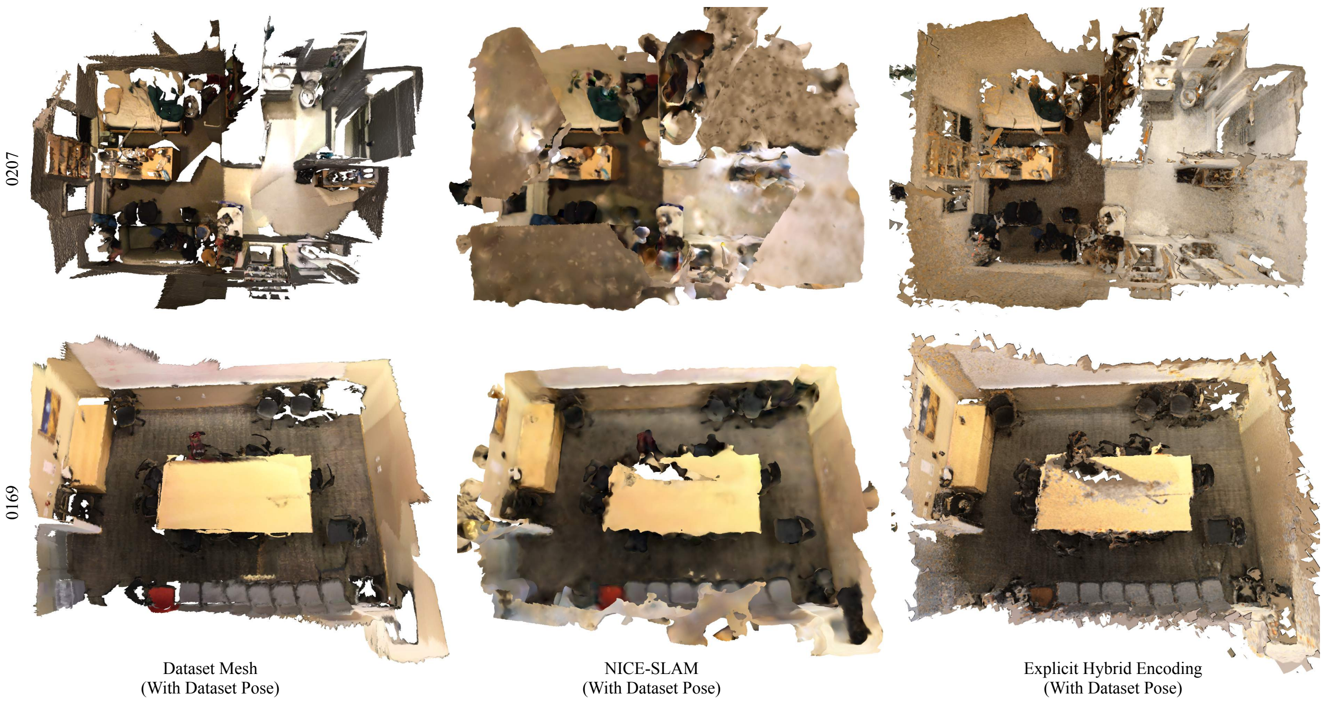

Figure 6. Qualitative evaluation of Hybrid Explicit Encoding on ScanNet Dataset. Both NICE-SLAM and Ours run on the posed RGB-D stream to simulate an externally provided tracker.

SLAM	0000	0059	0169	0181	0207	Avg.
NICE-SLAM* 3×32	4.65	3.74	4.39	3.28	3.20	3.83
Explicit Hybrid Encoding 1×2	**2.14**	**1.02**	**0.94**	**0.73**	**0.87**	**1.14**

Table 4. Runtime (explicit hybrid encodings) Comparison Measured in Seconds. The asterisk denotes NICE-SLAM runtimes obtained using ground truth poses, based on its open-source code.

Time	Hybrid	br	ck	gr	gwr	ma	tg	w	Avg.
Tracking	Dense+Tri 2×8	267	512	370	288	**275**	281	354	335
	Hash+Tri 2×8	**248**	**238**	**242**	242	279	**277**	**235**	**251**
Mapping	Dense+Tri 2×8	713	2441	1502	903	676	654	1381	1181
	Hash+Tri 2×8	**572**	**536**	**563**	**558**	**631**	**614**	**555**	**575**

Table 5. Runtime (hybrid encodings) Comparison Measured in Milliseconds.

4. Limitations

In summary, the scene representations explored in this paper primarily utilize spatial splitting along orthogonal coordinates. However, recent advancements in point-based methods [3, 6, 10] have demonstrated the effectiveness of newer scene representations like PointNeRF [9] and 3D Gaussian [5], offering greater adaptability. This adaptability paves the way for more sophisticated and robust systems that could integrate SLAM-centric and NeRF-centric methods, which are currently addressed separately in our work.

Point-based methods, whether modeled as 3D Gaussians with splatting-based rasterization [10] or as neural points with volume rendering [3, 6], essentially handle points in space. In neural point clouds, points directly represent spatial features, augmented by neural network processing. For 3D Gaussian representations, each point acts as the center of a Gaussian distribution, indicating not only a specific location but also a surrounding area of influence. Notably, splatting rasterization is generally less computationally demanding than volume rendering, which involves intricate integration across the volume as noted in [5, 10].

Looking ahead, we anticipate a more comprehensive evaluation that includes these recent approaches in implicit scene representation and geometric rendering, under a robustified SLAM system design, which incorporates essential components like loop closure and odometry, to unify both SLAM-centric and NeRF-centric methodologies.

References

[1] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF Conference*

on Computer Vision and Pattern Recognition, pages 6290–6301, 2022. 3

[2] 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. 3

[3] Jiarui Hu, Mao Mao, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Cp-slam: Collaborative neural point-based slam system. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 3, 4

[4] Mohammad Mahdi Johari, Camilla Carta, and François Fleuret. ESLAM: efficient dense SLAM system based on hybrid representation of signed distance fields. *CoRR*, abs/2211.11704, 2022. 2

[5] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4):1–14, 2023. 4

[6] Erik Sandström, Yue Li, Luc Van Gool, and Martin R Oswald. Point-slam: Dense neural point cloud-based slam. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18433–18444, 2023. 4

[7] Yijie Tang, Jiazhao Zhang, Zhinan Yu, He Wang, and Kai Xu. Mips-fusion: Multi-implicit-submaps for scalable and robust online neural rgb-d reconstruction. *arXiv preprint arXiv:2308.08741*, 2023. 3

[8] 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. 2, 3

[9] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022. 4

[10] Chi Yan, Delin Qu, Dong Wang, Dan Xu, Zhigang Wang, Bin Zhao, and Xuelong Li. Gs-slam: Dense visual slam with 3d gaussian splatting. *arXiv preprint arXiv:2311.11700*, 2023. 4

[11] Xingrui Yang, Hai Li, Hongjia Zhai, Yuhang Ming, Yuqian Liu, and Guofeng Zhang. Vox-fusion: Dense tracking and mapping with voxel-based neural implicit representation. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 499–507. IEEE, 2022. 1

[12] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 12776–12786. IEEE, 2022. 1, 3