

NGEL-SLAM: Neural Implicit Representation-based Global Consistent Low-Latency SLAM System

Yunxuan Mao¹, Xuan Yu¹, Zhuqing Zhang¹, Kai Wang², Yue Wang¹, Rong Xiong¹, and Yiyi Liao^{1*}

Abstract—Neural implicit representations have emerged as a promising solution for providing dense geometry in Simultaneous Localization and Mapping (SLAM). However, existing methods in this direction fall short in terms of global consistency and low latency. This paper presents NGEL-SLAM to tackle the above challenges. To ensure global consistency, our system leverages a traditional feature-based tracking module that incorporates loop closure. Additionally, we maintain a global consistent map by representing the scene using multiple neural implicit fields, enabling quick adjustment to the loop closure. Moreover, our system allows for fast convergence through the use of octree-based implicit representations. The combination of rapid response to loop closure and fast convergence makes our system a truly low-latency system that achieves global consistency. Our system enables rendering high-fidelity RGB-D images, along with extracting dense and complete surfaces. Experiments on both synthetic and real-world datasets suggest that our system achieves state-of-the-art tracking and mapping accuracy while maintaining low latency. The code is available at https://github.com/YunxuanMao/ngel_slam.

I. INTRODUCTION

Dense visual simultaneous localization and mapping (SLAM) is a fundamental and challenging problem in computer vision. It involves updating a map of an unknown environment while simultaneously tracking the location of an agent. In interactive applications such as AR/VR and robotics, it is crucial for a SLAM system to possess not only accurate tracking and mapping capabilities but also *global consistency* to ensure robustness and *low latency* for optimal responsiveness.

Traditional SLAM systems, such as [1], [2], [3], exhibit low latency, high-precision tracking, and employ loop detection to ensure global consistency. However, these systems are limited to constructing sparse point maps that lack dense geometry and texture information.

Recent advances in neural implicit representations have enabled accurate and dense 3D surface reconstruction. Consequently, several neural implicit representation-based SLAM systems have been proposed. As pioneer works in this direction, iMAP [5] and NICE-SLAM [4] achieve both tracking and mapping based on the neural representation, resulting in high-fidelity scene reconstruction. Nevertheless, the tracking based on neural representation lacks support for loop closure, leading to poor performance in large scenes

This work was supported by the National Key R&D Program of China under Grant 2021ZD0114500.

¹Yunxuan Mao, Xuan Yu, Zhuqing Zhang, Yue Wang, Rong Xiong, and Yiyi Liao are with Zhejiang University, Hangzhou, China.

²Kai Wang is with the Application Innovate Lab, Huawei Incorporated Company, Beijing, China.

*Corresponding author.

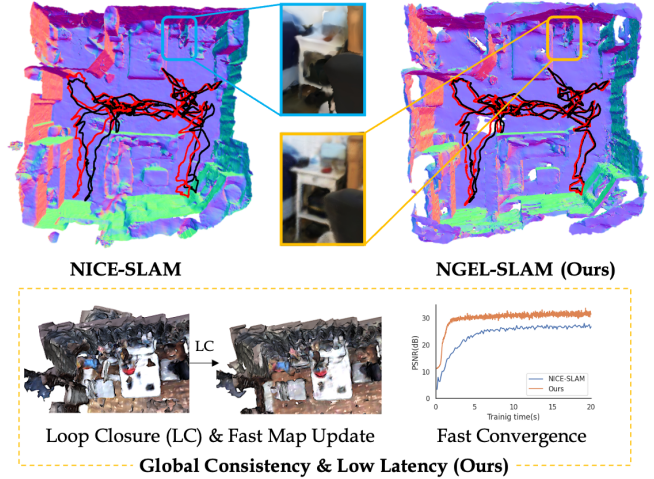


Fig. 1: **Rendering and tracking results.** Compared with NICE-SLAM [4], our method renders higher-fidelity images and provides more precise camera tracking results. Additionally, our method performs fast convergence and enables low-latency map updates after loop closure, enabling it to run 10x faster than NICE-SLAM. The ground truth camera trajectory is shown in black, and the estimated trajectory is shown in red.

due to the lack of global consistency, as illustrated in Fig. 1. Even if loop closure is integrated into their systems, e.g., by replacing tracking with traditional SLAM systems, re-training the entire map wrt. the updated pose is significantly time-consuming. Moreover, the slow convergence of their mapping networks further prevents them from meeting the requirement of low-latency mapping.

To address these challenges, we propose NGEL-SLAM, a Neural implicit representation-based Global consistEnt Low-latency SLAM system. NGEL-SLAM combines the tracking accuracy of the traditional SLAM system, ORB-SLAM3 [3], with the capability of neural implicit representations to extract dense meshes and generate high-fidelity images. The traditional feature-based tracking module allows us to easily incorporate loop closure, enabling tracking with global consistency. When a loop is detected, our mapping module immediately updates the map with low latency, thanks to the design of representing the scene as multiple neural implicit sub-maps. This avoids re-training the entire scene map upon loop closure but fixes most errors by simply updating the relative poses of the sub-maps. We further

selectively fine-tune each sub-map based on the updated pose when necessary. The mapping module updates the map every 14 ms and converges in a few iterations thanks to the octree-based sub-map representation, see Fig. 1. The fast convergence and rapid response to loop closure make our system a truly low-latency system that achieves global consistency. During inference, we introduce an uncertainty-based approach to select the best sub-maps for rendering an image at a given viewpoint. We evaluate our method on a variety of real-world and synthetic datasets of different sizes and demonstrate its robustness and accuracy.

Our main contributions are summarized as follows:

- Accurate tracking and mapping. Our system achieves accurate performance by running the traditional feature-based tracking module and neural implicit scene mapping module in parallel.
- Global consistency. Our system incorporates loop closure to ensure global consistency.
- Low latency. Our system achieves both low latency tracking and mapping. The low latency mapping includes the quick response to loop closure and fast convergence given a new frame.
- We conduct experiments on various datasets and demonstrate competitive performance compared to baselines.

II. RELATED WORKS

A. Visual SLAM

Visual SLAM (Simultaneous Localization and Mapping) is an important research area in robotics and computer vision. Over the past two decades, significant advancements have been made in SLAM research. Visual SLAM systems can be categorized into traditional SLAM systems [6], [7], [6], [8], [9], [10], [1], [2], [3] and learning-based SLAM systems [11], [12], [13]. During the evolution of traditional SLAM systems, PTAM [7] pioneers the separation of the SLAM task into tracking and mapping, achieving real-time performance. A significant milestone in the advancement of visual SLAM was ORB-SLAM [1], which employs an efficient feature-based approach to estimate camera trajectory and construct a 3D map. Traditional SLAM systems exhibit robustness to noise, allowing operation in diverse environments, scaling to large-scale environments, and handling of complex maps. However, they are limited to constructing sparse point-based maps while achieving accurate camera trajectory estimation. In recent years, a line of works proposes to replace a part of the traditional SLAM system as a learning-based module, e.g., employing convolutional neural networks to extract features [11], [12], [13]. These methods slow down the tracking process, while still facing the challenge of lacking a dense scene representation. In our work, we take advantage of the great tracking performance of traditional SLAM systems, using ORB-SLAM3 [3] to track the camera poses and perform loop closure for maintaining global consistency.

B. Neural Implicit Scene Representation

Neural implicit representations have gained popularity in various applications, including 3D reconstruction [14], [15],

[16], [17], [18], [19], [20], novel view synthesis [21], [22], [23] and 3D generative models [24], [25], [26]. In robotics, neural implicit representations have been used for object tracking and SLAM, enabling the construction of environment maps and estimation of robot or camera positions [5], [4], [27], [28], [29], [30], [31], which are closely related to our work. Among them, iMAP [5], NICE-SLAM [4], and ESLAM [28] achieve both tracking and mapping based on the neural representation. However, in large-scale scenes, these methods suffer from the absence of loop closure and global bundle adjustment (BA), resulting in reduced tracking accuracy, a lack of global consistency, and inadequate training speed to meet low latency requirements. While [30] and [31] enhance tracking accuracy by substituting neural tracking with a traditional method that incorporates loop closure, they fail to update the scene representation after loop closure, resulting in a lack of global consistency in their maps. In this paper, we also employ the traditional tracking module but enable low-latency map update after loop closure, as we represent the scene with multiple local maps. While [32], [33] also adopt the submap representation, they are offline systems. In contrast, we present a real-time SLAM system that consumes consecutive RGB-D frames and produces accurate poses and a global consistent map with low latency.

III. SYSTEM

Fig. 2 provides an overview of our system. In this section, we introduce our system from the following aspects: tracking and mapping module (III-A), dynamic local mapping (III-B), loop closing (III-C), and uncertainty-based image rendering (III-D).

A. Tracking and Mapping Module

Our system achieves simultaneous estimation of accurate camera poses and 3D scene geometry and appearance from RGB-D video input through the utilization of two modules: tracking and mapping. The tracking module is based on ORB-SLAM3, the outstanding traditional SLAM system, and the mapping module represents the scene using multiple implicit neural maps.

The system comprises three processes: tracking, dynamic local mapping, and loop closing. During the tracking process, the tracking process estimates the camera pose $[\mathbf{R}|\mathbf{t}]_i$ and determines if the input frame I_i is a keyframe. Each keyframe is fed into the dynamic local mapping process, which involves performing local BA by the tracking module and selecting the appropriate local map for training in the mapping module. Upon detecting a loop closure, the loop closing process begins, during which the tracking module optimizes all keyframe poses using global Bundle Adjustment (BA), and the mapping module promptly responds to significant changes in tracking poses by adjusting the sub-maps, followed by map fine-tuning. All three processes run in parallel. The high-speed training and quick response to loop closure make our system meet the low latency requirement for real-world applications. Note that we refer to *low latency*

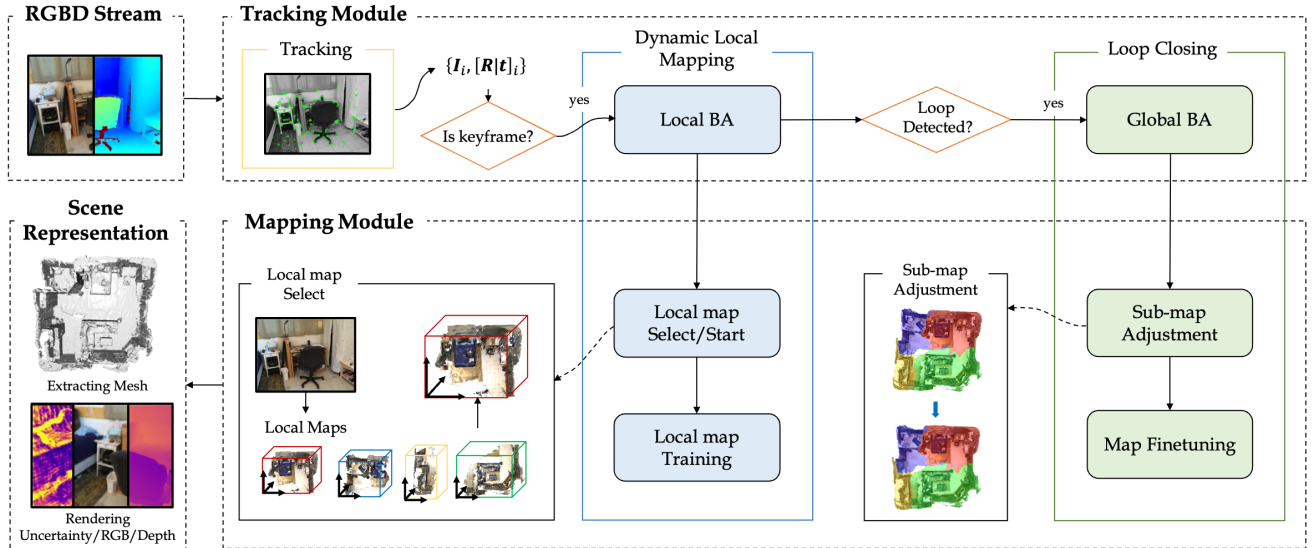


Fig. 2: **System overview.** Our proposed system comprises two primary modules: the tracking module and the mapping module. It can be further divided into three processes: tracking, dynamic local mapping, and loop closing. These three processes work together to ensure global consistency and low latency in our system. The tracking process takes an RGB-D stream as input and tracks the camera pose in real-time. If a frame is selected as a keyframe, it is passed to the dynamic local mapping process. In this process, the tracking module performs local BA while the mapping module trains the corresponding local map. When a loop is detected, the loop closing process optimizes the camera poses using global BA and updates the scene representation. All processes are executed in parallel.

as the capability of integrating most of the information of a keyframe into the map before receiving a new keyframe from the captured RGB-D stream, regardless of whether a loop closure is detected or not.

B. Dynamic Local Mapping

When a frame is decided to be a keyframe, the tracking module employs local BA to optimize the associated keyframes and provides the poses and the new keyframe to the mapping module. The mapping module first performs local map selection, in which it determines whether the new keyframe belongs to an existing local map by evaluating co-visible relations. This prevents the mapping module from generating redundant local maps, especially after loop closure. In cases where the keyframe does not belong to any existing local map, a new local map is initialized and anchored relative to the current keyframe, known as the anchor frame. After the local map is determined, the mapping module trains the local map with keyframes whose poses are optimized in local BA. The map is updated every 14ms, which ensures the low latency requirement.

C. Loop Closing

To ensure global consistency and rectify accumulated errors, loop detection is employed in our system. When a loop is detected, the tracking module performs global BA. However, global BA results in an immediate change in the previous predicted camera poses, commonly known as a trajectory jump. To address these changes, single volume-based implicit neural methods [4], [28] necessitate re-training

of the scene representation and updates to most of the previously trained parameters, which is time-consuming.

In our system, we represent the entire scene with multiple local maps. When a global BA is completed, the scene representation undergoes a two-stage optimization from coarse to fine. In the first stage, the mapping module performs sub-map adjustment, which updates the scene representation by transforming the maps with the anchor keyframe poses. In the second stage, the mapping module fine-tunes the previous local maps to rectify errors. The first stage is a real-time adjustment that corrects errors among the local maps. Our experimental results demonstrate that this stage effectively rectifies a significant portion of the errors in the scene representation. The second stage involves a sub-real-time optimization that eliminates small errors within the local maps, further enhancing the accuracy of the scene representation.

D. Uncertainty-based Image Rendering

As our system incorporates multiple sub-maps, there are two scenarios to consider when rendering images from a given viewpoint. The first scenario arises when the view frustum intersects a sub-map entirely, allowing us to render the image using that specific sub-map. The second scenario occurs when the view frustum lies at the boundary of different sub-maps, making it impossible to generate a complete image from a single sub-map. In such cases, we employ a pixel-wise fusion of the image based on the lowest uncertainty.

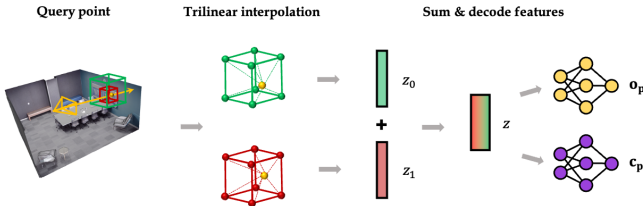


Fig. 3: **Mapping network.** The Mapping network employs a sparse octree structure to store multi-level features and two single MLPs.

IV. MAP REPRESENTATION AND TRAINING

A. Octree-based Implicit Neural Representation

Voxel grid-based NeRF architectures [4], [20] have been shown to converge quickly. However, since most of the space is unoccupied, the dense grid structure used in these methods is memory-wasting. To address this issue, we draw inspiration from NGLOD [34] and use a sparse octree-based grid. The octree only grows where the space is occupied, making it memory-efficient. Multi-level feature vectors, denoted by \mathbf{z} , are stored on the nodes of the octree. As shown in Fig. 3, when we query a point \mathbf{p} , the feature $\mathbf{z}_{\mathbf{p}}^{(i)}$ of \mathbf{p} in level i is obtained by trilinear interpolation. Then, similar to NGLOD, we sum the features in different active levels (i.e., levels that store features) to obtain $\mathbf{z}_{\mathbf{p}}$ as the feature of \mathbf{p} . Furthermore, we have modified the octree to an incremental paradigm, where a Morton code table is maintained to provide the position of new points added to the octree.

In our method, we employ two small MLP decoders, one for occupancy and another for color. To calculate the occupancy $o_{\mathbf{p}}$ and color $\mathbf{c}_{\mathbf{p}}$ at a given point $\mathbf{p} \in \mathbb{R}^3$ in space, we use the following equations:

$$o_{\mathbf{p}} = \sigma(f_{occ}(\mathbf{z}_{\mathbf{p}})), \quad \mathbf{c}_{\mathbf{p}} = f_{color}(\mathbf{z}_{\mathbf{p}}) \quad (1)$$

Here, $\mathbf{z}_{\mathbf{p}}$ is the feature vector at point \mathbf{p} , f_{occ} and f_{color} are the occupancy and color decoders, respectively, and σ is the sigmoid function.

B. Volume Rendering

To optimize our scene representation framework in section IV-A, we use differentiable volume rendering proposed in NeRF [21]. Given camera intrinsic parameters and current camera pose, we cast a ray \mathbf{r} from the camera center \mathbf{o} through the pixel along its normalized view direction \mathbf{v} .

1) *Depth and Color:* We sample N points along a ray, denoted as $\mathbf{x}_i = \mathbf{o} + d_i \mathbf{v}$, where d_i is the depth of point \mathbf{x}_i , and $i \in 1, \dots, N$. The predicted occupancy values and color values of these points are denoted as o_i and \mathbf{c}_i , respectively. For a given ray \mathbf{r} , we can calculate the depth \hat{D} and color $\hat{\mathbf{C}}$ as:

$$\hat{D}(\mathbf{r}) = \sum_{i=1}^N T_i \alpha_i d_i \quad \text{and} \quad \hat{\mathbf{C}}(\mathbf{r}) = \sum_{i=1}^N T_i \alpha_i \mathbf{c}_i \quad (2)$$

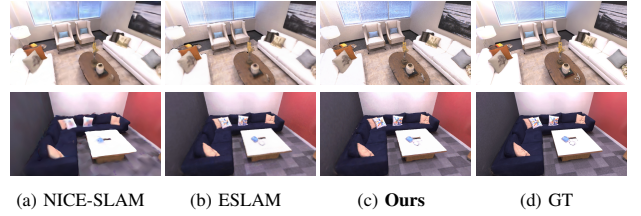


Fig. 4: **Rendering results on the Replica dataset.**

Here, $\alpha_i = o_i$ and $T_i = \prod_{j=1}^i (1 - o_j)$ correspond to the transmittance and alpha value of sample point i along the ray \mathbf{r} , respectively.

2) *Voxel-based sampling:* To fully utilize the octree structure, we adopt a voxel-based sampling policy. Given a ray \mathbf{r} , we query the voxels it intersects and sample points along the ray within these voxels. Since the octree only grows where the points are occupied, the sampled points are all either close to the object surface or inside the objects. We sample a fixed number of N_{point} points per voxel. Therefore, for a ray \mathbf{r} intersecting with N_{voxel} voxels, we sample $N = N_{point} \times N_{voxel}$ points along the ray.

3) *Optimizing:* To optimize the scene representation features and decoders in Section IV-A, we uniformly select M pixels in the current frame and train the scene representation using photometric loss and geometry loss. The photometric loss is the mean squared error (MSE) loss between the rendered and ground truth color images, while the geometry loss is a simple \mathcal{L}_1 loss between the rendered and ground truth depths. Specifically, we define the losses as follows:

$$\mathcal{L}_p = \frac{1}{M} \sum_{i=1}^M |\hat{\mathbf{C}} - \mathbf{C}_{gt}|_2 \quad \mathcal{L}_g = \frac{1}{M} \sum_{i=1}^M |\hat{D} - D_{gt}| \quad (3)$$

We optimize the features \mathbf{z} and decoder parameters θ jointly by minimizing the loss function:

$$\min_{\mathbf{z}, \theta} \lambda_p \mathcal{L}_p + \mathcal{L}_g \quad (4)$$

Here, λ_p is the weight of the photometric loss.

4) *Uncertainty:* Since occupancy follows a Bernoulli distribution, the occupancy value $o_{\mathbf{p}}$ of a point \mathbf{p} represents the probability that the point is occupied, and its variance can be calculated as $Var = o_{\mathbf{p}}(1 - o_{\mathbf{p}})$. The occupancy variance of a ray can then be rendered as:

$$\sigma_o^2(\mathbf{r}) = \frac{1}{N} \sum_{i=1}^N o_i(1 - o_i) \quad (5)$$

To account for the higher uncertainty in unobserved regions, we set the variance of these regions to be 0.25.

V. EXPERIMENT

In this section, we validate our SLAM framework on real and synthetic datasets of varying sizes and complexities. We also compare our method with existing implicit representation-based methods.

TABLE I: **Quantitative comparison of mapping on the Replica dataset.** Data averaged from eight scenes. GT pose and Est pose respectively represent rendering with ground truth poses and rendering with estimated poses.

	Evaluation @ GT Pose			Evaluation @ Est Pose		
	NICE-SLAM [4]	ESLAM [28]	Ours	NICE-SLAM [4]	ESLAM [28]	Ours
Depth L1 [cm] ↓	3.53	2.81	1.28	1.58	2.28	0.65
PSNR ↑	21.98	27.33	29.53	26.20	29.25	30.44
SSIM ↑	0.775	0.836	0.864	0.832	0.855	0.876
LPIPS ↓	0.247	0.204	0.156	0.233	0.190	0.151

TABLE II: **Quantitative comparison of tracking on TUM RGB-D.** ATE-RMSE [cm] is used as the metric.

	fr1/desk	fr2/xyz	fr3/office
iMAP [5]	4.9	2.0	5.8
NICE-SLAM [4]	2.7	1.8	3.0
ESLAM [28]	2.5	1.1	2.4
Ours	1.5	0.5	1.0

TABLE III: **Quantitative comparison of tracking on ScanNet.** ATE-RMSE [cm] is used as the metric.

Scene ID	0000	0059	0106	0169	0181	0207	Avg.
iMAP [5]	55.95	32.0	17.50	70.51	32.10	11.91	36.67
NICE-SLAM [4]	8.64	12.25	8.09	10.28	12.93	5.59	9.63
ESLAM [28]	7.41	8.64	7.48	6.73	9.03	5.72	7.50
Ours	7.23	6.98	7.95	6.12	10.14	6.27	7.44

A. Experiment Setup

1) *Datasets*: We consider 4 datasets for evaluation: Replica [35], ScanNet [36], TUM RGB-D dataset [37] for tracking and mapping capability analysis, and the apartment dataset provided by NICE-SLAM for ablation study.

2) *Baselines*: We compare our method to three existing open-source state-of-the-art RGB-D implicit neural SLAM systems: iMAP [5], NICE-SLAM [4], and ESLAM [28]. The experiments are designed following NICE-SLAM [4].

3) *Metrics*: To evaluate the geometry and appearance, we use L1 loss for rendered depth images and the PSNR, SSIM, and LPIPS for rendered color images on 100 randomly sampled poses. Additionally, we evaluate camera tracking using ATE RMSE.

4) *Implementation Details*: Our SLAM system runs on a single NVIDIA RTX 3090 GPU. We set the number of sampling points along a ray in every intersecting voxel to $N_{point} = 10$ and photometric loss weighting to $\lambda_p = 1$. We use sample $M = 5000$ pixels. Decoders are both MLPs with a hidden feature dimension of 32 and 2 fully-connected blocks. The RGBD stream is input at a rate of 10Hz.

B. Experiment Results

1) *Evaluation on Replica [35]*: We evaluate our method’s scene representation capabilities on 8 scenes from the Replica dataset and compare its performance in geometry and appearance representation to other baseline methods. Following other papers, we first evaluate the reconstruction performance by rendering the images with ground truth camera poses, which means the results include tracking errors. Furthermore, to better independently verify model reconstruction capabilities, we use the camera poses estimated by the tracking modules to render the images and evaluate the results. The results presented in TABLE I demonstrate that our method outperforms baseline methods significantly across all metrics. Qualitatively, as shown in Fig. 4, our method produces geometry that is more accurate and detailed, with higher-fidelity textures.



Fig. 5: **Rendering results on the ScanNet dataset.**

2) *Evaluation on TUM RGB-D [37]*: We evaluated our method’s camera tracking performance on the TUM RGB-D dataset, which is of a small scale. Our method significantly outperformed implicit neural scene representation methods as demonstrated in TABLE II. The result shows that the traditional tracking module with loop closure in our system is more accurate than the neural implicit tracking.

3) *Evaluation on ScanNet [36]*: We evaluated our method’s scene representation capabilities on a larger real-world scene dataset ScanNet [36], which is more challenging than synthetic ones in tracking and mapping. We selected the same scenes as NICE-SLAM.

The quantitative experiment of tracking on ScanNet is shown in TABLE III. Our method outperforms baseline methods in most scenes and on average. The qualitative experiment results on ScanNet are shown in Fig. 5 and the extracted mesh is shown in Fig. 1. The images rendered by our method are sharper and more detailed than those produced by NICE-SLAM and ESLAM.

C. Performance Analysis

In the previous subsection, We evaluate the geometry and appearance of scene reconstruction quality and camera tracking accuracy. However, a good SLAM system does not only perform well in these aspects. In the following, we evaluate other properties of our method and compare them with other implicit neural SLAM systems.

TABLE IV: **Runtime & Memory usage.** Mapping time refers to the duration of a single iteration.

	Mapping time [ms]	# Frames	Mem. [MB]
iMAP [5]	448	400	1.04
NICE-SLAM [4]	130	400	12.02
ESLAM [28]	19	400	6.79
Ours	14	120	5.32

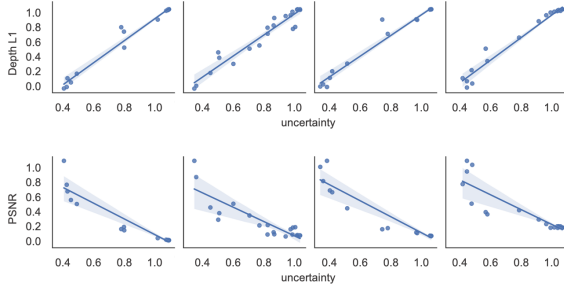


Fig. 6: **Relation between uncertainty and depth L1, PSNR** on four images. We normalize depth L1 and PSNR to $[0, 1]$ based on their worst and best values for better illustration.

1) *Runtime and Memory Usage:* We evaluate the mapping time, the number of mapping frames, and the memory usage on the eight scenes of the Replica dataset in TABLE IV. Here, mapping time refers to the duration to process $M = 1000$ pixels in a single iteration, which is the same as NICE-SLAM. As shown in the first two columns of TABLE IV, our mapping time of each single iteration is the shortest compared to the baselines. Further, our method requires significantly fewer frames for mapping while achieving better mapping performance, as we use only keyframes for mapping. This further speeds up training as the data I/O time is reduced. This allows our system to meet the low-latency requirement, i.e., the majority of information of a keyframe is integrated into the map before receiving a new keyframe.

Regarding memory usage, iMAP requires the lowest memory footprint as it only uses a single MLP to represent the scenes. Note that our method outperforms all the other baselines thanks to the octree structure.

2) *Uncertainty:* We propose an uncertainty calculation method in section III-D. The uncertainty is designed to identify pixels where occupancies are not well-trained. In this experiment, we investigate the relationship between uncertainty and depth L1 and PSNR. We randomly selected four viewpoints and rendered the depth map and color map of



(a) Before LC (b) After LC (c) After FT (d) GT

Fig. 7: **Ablation study on the sub-map adjustment and map fine-tuning.** LC refers to loop closure and FT refers to map fine-tuning.

TABLE V: **Ablation study on the map fine-tuning (FT).**

	Ours w/o FT	Ours
Depth L1 [cm] ↓	1.34	1.28
PSNR ↑	28.39	29.53
SSIM ↑	0.837	0.864
LPIPS ↓	0.203	0.156

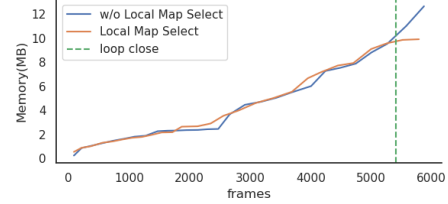


Fig. 8: **Ablation study on local map selection.**

the corresponding viewpoints in each volume. As shown in Fig. 6, depth L1 is positively correlated with uncertainty, and PSNR is negatively correlated with uncertainty, indicating our uncertainty-based image selection strategy is effective.

D. Ablation study

In this section, we show the importance of our sub-map adjustment for loop-closing scenes and how useful map fine-tuning and local map selection are.

1) *Sub-map adjustment:* We verify the effectiveness of sub-map adjustment. Given a sequence of trajectory with loop closure, we compared the RGB-D image rendered before and after loop closure. As shown in Fig. 7 (a) and (b), our method with sub-map adjustment can quickly respond to a loop closure without re-training the models and eliminate most of the errors.

2) *Map Finetuning:* We verify the effectiveness of our map finetuning. We compare our method with and without the map finetuning on the Replica dataset. As shown in TABLE V, we quantitatively show that the map finetuning improves the performance of our mapping module. Also, the rendering images with and without map finetuning are shown in Fig. 7. After map finetuning, the quality of rendering images is higher.

3) *Local Map Selection:* When a new keyframe comes in, our method determines whether it belongs to previous local maps or a new local map is required. This method ensures no redundant local maps for the same place. As shown in Fig. 8, the method keeps the memory usage from growing after loop closure.

VI. CONCLUSIONS

In this paper, we propose a global consistent neural implicit representation-based SLAM system, NGEL-SLAM, for indoor scenes. Combining the traditional tracking and neural implicit scene representation, our method generates high-precision mesh while tracking accurate camera poses. Compared to other neural implicit SLAM systems, our approach ensures global consistency and low latency which is more suitable for real-world applications.

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] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
- [3] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.
- [4] 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.
- [5] 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.
- [6] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.
- [7] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.
- [8] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.
- [9] 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.
- [10] J. Engel, T. Schöps, and D. 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*. Springer, 2014, pp. 834–849.
- [11] M. Yokozuka, S. Oishi, S. Thompson, and A. Banno, "Vitamin-e: Visual tracking and mapping with extremely dense feature points," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9641–9650.
- [12] M. R. U. Saputra, P. P. de Gusmao, C. X. Lu, Y. Almalioglu, S. Rosa, C. Chen, J. Wahlström, W. Wang, A. Markham, and N. Trigoni, "Deeptio: A deep thermal-inertial odometry with visual hallucination," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1672–1679, 2020.
- [13] R. Li, S. Wang, and D. Gu, "Deepslam: A robust monocular slam system with unsupervised deep learning," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 4, pp. 3577–3587, 2020.
- [14] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 165–174.
- [15] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4460–4470.
- [16] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7462–7473, 2020.
- [17] M. Oechsle, S. Peng, and A. Geiger, "Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5589–5599.
- [18] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su, "Mvs-nerf: Fast generalizable radiance field reconstruction from multi-view stereo," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 124–14 133.
- [19] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, 2020, pp. 523–540.
- [20] J. Wang, T. Bleja, and L. Agapito, "Go-surf: Neural feature grid optimization for fast, high-fidelity rgb-d surface reconstruction," *arXiv preprint arXiv:2206.14735*, 2022.
- [21] 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.
- [22] K. Zhang, G. Riegler, N. Snavely, and V. Koltun, "Nerf++: Analyzing and improving neural radiance fields," *arXiv preprint arXiv:2010.07492*, 2020.
- [23] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. 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*, 2021, pp. 7210–7219.
- [24] M. Niemeyer and A. Geiger, "Giraffe: Representing scenes as compositional generative neural feature fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 453–11 464.
- [25] K. Schwarz, Y. Liao, M. Niemeyer, and A. Geiger, "Graf: Generative radiance fields for 3d-aware image synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 20 154–20 166, 2020.
- [26] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein, "pigan: Periodic implicit generative adversarial networks for 3d-aware image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 5799–5809.
- [27] Z. Zhu, S. Peng, V. Larsson, Z. Cui, M. R. Oswald, A. Geiger, and M. Pollefeys, "Nicer-slam: Neural implicit scene encoding for rgb slam," *arXiv preprint arXiv:2302.03594*, 2023.
- [28] M. M. Johari, C. Carta, and F. Fleuret, "Eslam: Efficient dense slam system based on hybrid representation of signed distance fields," *arXiv preprint arXiv:2211.11704*, 2022.
- [29] X. Yang, H. Li, H. Zhai, Y. Ming, Y. Liu, and G. Zhang, "Vox-fusion: Dense tracking and mapping with voxel-based neural implicit representation," in *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2022, pp. 499–507.
- [30] C.-M. Chung, Y.-C. Tseng, Y.-C. Hsu, X.-Q. Shi, Y.-H. Hua, J.-F. Yeh, W.-C. Chen, Y.-T. Chen, and W. H. Hsu, "Orbeez-slam: A real-time monocular visual slam with orb features and nerf-realized mapping," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9400–9406.
- [31] Y. Haghghi, S. Kumar, J. P. Thiran, and L. Van Gool, "Neural implicit dense semantic slam," *arXiv preprint arXiv:2304.14560*, 2023.
- [32] X. Zhong, Y. Pan, J. Behley, and C. Stachniss, "Shine-mapping: Large-scale 3d mapping using sparse hierarchical implicit neural representations," *arXiv preprint arXiv:2210.02299*, 2022.
- [33] X. Yu, Y. Liu, S. Mao, S. Zhou, R. Xiong, Y. Liao, and Y. Wang, "Nf-atlas: Multi-volume neural feature fields for large scale lidar mapping," *arXiv preprint arXiv:2304.04624*, 2023.
- [34] T. Takikawa, J. Litalien, K. Yin, K. Kreis, C. Loop, D. Nowrouzezahrai, A. Jacobson, M. McGuire, and S. Fidler, "Neural geometric level of detail: Real-time rendering with implicit 3d shapes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 358–11 367.
- [35] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou, K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. D. Nardi, M. Goesele, S. Lovegrove, and R. Newcombe, "The Replica dataset: A digital replica of indoor spaces," *arXiv preprint arXiv:1906.05797*, 2019.
- [36] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3d reconstructions of indoor scenes," in *Proc. Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017.
- [37] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.