IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Spatial Coordinate Transformation for 3D Neural Implicit Mapping

Kyeongsu Kang, *Member, IEEE*, Seongbo Ha, *Graduate Student Member, IEEE*, Sibaek Lee, *Graduate Student Member, IEEE*, and Hyeonwoo Yu, *Member, IEEE*

***Abstract*—Implicit Neural Representation (INR)-based SLAM has a critical issue where all keyframes must be stored in memory for post-training whenever a remapping is needed due to the neural network's weights themselves representing the map. To address this, previous INR-based SLAM proposed methods to modify INR-based maps without changing the neural network's weights. However, these approaches suffer from low memory efficiency and increased space complexity. In this letter, we introduce a remapping method for INR-based maps that does not require post-training the neural network's weights and needed low space cost. The problem of function modification, such as updating a map defined as a neural network function, can be viewed as transforming the function's domain. Leveraging function domain transformation, we propose a method to update INR-based maps by identifying the transformation function between the post-optimization and pre-optimization domains. Additionally, to prevent cases where the transformation between the post-optimization and pre-optimization domains does not form a one-to-many relationship, we introduce a temporal domain and propose a method to find the spatial coordinate transformation function accordingly. Evaluations in INR-based techniques demonstrate that our proposed method effectively update to maps while requiring significantly less memory compared to existing remapping approaches.**

***Index Terms*—Mapping, SLAM.**

I. Introduction

IMPLICIT Neural Representation (INR)-based SLAM has gained significant attention since its introduction due to its capabilities in predicting unseen views and represent watertight maps [25]. In traditional SLAM, map representations such as point clouds or voxel-based maps struggled to generate maps for unobserved areas. However, INR offers the advantage of making rough predictions about unseen views, enabling the representation of unobserved areas in the map [25], [33], [34]. Consequently, INR-based SLAM has overcome the limitations of conventional map representation methods. As research on INR-based SLAM has progressed, further advancements have been made, including INR-based mapping in large-scale outdoor environments [4], [5], [13], [20], improvements in learning speed [5], [8], [27], and the generation of more precise maps [2], [12].

Despite its advantages, INR-based SLAM has a major limitation: remapping is challenging [12]. In traditional SLAM, map representations such as point clouds or voxel-based allow relatively easy update since these maps explicitly present landmark pose. When an explicit representation is used, remapping can be achieved by simply "transforming" the affected regions. In contrast, INR-based maps encode the map within the weights of the neural network. As a result, there is no direct way to "transform" a specific part of the map representation, and instead, optimal weight updates must be determined to achieve the desired remapping [2], [28], [29], [32]. Consequently, INR-based maps require post-training whenever updates are needed, making the process computationally expensive [30]. This poses a significant drawback in INR-based SLAM, where remapping is essential for optimization processes such as loop closing or global bundle adjustment (BA), leading to high computational resource requirements [12], [30]. Recent INR-based SLAM methods have recognized this challenge and proposed solutions such as using multiple submaps [10], [14], [26], [30], [31] or designing novel neural network architectures that allow updates without post-training [12], [20], [22]. However, these approaches suffer from increased memory usage and higher space complexity.

In this letter, we propose a new remapping approach for INR-based SLAM that is memory-efficient and does not require post-training. Our remapping method performs by approximating, via nonlinear regression a spatial coordinate transformation function that maps the pre-optimization domain X to the post-optimization domain X'. In addition, we introduce a temporal domain and utilize Gaussian Process (GP)-based uncertainty estimation [23] to resolve the one-to-many relationship, ensuring a robust transformation between X and X'. By applying this transformation to update the density and radiance fields at corrected positions rather than the original sampled positions, our approach maintains map quality while avoiding the need for post-training. Our approach does not require additional training for remapping. Moreover, it is compatible with single-network implicit neural representations and does not impose constraints on specific neural architectures. This allows it to be widely applicable to INR-based 3D representations such as Neural Radiance Fields (NeRF) [15].

Received 3 March 2025; accepted 16 July 2025. Date of publication 1 August 2025; date of current version 12 August 2025. This article was recommended for publication by Associate Editor P. Checchin and Editor S. Behnke upon evaluation of the reviewers' comments. This work was supported by the National Research Foundation of Korea (NRF) under Grant RS-2022-NR073038 and Grant RS-2024-00359937. *(Corresponding author: Hyeonwoo Yu.)*

The authors are with the Department of Intelligent Robotics, Sungkyunkwan University, Suwon 16419, South Korea (e-mail: thithin0821@skku.edu; sobo3607@skku.edu; lmjlss@skku.edu; hwyu@skku.edu).

The code is available at: https://github.com/Lab-of-AI-and-Robotics/SCT_NIM.

Digital Object Identifier 10.1109/LRA.2025.3595031

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

In summary, we propose a method that finds a nonlinear regression function between X and X' for remapping in INR-based SLAM. This enables remapping even in single-network INR-based SLAM without requiring post-training. Our contributions are as follows:

- We introduce a remapping method that utilizes a transformation function between the pre-optimization domain X and the post-optimization domain X'. This enables efficient remapping even in single-network INR-base SLAM without requiring additional training or specific network structures.
- We address the one-to-many relationship problem between X and X' by incorporating a temporal domain and leveraging uncertainty estimation. By converting the one-to-many relationship into a one-to-one mapping, we effectively solve the nonlinear regression problem, enabling robust remapping.

We evaluate our method on indoor datasets to assess its remapping performance. Furthermore, we examine its applicability to various INR-based techniques, demonstrating that our approach is not limited to a specific INR method but is broadly applicable. Additionally, experiments using single-network INR-based SLAM show that our method can be successfully applied to real SLAM scenarios.

II. Related Works

INR-based SLAM methods face challenges during loop closure events [30]. When loop closure occurs, a global optimization of the camera trajectory and 3D reconstruction is required. Consequently, updating the INR-based map involves modifying the neural network's weights, and a common approach to obtain an optimal INR-based map is post-training. Recent SLAM systems, such as GO-SLAM [32], use post-training methods that leverage multi-resolution hash encoding [7], [16] to enhance computing efficiency. However, as the amount of training data grows, post-training demands significant computational resources [2], [28], [29]. This process can be highly intensive, thereby hindering real-time performance. Without keyframe selection or active learning-based data filtering, updating an implicit neural representation map requires considerable time and memory. This challenge is especially critical for on-device robotics, where computational resources are limited, making extensive post-training infeasible.

To mitigate the challenges of post-training implicit neural networks during loop closure, some SLAM systems adopt a submap-based approaches that utilizes multiple implicit neural networks. In these methods, only the submaps affected by loop closure require updates, which significantly improves efficiency and scalability in large-scale environments [10], [14], [31]. NIM-REM exemplifies this strategy by modeling each submap with a separate implicit function, thereby facilitating efficient, real-time map corrections [30]. Similarly, MIPS-Fusion encodes local submaps in MLP networks representing Truncated Signed Distance Functions (TSDFs) and performs loop closure through rigid registration of these submaps [26]. However, MIPS-Fusion relies on co-visibility thresholds for loop closure detection, which may limit its ability to correct substantial drifts. To further improve efficiency and scalability, recent SLAM methods have introduced hybrid submap-based approaches that integrate elements of both implicit and explicit representations.

Loopy-SLAM constructs local submaps using neural point clouds and performs global pose optimization by aligning these submaps when a loop closure is detected, thereby enhancing global consistency without reprocessing previous frames [12], [22].

This strategy inherits core ideas from earlier dense map-base approaches that leverage volumetric submap partitioning for scalable, drift-free reconstruction, enabling loop closures without the need to reprocess entire map [6], [9]. These submap-based methods effectively reduce the computational overhead associated with global post-training and improve scalability in large-scale environments. Nevertheless, challenges remain in seamlessly integrating multiple submaps to maintain global consistency, often requiring costly retraining procedures, and in managing memory overhead as the number of submaps grows.

III. Method

A. Problem Redefine

To implement remapping in INR-based SLAM, we redefine the remapping problem from traditional SLAM. In traditional SLAM, map representations such as point clouds, voxels, or occupancy grids can be expressed as functions. One example of an explicit map representation is the point cloud map, which can be defined as follows:

$$M(X) = \begin{cases} 1, & \text{if point is located at X} \\ 0, & \text{otherwise} \end{cases}, \tag{1}$$

where X represents a 3D coordinate. In this formulation, locations containing points have a value of 1, while empty locations have a value of 0. To perform remapping in a function-based representation like point clouds, function transformation is required. When updating an explicit map using graph optimization, the process follows:

$$M^*(X') = M(C(X')), \text{ where } C(X') = X' + \Delta(X') = X, \tag{2}$$

where $M^*(X')$ denotes the post-optimization map, X' represents post-optimization 3D coordinates, and $C(X') = X$ defines the spatial coordinate transformation between the pre-optimization domain X and the post-optimization domain X'. Since the map is represented as a function, as shown in Fig. 2, we can obtain a new map function by transforming the coordinates through function composition. The function $\Delta(X')$ represents the transformation, varying based on X', and shifts each point's coordinates to a other location. This process is analogous to transforming domains in signal processing. Therefore, once the transformation function from the post-optimization domain X' to the pre-optimization domain X is known, an updated map can be generated [17], [18]. In other words, the remapping process can be formulated as finding $C(X')$ and composing a new function with the previous map function. Since $C(X')$ depends on X' in a nonlinear way, it can be viewed as a nonlinear function.

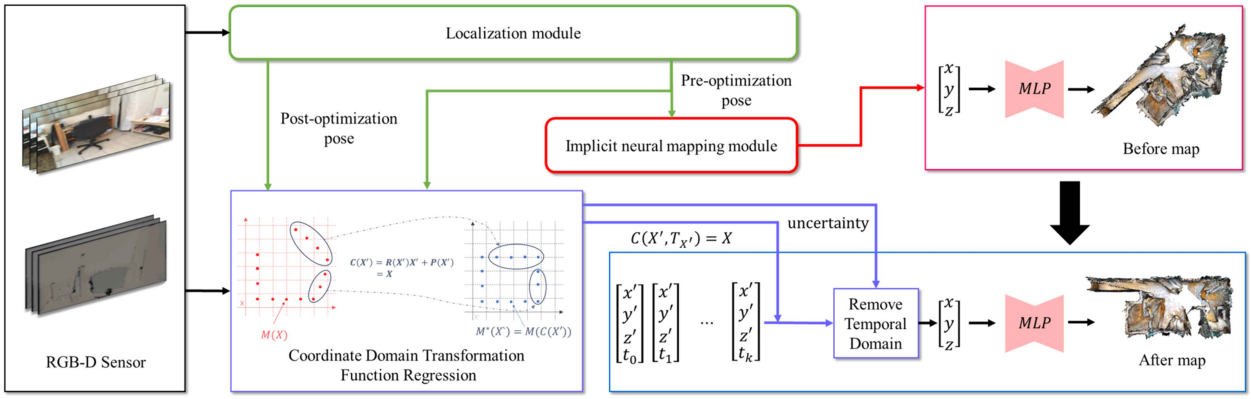
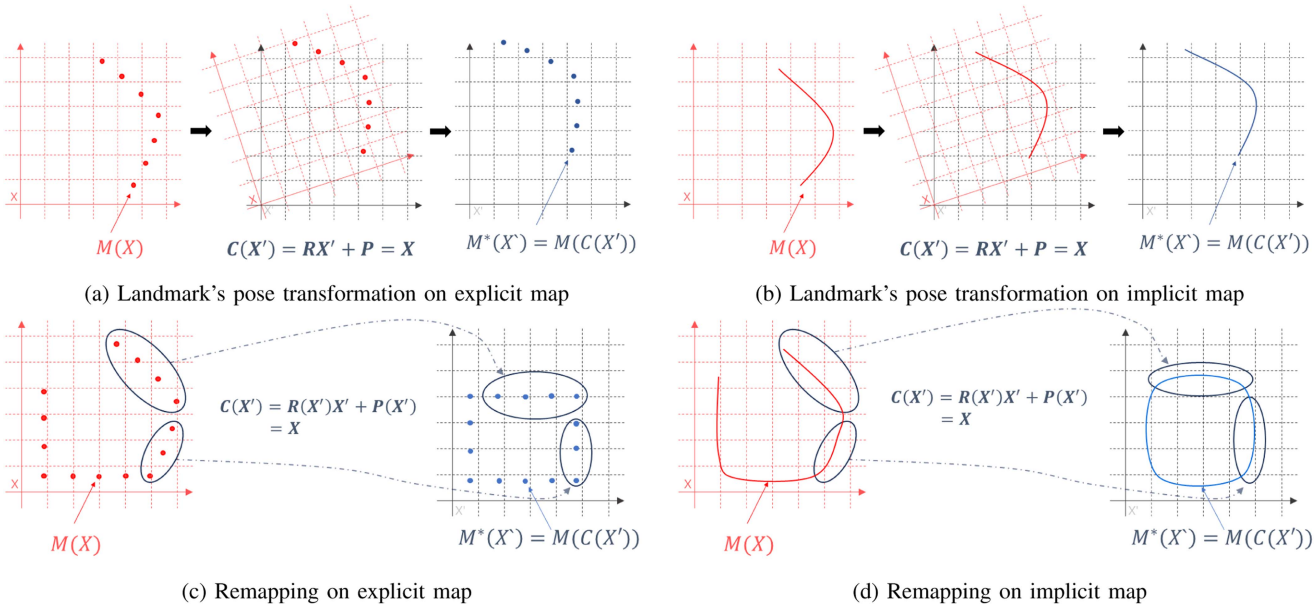


Fig. 1. Method overview. The proposed method estimate a spatial coordinate transform function using post-optimization and pre-optimization poses from the localization module. It then estimates a one-to-one function between X' and X by removing the temporal domain using uncertainty. The estimated function is composed with the before INR map to remapping.



(a) Landmark's pose transformation on explicit map

(b) Landmark's pose transformation on implicit map

(c) Remapping on explicit map

(d) Remapping on implicit map

Fig. 2. Pose transformation and remapping on the function map. The function map can be update using $C(X')$, which transforms the coordinate domain.

Remapping in INR-based SLAM can also be understood as transforming the function defined at the previous coordinates to a new function at the updated coordinates. However, unlike explicit maps, where knowing $C(X')$ values at sampled points is sufficient to remapping, INR-based maps require estimating a globally defined nonlinear function $C(X')$ over the entire domain X'. To address this, we estimate the spatial coordinate transformation function using nonlinear regression.

To achieve this, we use RGB and depth images from each frame, along with the camera poses tracked by a localization module. We obtain a point cloud map P_X by combining keyframe positions with depth images. This process runs independently of the implicit neural mapping module. When loop closing is detected, the localization module performs graph optimization and updates keyframe positions accordingly. By incorporating the updated keyframe positions and depth images, we generate a new point cloud map $P_{X'}$. Since $P_{X'}$ and P_X satisfy the transformation relationship $C(P_{X'}) = P_X$ as defined in (1) and (2), we estimate the nonlinear function $\hat{C}(X')$ using the following optimization:

$$\hat{C}(X') = \underset{C}{\operatorname{argmin}}\, \mathcal{L}(C(P_{X'}), P_X), \tag{3}$$

where $\mathcal{L}$ is the error function between $C(P_{X'})$ and P_X.

B. Temporal Domain

When loop closure occurs in SLAM, multiple transformation points may exist where X' maps to X . This results in a one-to-many relationship between the post-optimization domain X' and the pre-optimization domain X. Due to this one-to-many relationship in the coordinate transformation, the function $\hat{C}(X')$ may incorrectly approximate $C(X')$, leading to errors in mapping from X' to X

To address this issue, we propose to incorporate a temporal domain. In explicit maps, pose graph optimization (PGO) is commonly used during the remapping process to optimize

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

keyframe poses. During this optimization process, the observations associated with each keyframe are updated according to the optimized keyframe poses, under the assumption of rigid-body motion. This implies that the motion of the observations is governed by the transformation of the corresponding keyframe pose. Building on this, we introduce a temporal domain T to estimate $\hat{C}(X')$ in implicit maps. Similar to how keyframe poses and observations in explicit maps are assumed to undergo rigid-body motion in PGO, we assume that the mapping function $\hat{C}(X')$ in implicit representations is also related to the observation time T. By incorporating T, we effectively separate the mapping from X' to X, preserving the functional relationship and preventing conflicts caused by the one-to-many relationship.

Based on the method presented in Section III-A, we refine the estimation of the nonlinear function $\hat{C}(X')$ by integrating the temporal domain of the keyframe. This modification utilizes temporal from adjusted keyframes to separate multiple transformation points in the mapping between X' and X. The updated formulation of (3) incorporating the temporal domain is as follows:

$$\hat{C}(X', T_{X'}) = \underset{C}{\operatorname{argmin}}\, \mathcal{L}(C(P_{X'}, T_{X'}), P_X),$$
$$\text{where} \quad C(P_{X'}, T_{X'}) = P_{X'} + \Delta(P_{X'}, T_{X'}) = P_X. \tag{4}$$

C. Remove Temporal Domain Using Uncertainty Estimation

To apply our approach to INR-based SLAM and mapping, it is necessary to obtain density and radiance field independent of the temporal domain T. Ultimately, our goal is to estimate a stable transformation function $\hat{C}(X')$ that is independent of T. However, as described in Section III-B, since we perform nonlinear function regression using the temporal domain, we initially estimate $\hat{C}(X', T_{X'})$.

During this optimization process, the observations associated with each keyframe are updated according to the optimized keyframe poses, assuming rigid-body motion. This implies that the motion of the observations is governed by the transformation of the corresponding keyframe pose. There are two approaches to remove T: using the expectation of the function and maximizing the likelihood.

The expectation-based approach computes the expected value of the function as follows:

$$\hat{C}(X') = \mathbb{E}_T[\hat{C}(X', T_{X'})] = \int P(T_{X'})\, \hat{C}(X', T_{X'}) d_{T_{X'}}. \tag{5}$$

By computing the expectation of $\hat{C}(X', T_{X'})$ over $T_{X'}$, we obtain a stable estimate of $\hat{C}(X')$.

The likelihood maximization approach selects the most probable T that maximizes the likelihood function:

$$\hat{C}(X') = \hat{C}\left(X', \arg\max_{T_{X'}} P\left(\hat{C}(X', T_{X'})\right)\right). \tag{6}$$

By identifying the maximum likelihood value of $P(\hat{C}(X', T_{X'}))$, this approach yields a more reliable estimate of $\hat{C}(X')$.

To remove the temporal variable T, we utilize the GP-estimated transformation $\hat{C}(X', T_{X'})$ and its uncertainty. The regression using a GP is formulated as follows:

$$C(P_{X'}, T_{X'}) \sim \mathcal{GP}(0, K((P_{X'}, T_{X'}), (P_{X'}, T_{X'})))$$
$$\hat{C}(X', T_{X'}) = K_{PX}(K_{XX} + \sigma_n^2 I)^{-1}(P_X)$$
$$\hat{\Sigma}(X', T_{X'}) = K_{PP} - K_{PX}(K_{XX} + \sigma_n^2 I)^{-1} K_{XP}, \tag{7}$$

where $K(x, x')$ is $\sigma_0 \exp(-||x - x'||^2 / 2\sigma_f^2)$, which is well known as the Radial Basis Function (RBF). By performing a GP regression using the input $(P_{X'}, T_{X'})$ and the output P_X, we estimate $\hat{C}(X', T_{X'})$. Furthermore, the covariance $\hat{\Sigma}(X', T_{X'})$ provides the uncertainty. As a result, we can eliminate the temporal domain T using uncertainty-aware transformations based on (5) and (6).

Through empirical evaluations, we found that maximizing the likelihood provides better results than using expectation-based estimation. This observation is confirmed by the experimental results presented in Section IV-D, specifically in Table IV. Therefore, we adopt the likelihood maximization approach to obtain the density and radiance field

D. Remapping Framework

In this section, we present the specific components and algorithms of the proposed remapping framework. As overview shown in Fig. 1, our remapping process operates independently of the localization and mapping modules. Therefore, the remapping module remains in a "ready" state without actively performing remapping, as it does not need to be synchronized with the SLAM or mapping process.During this "ready" state, when events such as loop detection or PGO occur—resulting in changes to the global pose trajectory—the module records both the pre- and post-optimization pose trajectories, along with the inducing points required for the GP. This enables the remapping process to be executed later using the proposed method, guided by the stored inducing points. When spatial attributes queries are required—such as extracting a mesh from the INR-based map or generating a point cloud map—the proposed remapping process transforms the input coordinates accordingly. Intuitively, the GP acts as a coordinate corrector, modifying the inputs to the INR such that they reflect the appropriate positions in the updated map.

When querying spatial attributes from the INR, we are interested in obtaining the value at a post-optimized coordinate $X' \in \mathbb{R}^3$. However, since the INR has been trained on pre-optimized coordinates, it cannot directly provide spatial attributes at the desired post-optimized location. To address this, the GP serves as a regressor that maps the post-optimized coordinate X' to a corresponding pre-optimized coordinate $X \in \mathbb{R}^3$, which is then used as the input to the INR. As described in Section III-B, to resolve the one-to-many ambiguity inherent in this regression, we augment the input with a temporal index. Specifically, given a post-optimized spatial coordinate $\{x', y', z', t_0\}$, we construct K augmented query inputs by appending each keyframe timestamp t_k, forming inputs $\{x', y', z', t_k\}$ for $k = 0, \ldots, K-1$. This allows querying across the temporal domain associated

TABLE I
RECONSTRUCTION RESULTS IN INR METHODS (AVERAGE)

		HM3D					Replica				
		Error Pose	NN	GP(w/o T)	GP(w/ T)	GT Pose	Error Pose	NN	GP(w/o T)	GP(w/ T)	GT Pose
Acc. [cm]	NeRF [15]	65.46	35.24	**33.74**	33.77	27.86	63.99	58.15	**48.60**	**48.60**	22.70
	I-NGP+Unisurf [16], [19]	56.23	19.84	20.67	**19.72**	13.02	27.71	29.06	29.29	**25.18**	13.05
	Bayesian-NeRF [11]	81.15	45.61	**40.39**	40.71	38.26	83.37	73.34	63.39	**59.80**	32.92
Comp. [cm]	NeRF [15]	47.32	**14.14**	14.22	14.31	11.87	16.75	9.66	**9.41**	**9.41**	6.80
	I-NGP+Unisurf [16], [19]	40.52	5.04	4.98	**4.58**	4.49	10.24	3.46	3.52	**3.28**	3.83
	Bayesian-NeRF [11]	46.29	13.07	**11.06**	12.71	15.76	16.50	6.56	6.34	**6.19**	9.40
Comp Ratio. [%]	NeRF [15]	33.27	45.31	**45.88**	**45.88**	46.87	46.70	55.21	**55.50**	**55.50**	58.99
	I-NGP+Unisurf [16], [19]	45.73	67.68	68.00	**68.94**	62.83	67.85	80.91	80.69	**81.68**	72.07
	Bayesian-NeRF [11]	21.83	47.73	**55.72**	48.86	27.95	38.74	64.44	66.21	**66.90**	33.16

with keyframe acquisition. As explained in Section III-C, after performing regression over the temporal domain, we remove the temporal component and select the most appropriate pre-optimized coordinate among the candidates. This allows us to retrieve the most suitable X corresponding to the given post-optimized query X', enabling accurate spatial attributes queries from the INR.

IV. EXPERIMENTS

A. Implementation and Setup

1) Dataset: We evaluate on IMAP Replica sequences [24], [25], HM3D [21], ScanNet dataset [3]. The IMAP Replica sequences, a synthetic dataset, and the ScanNet dataset, a real-world dataset, consist of indoor environments captured from multiple viewpoints, making them well suited to evaluate reconstruction performance. Additionally, we use the HM3D synthetic dataset to assess remapping performance. To ensure a diverse evaluation, we randomly generate 38 sequences from the HM3D dataset to evaluate general performance in indoor environments.

2) Implemetation: For training and evaluation, we set the following configurations for nonlinear regression: Number of inducing points N_P=2048, σ_0=0.5, σ_f=0.1, sampling points along a ray in rendering $N_{sampling}$=24. For temporal modeling, we use the trajectory index as the temporal value. Unless otherwise specified, we remove the temporal domain using (6). In Section IV-B, for the localization task, ground-truth poses were used, and the keyframes were arbitrarily selected. In Section IV-C, both the localization and keyframe selection modules were adopted from Go-SLAM, and the INR was implemented using a hybrid of Instant-NGP and SDF representations. Implementation details for each experimental setup are described in the respective sections.

B. Evaluation of Remapping in INR-Method

In this section, we evaluate the remapping performance of our method in INR. We first train the INR with artificially introduced camera pose errors, then perform remapping using the estimated transformation function $\hat{C}(X')$, which is derived through GP regression. To simulate realistic pose errors, we introduce sudden trajectory drift at specific points during navigation, assuming a failure in odometry estimation. The timing of these errors is determined based on the highest drift points detected in the trajectory estimated using ORB SLAM3 [1]. We use ground-truth point cloud as $P_{X'}$, assuming an optimally processed loop closure and full bundle adjustment (BA). To evaluate reconstruction performance, we convert the rendered depth maps into point clouds and compute surface-based reconstruction metrics. We report the average performance across 8 sequences from the Replica dataset and 38 sequences from the HM3D dataset. Additionally, we compare GP regression approaches against neural network-based nonlinear regression approaches. Specifically, we evaluate a multi-layer neural network using 100,000 training points to estimate $\hat{C}(X')$.

The results are presented in Table I. Our proposed method consistently updates the INR-based map to closely match the ground truth.

C. Evaluation of Remapping in INR-Base SLAM

In this part, we evaluate the performance of our method when applied to a real SLAM pipeline. We compare our approach with Go-SLAM [32], which employs a post-training-based approach to update maps, and NIM-REM [30], which uses a submap-based method where each keyframe maintains a separate INR map. Go-SLAM includes a keyframe selection algorithm to optimize post-training. However, this algorithm is only effective when newly observed data is added and cannot be used as a post-processing step for remapping [32]. To ensure a fair comparison in remapping scenarios, we evaluate Go-SLAM without the keyframe selection process. NIM-REM does not have a built-in pose estimation or keyframe selection mechanism. Therefore, we use camera poses and keyframes estimated by Go-SLAM for evaluation. In addition, for mesh reconstruction, we generate query points with a resolution of 512 along each axis. For NIM-REM, voxel size is set to 0.2 when using the ScanNet dataset. Our method updates the map by applying $\hat{C}(X')$ to the mapping results obtained from Go-SLAM that don't use loop closing or global bundle adjustment (BA).

The results are presented in Tables III and II. For the ScanNet dataset, our proposed method achieves the highest performance across all average metrics except for completion. The visualization results in Fig. 3 show that maps generated using Go-SLAM and NIM-REM suffer from room merging or distortions. In contrast, our method successfully updates the map to be more

TABLE II
RECONSTRUCTION RESULTS IN INR-BASED SLAM (SCANNET DATASET)

	Scene ID	0000	0054	0059	0106	0169	0181	0233	0465	Avg.
ACC. [cm]	Go-SLAM (w/o Keyframe Selection)	5.65	15.86	13.97	36.79	11.83	**7.13**	6.50	**11.79**	13.69
	NIM-REM	8.72	**14.46**	19.27	**21.53**	13.17	12.19	11.01	13.34	14.21
	Ours	**5.55**	15.33	**11.56**	28.94	**10.75**	7.42	**5.68**	14.83	**12.51**
	Go-SLAM (w/ Keyframe Selection)	5.52	13.56	5.52	21.36	7.85	6.44	4.46	8.57	9.16
Comp. [cm]	Go-SLAM (w/o Keyframe Selection)	10.03	19.55	12.11	**8.17**	10.79	**6.79**	7.61	8.45	10.44
	NIM-REM	**6.91**	19.22	**7.76**	8.48	**6.42**	7.74	6.30	8.84	**8.96**
	Ours	9.69	**16.24**	10.15	8.42	10.18	7.21	**6.15**	**7.72**	9.47
	Go-SLAM (w/ Keyframe Selection)	8.65	15.90	53.84	8.33	145.71	7.65	5.50	8.41	31.75
Comp Ratio. [%]	Go-SLAM (w/o Keyframe Selection)	51.15	24.95	**47.75**	**46.02**	35.23	**58.40**	43.97	44.93	44.05
	NIM-REM	45.29	22.24	37.30	36.91	**53.63**	40.81	53.37	34.96	40.56
	Ours	**52.54**	**34.42**	46.86	42.81	37.87	57.18	**57.18**	**52.45**	**47.66**
	Go-SLAM (w/ Keyframe Selection)	56.15	32.98	36.43	42.46	19.40	55.04	63.90	52.33	44.84
F Score [%]	Go-SLAM (w/o Keyframe Selection)	**56.57**	26.58	**46.16**	**34.84**	33.66	**59.11**	46.61	42.54	43.26
	NIM-REM	44.23	23.01	27.90	29.22	**45.50**	37.44	53.49	33.21	36.75
	Ours	56.40	**33.40**	44.26	31.93	36.36	56.30	**57.76**	**44.74**	**45.14**
	Go-SLAM (w/ Keyframe Selection)	60.46	35.30	48.38	35.54	30.52	57.52	66.72	52.60	48.38
Space Cost [MB]	Go-SLAM	148.08	181.03	88.74	95.34	83.23	96.44	189.82	219.12	137.73
	NIM-REM	7971.6	10599.6	3241.2	3766.8	2803.2	3854.4	11300.4	13636.4	7146.7
	Ours	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**

TABLE III
RECONSTRUCTION RESULTS IN INR-BASED SLAM (REPLICA DATASET)

	Scene ID	office0	office1	office2	office3	office4	room0	room1	room2	Avg.
ACC. [cm]	Go-SLAM (w/o Keyframe Selection)	2.48	2.04	3.82	4.63	4.48	5.07	2.60	2.83	3.49
	NIM-REM	5.75	3.21	3.92	5.40	7.86	7.10	6.08	6.28	5.70
	Ours	**2.37**	**1.98**	**3.78**	**4.50**	**3.39**	**3.76**	**2.45**	**2.82**	**3.13**
	Go-SLAM (w/ Keyframe Selection)	2.24	1.80	3.52	3.79	3.64	3.15	2.39	3.03	2.95
Comp. [cm]	Go-SLAM (w/o Keyframe Selection)	2.79	2.46	3.86	4.92	4.99	4.67	2.41	6.96	4.13
	NIM-REM	2.82	**2.40**	**3.07**	**3.91**	**3.85**	4.55	3.40	5.61	**3.70**
	Ours	**2.67**	2.42	3.80	5.29	4.84	**4.00**	**2.28**	**5.26**	3.82
	Go-SLAM (w/ Keyframe Selection)	2.52	2.27	3.64	4.53	4.92	3.80	2.21	5.95	3.73
Comp Ratio. [%]	Go-SLAM (w/o Keyframe Selection)	88.88	92.65	82.94	76.34	79.98	78.34	92.85	82.82	84.35
	NIM-REM	90.41	90.70	**88.99**	**79.15**	76.48	66.22	79.05	64.65	79.46
	Ours	**90.47**	**92.89**	83.49	78.42	**83.01**	**82.09**	**94.17**	**83.11**	**85.96**
	Go-SLAM (w/ Keyframe Selection)	91.85	92.97	86.29	80.81	81.59	84.86	94.39	84.18	86.99
F Score [%]	Go-SLAM (w/o Keyframe Selection)	89.49	93.37	82.13	75.84	78.78	78.04	92.12	**85.94**	84.46
	NIM-REM	80.06	86.81	**84.55**	74.32	65.13	58.26	69.60	57.81	72.07
	Ours	**91.13**	**93.60**	82.68	**78.57**	**84.62**	**82.92**	**93.49**	85.78	**86.60**
	Go-SLAM (w/ Keyframe Selection)	91.33	93.81	85.95	81.82	82.84	86.64	93.65	86.43	87.81
Space Cost [MB]	Go-SLAM	102.53	82.03	109.38	83.00	107.42	81.05	114.25	96.68	97.042
	NIM-REM	3066.0	2452.8	3270.4	2482.0	3212.0	2423.6	3416.4	2890.8	2901.75
	Ours	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**	**48.15**

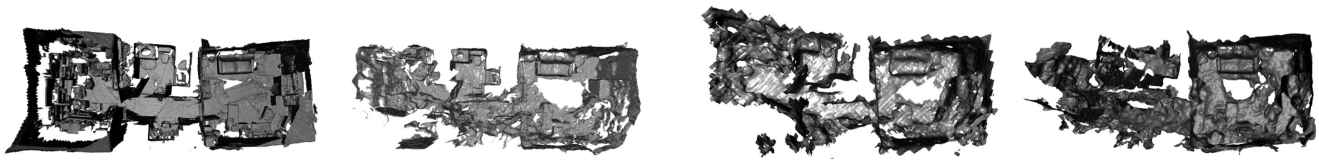
consistent with the ground truth. In particular, despite the potential inaccuracy of the INR trained in the pre-optimization domain, our approach demonstrates robust performance, suggesting that the proposed method is capable of performing effective remapping.

Additionally, in terms of memory efficiency, our approach demonstrates the lowest memory consumption across all datasets. In the case of Go-SLAM, both the INR map and the RGB-D keyframe dataset must be stored in memory. NIM-REM, on the other hand, maintains a separate INR map for each keyframe, resulting in high memory usage. In contrast, our method only requires a single INR map and two point cloud maps for updating the scene representation, leading to significantly reduced memory requirements.

D. Ablation Study

1) Temporal Domain Remove: We validate the remapping performance by comparing (5) $\mathbb{E}_T$ and (6) argmax_T as proposed in Section III-C for removing the temporal domain. The experiments were conducted under the same settings as Section IV-B, utilizing instance-NGP [16] and UniSurf [19] as the implicit neural representation methods. The results are presented in Table IV. The experimental findings confirm that, in most cases, argmax_T achieves superior performance.

2) Number of Inducing Points: We also report the impact of the number of inducing points in the our method. Specifically, we evaluate metrics on the ScanNet dataset by varying the number of inducing points used during the remapping. The



(a) GT (b) Ours (c) NIM-REM (d) Go-SLAM (w/o K.S)

Fig. 3. Remapping result on ScanNet sequence 0054.

TABLE IV
COMPARE OF ARGMAX$_T$ AND $\mathbb{E}_T$

	HM3D		Replica	
	argmax$_T$ (6)	$\mathbb{E}_T$ (5)	argmax$_T$ (6)	$\mathbb{E}_T$ (5)
ACC. [cm]↓	**19.72**	28.95	25.18	**22.51**
Comp. [cm]↓	**4.58**	14.02	**3.28**	5.23
Comp Ratio. [%]↑	**68.94**	49.24	**81.68**	74.01

TABLE V
RENDERING PERFORMANCE OF IN INR METHOD

	NN	GP(w/o T)	GP(w/ T)	GT Pose
PSNR	20.49	19.93	**20.93**	24.34
SSIM	0.588	0.562	**0.592**	0.717
Space Cost [MB]	69.34	**64.08**	**64.08**	-
Render Time [s]	**0.214***	**0.512**	18.130	0.198

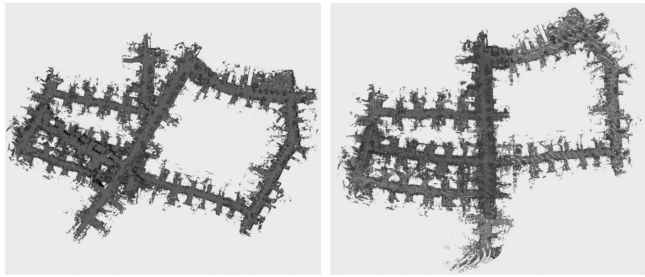
* Not include post-training time.

TABLE VI
COMPARE OF COMPUTATIONAL EFFICIENCY

	Ours	Go-SLAM (w/o Keyframe Selection)
ACC. [cm]	**15.33**	16.05
Comp. [cm]	**16.24**	19.04
Comp Ratio. [%]	**34.42**	27.02
F score [%]	**33.40**	27.46

TABLE VII
EFFICIENCY ABOUT NUMBER OF INDUCING POINTS

	Ours				**NIM-REM**
	256	512	1024	2048	—
ACC. [cm]	12.63	12.87	12.71	**12.51**	12.60
Comp. [cm]	10.16	10.32	9.93	9.47	**8.52**
Comp Ratio. [%]	43.69	43.78	44.51	**47.66**	45.55
F Score [%]	42.23	41.88	42.28	**45.14**	40.50
Space Cost [MB]	48.15	48.15	48.15	**48.15**	7146.7
Time [$\log_{10}$ s]	2.86	2.89	3.12	3.32	**-0.70**



(a) Not loop closing (b) Ours

Fig. 4. Remapping result on Kitti 05 dataset.

results are presented in Table VII. In most metrics, a larger number of inducing points leads to improved performance. These results suggest that the more accurately the nonlinear regression function $\hat{C}(X')$ approximates the target coordinate transformation $C(X')$, the better the remapping performance becomes.

3) Computational Efficiency: To evaluate the computational efficiency of our method, we additionally compare remapping performance on ScanNet scene 0054, where trajectory errors are particularly prominent. For Go-SLAM, it is difficult to explicitly define the remapping duration, so we perform post-training for the same amount of time as our method—approximately 35 minutes (2,500 iterations)—to assess its reconstruction performance. The results are presented in Tables VI and VII. While NIM-REM exhibits faster runtime compared to our method, it should be noted that remapping is a post-processing step where accuracy takes precedence over speed. Our approach achieves higher reconstruction accuracy than NIM-REM, while also requiring significantly less memory. Compared to Go-SLAM, despite consuming equal remapping time, our method yields superior reconstruction performance.

4) Rendering Performance: To evaluate the rendering performance of our method, we conducted experiments on the various versions assessed in Section IV-B. All experiments were performed on images of resolution 320 × 240 using the HM3D dataset, under the same settings described in Section IV-B. The results are reported in Table V. In terms of image rendering quality, the GP (w/ T) variant achieved the highest performance. Although the GP (w/ T) configuration results in the highest runtime, the objective of our method is not real-time rendering but asynchronous processing tailored for SLAM systems. Since our method is decoupled from the real-time estimation loop and is only executed during the post-processing stage when querying spatial information from the INR, the increased inference time does not affect the system's real-time performance in practice.

5) Large Scale Remapping: To demonstrate that our method is applicable to large-scale environments, we conducted experiments using scan data from the KITTI dataset, which represents a city-scale setting. Since most INR-based SLAM systems rely on hybrid approaches in large environments due to memory and training limitations of a single-network INR, we also adopted a hybrid approach for evaluation. Specifically, we used NeRF-LOAM [4] as the base framework, where localization was performed using ICP and Scan Context. Other components,

such as keyframe selection, followed the original NeRF-LOAM implementation. Remapping was conducted using our proposed method based on both loop closure and non-loop closure scenarios detected by Scan Context. As the INR adopts a hybrid representation, we performed remapping by transforming SDF values of the observed voxels via a Gaussian Process. The results are shown in Fig. 4. Our method successfully enables remapping in large-scale environments, demonstrating its applicability to hybrid INR-based SLAM systems in city-scale mapping scenarios.

V. Conclusion

In this letter, we propose remapping method for INR-based SLAM by formulating the problem as a function transformation task. Since the remapping of an INR-based map, represented by a neural network, can be interpreted as finding a transformation between different domains, we show that a domain transformation function between the post-optimization domain and the pre-optimization domain can be used to update the map without modifying the neural network's weights. To prevent errors caused by the one-to-many relationship between the post-optimization domain and the pre-optimization domain, we introduce a temporal domain as an additional constraint. This ensures that the estimated transformation function remains valid and accurate. We validate our approach can be applied to single-network INR-based SLAM, offering significantly more memory efficiency than existing methods while maintaining high remapping performance.

References

[1] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021.

[2] C.-M. Chung et al., "Orbeez-SLAM: A real-time monocular visual SLAM with ORB features and NeRF-realized mapping," in *Proc. 2023 IEEE Int. Conf. Robot. Automat.*, 2023, pp. 9400–9406.

[3] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2432–2443.

[4] J. Deng et al., "NeRF-LOAM: Neural implicit representation for large-scale incremental LiDAR odometry and mapping," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 8218–8227.

[5] T. Deng et al., "Compact 3D Gaussian splatting for dense visual SLAM," 2024, *arXiv:2403.11247*.

[6] N. Fioraio, J. Taylor, A. Fitzgibbon, L. Di Stefano, and S. Izadi, "Large-scale and drift-free surface reconstruction using online subvolume registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 4475–4483.

[7] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5501–5510.

[8] S. Ha, J. Yeon, and H. Yu, "RGBD GS-ICP SLAM," 2024, *arXiv:2403.12550*.

[9] O. Kähler, V. A. Prisacariu, and D. W. Murray, "Real-time large-scale dense 3D reconstruction with loop closure," in *Proc. 14th Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands, 2016, pp. 500–516.

[10] X. Kong, S. Liu, M. Taher, and A. J. Davison, "vMAP: Vectorised object mapping for neural field SLAM," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 952–961.

[11] S. Lee, K. Kang, S. Ha, and H. Yu, "Bayesian NeRF: Quantifying uncertainty with volume density for neural implicit fields," *IEEE Robot. Automat. Lett.*, vol. 10, no. 3, pp. 2144–2151, Mar. 2025.

[12] L. Liso, E. Sandström, V. Yugay, L. Van Gool, and M. R. Oswald, "Loopy-SLAM: Dense neural slam with loop closures," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 20363–20373.

[13] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 7210–7219.

[14] H. Matsuki, K. Tateno, M. Niemeyer, and F. Tombari, "NEWTON: Neural view-centric mapping for on-the-fly large-scale SLAM," *IEEE Robot. Automat. Lett.*, vol. 9, no. 4, pp. 3704–3711, Apr. 2024.

[15] 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," *Commun. ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[16] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 1–15, 2022.

[17] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[18] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[19] M. Oechsle, S. Peng, and A. Geiger, "UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 5589–5599.

[20] Y. Pan, X. Zhong, L. Wiesmann, T. Posewsky, J. Behley, and C. Stachniss, "PIN-SLAM: LiDAR SLAM using a point-based implicit neural representation for achieving global map consistency," 2024, *arXiv:2401.09101*.

[21] S. K. Ramakrishnan et al., "Habitat-matterport 3D dataset (HM3D): 1000 large-scale 3D environments for embodied AI," 2021, *arXiv:2109.08238*.

[22] E. Sandström, Y. Li, L. Van Gool, and M. R. Oswald, "Point-SLAM: Dense neural point cloud-based SLAM," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 18433–18444.

[23] M. Seeger, "Gaussian processes for machine learning," *Int. J. Neural Syst.*, vol. 14, no. 02, pp. 69–106, 2004.

[24] J. Straub et al., "The replica dataset: A digital replica of indoor spaces," 2019, *arXiv:1906.05797*.

[25] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "iMAP: Implicit mapping and positioning in real-time," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 6229–6238.

[26] Y. Tang, J. Zhang, Z. Yu, H. Wang, and K. Xu, "MIPS-fusion: Multi-implicit-submaps for scalable and robust online neural RGB-D reconstruction," *ACM Trans. Graph.*, vol. 42, no. 6, pp. 1–16, 2023.

[27] A. L. Teigen, Y. Park, A. Stahl, and R. Mester, "RGB-D mapping and tracking in a plenoxel radiance field," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2024, pp. 3342–3351.

[28] H. Wang, J. Wang, and L. Agapito, "Co-SLAM: Joint coordinate and sparse parametric encodings for neural real-time SLAM," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2023, pp. 13293–13302.

[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 *Proc. 2022 IEEE Int. Symp. Mixed Augmented Reality*, 2022, pp. 499–507.

[30] Y. Yuan and A. Nüchter, "An algorithm for the SE(3)-transformation on neural implicit maps for remapping functions," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 7763–7770, Jul. 2022.

[31] H. Zhai et al., "Vox-fusion: Voxel-based neural implicit dense tracking and mapping with multi-maps," 2024, *arXiv:2403.12536*.

[32] Y. Zhang, F. Tosi, S. Mattoccia, and M. Poggi, "GO-SLAM: Global optimization for consistent 3D instant reconstruction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 3727–3737.

[33] Z. Zhu et al., "NICER-SLAM: Neural implicit scene encoding for RGB SLAM," in *Proc. 2024 Int. Conf. 3D Vis.*, 2024, pp. 42–52.

[34] Z. Zhu et al., "NICE-SLAM: Neural implicit scalable encoding for SLAM," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12786–12796.