UNIKD: UNcertainty-filtered Incremental Knowledge Distillation for Neural Implicit Representation

Mengqi Guo, Chen Li, Hanlin Chen, and Gim Hee Lee

Department of Computer Science, National University of Singapore
{mengqi, gimhee.lee}@comp.nus.edu.sg
https://dreamguo.github.io/projects/UNIKD

Abstract. Recent neural implicit representations (NIRs) have achieved great success in the tasks of 3D reconstruction and novel view synthesis. However, they require the images of a scene from different camera views to be available for one-time training. This is expensive especially for scenarios with large-scale scenes and limited data storage. In view of this, we explore the task of incremental learning for NIRs in this work. We design a student-teacher framework to mitigate the catastrophic forgetting problem. Specifically, we iterate the process of using the student as the teacher at the end of each time step and let the teacher guide the training of the student in the next step. As a result, the student network is able to learn new information from the streaming data and retain old knowledge from the teacher network simultaneously. Although intuitive, naively applying the student-teacher pipeline does not work well in our task. Not all information from the teacher network is helpful since it is only trained with the old data. To alleviate this problem, we further introduce a random inquirer and an uncertainty-based filter to filter useful information. Our proposed method is general and thus can be adapted to different implicit representations such as neural radiance field (NeRF) and neural surface field. Extensive experimental results for both 3D reconstruction and novel view synthesis demonstrate the effectiveness of our approach compared to different baselines.

Keywords: NIRs · Incremental learning · Knowledge distillation

1 Introduction

Recent neural implicit representations (NIRs) [34, 50, 55, 59] such as NeRF and neural surface field have attracted increasing attention in the last few years because of their great success in novel view synthesis and 3D reconstruction. The key to these representations is to memorize the volume density or SDF value and view-dependent color of every spatial point in the scene with a multi-layer perceptron (MLP). Although the simple MLP networks implicitly represent the 3D scenes precisely, they require all images of a scene from different camera views to be available for a one-time training. This is expensive especially for scenarios with large-scale scenes and limited data storage. In view of this limitation, we

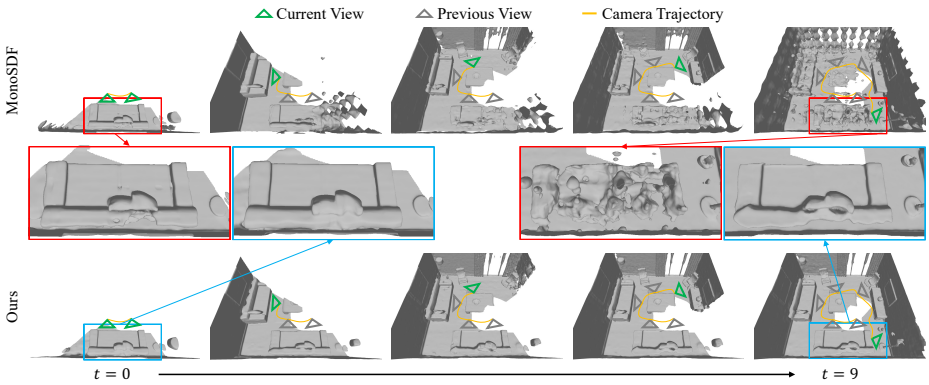

Fig. 1: Visualization of the 3D reconstruction by MonoSDF [59] and our approach under the incremental setting. MonoSDF fails to reconstruct 3D surface observed at $t = 0$ after being trained with new data because of the forgetting problem. In comparison, our approach is able to reconstruct both previously seen and new data.

explore an important task of incremental learning for NIRs in this work. In the incremental setting, the model trains on the current data without accessing any previous data, but tests on both current and previous data.

The main challenge for incremental learning is the catastrophic forgetting problem [41], where the network trained on only new incoming images drastically forgets the previously learned knowledge. This is evident from the result of MonoSDF [59] in Fig. 1, where the network is trained continuously with new incoming data captured along the yellow trajectory. The triangles represent camera views at different time steps, and green denotes the camera view at the current time step and gray ones for previous steps. We can see that the model fails to reconstruct the 3D scene observed at $t = 0$ after being trained with new data. The catastrophic forgetting problem is widely discussed in the incremental learning literature [1, 2, 27, 31, 62], and the most related work to ours is Continual Neural Mapping (CNM) [54]. CNM is the first work that introduces the incremental setting for 3D reconstruction using the Signed Distance Function (SDF). A data-replay strategy [25] is adopted in CNM to mitigate the forgetting problem. However, the data-replay still requires part of the previous training data to be stored. Additionally, CNM only shows results for SDF-based 3D reconstruction, while we aim for a general pipeline for different NIRs such as NeRF and neural surface field.

In this work, we propose a student-teacher pipeline to tackle the catastrophic forgetting problem in incremental NIRs. Specifically, we first train the model with currently available data, and then use the trained model as the teacher model with knowledge distillation strategy [17] to self-supervise the student network. We iterate this process by using the student as the teacher at the end of each time step, and letting the teacher guide the training of the student in the next step. As a result, the student is able to learn from the newly available data and preserve the old knowledge from the teacher simultaneously. Furthermore,

we also propose an alternate optimization strategy such that the new data and knowledge from the teacher can be effectively imparted to the student.

The aim of introducing the teacher network is to impart the knowledge obtained from the previous training steps to the current student network. R2L [49] uses random input views to distill information from a well-trained NeRF to a compact network. However, the teacher network is trained only with the old views in our case and thus is unable to generate useful knowledge for the unseen views. To solve this problem, we further introduce a random inquirer and an uncertainty-based filter for filtering useful knowledge. We adopt the self-supervised uncertainty modeling from [20] to predict the uncertainty of the network for each input ray. The inquirer randomly generates camera views for the uncertainty module and the filter removes the uncertain queries based on a confidence score. Intuitively, the uncertainty module would only have high confidence for the previously seen or similar data, and hence it is able to filter out the incorrect knowledge generated from the random query.

We evaluate the effectiveness of our proposed approach on two popular NIRs, NeRF [34] and MonoSDF [59]. Extensive experimental results on both 3D reconstruction and novel view synthesis show that our approach mitigates the catastrophic forgetting problems effectively without storing previous training data. Specifically, our method significantly improves 39.6% and 61.3% over MonoSDF in terms of F1 on the large-scale datasets ICL-NUIM [16] and Replica [45]. Moreover, our approach outperforms NeRF by 36.3% and 63.9% in terms of PSNR on the object-scale 360Capture [34] and large-scale ScanNet [11] datasets, respectively. Our contributions are summarized as follows:

- We explore the incremental learning task for general NIRs.
- We propose a student-teacher pipeline to mitigate catastrophic forgetting in incremental learning.
- We design the uncertainty filter and the random inquirer to generate and select useful information for the student network.
- We significantly outperform baselines by a large margin for both 3D reconstruction and novel view synthesis.

2 Related Work

Neural Implicit Representation. The neural implicit representations (NIRs) [10, 26, 28, 52, 56] have shown remarkable potential in various computer vision tasks, such as novel view synthesis [5, 6, 34] and 3D reconstruction [3, 47, 59]. The pioneering work NeRF [34] introduced a simple yet effective MLP network to implicitly capture the 3D scene and propose a differentiable rendering method for generating novel view images. Many follow-up works have attempted to enhance NeRF to fully exploit their potential, such as real-time rendering [40, 57], faster training [8, 15, 29, 35], sparse view [42, 58], generalizable model [9, 51], lightning changing [32, 33], better representation [4, 60], *etc*. Some recent works [50, 55] proposed neural implicit surfaces and incorporated the signed distance function (SDF) into NeRF for smooth and accurate surface reconstruction. MonoSDF [59]

further leveraged monocular depth and normal priors to achieve more detailed reconstruction for larger 3D scenes. Despite the great success, existing NIRs suffer from the catastrophic forgetting problem when continuously learning from streaming data. In view of this problem, we focus on the under-explored and yet important incremental settings for NIRs in this paper.

Incremental Learning. Incremental learning is a classical machine learning problem where only partial data is available for training at each step. Existing methods typically fall into three categories [12]: data replay [7, 22, 30, 39, 43, 44], parameter regularization [1,21,27,38], and parameter isolation [2,14,31,53]. In this paper, we revisit some classical incremental approaches to build strong baselines. Specifically, PTAM [22] introduced keyframes replaying (KR) to avoid forgetting, MAS [1] measured the parameter importance for each task and regularized the important parameters, PackNet [31] assigned parameters subsets explicitly to different tasks by constituting binary masks, POD [13] and AFC [19] employed knowledge distillation on the intermediate network features. CNM [54] is the first work on incremental learning for neural surface field, which reconstructs 3D surface from streaming depth inputs using a reply-based method [25]. CLNeRF [63] applies the data replay on NeRF for novel view synthesis. However, their method still requires access to some of the previous data to prevent forgetting and they mainly focus on the SDF or NeRF representation. In comparison, our approach can be adapted to different NIRs without access to any previous data.

NIR-SLAM and Large-scale NeRF. Traditional simultaneous localization and mapping (SLAM) [23, 36, 37] is able to reconstruct 3D scenes with streaming data, which share similar spirits with our incremental setting. Recently, some works such as iMAP [46] and NICE-SLAM [64] have adopted neural implicit representation as the scene representation in SLAM and achieved promising performance. These approaches mitigate the forgetting problem by storing keyframes, as done in traditional SLAM. The drawback of using keyframes is that memory usage will increase accordingly as the scene gets larger. On the other hand, Recent NeRFusion [61] and Block-NeRF [48] handle large-scale scenes by incrementally reconstructing a global scene representation by fusing local voxel representations. However, the voxel-based representation requires substantial storage and the fusion stage requires all the images of a scene. Moreover, all those approaches work on one specific neural implicit representation while we propose a general approach, which is also memory-efficient.

3 Preliminaries

3.1 Neural Implicit Representations

In this section, we present a unified formulation for the currently dominant NIRs including NeRF [34] and neural surface field [55, 59]. The principal idea is to use a simple neural network such as MLPs to memorize the color $\mathbf{c} = (r, g, b)$, volume density σ for each location $\mathbf{x} = (x, y, z)$ and camera view direction $\mathbf{d} = (\theta, \phi)$ in a 3D scene. While existing neural surface field predicts SDF value

which is then converted to density, we use $(\mathbf{c}, \sigma) = F(\mathbf{x}, \mathbf{d})$ to represent both NeRF and neural surface field networks in this paper for simplicity. The per-pixel RGB $c(\mathbf{r})$ value of an image can be rendered with N 3D points taken along the ray $\mathbf{r}$ from the camera center to the pixel as:

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

where $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$ is the accumulated transmittance along the ray $\mathbf{r}$ from camera center to i^{th} 3D point, δ_i indicates the distance between i^{th} sample and $(i+1)^{\text{th}}$ sample. Since the whole pipeline is differentiable, the rendering output can be directly supervised by the RGB image:

$$\mathcal{L} = \mathcal{L}_{rgb} + [\mathcal{L}_{eik}] + [\mathcal{L}_{prior}], \quad \text{where} \quad \mathcal{L}_{rgb} = \sum_{\mathbf{r} \in R} \Big(||c^*(\mathbf{r}) - c(\mathbf{r})||_2^2 \Big) \tag{2}$$

represents the rendering loss. $c^*(\mathbf{r})$ denotes the ground truth color and R represents a group of rays from one or more camera views. $\mathcal{L}_{eik}$ is the SDF regularizer for neural surface field [50, 55, 59], and $\mathcal{L}_{prior}$ represents the geometry prior term [59]. Generally, the prior term consists of depth and normal priors. More details about the depth and normal priors are provided in supplementary material. Note that $\mathcal{L}_{rgb}$ is the fundamental term for both NeRF and neural surface field. $\mathcal{L}_{eik}$ and $\mathcal{L}_{prior}$ are prior terms used in [59], which we denote with [.].

3.2 Catastrophic Forgetting of NIRs

Despite the impressive performance, existing NIRs require training on the entire set of images covering all views. In practice, this may not be feasible in the scenario of limited data storage or streaming data, which may require the network to be trained on new data without revisiting old ones. This may lead to the catastrophic forgetting of existing NIRs, where the network quickly forgets previously learned knowledge while acquiring new knowledge. To address this issue, we explore the task of incremental learning for NIRs with the goal of mitigating the catastrophic forgetting problem.

4 Our Method

In this section, we first introduce the incremental setting for NIRs. We then represent our proposed student-teacher pipeline with an uncertainty based filter and an alternative optimization strategy.

4.1 Problem Definition

We consider a common scenario in the robotics or vision community, where $T+1$ groups of data $\mathcal{D} = \{\mathcal{D}^0, \mathcal{D}^1, \cdots, \mathcal{D}^T\}$ come in sequentially. The data for each time step t consists of N pairs of images I^t and the corresponding

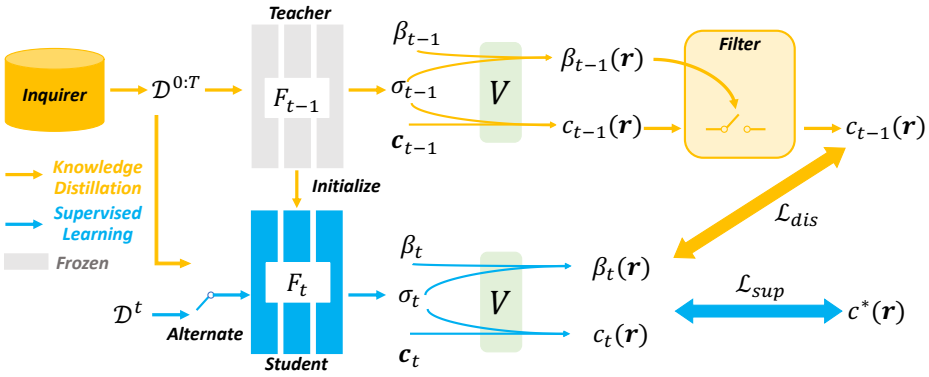

Fig. 2: The overall framework of our proposed student-teacher pipeline. At time step t, The student network learns simultaneously from the currently available data $\mathcal{D}^t$ and the previously learned knowledge from the teacher network. The input of the teacher network is generated with the random inquirer. The output is filtered with an uncertainty based filter for useful information selection. V denotes the differentiable volume renderer.

camera poses P^t, *i.e.*, $\mathcal{D}^t = \{d_0^t, d_1^t, \cdots, d_N^t\}$, where $d_n^t = (I_n^t, P_n^t)$. Generally, the camera poses do not overlap between different time steps. The task of incremental learning for NIRs aims to continually learn from the newly arriving data $\mathcal{D}^t$ and also preserve the knowledge from previously seen data $\mathcal{D}^{0:t-1}$.

4.2 Overview

The overall framework of our approach is illustrated in Fig. 2. It comprises a student and a teacher network, both sharing the same architecture with a density branch, a color branch, and an uncertainty branch. At each time step t, the student learns simultaneously from the currently available data $\mathcal{D}^t$ and the previously learned knowledge from the teacher network. The student model trained in this step is then utilized as the teacher model in the next step and imparts its acquired knowledge to the next student. We iterate this process throughout the training process. To explore the knowledge space of the teacher network, we design a random inquirer that generates camera views for the teacher network. However, the teacher network can generate erroneous information for randomly generated views because it only trains on previously seen data $\mathcal{D}^{0:t-1}$. We further design an uncertainty branch to predict uncertainty scores and select only reliable information from the teacher network.

4.3 Supervised Learning

At each time step t, we utilize the available data $\mathcal{D}^t$ to train the student network directly with rendering loss. Except for the density and color, we also predict the uncertainty value for each input to measure the confidence.

Uncertainty Modeling. We adopt the self-supervised uncertainty formulation from [20] to model the uncertainty of the network for each input ray. This formulation has also been used in previous NeRF-W [32] to distinguish the static

and transient scenes. With a different objective, we aim to indicate the confidence of the network on the current input. Specifically, we build an additional branch that shares the same input as the color branch to predict the uncertainty $(\mathbf{c}, \sigma, \beta) = F(\mathbf{x}, \mathbf{d})$. We adopt Softplus as the activation function on the uncertainty for stable training. Finally, we compute the pixel-wise uncertainty from each sample point using the same volume rendering technique as the color:

$$\beta(\mathbf{r}) = \sum_{n=1}^{N} T_i\big(1 - exp(-\sigma_j \delta_j)\big)\hat{\beta}_i + \beta_{min}, \quad \text{where} \quad \hat{\beta}_i = \log\left(1 + e^{\beta_i - 1}\right), \quad (3)$$

T_i represents the accumulated transmittance expressed by Eqn. (1), and β_{min} denotes a hyper-parameter that ensures the minimum uncertainty following [32].

Supervised Optimization. The objective function of supervised training is:

$$\mathcal{L}_{sup} = \sum_{\mathbf{r} \in R} \Big(\frac{||c^*(\mathbf{r}) - c_t(\mathbf{r})||_2^2}{2} + \frac{||c^*(\mathbf{r}) - c_t(\mathbf{r})||_2^2}{2 * \beta_t(\mathbf{r})^2} + \log\big(\beta_t(\mathbf{r})\big) + \eta \Big), \quad (4)$$

where η denotes the margin of the uncertainty regular term to avoid negative values. Note that we only need to supervise the color and the uncertainty is implicitly learned from the loss function. Intuitively, on one hand, the network needs to predict a high uncertainty value when the color prediction is inaccurate in order to minimize the loss function. On the other hand, the regularization term $\log(\beta_t(\mathbf{r}))$ prevents the network from predicting infinite uncertainty.

4.4 Knowledge Distillation

The network trained with only the currently available data at each time step tends to forget the previously learned knowledge, referred as the catastrophic forgetting problem. To prevent this, we further introduce a teacher network to impart the previously learned knowledge to the current model.

Student-teacher Modeling. At each time step t, the student network F_t concurrently learns from the teacher network F_{t-1} and the new coming data $\mathcal{D}^t$. The student network is then used as the teacher network after each step. We iterate the process of using the student as the teacher at the end of each step, and let the teacher guide the student in the next step. As a result, the student network can learn both new knowledge from $\mathcal{D}^t$ and old knowledge from F_{t-1} and hence mitigate the forgetting problem. To facilitate the knowledge imparting from the teacher to the student network, we introduce a knowledge distillation loss [17]. Moreover, we initialize the parameters of the student network with that of the teacher network. This initialization strategy also helps to mitigate the forgetting problem since the parameters are learned from previously seen data.

Random Inquirer. The role of the teacher network is to impart old knowledge, which means the inputs should be the same as the training data from the previous time steps. However, the previous training data is not accessible under the incremental setting. To solve this problem, we design a constrained

random inquirer to generate inputs for the teacher network. For scenes where the camera moves along a trajectory, we randomly generate camera views in the range of each degree of freedom of the camera matrix from the previous data. Specifically, we store the range of six values in the camera matrix, *i.e.*, $r = (x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}, \alpha_{min}, \alpha_{max}, \beta_{min}, \beta_{max}, \gamma_{min}, \gamma_{max})$, at each time step t. We first randomly choose a time step k from $0 : t-1$ at time step t, and then randomly generate a group of the six values in the range of r_k to compute the camera matrix.

Uncertainty-based Filter. The role of the random inquirer is to explore the knowledge space of the teacher network such that we can extract useful information, *i.e.* the knowledge from previous time steps. However, the teacher network might output incorrect knowledge since it has been trained only on $D^{0:t-1}$, while the input generated from the random inquirer covers the whole dataset. To overcome this issue, we utilize the uncertainty module as described in Sec. 4.3 for useful knowledge selection. Specifically, we take the average of the output uncertainty value over rays from one camera view, and only select camera views with an uncertainty smaller than a threshold β^{thr}, *i.e.*:

$$R^* \leftarrow R^* \cup R_v, \quad \text{if} \quad \frac{1}{\mathcal{N}_{R_v}} \sum_{\mathbf{r} \in R_v} \big(\beta_t(\mathbf{r})\big) < \beta^{thr}. \tag{5}$$

R_v is the rays for camera view v generated from the random inquirer, $\mathcal{N}_{R_v}$ is the number of rays samples, and R^* is the collection of data samples we use for knowledge distillation. Intuitively, the network tends to output lower uncertainty for previously seen data compared to unseen ones, and thus we can use R^* to approximate the unavailable data from the previous training step. Note that the selection is conducted in terms of camera views, *i.e.* average over all ray samples instead of a single ray. This is empirically shown to better distinguish the seen and unseen images.

Distilled Optimization. Finally, we use the teacher network to guide the student network via a knowledge distillation loss:

$$\mathcal{L}_{dis} = \sum_{\mathbf{r} \in R^*} \Big(\frac{||c_{t-1}(\mathbf{r}) - c_t(\mathbf{r})||_2^2}{2} + \frac{||c_{t-1}(\mathbf{r}) - c_t(\mathbf{r})||_2^2}{2 * \beta_t(\mathbf{r})^2} + \log\big(\beta_t(\mathbf{r})\big) + \eta \Big), \tag{6}$$

where $c_{t-1}(\mathbf{r})$ and $c_t(\mathbf{r})$ represent the output color of the teacher and student model, respectively. R^* denotes useful data selection from Eqn. (5). With knowledge distillation, the student network is able to preserve the previously learned knowledge throughout the whole training process.

4.5 Iterative Optimization

We propose an iterative optimization mechanism to enable the student network to learn simultaneously from the current data $\mathcal{D}^t$ and knowledge from the teacher

Table 1: Comparison with baselines on the ICL-NUIM. (Best and second best results are highlighted in bold and underlined, respectively.)

	MonoSDF [59]	MonoSDF* [59]	CNM [54]	MAS [1]	PackNet [31]	KR [22]	POD [13]	AFC [19]	Ours
F1↑	64.71	89.68	69.93	66.40	76.03	<u>86.78</u>	84.52	86.17	**90.32**
CD↓	5.94	2.64	5.42	5.84	4.39	<u>3.02</u>	3.32	3.10	**2.60**

Table 2: Comparison with baselines on the Replica. (Best and second best results are highlighted in bold and underlined, respectively.)

	MonoSDF [59]	MonoSDF* [59]	iMAP [46]	NICE-SLAM [64]	CNM [54]	MAS [1]	PackNet [31]	KR [22]	POD [13]	AFC [19]	Ours
F1↑	53.63	86.18	-	-	67.52	58.75	61.99	<u>79.67</u>	72.59	74.96	**86.52**
CD↓	8.58	2.94	4.99	**2.93**	6.54	7.98	7.49	3.99	4.56	4.20	<u>3.11</u>

network. Specifically, we alternatively optimize the supervised loss Eqn. (4) and the knowledge distillation loss Eqn. (6), *i.e.*:

$$\mathcal{L} = \begin{cases} \mathcal{L}_{sup} + [\mathcal{L}_{eik}] + [\mathcal{L}_{prior}], & \text{if } i \text{ is even} \\ \mathcal{L}_{dis} + [\mathcal{L}_{eik}] + [\mathcal{L}_{prior\_dis}], & \text{otherwise} \end{cases}, \tag{7}$$

where i denotes the iteration number, $\mathcal{L}_{prior\_dis}$ represents that the prior comes from the teacher network instead of the pre-trained model as in the $\mathcal{L}_{prior}$.

5 Experiments

5.1 Experimental Settings

We apply our approach to currently dominant implicit representations NeRF [34] and neural surface field [59], and show results for both novel view synthesis and 3D reconstruction.

Dataset. The previous NIRs conducted experiments on different types of scenes, thus we consider the following datasets to cover: a) Object-scale scenes, *i.e.* 360Capture [34]; b) Large-scale synthetic scenes, *i.e.* ICL-NUIM [16] and Replica [45]; c) Large-scale real-world scenes, *i.e.* ScanNet [11]. For incremental setting, we divide the images of each scene and the corresponding camera poses into 10 time steps $\mathcal{D} = \{\mathcal{D}^0, \mathcal{D}^1, \cdots, \mathcal{D}^9\}$.

Baselines. We compare against a) the main baselines NeRF [34] and MonoSDF [59] under both incremental and batch training settings; b) SLAM-based NIRs iMAP [46] and NICE-SLAM [64]; c) Reply-based NIRs CNM [54] and KR [22] (replay 10 keyframes following iMAP [46]), ; d) Four representative incremental learning baselines MAS [1], PackNet [31], POD [13], AFC [19].

Evaluation Metrics. For 3D reconstruction, we follow MonoSDF [59] to report Chamfer Distance (CD) and F1 score with a threshold of 5cm. For novel view synthesis, we follow NeRF [34] to report PSNR, SSIM, and LPIPS.

Backbone for 3D Reconstruction. In the 3D reconstruction experiments, we adopt MonoSDF [59] as our backbone. Our network F consists of an SDF network, a color network, and an uncertainty network. The density network consists of eight fully connected (FC) layers with 256-channel, the color and uncertainty networks are both two FC layers with 256-channel. We train our network with a batch size of 1024 rays, where each ray samples 96 points.

Table 3: Quantitative comparison with baselines on the 360Capture dataset. We show results for each step test datasets $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ and the average performance over $\mathcal{D}^{0:9}$, respectively. All models are incrementally trained on the 10-step training datasets.

Method	Test Dataset (PSNR↑ / SSIM↑ / LPIPS↓)				
	$\mathcal{D}^0$	$\mathcal{D}^3$	$\mathcal{D}^6$	$\mathcal{D}^9$	Average on $\mathcal{D}^{0:9}$
NeRF [34]	16.10 / 0.468 / 0.276	14.97 / 0.407 / 0.357	15.50 / 0.512 / 0.323	18.18 / 0.636 / 0.217	15.56 / 0.468 / 0.317
NeRF* [34]	24.27 / 0.781 / 0.177	23.81 / 0.767 / 0.184	22.86 / 0.738 / 0.188	20.75 / 0.684 / 0.227	22.81 / 0.741 / 0.198
MAS [1]	18.15 / 0.552 / 0.273	16.42 / 0.500 / 0.332	16.95 / 0.525 / 0.374	17.35 / 0.551 / 0.340	17.02 / 0.513 / 0.341
PackNet [31]	17.21 / 0.497 / 0.349	17.09 / 0.491 / 0.378	17.01 / 0.489 / 0.390	14.45 / 0.417 / 0.447	16.52 / 0.474 / 0.388
KR [22]	18.76 / 0.629 / 0.225	19.50 / 0.625 / 0.238	**21.14** / **0.682** / 0.213	19.46 / 0.657 / 0.328	19.68 / 0.642 / 0.240
POD [13]	18.54 / 0.585 / 0.269	17.29 / 0.502 / 0.317	18.17 / 0.542 / 0.270	18.31 / 0.573 / 0.291	17.79 / 0.561 / 0.291
AFC [19]	19.01 / 0.603 / 0.258	19.19 / 0.621 / 0.252	19.37 / 0.639 / 0.240	19.45 / 0.643 / 0.233	19.30 / 0.632 / 0.241
Ours	**22.48** / **0.701** / **0.188**	**22.16** / **0.694** / **0.208**	21.07 / **0.682** / **0.173**	**19.82** / **0.658** / **0.221**	**21.21** / **0.672** / **0.211**

Table 4: Quantitative comparison with baselines on the ScanNet dataset. We show results for each step test datasets $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ and the average performance over $\mathcal{D}^{0:9}$, respectively. All models are incrementally trained on the 10-step training datasets.

Method	Test Dataset (PSNR↑ / SSIM↑ / LPIPS↓)				
	$\mathcal{D}^0$	$\mathcal{D}^3$	$\mathcal{D}^6$	$\mathcal{D}^9$	Average on $\mathcal{D}^{0:9}$
NeRF [34]	12.61 / 0.580 / 0.396	11.59 / 0.505 / 0.571	15.65 / 0.578 / 0.470	25.95 / 0.876 / 0.145	13.78 / 0.576 / 0.460
NeRF* [34]	22.55 / 0.824 / 0.244	22.23 / 0.850 / 0.231	24.53 / 0.860 / 0.211	25.51 / 0.881 / 0.150	23.55 / 0.852 / 0.212
iMAP [46]	10.85 / 0.581 / 0.472	19.53 / 0.808 / 0.300	19.19 / 0.789 / 0.326	20.88 / 0.794 / 0.328	18.80 / 0.761 / 0.340
MAS [1]	16.40 / 0.669 / 0.366	12.15 / 0.595 / 0.512	15.76 / 0.684 / 0.455	22.22 / 0.812 / 0.303	15.76 / 0.673 / 0.402
PackNet [31]	12.74 / 0.535 / 0.488	11.99 / 0.604 / 0.456	13.60 / 0.630 / 0.429	11.49 / 0.550 / 0.457	12.75 / 0.592 / 0.439
KR [22]	14.03 / 0.598 / 0.392	12.72 / 0.583 / 0.525	16.69 / 0.691 / 0.396	22.02 / 0.801 / 0.271	16.04 / 0.663 / 0.397
POD [13]	15.23 / 0.639 / 0.399	12.83 / 0.597 / 0.472	15.27 / 0.660 / 0.420	22.10 / 0.812 / 0.268	15.35 / 0.661 / 0.407
AFC [19]	16.28 / 0.657 / 0.412	14.92 / 0.640 / 0.437	16.32 / 0.663 / 0.408	22.58 / 0.843 / 0.254	16.38 / 0.684 / 0.382
Ours	**21.74** / **0.812** / **0.224**	**20.21** / **0.825** / **0.296**	**23.90** / **0.851** / **0.224**	**25.30** / **0.876** / **0.162**	**22.59** / **0.841** / **0.230**

Backbone for Novel View Synthesis. In the novel view synthesis experiments, we adopt NeRF [34] as our backbone. Specifically, our network F consists of a density network, a color network, and an uncertainty network. The density network consists of eight fully connected (FC) layers with 256-channel, the color and uncertainty networks are both one FC layer with 128-channel. We train our network with a batch size of 1024 rays, 64 points per ray for the coarse network and $64 + 128$ points per ray for the fine network. We adopt AlexNet [24] to compute LPIPS.

5.2 Results on 3D Reconstruction

We first evaluate our approach for the neural surface field based representation. We adopt MonoSDF as the backbone and show 3D reconstruction results on the large-scale datasets ICL-NUIM and Replica.

ICL-NUIM. We show the results of our approach and the baselines on the ICL-NUIM dataset in Tab. 1. We can see that the performance of MonoSDF drops significantly when trained under incremental setting compared to the results under batch training (MonoSDF*). Note that batch training means that all data are available for one-time training, which is the upper bound of incremental training. The incremental baselines only achieve minor improvement compared with MonoSDF with the exception of KR. However, KR requires more memory as shown in Tab. 7. In comparison, our approach improves over the MonoSDF

baseline by 39.6% in F1 and is even slightly better than batch training while keeping a low memory usage.

Replica. We further show results on the Replica dataset in Tab. 2. We can see that MonoSDF and incremental baselines (MAS, PackNet, and KR) suffer from the forgetting problem, which can be evident from the performance drop compared to MonoSDF*. Our approach improves MonoSDF by 61.3% for F1 and also achieves similar performance with batch training. Compared to the SLAM-based baselines, our model outperforms iMAP by a large margin and is only slightly worse than NICE-SLAM. The better performance of NICE-SLAM can be attributed to the use of more powerful representations compared to our backbone MonoSDF. Moreover, the memory usage of SLAM-based baselines is larger than ours and increases as the scene gets larger, as discussed in Sec. 5.5. Note that we compare with different baselines on the two datasets since CNM and SLAM-based NIRs (iMAP, NICE-SLAM) show results on the ICL-NUIM and Replica, respectively.

Qualitative Results. We further show the qualitative comparison of the ICL-NUIM and Replica datasets in Fig. 3. We can see that the MonoSDF baseline learns well for the current scene (scenes outside the red box) but fails on previously seen scenes (highlighted with red boxes) completely. In comparison, our approach is able to reconstruct detailed geometries for both current and previous scenes, achieving similar quality with "MonoSDF*" and ground truth.

5.3 Results on Novel View Synthesis

We then evaluate our approach for the novel view synthesis task. We adopt NeRF as our backbone and show results for the 360Capture and ScanNet datasets. The four columns $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ denote the testing dataset at the corresponding time step. The results are obtained from the final model, which has been trained incrementally on all views $\mathcal{D}^{0:9}$. Thus $\mathcal{D}^9$ is the test dataset of current views and $\mathcal{D}^{0:8}$ are the test datasets of previous views.

360Capture. We show the results of our approach and baselines on the 360Capture dataset in Tab. 3. We can see that the NeRF baseline suffers from the forgetting problem, leading to a large performance drop on the testing data at previous steps $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6$. The incremental baselines (MAS, PackNet, and KR) mitigate the forgetting problem to some extent with limited improvement. In comparison, our approach is able to perform consistently well on previous testing data with improvements over the NeRF baseline by 39.6%, 48.0%, and 35.9% for $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6$, and 36.3% for $\mathcal{D}^{0:9}$ in PSNR.

ScanNet. We further show results for the more challenging large-scale ScanNet dataset in Tab. 4. We can see that both the NeRF baseline and incremental baselines perform poorly on the testing datasets of previous time steps because of the severe forgetting problem caused by little overlap between images in this dataset. Benefiting from the student-teacher pipeline, our method still achieves promising results with significant improvements over incremental baselines (MAS, PackNet, and KR) and SLAM-based baseline iMAP. Comparable performance

Table 5: Ablation studies of proposed modules on ICL-NUIM and 360Capture datasets.

Method	ICL-NUIM		360Capture		
	F1↑	CD↓	PSNR↑	SSIM↑	LPIPS↓
w/o s-t	64.71	5.94	15.56	0.452	0.405
w/o filter	81.63	3.82	19.39	0.639	0.248
Ours	**90.32**	**2.60**	**21.21**	**0.672**	**0.211**
w/ $P^{0:t}$	89.50	2.65	21.38	0.673	0.207

Table 6: Ablation studies of generalizability to different models on ScanNet dataset.

Method	Baseline	Ours
NeRF [34]	13.78 / 0.576 / 0.460	**22.59** / **0.841** / **0.230**
Tri-MipRF [18]	20.34 / 0.712 / 0.339	**27.02** / **0.854** / **0.190**
ZipNeRF [6]	21.78 / 0.739 / 0.302	**27.85** / **0.876** / **0.172**

is also achieved with batch training NeRF*, which further demonstrates the effectiveness of our approach.

Qualitative Results. We further show qualitative comparison on the ScanNet scene 101 and 360Capture scene *Vasdeck* in Fig. 4. As we can see, the original NeRF suffers from the catastrophic forgetting problem and outputs images on previous time steps $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6$ with severe artifacts including noise and blur. In comparison, our approach generates realistic images with comparable quality to the batch training. This suggests the effectiveness of our proposed approach in mitigating the forgetting problem.

5.4 Ablation Study

Proposed Module. As shown in Tab. 5, we conduct ablation studies for both neural surface field and NeRF representations on the ICL-NUIM and 360Capture datasets, respectively. We verify the contribution of our proposed components student-teacher modeling (s-t) and uncertainty-based filter (filter) by removing each component at a time. As can be seen that the performance drops when each component is removed. Specifically, our approach becomes the original NeRF or MonoSDF when the student-teacher modeling is removed, and the model fails completely. Without the uncertainty-based filter, the performance drops significantly for incremental 3D reconstruction task on the ICL-NUIM dataset. This is because the output of the teacher network is not necessarily correct for any input generated from the random inquirer, and the incorrect information can mislead the student during the knowledge distillation. Additionally, we also show results when the camera poses of previous time steps are stored, denoted as w/ $P^{0:t}$. We can see that we achieve comparable performance with this scenario although we do not store any data from the previous time step.

Generalize to different NeRF models. We also apply our proposed approach to the most recent NeRF models, including Tri-MipRF [18] and ZipNeRF [6], to show the generalization over different backbones. As shown in Tab. 6, we consistently outperform different backbones on ScanNet.

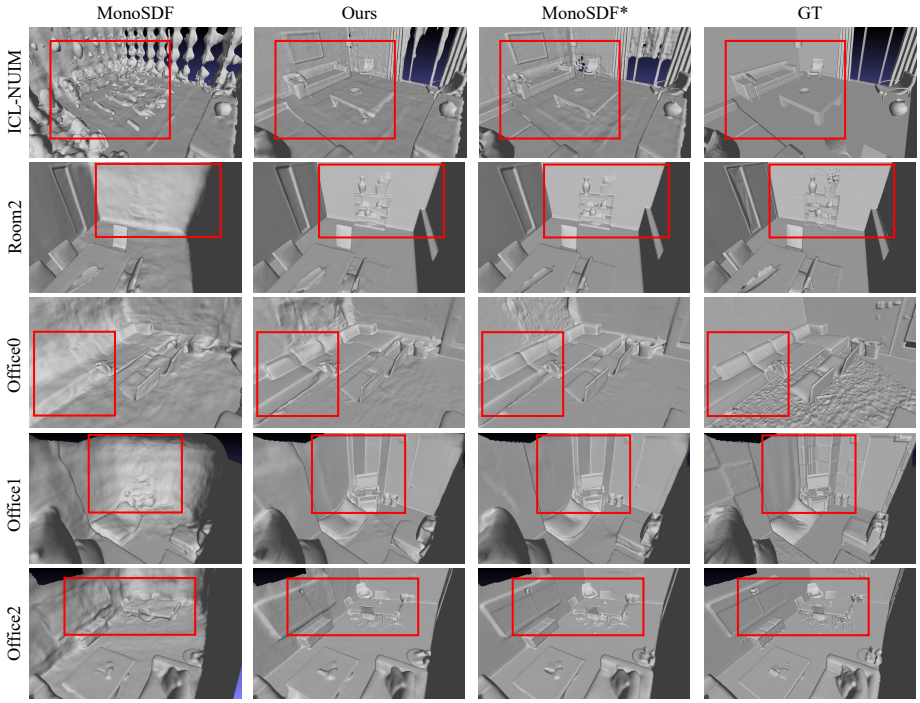

Fig. 3: Qualitative comparison on the ICL-NUIM and Replica datasets. Both 'MonoSDF' and 'Ours' models are incrementally trained on the 10-step training datasets. The red boxes are the previously learned views.

Table 7: The memory usage analysis for all baselines and Ours on Replica dataset.

	MonoSDF [59]	MonoSDF* [59]	iMAP [46]	NICE-SLAM [64]	CNM [54]	MAS [1]	PackNet [31]	KR [22]	POD [13]	AFC [19]	Ours
Memory(MB)	0	300	30	30	**2**	16	11	30	33	19	3

5.5 Memory Analysis

As Tab. 7 shown, we further provide the memory usage analysis for all methods presented in Tab. 1 and Tab. 2. Among the baselines, iMAP, NICE-SLAM, CNM, KR, and POD require additional space to store keyframes, MAS, PackNet, and AFC need memory to save the importance score or masks for network parameters. POD, AFC, and ours use a distillation strategy that stores the teacher model. The batch training method MonoSDF* requires much more memory than other baselines for storing a full batch of data. We can conclude that our model achieves better performance with smaller memory usage based on the memory usage (Tab. 7) and the performance comparison (Tab. 2).

Memory replay technique plays a crucial role in mitigating the forgetting problem in existing SLAM-based methods [46, 64] and incremental NIRs [54, 63]. These approaches require additional memory allocation to store the previously seen data. To understand the relationship between memory utilization and performance, we conducted experiments using the strongest baselines NICE-SLAM with varying memory capacities on the large-scale scene Apartment [64], as shown in Fig. 5. We can see that performance improves as memory usage

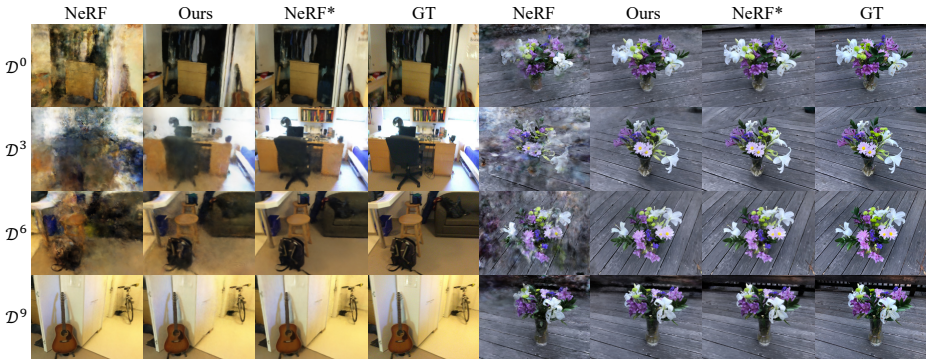

Fig. 4: Qualitative comparison on the ScanNet and 360Capture datasets. 'NeRF' and 'Ours' models are incrementally trained on the 10-step training datasets. $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6$ denote the results of previous views from each time step test datasets and $\mathcal{D}^9$ is the results of current views from the latest test dataset.

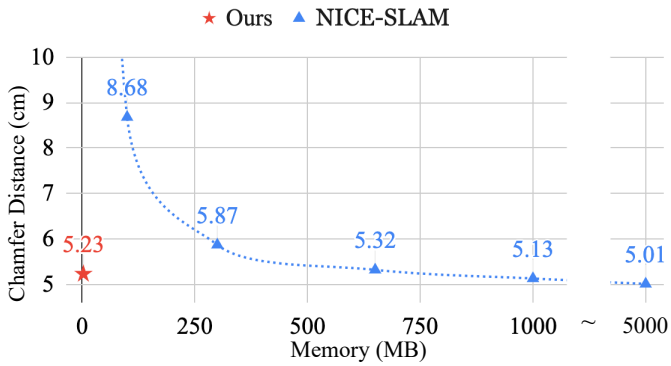

Fig. 5: Comparison of Ours and NICE-SLAM with different memory usage.

increases from 0 to 5,000MB (equivalent to the batch training method MonoSDF*). In comparison, our approach performs similarly to MonoSDF* (5.12cm) while maintaining minimal memory consumption (around 3MB).

6 Conclusion

In this paper, we explore the task of incremental learning for Neural Implicit Representations (NIRs). We propose a student-teacher pipeline for mitigating the catastrophic forgetting problem. To improve the effectiveness of the data provided by the teacher network, we further design a random inquirer and an uncertainty-based filter for useful knowledge distillation. Supervised learning and knowledge distillation are iteratively utilized for the combination of preserving old information and learning current new data. Experiments on both 3D reconstruction and novel view synthesis demonstrate that our model achieves great improvement compared to baselines under the incremental setting.

Acknowledgement. This research work is supported by the Agency for Science, Technology and Research (A*STAR) under its MTC Programmatic Funds (Grant No. M23L7b0021).

References

1. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: Learning what (not) to forget. In: ECCV (2018)
2. Aljundi, R., Chakravarty, P., Tuytelaars, T.: Expert gate: Lifelong learning with a network of experts. In: CVPR (2017)
3. Azinović, D., Martin-Brualla, R., Goldman, D.B., Nießner, M., Thies, J.: Neural rgb-d surface reconstruction. In: CVPR (2022)
4. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: ICCV (2021)
5. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: CVPR (2022)
6. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Zip-nerf: Anti-aliased grid-based neural radiance fields. CVPR (2023)
7. Chaudhry, A., Ranzato, M., Rohrbach, M., Elhoseiny, M.: Efficient lifelong learning with a-gem. arXiv preprint arXiv:1812.00420 (2018)
8. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. ECCV (2022)
9. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In: ICCV (2021)
10. Chen, H., Li, C., Guo, M., Yan, Z., Lee, G.H.: Gnesf: Generalizable neural semantic fields. NeurIPS (2023)
11. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: CVPR (2017)
12. De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T.: A continual learning survey: Defying forgetting in classification tasks. TPAMI (2021)
13. Douillard, A., Cord, M., Ollion, C., Robert, T., Valle, E.: Podnet: Pooled outputs distillation for small-tasks incremental learning. In: ECCV (2020)
14. Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A.A., Pritzel, A., Wierstra, D.: Pathnet: Evolution channels gradient descent in super neural networks. arXiv preprint arXiv:1701.08734 (2017)
15. Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. In: CVPR (2022)
16. Handa, A., Whelan, T., McDonald, J., Davison, A.: A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In: ICRA (2014)
17. Hinton, G., Vinyals, O., Dean, J., et al.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
18. Hu, W., Wang, Y., Ma, L., Yang, B., Gao, L., Liu, X., Ma, Y.: Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In: CVPR (2023)
19. Kang, M., Park, J., Han, B.: Class-incremental learning by knowledge distillation with adaptive feature consolidation. In: CVPR (2022)
20. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? NeurIPS (2017)

21. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences (2017)
22. Klein, G., Murray, D.: Parallel tracking and mapping for small ar workspaces. In: ISMAR (2007)
23. Klein, G., Murray, D.: Parallel tracking and mapping on a camera phone. In: ISMAR (2009)
24. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Communications of the ACM (2017)
25. Lee, S., Ha, J., Zhang, D., Kim, G.: A neural dirichlet process mixture model for task-free continual learning. ICLR (2020)
26. Li, J., Feng, Z., She, Q., Ding, H., Wang, C., Lee, G.H.: Mine: Towards continuous depth mpi with nerf for novel view synthesis. In: CVPR (2021)
27. Li, Z., Hoiem, D.: Learning without forgetting. TPAMI (2017)
28. Lin, C.H., Ma, W.C., Torralba, A., Lucey, S.: Barf: Bundle-adjusting neural radiance fields. In: ICCV (2021)
29. Liu, L., Gu, J., Zaw Lin, K., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. NeurIPS (2020)
30. Lopez-Paz, D., Ranzato, M.: Gradient episodic memory for continual learning. NeurIPS (2017)
31. Mallya, A., Lazebnik, S.: Packnet: Adding multiple tasks to a single network by iterative pruning. In: CVPR (2018)
32. Martin-Brualla, R., Radwan, N., Sajjadi, M.S., Barron, J.T., Dosovitskiy, A., Duckworth, D.: Nerf in the wild: Neural radiance fields for unconstrained photo collections. In: CVPR (2021)
33. Mildenhall, B., Hedman, P., Martin-Brualla, R., Srinivasan, P.P., Barron, J.T.: Nerf in the dark: High dynamic range view synthesis from noisy raw images. In: CVPR (2022)
34. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
35. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. SIGGRAPH (2022)
36. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A.: Kinectfusion: Real-time dense surface mapping and tracking. In: ISMAR (2011)
37. Newcombe, R.A., Lovegrove, S.J., Davison, A.J.: Dtam: Dense tracking and mapping in real-time. In: ICCV (2011)
38. Rannen, A., Aljundi, R., Blaschko, M.B., Tuytelaars, T.: Encoder based lifelong learning. In: ICCV (2017)
39. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: CVPR (2017)
40. Reiser, C., Peng, S., Liao, Y., Geiger, A.: Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In: ICCV (2021)
41. Robins, A.: Catastrophic forgetting, rehearsal and pseudorehearsal. Connection Science (1995)
42. Roessle, B., Barron, J.T., Mildenhall, B., Srinivasan, P.P., Nießner, M.: Dense depth priors for neural radiance fields from sparse input views. In: CVPR (2022)
43. Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., Wayne, G.: Experience replay for continual learning. NeurIPS (2019)

44. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. NeurIPS (2017)
45. Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., et al.: The replica dataset: A digital replica of indoor spaces. arXiv preprint arXiv:1906.05797 (2019)
46. Sucar, E., Liu, S., Ortiz, J., Davison, A.J.: imap: Implicit mapping and positioning in real-time. In: ICCV (2021)
47. Sun, J., Xie, Y., Chen, L., Zhou, X., Bao, H.: Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In: CVPR (2021)
48. Tancik, M., Casser, V., Yan, X., Pradhan, S., Mildenhall, B., Srinivasan, P.P., Barron, J.T., Kretzschmar, H.: Block-nerf: Scalable large scene neural view synthesis. In: CVPR (2022)
49. Wang, H., Ren, J., Huang, Z., Olszewski, K., Chai, M., Fu, Y., Tulyakov, S.: R2l: Distilling neural radiance field to neural light field for efficient novel view synthesis. In: ECCV (2022)
50. Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W.: Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. NeurIPS (2021)
51. Wang, Q., Wang, Z., Genova, K., Srinivasan, P.P., Zhou, H., Barron, J.T., Martin-Brualla, R., Snavely, N., Funkhouser, T.: Ibrnet: Learning multi-view image-based rendering. In: CVPR (2021)
52. Wei, Y., Liu, S., Rao, Y., Zhao, W., Lu, J., Zhou, J.: Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In: ICCV (2021)
53. Xu, J., Zhu, Z.: Reinforced continual learning. NeurIPS (2018)
54. Yan, Z., Tian, Y., Shi, X., Guo, P., Wang, P., Zha, H.: Continual neural mapping: Learning an implicit scene representation from sequential observations. In: ICCV (2021)
55. Yariv, L., Gu, J., Kasten, Y., Lipman, Y.: Volume rendering of neural implicit surfaces. NeurIPS (2021)
56. Yen-Chen, L., Florence, P., Barron, J.T., Rodriguez, A., Isola, P., Lin, T.Y.: inerf: Inverting neural radiance fields for pose estimation. In: IROS (2021)
57. Yu, A., Li, R., Tancik, M., Li, H., Ng, R., Kanazawa, A.: Plenoctrees for real-time rendering of neural radiance fields. In: ICCV (2021)
58. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelnerf: Neural radiance fields from one or few images. In: CVPR (2021)
59. Yu, Z., Peng, S., Niemeyer, M., Sattler, T., Geiger, A.: Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. NeurIPS (2022)
60. Zhang, K., Riegler, G., Snavely, N., Koltun, V.: Nerf++: Analyzing and improving neural radiance fields. arXiv preprint arXiv:2010.07492 (2020)
61. Zhang, X., Bi, S., Sunkavalli, K., Su, H., Xu, Z.: Nerfusion: Fusing radiance fields for large-scale scene reconstruction. In: CVPR (2022)
62. Zhao, N., Lee, G.H.: Static-dynamic co-teaching for class-incremental 3d object detection. In: AAAI (2022)
63. Zhipeng Cai, M.M.: Clnerf: Continual learning meets nerf. In: ICCV (2023)
64. Zhu, Z., Peng, S., Larsson, V., Xu, W., Bao, H., Cui, Z., Oswald, M.R., Pollefeys, M.: Nice-slam: Neural implicit scalable encoding for slam. In: CVPR (2022)