Incremental Learning for Neural Radiance Field with Uncertainty-Filtered Knowledge Distillation

Mengqi Guo, Chen Li, Gim Hee Lee
Department of Computer Science, National University of Singapore
{mengqi, lic, gimhee.lee}@comp.nus.edu.sg

Abstract

Recent neural radiance field (NeRF) representation has achieved great success in the tasks of novel view synthesis and 3D reconstruction. However, they suffer from the catastrophic forgetting problem when continuously learning from streaming data without revisiting the previous training data. This limitation prohibits the application of existing NeRF models to scenarios where images come in sequentially. In view of this, we explore the task of incremental learning for neural radiance field representation in this work. We first propose a student-teacher pipeline to mitigate the catastrophic forgetting problem. Specifically, we iterate the process of using the student as the teacher at the end of each incremental step and let the teacher guide the training of the student in the next step. In this way, the student network is able to learn new information from the streaming data and retain old knowledge from the teacher network simultaneously. Given that not all information from the teacher network is helpful since it is only trained with the old data, we further introduce a random inquirer and an uncertainty-based filter to filter useful information. We conduct experiments on the NeRF-synthetic360 and NeRF-real360 datasets, where our approach significantly outperforms the baselines by 7.3% and 25.2% in terms of PSNR. Furthermore, we also show that our approach can be applied to the large-scale camera facing-outwards dataset ScanNet, where we surpass the baseline by 60.0% in PSNR.

1. Introduction

Recent neural radiance field (NeRF) representation [27] has attracted increasing attention in the last few years because of its great success in novel view synthesis and 3D reconstruction. The key of NeRF is to memorize the volume density 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 images of a scene at different camera views to be available for a one-time training. This is not applicable to real-world scenarios where streaming data comes in sequentially or where only limited data storage is available. In view of this, we explore an important task of incremental learning for neural radiance field representation 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.

Figure 1. Visualization of rendered images from the incremental learning novel view synthesis task. Our method alleviates catastrophic forgetting and significantly surpasses NeRF [27] baseline.

The main challenge for this task is the catastrophic forgetting problem [34], where the network trained with new incoming images drastically forgets the previously learned knowledge. This can be evident from the result of NeRF in Fig. 1, where the NeRF model is trained in the incremental setting. We can see that NeRF forgets the appearances of previous scenes that are not visible under the current views resulting in the rendering of noisy and blurry images with severe artifacts. The catastrophic forgetting problem is widely discussed in the incremental learning literature [1, 2, 20, 24, 54], and the most related work to ours is Continual Neural Mapping (CNM) in [47]. CNM first introduces the incremental setting for Signed Distance Function (SDF) based reconstruction. A data-replay strategy [18] is adopted in CNM to mitigate the forgetting problem. However, the data-replay still requires part of the previous training data to be stored and the amount of stored data increases as the incremental training proceeds. Additionally,

the SDF-based representation in [30] requires ground truth depth for supervision, while our NeRF-based representation focuses on supervision with only RGB images.

In this work, we propose a student-teacher pipeline to tackle the catastrophic forgetting problem in the incremental NeRF. Specifically, we first train the NeRF model with the currently available data, and then use the trained model as the teacher model to generate supervision for the student model. We iterate this process by using the student as the teacher at the end of each incremental step, and letting the teacher guide the training of the student in the next step. In this way, the student network is able to learn from the newly available data and preserves the old knowledge from the teacher network. Furthermore, we also propose an alternate optimization strategy such that the new data and knowledge from the teacher network can be effectively imparted to the student network.

The aim of introducing the teacher network is to impart the knowledge obtained from the previous training steps to the current student network. However, the teacher network which has only been trained with the old data is unable to generate useful knowledge for the unseen data. In view of this, we further introduce a random inquirer and an uncertainty-based filter for filtering useful knowledge. 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 show the result of our proposed approach in Fig. 4, where we can see that our model effectively preserves the knowledge from previously seen data, and accurately renders realistic images from both current and previous views.

We validate our proposed approach on both object-centric and large-scale camera facing-outwards datasets, where we achieve better results and largely reduce the catastrophic forgetting problems. Without storing any images from previous data, our model significantly outperforms NeRF by 7.3%, 25.2%, and 60.0% in terms of PSNR on the NeRF-synthetic, NeRF-real, and ScanNet datasets, respectively. Our contributions are as follows:

- We introduce the task of incremental learning for neural radiance fields.
- We propose a teacher-student 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's network.
- We significantly outperform the NeRF baseline by a large margin on various datasets.

2. Related Work

Neural Radiance Fields. The neural radiance field (NeRF) [27] exhibits strong capabilities on novel view synthesis [4, 5], 3D reconstruction [3, 41], depth estimation [19,45], and camera pose estimation [21,49]. They [27] adopt a simple yet effective MLP network to implicitly memorize the 3D scene and propose a differentiable rendering method for generating images from 3D color and density. Due to its intriguing ability to represent complex 3D scenes with a simple neural network, many follow-up works have been trying to improve NeRF to unleash its full potential. These works include real-time rendering [33, 50], faster training [7, 12, 22, 28], sparse view input [35,51], scene-agnostic model [8,44], scalable to large-scale scenes [42, 53], lightning changing [25, 26], density representation [29, 43, 48], robust training [21], better representation [4, 52], *etc*. In this paper, we focus on the under-explored and yet important incremental settings for NeRF [27] on the task of novel view synthesis, where NeRF encounters the catastrophic forgetting problem. To this end, we introduce incremental learning into neural radiance fields to prevent the forgetting of previous knowledge.

Incremental Learning. Incremental learning is a classical machine learning problem, which refers to the setting where only partial data is available for training at each step. Most existing methods focus on three fields: data replay [6, 23, 32, 36, 39], parameter regularization [1, 16, 20, 31, 54], and parameter isolation [2, 11, 24, 46] according to [10]. The first work in incremental learning for SDF-based implicit neural reconstruction is Continual Neural Mapping [47]. They adopt a reply-based method [18] for 3D indoor scene reconstruction from incremental depth images. However, their method requires some of the previous data to prevent forgetting and their backbone [40] needs the ground truth depth for supervision. In contrast, we focus on incremental learning for novel view synthesis with NeRF trained on the current RGB images without access to any previous data. Specifically, we propose a regularization-based method where the neural implicit function is the representation of previously seen images. We rely on the network from the last step as the teacher model and utilize the knowledge distillation strategy [14] to self-supervise the current network noted as the student model. To generate pseudo data from the teacher network, we propose a random inquirer and an uncertainty-based filter.

Uncertainty Modeling. It is straightforward to use the variance of the predicted class probability distribution in the classification task [13]. For the regression task, [15] proposed an uncertainty prediction branch by self-supervised learning. This formulation has been explored in NeRF-in-the-Wild [25] to distinguish the static and transient scenes at the pixel-level. However, in this paper, we aim to model the

network confidence of the output at the image-level for the selection of useful generated camera views from our random inquirer. Consequently, the student network can always learn from the previously seen camera views from the teacher network, *i.e.* the previous knowledge.

3. Preliminaries: NeRF

Neural Radiance Fields (NeRF) [27] is an effective neural implicit representation of 3D scenes for the novel view synthesis task. The principal idea of NeRF is to memorize the 3D scene as color $\mathbf{c} = (r, g, b)$ and volume density σ for each 3D location $\mathbf{x} = (x, y, z)$ and camera view direction $\mathbf{d} = (\theta, \phi)$ with a simple neural network $F(\cdot)$, *e.g.* MLPs. 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:

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

where δ_j indicates the distance between i^{th} sample and $(i+1)^{\text{th}}$ sample, and T_i is the accumulated transmittance along the ray $\mathbf{r}$ from camera center to i^{th} 3D point, represented as:

$$T_i = \exp\Big(-\sum_{j=1}^{i-1} \sigma_j \delta_j\Big). \tag{2}$$

The model is trained in a coarse-to-fine manner on the coarse and fine MLPs denoted as F_c and F_f, respectively, for faster convergence. Since the whole pipeline is differentiable, the rendering output can be directly supervised by RGB image label with the L2 distance:

$$\mathcal{L} = \sum_{\mathbf{r} \in R} \Big(||c^*(\mathbf{r}) - c_c(\mathbf{r})||_2^2 + ||c^*(\mathbf{r}) - c_f(\mathbf{r})||_2^2 \Big), \tag{3}$$

where $c^*(\mathbf{r})$ denoted the ground truth color, $c_c(\mathbf{r})$ and $c_f(\mathbf{r})$ indicates the rendered color of the coarse network F_c and fine network F_f, and R represents a group of rays from one or more camera views.

3.1. Catastrophic Forgetting of NeRF

Although NeRF [27] has achieved impressive performance, it has to be trained simultaneously on the entire set of images that covers all views as shown in Fig. 2. In practice, the entire set of images might not be available simultaneously due to streaming data or limited data storage. As a result, the network $F(\cdot)$ has to be trained on new coming data without revisiting old ones and thus suffers from catastrophic forgetting, where it quickly forgets previously learned knowledge while acquiring new ones. In view of this, we propose a new task of incremental learning for neural radiance fields and aim to mitigate the catastrophic forgetting problem in this task.

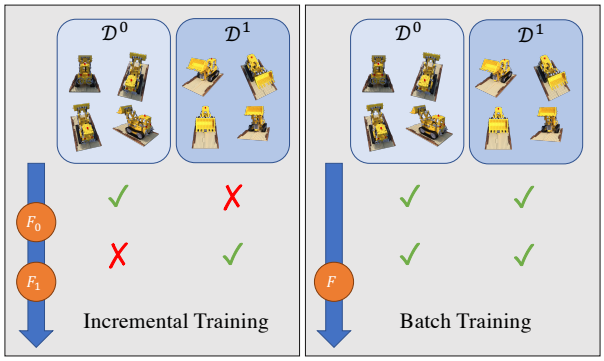

Figure 2. Comparison of incremental training and batch training. The blue arrow denotes the training time flow. The incremental training only has the access to the current data such as $\mathcal{D}^0$ in time step 0, while all the data are available in batch training.

4. Our Method

In this section, we first introduce the incremental setting for NeRF-based novel view synthesis. We then introduce our proposed student-teacher pipeline the uncertainty 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 consists of N pairs of images I^t and the corresponding 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 $P^i \cap P^j = \phi, \forall i, j \in \{0, 1, \cdots, T\}, i \neq j$. The task of incremental learning for NeRF aims to continually learn from the newly arriving data $\mathcal{D}^t$ while also preserve the knowledge from previously seen data $\mathcal{D}^{0:t-1}$.

4.2. Overview

We show the overall framework of our approach in Fig. 3, which consists of a student and teacher network. The student and teacher networks share the same structure, which consists of a density branch, a color branch and a uncertainty branch. At each incremental step t, the student learns simultaneously from the currently available data $\mathcal{D}^t$ and the old knowledge from the teacher network. The trained student model is then used as the teacher model at the next step and imparts the learned knowledge to the student. We iterate this process through the whole training process. To explore the knowledge space of the teacher network, we design an inquirer to generate camera views for the teacher network. Given that the teacher network can generate incorrect information for randomly generated views since it only trains on seen data $\mathcal{D}^{0:t-1}$, we select the

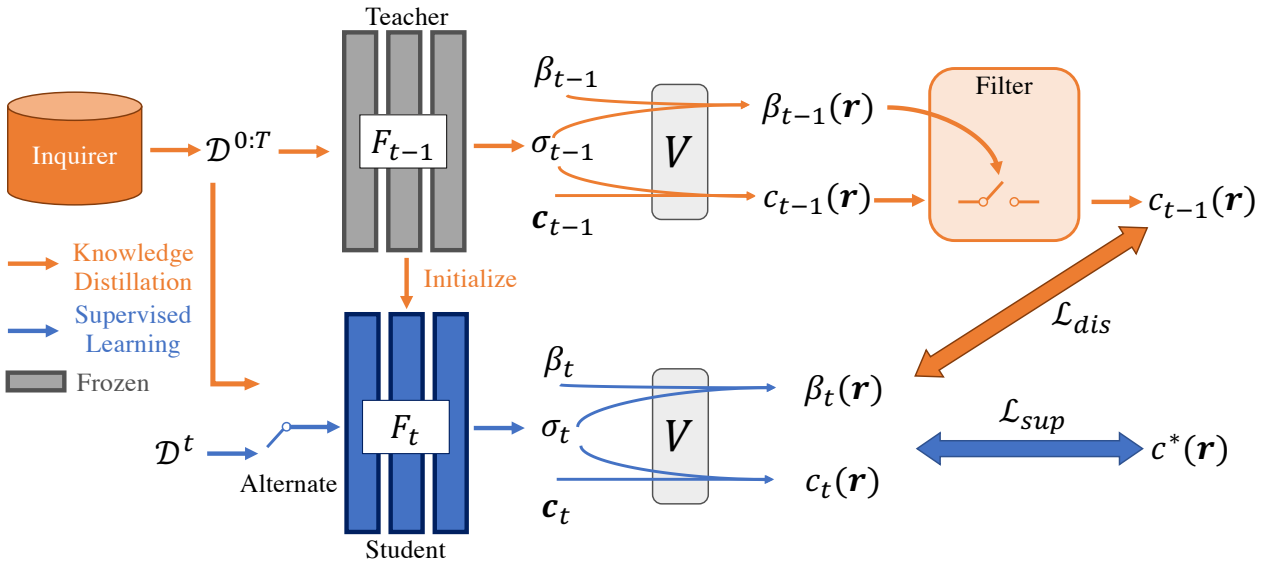

Figure 3. Illustration of our teacher-student framework. For the current data D^t, we utilize a supervised learning strategy with ground truth label $c^*(\mathbf{r})$. For the knowledge distillation from a feature-frozen teacher network, we propose a random inquirer and uncertainty-based filter to teach rich and useful information. V refers to the differentiable volume renderer with the weights computed from σ, and it works for both color $\mathbf{c}$ from Eqn. (1) and uncertainty β from Eqn. (5).

reliable information from the teacher network based on the uncertainty score predicted by the uncertainty branch.

4.3. Supervised learning

At each incremental time step t, we directly utilize the currently available data $\mathcal{D}^t$ to supervise the student network as in Sec. 3 such that the network continually learns from new coming data. Our student network is built upon the original NeRF by introducing an uncertainty branch. This branch is used to model the uncertainty of the model for the input of each ray and is also used to select useful information from the teacher network during knowledge distillation.

Uncertainty Modeling. We adopt the self-supervised uncertainty formulation from [15] to our case to model the uncertainty of the network for each input ray. This formulation has also been exploited in previous NeRF-in-the-Wild [25] to distinguish the static and transient scenes. With a different objective, we aim to utilize the output of the uncertainty branch to indicate the confidence of the network about the current input. Specifically, we build an additional branch, which shares the same input as the color branch to predict the uncertainty $\beta = F(\mathbf{x}, \mathbf{d})$. We adopt Softplus as the activation function for uncertainty for stable training, *i.e.*:

$$\hat{\beta}_i = log(1 + e^{\beta_i - 1}). \tag{4}$$

Finally, we compute the pixel-wise uncertainty from each sample point using the same volume rendering as the color:

$$\beta(\mathbf{r}) = \sum_{n=1}^{N} T_i(1 - exp(-\sigma_j \delta_j))\hat{\beta}_i + \beta_{min}, \tag{5}$$

where T_i represents the accumulated transmittance expressed by Eqn. (2), and β_{min} denotes a hyper-parameter that ensures the minimum uncertainty following [25].

Supervised Optimization. The objective function of the supervised training is expressed as:

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

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, which we refer as the catastrophic forgetting problem. This problem manifests itself in struggling to synthesize high-quality images for previously seen views. To prevent this, we further introduce a teacher network to impart the previously learned knowledge to the current model.

Teacher-student Modeling. At each time step t, the student network F_t concurrently learns from the teacher network F_{t-1}, except for the new coming data $\mathcal{D}^t$. The student network is then used as the teacher network after each time step. We iterate the process of using the student as the teacher at the end of each incremental time step, and let the teacher guide the student in the next step. In this way, the student network can learn both new knowledge from the new data and old knowledge from the teacher network and hence mitigating the forgetting problem. To facilitate the knowledge imparting from the teacher to the student network, we introduce a knowledge distillation loss [14] for both the coarse and fine stages. 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 all previously seen data.

Random Inquirer. The teacher network shares the same input as the student network with different augmentations in the traditional student-teacher pipeline [54]. In our case, the role of the teacher network is to impart old knowledge, which means the input should be training data from previous time steps. However, the previous training data is not accessible. To circumvent this problem, we design a constrained random inquirer to generate inputs for knowledge distillation. Specifically, we randomly generate camera views as the inputs, which needs to fulfill some constraints specific to different types of datasets. For the object-centric datasets, all cameras locate on one hemisphere and point toward the origin at the same distance. The degree of freedom of the camera matrix is reduced to two in this case, and hence we can generate the input by randomly sampling from the range of the whole dataset, such as on a hemisphere. For indoor-scene datasets (camera facing-outwards) 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. Refer to our supplementary for more details on the random inquirer.

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 only been trained on $D^{0:t-1}$, while the input generated from the random inquirer covers the whole dataset. To overcome this, 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:

$$R^* \leftarrow R^* \cup R_v, \quad \text{if} \quad \frac{1}{\mathcal{N}_{R_v}} \sum_{\mathbf{r} \in R_v} (\beta_t(\mathbf{r})) < \beta^{thr}. \tag{7}$$

where β^{thr} denotes the threshold. R_v denotes the rays for camera view v generated from the random inquirer, $\mathcal{N}_{R_v}$ is the number of rays samples. R^* represents a 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:

$$\begin{aligned} \mathcal{L}_{dis} = \sum_{\mathbf{r} \in R^*} \Big(& \frac{||c_{t-1}(\mathbf{r}) - c_t(\mathbf{r})||_2^2}{2} \\ & + \frac{||c_{t-1}(\mathbf{r}) - c_t(\mathbf{r})||_2^2}{2 * \beta_t(\mathbf{r})^2} + \log(\beta_t(\mathbf{r})) + \eta \Big), \end{aligned} \tag{8}$$

where $c_{t-1}(\mathbf{r})$ and $c_t(\mathbf{r})$ represent the output color of the teacher or student model, respectively. R^* denotes useful data selection from Eqn. (7). 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 network. Specifically, we alternatively optimize the supervised loss Eqn. (6) and the knowledge distillation loss, *i.e*.:

$$\mathcal{L} = \begin{cases} \mathcal{L}_{sup}, & \text{if } i \,\%\, 2 == 0 \\ \mathcal{L}_{dis}, & \text{otherwise} \end{cases}, \tag{9}$$

where i denotes the iteration number. Note that this optimization function is used for both the coarse network and fine network.

5. Experiments

5.1. Experimental settings

We conduct experiments on three datasets, two object-centric datasets, NeRF-synthetic360 [27] and NeRF-real360 [27], and a large-scale camera facing-outwards dataset, ScanNet [9].

NeRF-synthetic360. We use all eight scenes, each containing 100 training images and 200 testing images with a resolution of 800×800. For incremental setting, we divide the images of each scene and the corresponding camera poses into four time steps in four quadrants $\mathcal{D} = \{\mathcal{D}^0, \mathcal{D}^1, \mathcal{D}^2, \mathcal{D}^3\}$.

Scene	Method	Test Dataset (PSNR ↑ /SSIM ↑ /LPIPS ↓)				
		$\mathcal{D}^0$	$\mathcal{D}^1$	$\mathcal{D}^2$	$\mathcal{D}^3$	Average on $\mathcal{D}^{0:3}$
Chair	NeRF	25.51 / 0.935 / 0.066	24.88 / 0.926 / 0.070	27.32 / 0.943 / **0.038**	**32.14 / 0.975 / 0.018**	27.46 / 0.945 / 0.048
	w/o filter	30.49 / 0.969 / 0.031	24.37 / 0.949 / 0.057	22.55 / 0.918 / 0.079	24.43 / 0.933 / 0.070	25.46 / 0.942 / 0.059
	w/o init	28.36 / 0.955 / 0.056	27.83 / 0.948 / 0.055	26.14 / 0.915 / 0.080	28.22 / 0.944 / 0.064	27.64 / 0.941 / 0.064
	Ours	**31.29 / 0.973 / 0.027**	**31.62 / 0.973 / 0.025**	**28.13 / 0.947** / 0.044	27.07 / 0.939 / 0.057	**29.53 / 0.958 / 0.038**
Lego	NeRF	24.99 / 0.917 / 0.049	23.15 / 0.896 / 0.072	25.31 / 0.930 / 0.049	**34.23 / 0.981 / 0.011**	26.92 / 0.931 / 0.045
	w/o filter	26.11 / 0.933 / 0.044	13.93 / 0.842 / 0.127	13.64 / 0.834 / 0.170	21.13 / 0.919 / 0.086	18.70 / 0.882 / 0.107
	w/o init	19.34 / 0.840 / 0.141	15.85 / 0.756 / 0.241	19.88 / 0.843 / 0.150	30.52 / 0.962 / 0.030	21.40 / 0.850 / 0.141
	Ours	**28.12 / 0.949 / 0.031**	**29.81 / 0.962 / 0.025**	**30.25 / 0.965 / 0.026**	31.94 / 0.970 / 0.022	**30.03 / 0.962 / 0.026**
Ship	NeRF	24.62 / 0.874 / 0.123	21.67 / 0.826 / 0.143	25.00 / 0.880 / 0.103	**29.49** / 0.929 / **0.080**	25.20 / 0.877 / 0.112
	w/o filter	24.35 / 0.880 / 0.157	18.27 / 0.809 / 0.202	11.21 / 0.732 / 0.256	19.93 / 0.854 / 0.177	18.44 / 0.819 / 0.198
	w/o init	24.46 / 0.872 / 0.165	23.39 / 0.843 / 0.185	24.47 / 0.871 / 0.152	26.98 / 0.904 / 0.129	24.83 / 0.873 / 0.158
	Ours	**26.92 / 0.900 / 0.117**	**26.02 / 0.884 / 0.111**	**27.12 / 0.903 / 0.096**	28.49 / **0.967** / 0.097	**27.14 / 0.914 / 0.105**
Average	NeRF	25.08 / 0.926 / 0.056	23.18 / 0.907 / 0.067	25.54 / 0.931 / 0.049	**31.09 / 0.966 / 0.026**	26.22 / 0.932 / 0.050
	w/o filter	27.76 / 0.946 / 0.053	21.49 / 0.908 / 0.089	18.21 / 0.879 / 0.125	22.85 / 0.923 / 0.083	22.58 / 0.914 / 0.088
	w/o init	24.69 / 0.915 / 0.089	23.22 / 0.897 / 0.104	24.50 / 0.910 / 0.089	27.81 / 0.943 / 0.060	25.05 / 0.916 / 0.086
	Ours	**28.70 / 0.953 / 0.044**	**28.07 / 0.951 / 0.042**	**28.23 / 0.949 / 0.043**	27.52 / 0.951 / 0.053	**28.13 / 0.951 / 0.045**

Table 1. Comparison with the baseline and ablation studies on the NeRF-synthetic360 dataset measured by PSNR, SSIM, and LPIPS. We show the average and three scenes results on each step test datasets $\mathcal{D}^0, \mathcal{D}^1, \mathcal{D}^2, \mathcal{D}^3$ and the average performance, respectively. 'NeRF' is our baseline, "w/o filter" and "w/o init" denote the two ablations, and 'Ours' represents our full pipeline. All the models are incrementally trained on the 4-step training datasets.

NeRF-real360. We use both two scenes with around 100 images with camera poses in each scene. Following the setting in NeRF [27], we randomly sample 12.5% of images as the training set and use the rest for testing. For incremental setting, we split one scene into 10-time steps $\mathcal{D} = \{\mathcal{D}^0, \mathcal{D}^1, \cdots, \mathcal{D}^9\}$ in the temporal order.

ScanNet. Following [22], we use scene 101 and scene 241 with > 1000 images at a resolution of 648×484. We randomly sample 20% of images as the training set and use the rest for testing. The camera poses are optimized by COLMAP [37,38]. For incremental setting, we split a scene into 10-time steps $\mathcal{D} = \{\mathcal{D}^0, \mathcal{D}^1, \cdots, \mathcal{D}^9\}$ in the temporal order.

Implementation details. We train the network for 50k (for ScanNet and NeRF-real360) or 100k (for NeRF-synthetic360) iterations in each incremental step. During optimization, we employ the Adam optimizer with a learning rate that starts at 5×10^4 and exponentially decays to 5×10^5. The hyperparameters are $\beta_{min} = 0.1$ and $\eta = 3$, and the uncertainty threshold β^{thr} is set differently for different scenes.

5.2. Comparison

Baseline. We compare to original NeRF [27] with default setting in all three benchmarks. The implementation details for the baseline and ours are the same for a fair comparison.

NeRF-synthetic360. We first show the results of our model and the baseline 'NeRF' on the NeRF-synthetic360 dataset in Tab. 1. Results of ablation models 'w/o filter' and 'w/o init' are also included, which we will discuss in Sec. 5.3. The four columns $\mathcal{D}^0, \mathcal{D}^1, \mathcal{D}^2, \mathcal{D}^3$ denote the testing dataset at the corresponding time step. The results are obtained from the final model, which has been trained incrementally on all four datasets $\mathcal{D}^{0:3}$. We can see that the performance of the NeRF baseline drops significantly on the previous testing data, *i.e.* $\mathcal{D}^0, \mathcal{D}^1, \mathcal{D}^2$ despite the good performance on $\mathcal{D}^4$. This indicates that the baseline only learns from current data and forgets previously learned knowledge. In comparison, our approach achieves stable performance across testing data of different time steps, and we achieve better average performance on datasets across all time steps (last column). Moreover, our average performance of all scenes (last row) improves the baseline by 7.3% in PSNR.

NeRF-real360. To evaluate the effectiveness of our method on real-world scenes, we also conduct experiments on the NeRF-real360 [27] dataset. We can see from Tab. 2 that the baseline suffers from the forgetting problem and the performance drops significantly on the testing data at previous time steps $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6$. In comparison, our model is able to perform well on previous testing data with improvements on the baseline by 25.7%, 33.7%, and 24.0% in PSNR. Furthermore, the overall results of our approach for all testing data $\mathcal{D}^{0:9}$ outperform the baseline by 25.2% in PSNR and 28.8% in SSIM.

Scene	Method	Test Dataset (PSNR ↑ /SSIM ↑ /LPIPS ↓)				
		$\mathcal{D}^0$	$\mathcal{D}^3$	$\mathcal{D}^6$	$\mathcal{D}^9$	Average on $\mathcal{D}^{0:9}$
Pinecone	NeRF	15.18 / 0.351 / 0.289	14.61 / 0.294 / 0.319	15.06 / 0.447 / 0.322	17.45 / **0.590** / **0.221**	14.73 / 0.347 / 0.319
	Ours	**19.35** / **0.519** / **0.250**	**19.07** / **0.518** / **0.278**	**18.30** / **0.535** / **0.243**	**17.99** / 0.585 / 0.272	**18.20** / **0.498** / **0.275**
Vasdeck	NeRF	17.02 / 0.584 / 0.263	15.32 / 0.519 / 0.394	15.94 / 0.577 / 0.323	18.91 / 0.681 / **0.213**	16.39 / 0.589 / 0.314
	Ours	**21.13** / **0.709** / **0.234**	**20.94** / **0.721** / **0.241**	**20.14** / **0.715** / **0.261**	**20.15** / **0.708** / 0.227	**20.76** / **0.709** / **0.249**
Average	NeRF	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
	Ours	**20.24** / **0.614** / **0.242**	**20.01** / **0.620** / **0.260**	**19.22** / **0.625** / **0.252**	**19.07** / **0.647** / 0.250	**19.48** / **0.603** / **0.262**

Table 2. Comparison with the baseline on NeRF-real360 dataset measure by PSNR, SSIM, and LPIPS. We show the average and two scenes results on each step test datasets $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ and the average performance, respectively. All the models are incrementally trained on the 10-step training datasets.

Scene	Method	Test Dataset (PSNR ↑ /SSIM ↑ /LPIPS ↓)				
		$\mathcal{D}^0$	$\mathcal{D}^3$	$\mathcal{D}^6$	$\mathcal{D}^9$	Average on $\mathcal{D}^{0:9}$
101	NeRF	12.14 / 0.580 / 0.396	12.91 / 0.597 / 0.529	18.42 / 0.763 / 0.313	**29.77** / **0.903** / **0.120**	14.83 / 0.657 / 0.403
	Ours	**23.53** / **0.822** / **0.230**	**20.04** / **0.787** / **0.344**	**26.09** / **0.877** / **0.190**	28.19 / 0.887 / 0.160	**23.08** / **0.833** / **0.248**
241	NeRF	13.07 / 0.580 / 0.396	10.27 / 0.413 / 0.612	12.87 / 0.393 / 0.627	22.12 / 0.849 / 0.169	12.74 / 0.495 / 0.517
	Ours	**19.95** / **0.801** / **0.217**	**20.38** / **0.863** / **0.248**	**21.70** / **0.824** / **0.257**	**22.41** / **0.865** / **0.163**	**21.02** / **0.833** / **0.244**
Average	NeRF	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
	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	**22.05** / **0.833** / **0.246**

Table 3. Comparison with the baseline on ScanNet dataset measure by PSNR, SSIM, and LPIPS. We show the average and two scenes results on each step test datasets $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ and the average performance, respectively. All the models are incrementally trained on the 10-step training datasets.

ScanNet. We further conduct experiments on the more challenging large-scale camera facing-outwards ScanNet [9] dataset. We report the results in Tab. 3, which includes the performance of the final model on $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$. We note three findings. **First**, the baseline performs 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 teacher-student pipeline, our method still achieves promising results with improvements of 72.4%, 74.4%, and 52.7% in PSNR. **Second**, our model achieves comparable results on current testing data $\mathcal{D}^9$. The reason is that our random inquirer and uncertainty-based filter provide more effective training for views from $\mathcal{D}^0$, which includes the same scene as $\mathcal{D}^9$. **Third**, our model significantly improves the baseline by 60.0% in terms of PSNR since the baseline almost completely forgets the previous data.

5.3. Ablation Study

We conduct ablations studies on the NeRF-synthetic360 dataset to verify the effectiveness of our uncertainty-based filter and initialization as we discussed in Sec. 4.4 (Uncertainty-based Filter) and Sec. 4.4 (Teacher-student Modeling), respectively.

Uncertainty-based Filter. To investigate the effectiveness of the uncertainty-based filter, we compare with our network without the filter, denoted as 'w/o filter' in Tab. 1. We can see that the performance drops significantly without the uncertainty-based filter, only achieving 22.58dB in terms of PSNR. 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.

Initialization. We initialize the student teacher with parameters of the teacher network at the beginning of each incremental time step. To verify the effectiveness, we compare with our model trained from scratch, denoted as 'w/o init' in Tab. 1. We can see that the model without initialization performs well on the current time step $\mathcal{D}^3$, but drops 4.01dB, 4.85dB, and 3.73dB in PSNR for the previous time-step testing data. This demonstrates the effectiveness of our initialization strategy in preserving previously learned knowledge.

5.4. Visualization

We show the qualitative comparison for the ScanNet scene 101 and NeRF-real360 scene *Vasdeck* shown in Fig. 4, where $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ represent the testing data at

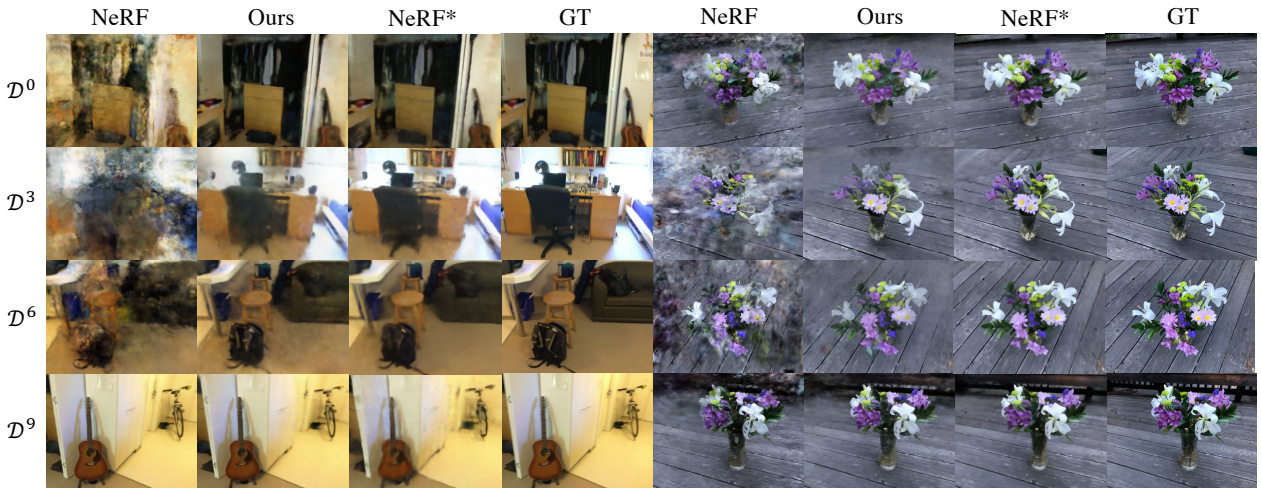

Figure 4. Qualitative comparison with the baseline 'NeRF' and upper bound batch training method 'NeRF*' on ScanNet and NeRF-real360 datasets. 'GT' represents the ground truth images and $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ denote the results from each time step test views. 'NeRF' and 'Ours' models are incrementally trained on the 10-step training datasets.

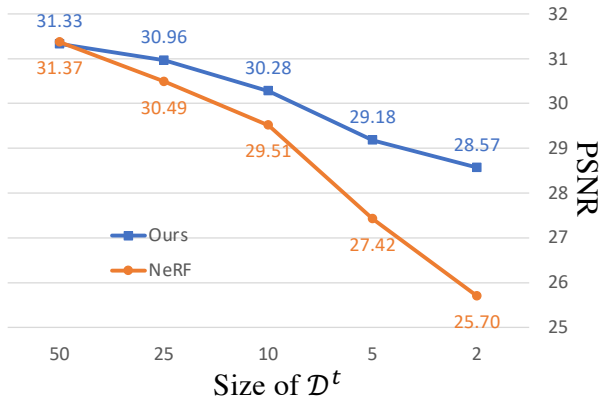

Figure 5. Comparison of the impact of $\mathcal{D}^t$ size on performance measured by PSNR with the baseline 'NeRF'.

the corresponding time step. The results of our model and 'NeRF' are obtained from incremental training, while the results of 'NeRF*' are from batch training, *i.e.* training with all data $\mathcal{D}$. Note that the batch training is the upper bound of incremental learning. As we can see, the original NeRF suffers from the catastrophic forgetting problem and outputs images on $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6$ 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.

5.5. Limited Storage Analysis

Since we only need to store the data at the current time step, incremental learning is useful for scenarios where only limited data storage is available. We hence investigate the performance of our model when the data storage gets smaller, *i.e.* the number of images in $\mathcal{D}^t$ becomes small. We compare our method with the baseline on the *Lego* scene of the NeRF-synthetic360 dataset, which has 100 training images in total. As shown in Fig. 5, the performance of our model stays stable when the number of images in $\mathcal{D}^t$ reduces from 50 to 2, while the performance of NeRF drops significantly. This suggests the advantage of our model when applied to cases with extremely low data storage.

6. Limitation

Although our method achieves significant improvement based on the NeRF [27] baseline in incremental training, our performance still has a margin compared to the models with batch training. As there is plentiful information in the large-scale scene and there is much less overlap between views in the camera facing-outwards dataset, our method may not memorize all the knowledge from the limited images and even the batch training model can only learn limited information about the scene as shown in Fig. 4.

7. Conclusion

In this paper, we introduce the task of incremental learning for neural radiance fields and propose a teacher-student pipeline for mitigating the catastrophic forgetting problem. To improve the effectiveness of the data provided by the teacher network, we design a random inquirer and an uncertainty-based filter for rich and useful knowledge distillation. Supervised learning and knowledge distillation are iteratively utilized for the combination of preserving old information and learning from current new data. Experiments on three synthetic and real-world datasets demonstrate that our model achieves great improvement comparing to the baseline with incremental data.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, 2018. 1, 2

[2] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *CVPR*, 2017. 1, 2

[3] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *CVPR*, 2022. 2

[4] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021. 2

[5] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022. 2

[6] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018. 2

[7] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. *ECCV*, 2022. 2

[8] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *ICCV*, 2021. 2

[9] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 5, 7, 11

[10] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *TPAMI*, 2021. 2

[11] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017. 2

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022. 2

[13] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *ICML*, 2017. 2

[14] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2, 5

[15] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *NeurIPS*, 2017. 2, 4

[16] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 2017. 2

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 2017. 11

[18] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. *ICLR*, 2020. 1, 2

[19] Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. Mine: Towards continuous depth mpi with nerf for novel view synthesis. In *CVPR*, 2021. 2

[20] Zhizhong Li and Derek Hoiem. Learning without forgetting. *TPAMI*, 2017. 1, 2

[21] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *ICCV*, 2021. 2

[22] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 2, 6

[23] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *NeurIPS*, 2017. 2

[24] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *CVPR*, 2018. 1, 2

[25] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *CVPR*, 2021. 2, 4

[26] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P Srinivasan, and Jonathan T Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *CVPR*, 2022. 2

[27] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 3, 5, 6, 8, 11

[28] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *SIGGRAPH*, 2022. 2

[29] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *ICCV*, 2021. 2

[30] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[31] Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *ICCV*, 2017. 2

[32] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. 2

[33] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *ICCV*, 2021. 2

[34] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 1995. 1

[35] Barbara Roessle, Jonathan T Barron, Ben Mildenhall, Pratul P Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *CVPR*, 2022. 2

[36] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *NeurIPS*, 2019. 2

[37] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 6

[38] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 6

[39] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *NeurIPS*, 2017. 2

[40] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *NeurIPS*, 2020. 2

[41] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In *CVPR*, 2021. 2

[42] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *CVPR*, 2022. 2

[43] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *NeurIPS*, 2021. 2

[44] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, 2021. 2

[45] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *ICCV*, 2021. 2

[46] Ju Xu and Zhanxing Zhu. Reinforced continual learning. *NeurIPS*, 2018. 2

[47] Zike Yan, Yuxin Tian, Xuesong Shi, Ping Guo, Peng Wang, and Hongbin Zha. Continual neural mapping: Learning an implicit scene representation from sequential observations. In *ICCV*, 2021. 1, 2

[48] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *NeurIPS*, 2021. 2

[49] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *IROS*, 2021. 2

[50] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *ICCV*, 2021. 2

[51] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 2

[52] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 2

[53] Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Nerfusion: Fusing radiance fields for large-scale scene reconstruction. In *CVPR*, 2022. 2

[54] Na Zhao and Gim Hee Lee. Static-dynamic co-teaching for class-incremental 3d object detection. In *AAAI*, 2022. 1, 2, 5

Supplementary Material

A. Our Method

Random Inquirer. As we explained in the main paper, our random inquirer has to obey several dataset-specific constraints. The object-centric datasets such as NeRF-synthetic360 and NeRF-real360 [27] satisfy: (1) all cameras point toward the origin in the global coordinate system, and (2) the distance from the camera center to the origin is a constant number. According to (1), once the camera center position $\mathbf{x}_c = (x, y, z)$ is known, the camera rotation representation, Euler angles (α, β, γ), are also determined. Based on (2), the camera matrix further reduces one freedom degree. We thus randomly generate the camera centers on a hemisphere, and then compute the full camera matrices based on the generated camera centers. For indoor-scene datasets (camera facing-outwards) such as ScanNet [9] where the camera moves along a trajectory, we store the range of six values in the camera matrix, *i.e.*, $r_t = (x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}, \alpha_{min}, \alpha_{max}, \beta_{min}, \beta_{max}, \gamma_{min}, \gamma_{max})$, at each time step t. Specifically, 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.

B. Experiments

Implementation Details. Our coarse networks F_c and fine networks F_f have the same structure, which 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 coarse network and $64 + 128$ points per ray for the fine network. For datasets with white backgrounds, such as NeRF-synthetic360 [27], we crop the center of images for sampling rays in the first 500 iterations of the first time step. This is to avoid the non-convergence problem caused by sampling a large number of rays on white backgrounds. We adopt AlexNet [17] to compute LPIPS.

B.1. Comparison

We only present quantitative results for three scenes $(Chair, Lego, Ship)$, and the average performances over all eight scenes of the NeRF-synthetic360 [27] dataset in Tab. 1 of the main paper due to the limited space. We further show the quantitative results for all eight scenes in Tab. 4. We can see that our approach consistently outperforms the baseline on the test data of previous time steps $\mathcal{D}^0$ to $\mathcal{D}^2$ and also achieves better performance for the average results.

B.2. Visualization

As shown in Fig. 6 to Fig. 11, we show the visualization results of the baseline and ours on six scenes of three datasets. We can conclude that our method consistently alleviates the catastrophic forgetting problem in all scenarios, *i.e.*, synthetic object-centric, real-world object-centric, and real-world large-scale (camera facing outwards) scenarios.

B.3. Rendered Video

We further provide rendered videos on ScanNet [9] dataset in 'scannet101.mp4' and 'scannet241.mp4' files, where 'NeRF' and 'Ours' denote the results from baseline and our model, respectively. The videos are rendered views from $\mathcal{D}^0$ to $\mathcal{D}^9$. We can see from the comparison that the baseline (NeRF [27]) suffers from the catastrophic forgetting problem, resulting in poor-quality videos with severe artifacts including noise and blur. In comparison, our approach greatly mitigates this problem with the proposed teacher-student pipeline.

B.4. Ablation Study

We provide more ablation studies on the ScanNet dataset in Tab. 5. The 'w/o filter' and 'w/o init' denote our model without the uncertainty module or the initialization (Discussed in Sec. 4.4 of the main paper). We can see that the model without the uncertainty-based filter still achieves reasonable performance on the ScanNet dataset. This is because that the random inquirer already has some prior range when generating camera pose, which can be regarded as a 'prior filter'. The model without initialization decreases by 26.3% in PSNR compared to the full model on average. The drop on the average scores is mainly caused by the degraded performance on the test data of previous time steps $\mathcal{D}^0$ to $\mathcal{D}^8$. This verifies the effectiveness of the initialization in preserving previously learned knowledge.

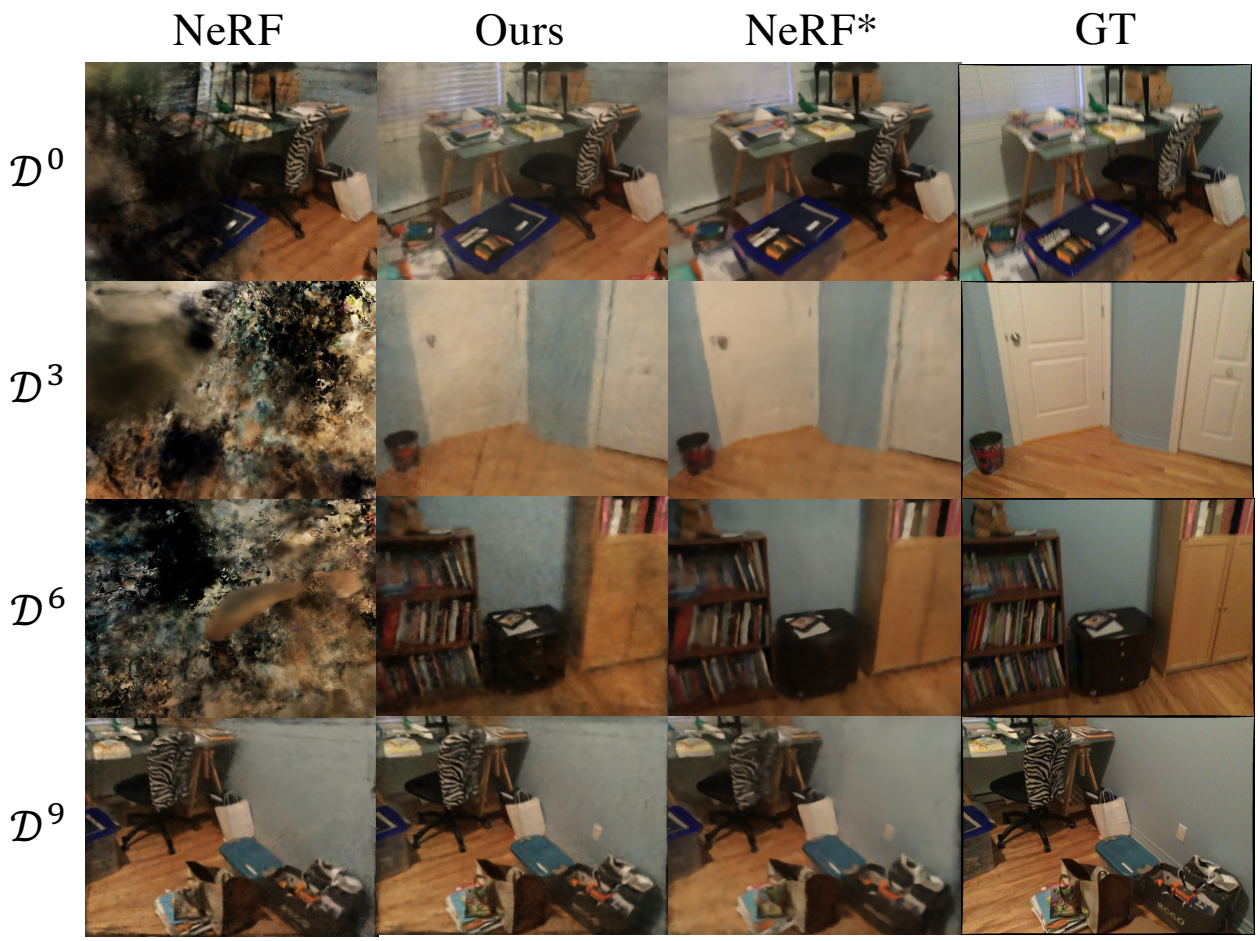

Figure 6. Qualitative comparison with the baseline 'NeRF' and upper bound batch training method 'NeRF*' on scene 241 of ScanNet dataset. 'GT' represents the ground truth images and $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ denote the results from each time step test views. 'NeRF' and 'Ours' models are incrementally trained on the 10-step training datasets.

Scene	Method	Test Dataset (PSNR ↑ /SSIM ↑ /LPIPS ↓)				
		$\mathcal{D}^0$	$\mathcal{D}^1$	$\mathcal{D}^2$	$\mathcal{D}^3$	Average on $\mathcal{D}^{0:3}$
Chair	NeRF	25.51 / 0.935 / 0.066	24.88 / 0.926 / 0.070	27.32 / 0.943 / **0.038**	**32.14 / 0.975 / 0.018**	27.46 / 0.945 / 0.048
	w/o filter	30.49 / 0.969 / 0.031	24.37 / 0.949 / 0.057	22.55 / 0.918 / 0.079	24.43 / 0.933 / 0.070	25.46 / 0.942 / 0.059
	w/o init	28.36 / 0.955 / 0.056	27.83 / 0.948 / 0.055	26.14 / 0.915 / 0.080	28.22 / 0.944 / 0.064	27.64 / 0.941 / 0.064
	Ours	**31.29 / 0.973 / 0.027**	**31.62 / 0.973 / 0.025**	**28.13 / 0.947** / 0.044	27.07 / 0.939 / 0.057	**29.53 / 0.958 / 0.038**
Drums	NeRF	20.71 / 0.896 / 0.077	19.66 / 0.881 / 0.081	20.66 / 0.892 / 0.078	**23.82 / 0.934 / 0.050**	21.21 / 0.901 / 0.072
	Ours-filter	23.75 / 0.926 / 0.075	22.49 / 0.914 / 0.084	20.97 / 0.897 / 0.101	21.55 / 0.909 / 0.088	22.19 / 0.912 / 0.087
	Ours-init	22.32 / 0.907 / 0.108	21.55 / 0.901 / 0.106	21.83 / 0.901 / 0.110	22.17 / 0.910 / 0.098	21.97 / 0.905 / 0.106
	Ours	**23.95 / 0.930 / 0.071**	**23.84 / 0.931 / 0.063**	**23.19 / 0.920 / 0.072**	21.66 / 0.912 / 0.084	**23.16 / 0.923 / 0.073**
Ficus	NeRF	23.56 / 0.938 / 0.036	22.89 / 0.929 / 0.038	24.61 / 0.943 / 0.032	**29.87 / 0.972 / 0.015**	25.23 / 0.946 / 0.030
	Ours-filter	**27.32 / 0.956** / 0.029	21.61 / 0.924 / 0.070	14.95 / 0.885 / 0.143	23.78 / 0.941 / 0.050	21.92 / 0.927 / 0.073
	Ours-init	22.29 / 0.912 / 0.065	21.91 / 0.916 / 0.063	21.19 / 0.907 / 0.071	25.54 / 0.939 / 0.045	22.73 / 0.919 / 0.061
	Ours	27.11 / 0.955 / **0.026**	**27.15 / 0.958 / 0.027**	**25.62 / 0.954 / 0.030**	24.91 / 0.949 / 0.034	**26.20 / 0.954 / 0.029**
Hotdog	NeRF	31.05 / 0.969 / 0.029	27.00 / 0.958 / 0.042	30.24 / 0.971 / 0.028	**37.68 / 0.988 / 0.010**	31.49 / 0.972 / 0.027
	Ours-filter	31.84 / 0.971 / 0.037	22.95 / 0.941 / 0.080	16.42 / 0.909 / 0.111	21.66 / 0.942 / 0.086	23.22 / 0.941 / 0.079
	Ours-init	31.30 / 0.968 / 0.044	27.66 / 0.963 / 0.043	32.03 / 0.976 / 0.031	34.39 / 0.979 / 0.028	31.35 / 0.972 / 0.037
	Ours	**33.89 / 0.978 / 0.027**	**30.36 / 0.975 / 0.026**	**35.04 / 0.985 / 0.017**	34.46 / 0.982 / 0.024	**33.44 / 0.980 / 0.024**
Lego	NeRF	24.99 / 0.917 / 0.049	23.15 / 0.896 / 0.072	25.31 / 0.930 / 0.049	**34.23 / 0.981 / 0.011**	26.92 / 0.931 / 0.045
	w/o filter	26.11 / 0.933 / 0.044	13.93 / 0.842 / 0.127	13.64 / 0.834 / 0.170	21.13 / 0.919 / 0.086	18.70 / 0.882 / 0.107
	w/o init	19.34 / 0.840 / 0.141	15.85 / 0.756 / 0.241	19.88 / 0.843 / 0.150	30.52 / 0.962 / 0.030	21.40 / 0.850 / 0.141
	Ours	**28.12 / 0.949 / 0.031**	**29.81 / 0.962 / 0.025**	**30.25 / 0.965 / 0.026**	31.94 / 0.970 / 0.022	**30.03 / 0.962 / 0.026**
Materials	NeRF	23.32 / 0.919 / 0.045	21.12 / 0.894 / 0.056	24.15 / 0.922 / 0.039	**29.54 / 0.962 / 0.016**	24.53 / 0.924 / **0.039**
	Ours-filter	27.74 / 0.958 / **0.028**	21.88 / 0.925 / 0.057	19.14 / 0.895 / 0.098	21.36 / 0.912 / 0.074	22.53 / 0.923 / 0.064
	Ours-init	23.36 / 0.913 / 0.065	22.04 / 0.903 / 0.073	24.52 / 0.921 / 0.055	27.05 / 0.947 / 0.037	24.24 / 0.921 / 0.058
	Ours	**27.88 / 0.959 / 0.028**	**26.84 / 0.953 / 0.030**	**27.41 / 0.948 / 0.033**	22.68 / 0.919 / 0.068	**26.20 / 0.945** / 0.040
Mic	NeRF	26.89 / 0.962 / 0.025	25.07 / 0.949 / 0.035	27.01 / 0.963 / 0.025	**31.96 / 0.983 / 0.011**	27.73 / 0.964 / **0.024**
	Ours-filter	**30.50 / 0.976 / 0.023**	26.40 / 0.961 / 0.037	26.80 / 0.961 / 0.041	28.97 / 0.971 / 0.032	28.17 / 0.967 / 0.033
	Ours-init	26.05 / 0.951 / 0.065	25.50 / 0.948 / 0.068	25.92 / 0.948 / 0.065	27.63 / 0.959 / 0.051	26.28 / 0.952 / 0.062
	Ours	30.40 / **0.976** / 0.025	**28.89 / 0.971 / 0.027**	**29.04 / 0.970 / 0.029**	28.95 / 0.970 / 0.034	**29.32 / 0.972** / 0.029
Ship	NeRF	24.62 / 0.874 / 0.123	21.67 / 0.826 / 0.143	25.00 / 0.880 / 0.103	**29.49** / 0.929 / **0.080**	25.20 / 0.877 / 0.112
	w/o filter	24.35 / 0.880 / 0.157	18.27 / 0.809 / 0.202	11.21 / 0.732 / 0.256	19.93 / 0.854 / 0.177	18.44 / 0.819 / 0.198
	w/o init	24.46 / 0.872 / 0.165	23.39 / 0.843 / 0.185	24.47 / 0.871 / 0.152	26.98 / 0.904 / 0.129	24.83 / 0.873 / 0.158
	Ours	**26.92 / 0.900 / 0.117**	**26.02 / 0.884 / 0.111**	**27.12 / 0.903 / 0.096**	28.49 / **0.967** / 0.097	**27.14 / 0.914 / 0.105**
Average	NeRF	25.08 / 0.926 / 0.056	23.18 / 0.907 / 0.067	25.54 / 0.931 / 0.049	**31.09 / 0.966 / 0.026**	26.22 / 0.932 / 0.050
	w/o filter	27.76 / 0.946 / 0.053	21.49 / 0.908 / 0.089	18.21 / 0.879 / 0.125	22.85 / 0.923 / 0.083	22.58 / 0.914 / 0.088
	w/o init	24.69 / 0.915 / 0.089	23.22 / 0.897 / 0.104	24.50 / 0.910 / 0.089	27.81 / 0.943 / 0.060	25.05 / 0.916 / 0.086
	Ours	**28.70 / 0.953 / 0.044**	**28.07 / 0.951 / 0.042**	**28.23 / 0.949 / 0.043**	27.52 / 0.951 / 0.053	**28.13 / 0.951 / 0.045**

Table 4. Comparison with the baseline and ablation studies on the NeRF-synthetic360 dataset measured by PSNR, SSIM, and LPIPS. We show the average and three scenes results on each step test datasets $\mathcal{D}^0$, $\mathcal{D}^1$, $\mathcal{D}^2$, $\mathcal{D}^3$ and the average performance, respectively. 'NeRF' is our baseline, "w/o filter" and "w/o init" denote the two ablations, and 'Ours' represents our full pipeline. All the models are incrementally trained on the 4-step training datasets.

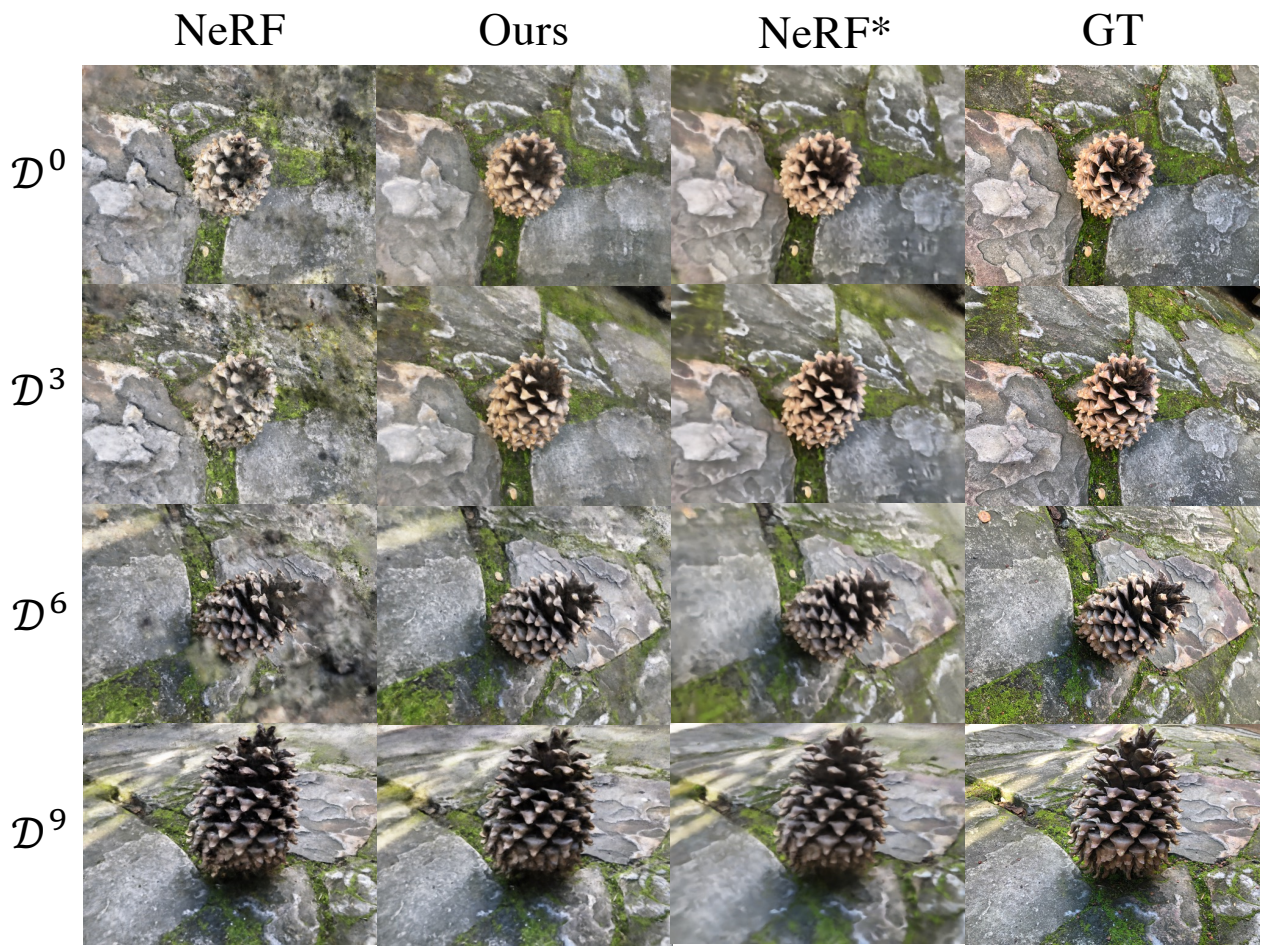

Figure 7. Qualitative comparison with the baseline 'NeRF' and upper bound batch training method 'NeRF*' on scene *Pinecone* of NeRF-real360 dataset. 'GT' represents the ground truth images and $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ denote the results from each time step test views. 'NeRF' and 'Ours' models are incrementally trained on the 10-step training datasets.

Scene	Method	Test Dataset (PSNR ↑ /SSIM ↑ /LPIPS ↓)				
		$\mathcal{D}^0$	$\mathcal{D}^3$	$\mathcal{D}^6$	$\mathcal{D}^9$	Average on $\mathcal{D}^{0:9}$
101	NeRF	12.14 / 0.580 / 0.396	12.91 / 0.597 / 0.529	18.42 / 0.763 / 0.313	**29.77 / 0.903 / 0.120**	14.83 / 0.657 / 0.403
	w/o filter	22.58 / 0.798 / 0.275	**21.46 / 0.802 / 0.335**	25.89 / 0.873 / 0.206	27.52 / 0.876 / 0.205	23.07 / 0.828 / 0.270
	w/o init	18.58 / 0.734 / 0.400	12.68 / 0.697 / 0.496	16.52 / 0.791 / 0.363	27.87 / 0.883 / 0.175	15.18 / 0.729 / 0.423
	Ours	**23.53 / 0.822 / 0.230**	20.04 / 0.787 / 0.344	**26.09 / 0.877 / 0.190**	28.19 / 0.887 / 0.160	**23.08 / 0.833 / 0.248**
241	NeRF	13.07 / 0.580 / 0.396	10.27 / 0.413 / 0.612	12.87 / 0.393 / 0.627	22.12 / 0.849 / 0.169	12.74 / 0.495 / 0.517
	w/o filter	19.86 / 0.799 / 0.221	19.88 / 0.851 / 0.271	**22.00** / 0.821 / 0.267	22.39 / 0.862 / 0.166	20.80 / 0.827 / 0.256
	w/o init	19.10 / 0.741 / 0.365	19.68 / 0.862 / 0.267	19.70 / 0.775 / 0.370	21.69 / 0.831 / 0.228	19.75 / 0.772 / 0.336
	Ours	**19.95 / 0.801 / 0.217**	**20.38 / 0.863 / 0.248**	21.70 / **0.824 / 0.257**	**22.41 / 0.865 / 0.163**	**21.02 / 0.833 / 0.244**
Average	NeRF	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
	w/o filter	21.22 / 0.799 / 0.248	**20.67 / 0.827** / 0.303	**23.95** / 0.847 / 0.237	24.96 / 0.869 / 0.186	21.94 / 0.828 / 0.263
	w/o init	18.84 / 0.738 / 0.383	16.18 / 0.780 / 0.386	18.11 / 0.783 / 0.367	24.78 / 0.857 / 0.202	17.46 / 0.750 / 0.380
	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	**22.05 / 0.833 / 0.246**

Table 5. Comparison with the baseline on ScanNet dataset measure by PSNR, SSIM, and LPIPS. We show the average and two scenes results on each step test datasets $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ and the average performance, respectively. All the models are incrementally trained on the 10-step training datasets.

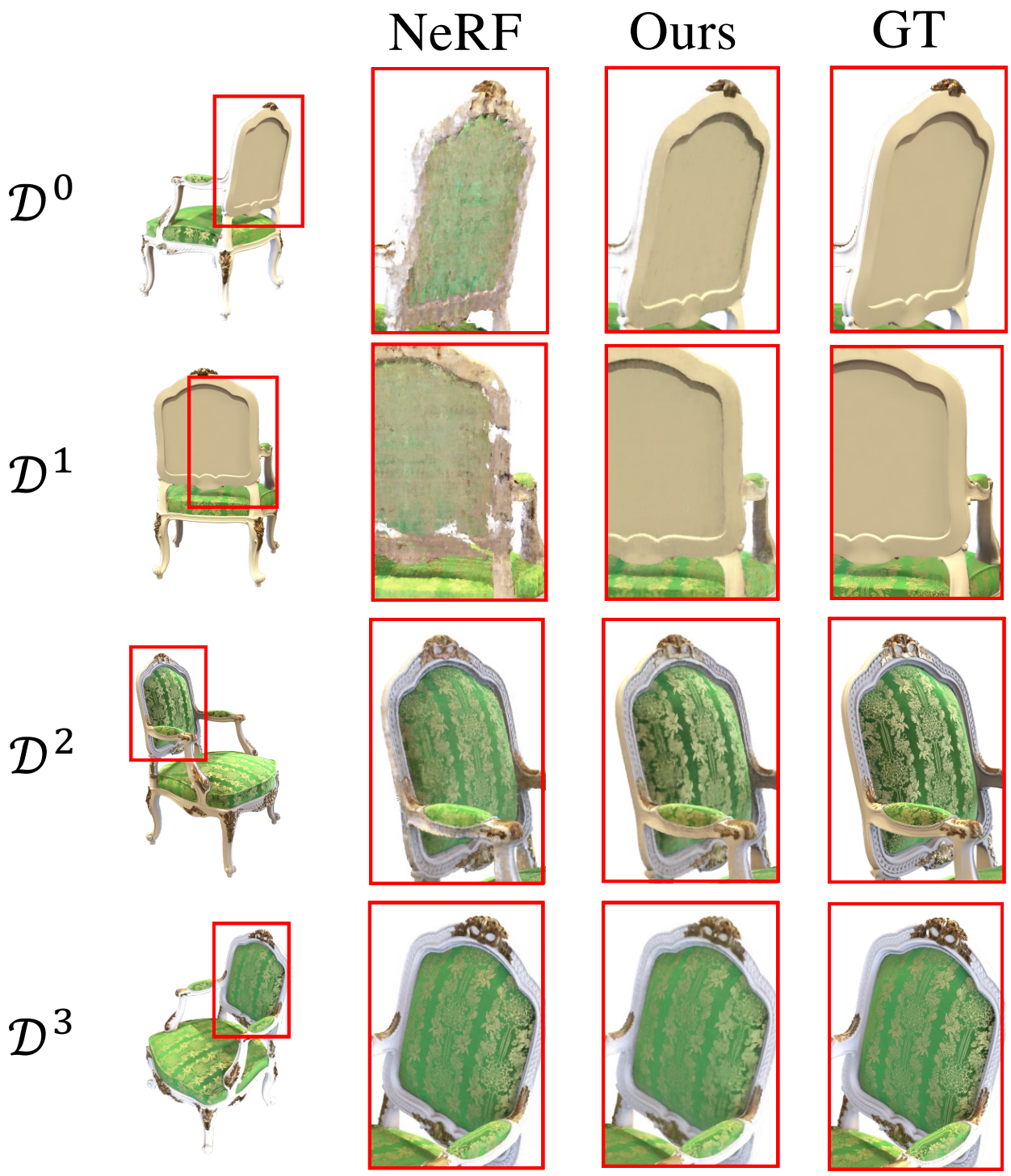

Figure 8. Qualitative comparison with the baseline 'NeRF' on scene $Chair$ of NeRF-synthetic360 dataset. 'GT' represents the ground truth images and $\mathcal{D}^0, \mathcal{D}^1, \mathcal{D}^2, \mathcal{D}^3$ denote the results from each time step test views. 'NeRF' and 'Ours' models are incrementally trained on the 4-step training datasets.

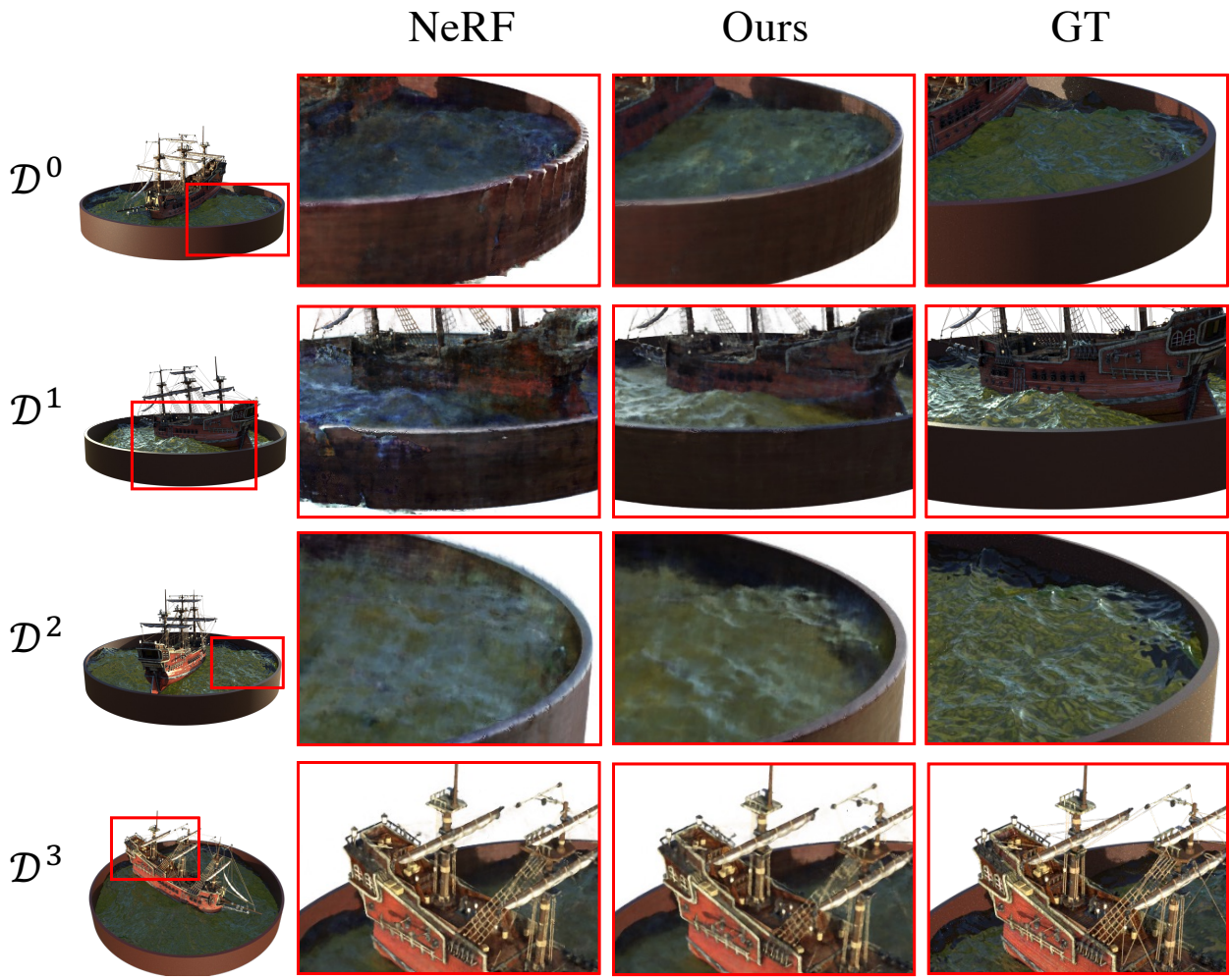

Figure 9. Qualitative comparison with the baseline 'NeRF' on scene *Ship* of NeRF-synthetic360 dataset. 'GT' represents the ground truth images and $\mathcal{D}^0, \mathcal{D}^1, \mathcal{D}^2, \mathcal{D}^3$ denote the results from each time step test views. 'NeRF' and 'Ours' models are incrementally trained on the 4-step training datasets.

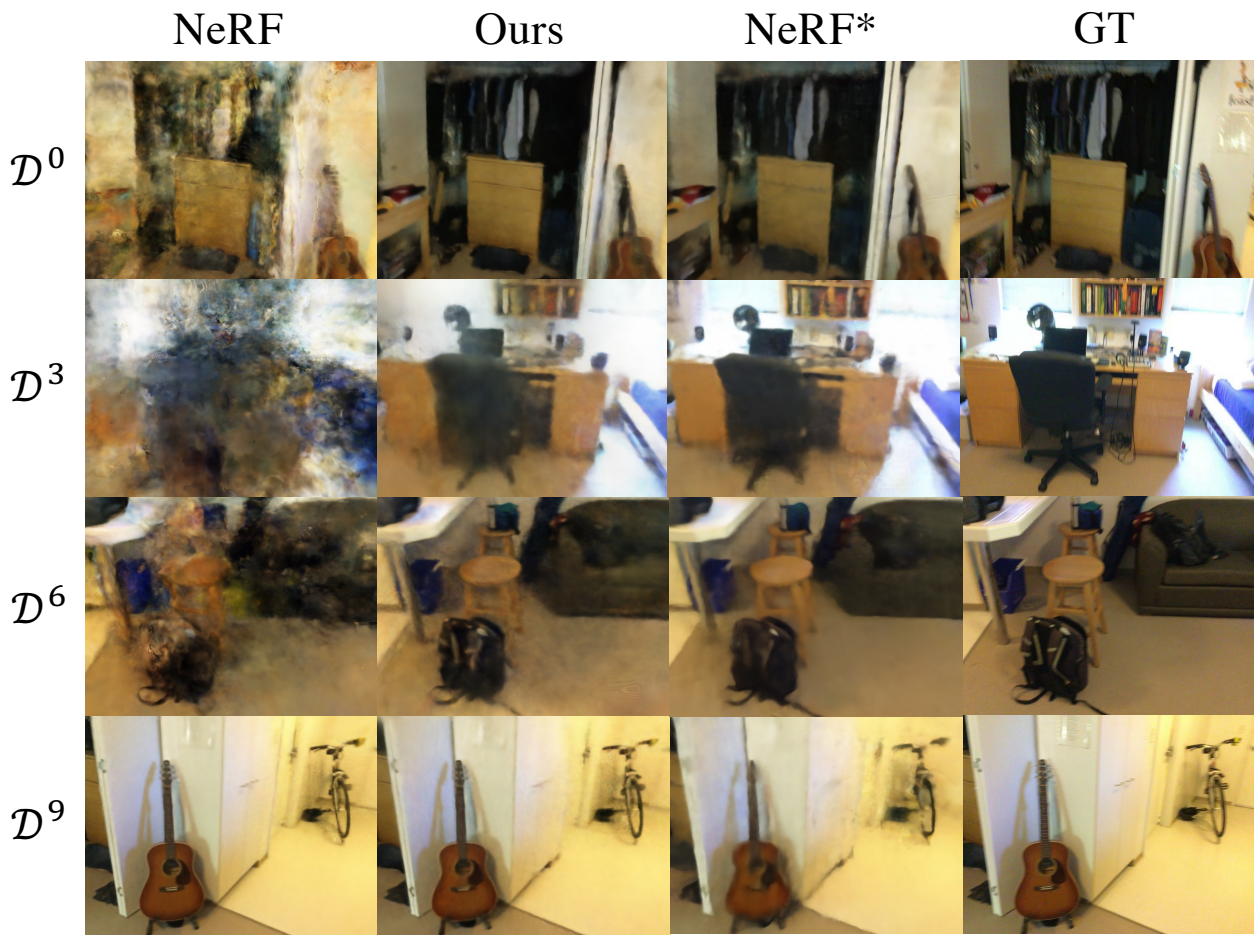

Figure 10. Qualitative comparison with the baseline 'NeRF' and upper bound batch training method 'NeRF*' on scene 101 of ScanNet dataset. 'GT' represents the ground truth images and $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ denote the results from each time step test views. 'NeRF' and 'Ours' models are incrementally trained on the 10-step training datasets.

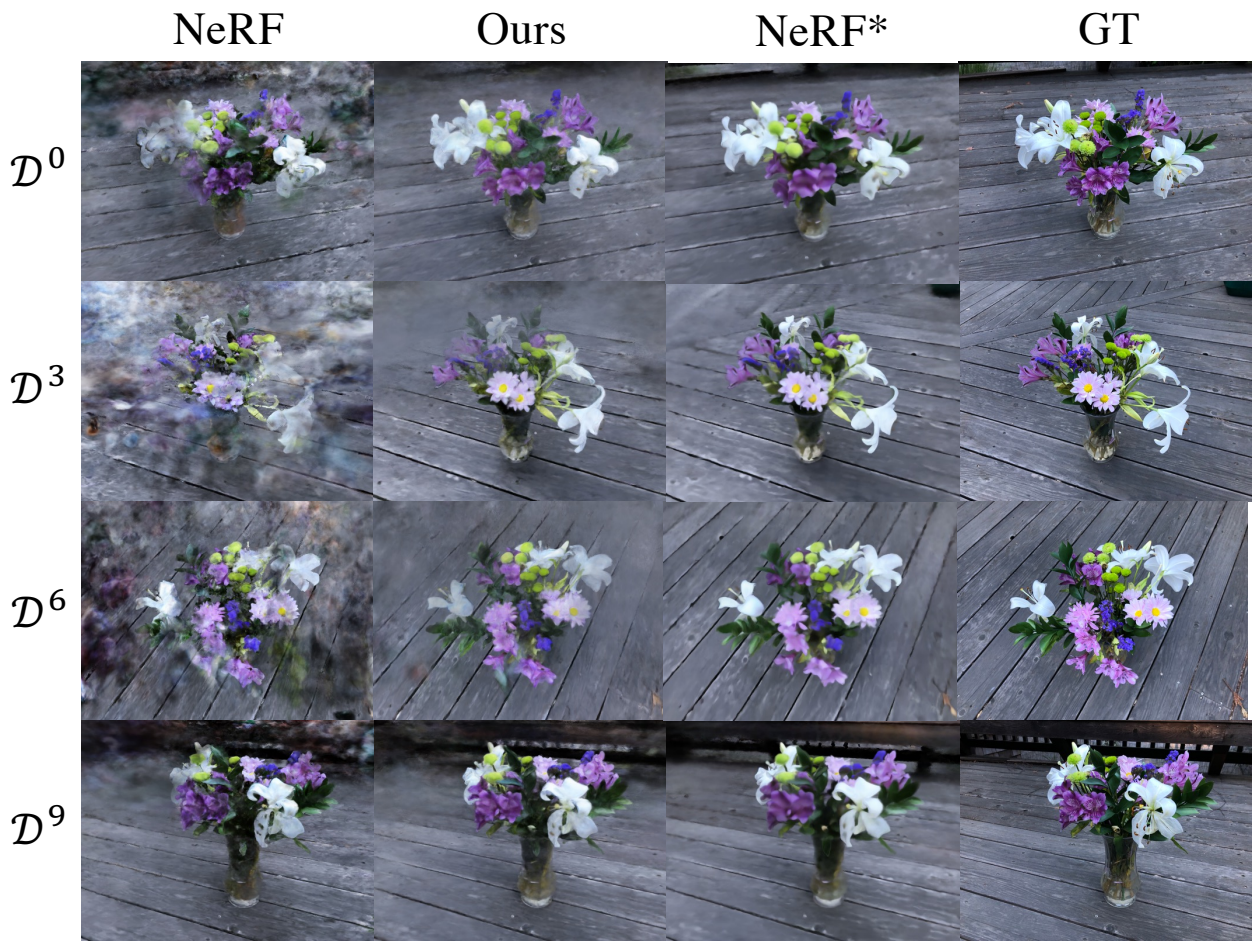

Figure 11. Qualitative comparison with the baseline 'NeRF' and upper bound batch training method 'NeRF*' on scene $Vasdeck$ of NeRF-real360 dataset. 'GT' represents the ground truth images and $\mathcal{D}^0, \mathcal{D}^3, \mathcal{D}^6, \mathcal{D}^9$ denote the results from each time step test views. 'NeRF' and 'Ours' models are incrementally trained on the 10-step training datasets.