2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Incremental 3D Reconstruction through a Hybrid Explicit-and-Implicit Representation

Feifei Li, Panwen Hu, Qi Song, Rui Huang†

Abstract—
3D reconstruction is an important task in computer vision and is widely used in robotics and autonomous driving. When building large-scale scenes, limitations in computing resources and the difficulty of accessing the entire dataset in a single task are inevitable. Therefore, an incremental reconstruction approach is desired. On the one hand, traditional explicit 3D reconstruction methods such as SLAM and SFM require global optimization, which means that time and space resources increase dramatically with the growth of training data. On the other hand, implicit methods like Neural Radiation Fields (NeRF) suffer from catastrophic forgetting if trained incrementally. In this paper, we incrementally reconstruct 3D models in a hybrid representation, where the density of the radiation field is formulated by a voxel grid, and the view-dependent color information of the points is inferred by a shallow MLP. The expansion of the voxel grid and the distillation of the shallow MLP are efficient in this case. Experimental results demonstrate that our incremental method achieves a level of accuracy on par with approaches employing global optimization techniques.

I. Introduction

3D scene reconstruction from given images and corresponding camera poses is a complex and long-standing problem in the field of computer vision. There are two mainstream approaches to 3D scene reconstruction: 1) explicit novel view synthesis methods, such as SLAM [1]–[3], which try to match key points in adjacent frames and globally optimize the 3D positions of these points to get an explicit 3D model. 2) implicit novel view synthesis approaches, like Neural Radiation Fields (NeRF), which create an implicit representation of the 3D scene with the utilization of neural networks. However, both of these approaches are confronted with a fundamental challenge—the memory demand significantly rises as the scene size scales up and the representation power is inherently constrained. Consequently, an incremental reconstruction approach is desired.

A recently proposed NeRF-based reconstruction work Block-NeRF [4], as illustrated in Fig. 1(a), splits the large-scale scene into multiple blocks and utilizes independent NeRFs to jointly represent the entire scene, thereby alleviating the limitations in representation power. But the rendering time increases dramatically. Alternatively, MEIL-NeRF, as depicted in Fig. 1(b), leverages the knowledge obtained from selected rays in previous tasks through knowledge distillation to mitigate the catastrophic forgetting issue, which arises when aggregating the existing models to encompass the entire region of interest. Nonetheless, the forgetting problem still persists, as rendering errors tend to accumulate with each new task. In other words, the student model (the orange block in Fig. 1(b)) may acquire inaccurate knowledge from the teacher model (the blue block in Fig. 1(b)). To avoid accumulated errors, iMAP [5] saves frames with accurate depth estimation as keyframes, which retain cues about frequently occurring parts of the training data. Nevertheless, iMAP runs the risk of forgetting information about other regions with poor depth estimation and also increases memory usage.

In this paper, our objective is to develop an effective framework that addresses the challenges outlined above. Firstly, considering that incremental construction of explicit voxel grids offers notable advantages in updating weights of the current task without affecting the parameters of irrelevant previous tasks, we employ an explicit voxel grid to effectively represent the scene characteristics as well as prevent the forgetting issue. As illustrated in Fig. 1(c), we can seamlessly expand the existing voxel grid to a larger one with the newly added data. After that, in order to reduce computational costs, we choose a relatively low solution voxel grid and utilize shallow MLPs to predict colors. Meanwhile, we employ knowledge distillation to continuously improve the performance of these shallow MLPs, which also achieves a good trade-off between accuracy and efficiency.

To mitigate the accumulated errors caused by knowledge distillation in the shallow MLPs, we further propose to preserve frames that are geometrically significant. In particular, unlike existing methods that retain frames based on training loss, we select the frames that encompass a greater number of newly added valid voxels as keyframes in our work. We follow a two-stage coarse-to-fine training strategy, similar to the one described in [6], and mark empty voxels as invalid during the coarse stage. The entire pipeline is supervised by the pixel colors like NeRF, which involves accumulating color points along the ray.

In summary, the main contributions of this work are as follows:

- A hybrid 3D incremental reconstruction pipeline is proposed to represent 3D scenes with voxel grids and neural networks.
- A new keyframe selection method is put forward using the explicit voxel grid, which is not available in implicit representation methods.
- A new continual reconstruction method optimized through random distillation and keyframe replay is raised in this work.

The authors are with School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, Guangdong, 518172, China. This work was supported in part by Shenzhen Science and Technology Program under Grants JCYJ20220818103006012 and ZDSYS20211021111415025.

† Corresponding author, ruihuang@cuhk.edu.cn

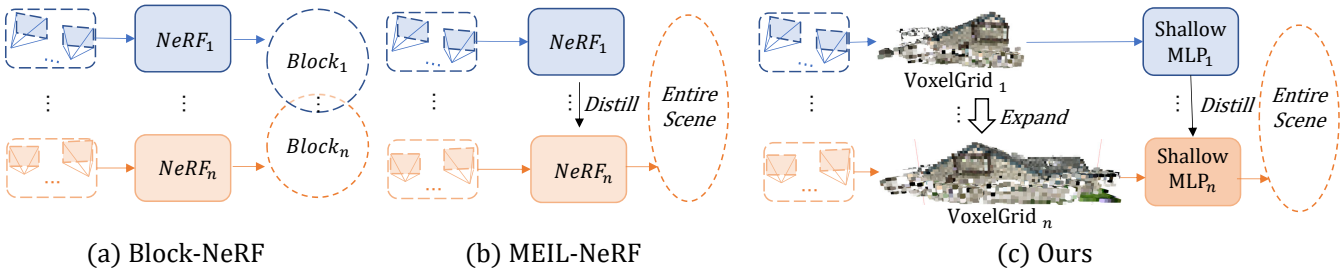

Fig. 1. Comparison of different incremental reconstruction methods.(a). Block-NeRF: Training several NeRFs in the geometric blocks of the scene. The joint section of multiple blocks is rendered by the weighted average of the NeRFs of blocks; (b). MEIL-NeRF: The training data is sequentially divided into different tasks, and the current model (NeRF) is built using the knowledge distilled from the previous models; (c). Our proposed method: a voxel grid is constructed to represent density, color, and other features. A shallow MLP is trained to estimate orientation-dependent colors given the features from the voxel grid. Incremental reconstruction is conducted through voxel grid expansion and MLP distillation.

II. Related Work

A. Novel view synthesis

Given a set of images and corresponding camera poses, Novel View Synthesis (NVS) attempts to render novel images from novel viewpoints. Many works try to solve this problem by reconstructing 3D models [7]–[9] or implicitly representing the position of color&density mapping through deep neural networks [6], [10], [11].

Explicit novel view synthesis. Traditional explicit novel view synthesis methods use mesh [12], [13] or light field [14] to build 3D models and obtain novel views by interpolating input images. Recently, convolutional neural networks (CNN) have been used to estimate voxel grids [15]–[17], and some progress has been made in accuracy and efficiency. However, these methods either require global optimization or are very time-consuming.

Implicit novel view synthesis. Neural Radiance Fields (NeRF) [10] is one of the most promising implicit novel view synthesis methods in recent years. NeRF uses MLP to represent the density and color of the 3D point sampled from the rays transmitted from the camera to the scene. In this case, NeRF can be seen as the end-to-end mapping from locations & directions to color & density. Several works [4], [18]–[22] try to build NeRF in large-scale outdoor scenes and solve problems like occlusion by dynamic objects, and memory limit when representing large-scale scenes. To reach the purpose of light-weight and real-time rendering, [5], [23]–[26] try to reform the architecture of NeRF and achieve some progress. Generally, the representational power of implicit novel view synthesis is limited by the number of the parameters.

Mixed novel view synthesis. Some other works try to combine the use of explicit and implicit structures to gain the advantages of both approaches. NSVF [11] uses deep neural networks to represent sparse voxel fields. The entire process is differentiable and more efficient because points from empty space are not sampled for rendering. DVGO [6] further simplifies the representation using a shallow MLP to map the viewpoint-dependent colors of sample points. Our proposed method is based on a hybrid approach since incremental construction is easier and more reliable for a 3D voxel grid than deep MLP. We apply continuous learning in shallow MLP to regress density and color.

B. Continual learning in 3D reconstruction

Continuous learning. Continuous learning is a fundamental problem in machine learning, where multiple tasks are trained sequentially, and data from previous tasks is unavailable for the current task. The purpose of continuous learning is to transfer knowledge from previous tasks to the current task. Neural networks often face catastrophic forgetting when learning tasks in a sequential manner. Many methods have been proposed to solve this problem. Knowledge distillation [27]–[29], first proposed by [30], uses knowledge from a collection of expert models to train a model. Elastic Weight Consolidation (EWC) [31]–[33] selectively slows down the update rate of important weights to avoid catastrophic forgetting. PackNet [34] uses network pruning to free up some parameters for use by new tasks. Therefore, PackNet packs all parameters of different tasks into the entire network and uses filters to select the corresponding parameters for evaluation.

Incremental 3D reconstruction. Many works apply continual learning to 3D reconstruction to achieve incremental building. MEIL-NeRF [35] uses knowledge distillation to transfer knowledge from previous to current tasks. CLNeRF [36] uses replay buffer for continual learning. iMAP [5] saves the keyframes of previous tasks and uses these keyframes to optimize subsequent tasks. Other works, such as Block-NeRF [4] and Switch-NeRF [20], divide large-scale scenes into multiple physical blocks or implicit expert systems represented by NeRF. These methods either require a lot of computing resources or take a long time to train. Instead, our proposed method only requires a low-resolution voxel grid and a shallow MLP.

III. Preliminaries

In this section, we introduce the principles of Neural Radiation Fields (NeRF) and DVGO [6]. NeRF has achieved great success by using multi-layer perceptrons (MLPs) for implicit novel view synthesis. Based on the pose and orientation of the input view, the colors and densities of sampled points

along the ray from the camera to the 3D scene are estimated. The principle of NeRF can be explained as:

$$\sigma, c = MLP_{\Theta}(\boldsymbol{x}, \boldsymbol{d}) \tag{1}$$

where $\boldsymbol{x}$ denotes the position of the sampled point, $\boldsymbol{d}$ represents the direction of rays, σ represents the density. Density and color are implicitly represented by two deep MLPs.

However, the ray marching process in NeRF is time-consuming. To address this problem, NSVF [37] designs a hybrid model of the explicit hierarchical octree structure and an implicit deep MLP to render the novel views. This architecture enables us to self-prune unnecessary empty voxels. A threshold density is set to remove the empty voxels. This will significantly reduce the training and rendering time. As claimed in the original paper, NSVF is over 10 times faster than NeRF for rendering time. However, NSVF still uses deep MLPs as the implicit representation, which is slow to train.

To further reduce the training and rendering time, DVGO [6] proposes a more efficient direct voxel grid optimization approach. For DVGO,

$$\begin{aligned} \sigma &= interp(\boldsymbol{x}, VG^{(dens)}) \\ c &= MLP_{\Theta}(interp(\boldsymbol{x}, VG^{(feat)}), \boldsymbol{x}, \boldsymbol{d}) \end{aligned} \tag{2}$$

where VG is the voxel grid of density or implicit feature. Trainable parameters in this case are the voxel grid and the MLP_{θ}.

IV. Method

A. Problem statement

We discuss two main incremental-related situations that would usually happen in real-world 3D reconstruction:

- **Q1. Sequential Input.** Assuming all the data of the entire scene has been gathered, due to the constraints imposed by limited computing resources and/or real-time requirements, it becomes imperative to partition the dataset into multiple sequences and continuously train on each sequence.
- **Q2. Continual Build.** Assuming a partial representation of the scene has been constructed, we now have access to the remaining data pertaining to the scene. This new data is utilized to augment the existing model and construct a comprehensive representation of the entire scene without necessitating the retrieval of training data from previous parts of the scene.

The two problems are similar in that there is a requirement to transfer knowledge from an existing model to a new model without accessing the existing model's training data. In this scenario, when the dataset of **Q1** is partitioned into N sequences, **Q1** can transition into an N-step instant reconstruction problem. Meanwhile, **Q2** can be regarded as a specific instance with N = 2, wherein the emphasis lies on the rendering quality and the issue of catastrophic forgetting of the data in task 1 rather than real-time performance and computational resources.

B. Pipeline of our proposed method

In the NeRF method, multiple rays are sampled from the camera to pixels on the image plane. We uniformly sample multiple points to estimate the density and color along each ray. The RGB estimation for a specific pixel is obtained by summing the values of the points along the corresponding ray. To ensure computational efficiency, our approach avoids computing arbitrarily distributed points within the scene. Instead, we estimate the color and density of a grid of voxels that encompasses a specified bounding box determined by the camera pose and intrinsic matrix. For each ray sample, we uniformly sample points following the same approach as NeRF. Subsequently, we interpolate the voxel grid to obtain estimations of the color and density for these sampled points. The remaining operations in our method align with those in NeRF.

In summary, the proposed architecture is depicted in Algorithm 1, where $VG^{(co)}$ and $VG^{(fi)}$ denote the voxel grids utilized for the coarse and fine stages, respectively. The key frame list is represented by KF, with subscripts C and P referring to the current and previous tasks, respectively. Additionally, a shallow multi-layer perceptron, denoted as MLP, is employed at the fine stage. The specific details regarding each step are elaborated upon in subsequent sections of this chapter.

Algorithm 1 Training process of our approach

Require: MLP_P, $VG_P^{(co)}$, $VG_P^{(fi)}$, KF
1: **while** input new I_C&P_C **do**
2: Calculate new bound of expanded $VG_C^{(co)}$ according to P_C
3: Copy values of $VG_P^{(co)}$ into $VG_C^{(co)}$
4: Coarse stage training & update $VG_C^{(co)}$ ▷ Sec: IV-D
5: Compute KF_C ▷ Sec: IV-E
6: Initiate $VG_C^{(fi)}$ based on $VG_C^{(co)}$
7: Training MLP_C and $VG_C^{(f)}$ & distilling knowledge from MLP_P ▷ Sec: IV-F
8: Append KF_C to KF
9: Replace MLP_P, $VG_P^{(co)}$, $VG_P^{(fi)}$ by the current data.
10: **end while**

C. Coarse-to-Fine Training Method

Motivated by DVGO, our training approach follows a two-stage methodology. We initially focus on the coarse-stage reconstruction of explicit voxels, which represent density and RGB information. The scene boundary encompasses all camera frustums from the training data, defined by the 6DoF camera pose, as well as the near and far distances along the ray. To establish this boundary, we compute all rays originating from each camera, similar to the standard NeRF approach. Subsequently, we sample the near and far points along each ray, marking the cuboid boundaries with $(x_{max}, y_{max}, z_{max})$ and $(x_{min}, y_{min}, z_{min})$ respectively. Once the boundaries are determined, a voxel grid can be defined within the corresponding area, representing

the density and color information of the scene. Two key differences exist between our coarse stage and the standard NeRF training procedure. Firstly, in this paper, we query the voxel grid to obtain the density and color values for each sampled point along the ray, while NeRF implicitly regresses these values. Secondly, we do not consider the ray's orientation during the coarse voxel grid training. In the real-world scenario, rays traverse through the air before reaching surfaces, where the density of points along this part of the ray is expected to be zero. To simplify the training process, we mask the voxels whose density estimations fall below a certain threshold once the coarse model training is complete. Throughout this paper, we refer to the masked voxels as 'invalid voxels' and the remaining voxels as 'valid voxels'.

Then, we introduce direction-dependent implicit representations at fine stages. Unlike DVGO, we use the same voxel size in the fine stage as in the coarse stage and reload the density of the coarse voxels to initiate the fine voxels. We randomly rescale the voxel and interpolate values to train a more precise model as DVGO. High-dimensional features are saved in voxels instead of only color in the fine stage. A shallow MLP is used to regress view-related colors based on features. We will deepen the MLP in incremental reconstruction because the representation power should improve when the scene becomes larger.

D. Coarse stage incremental reconstruction

The incremental building method in this coarse stage consists of two parts. First, the expansion of the voxel grid when we have new training data; Second, the estimation of the density and color of the expanded voxels according to the existing voxels. For the expansion phase, the new boundary of the scene can be obtained with the same method as the existing boundary. Let $(x^P_{max}, y^P_{max}, z^P_{max})$ and $(x^P_{min}, y^P_{min}, z^P_{min})$ mark the existing boundary, Then we can calculate $(x^C_{max}, y^C_{max}, z^C_{max})$ and $(x^C_{min}, y^C_{min}, z^C_{min})$ using new training data. The new boundary can be marked as $(\max(x^C_{max}, x^P_{max}), \max(y^C_{max}, y^P_{max}), \max(z^C_{max}, z^P_{max}))$ and $(\min(x^C_{min}, x^P_{min}), \min(y^C_{min}, y^P_{min}), \min(z^C_{min}, z^P_{min}))$. Multiple cuboid areas would be added around the existing cuboid. We set the size of the new voxels the same as the existing ones and then initiate the density and color of the new voxels. When we train the coarse voxels only with new data, a significant problem is that we would update the existing grid if the new training rays go through the existing voxels, which would cause catastrophic forgetting of existing training data. To address this in the coarse model, we train the model with the saved keyframes and the new data.

E. Geometric-based keyframe selection

In the coarse model training stage, we introduce a keyframe selection method. Frames that result in rays passing through a larger number of valid voxels contain more valuable information about the scene compared to others. During training, we sample rays and points for each pair of picture and camera pose and subsequently calculate the number of valid voxels queried by these points. However, the selection of keyframes is not based solely on the absolute number of queried voxels, as adjacent frames often cover the same voxel regions with slight differences. Therefore, the frame that queries the highest number of new voxels compared to the previous keyframe among the F (sampling frequency) frames is selected as the new keyframe.

F. Fine stage incremental reconstruction

In the fine stage, a shallow MLP is designed to estimate colors based on location and orientation. In this case, even if the rays from the keyframes of the model would cover all valid voxels, the orientations of the rays sampled from the keyframes will not include all the orientations of the previous training data. To solve this problem, in addition to using keyframes to avoid catastrophic forgetting, we also use knowledge distillation to further extract information from past models. We save all camera poses of the training data for each training task for distillation. We load and fix the model from the previous task as the teacher model and distill information from saved camera poses from the teacher model. The loss function of distillation can be designed as:

$$\mathcal{L} = \mathcal{L}_C + \alpha \mathcal{L}_P \tag{3}$$

where $\mathcal{L}_P$ denotes the distilling loss from the previous task with the saved camera poses, $\mathcal{L}_C$ denotes the loss of the current task:

$$\begin{aligned} \mathcal{L}_C &= ||\boldsymbol{c} - MLP^S(interp(\boldsymbol{x_C}, V^{feat}), \boldsymbol{x_C}, \boldsymbol{d_C})|| \\ \mathcal{L}_P &= ||MLP^S(interp(\boldsymbol{x_P}, V^{feat}), \boldsymbol{x_P}, \boldsymbol{d_P}) - \\ &\qquad MLP^T(interp(\boldsymbol{x_P}, V^{feat}), \boldsymbol{x_P}, \boldsymbol{d_P})|| \end{aligned} \tag{4}$$

where T and S denote shallow MLP from the teacher and student model, respectively. $interp$ means interpolate the feature from the voxel grid. $\boldsymbol{x}$ and $\boldsymbol{d}$ represent the location of the point and the orientation of the ray. V^{feat} denotes the voxel grid of features from the fine model.

G. Different update rules for existing and current voxels

We will also discuss the rendering quality of joint regions for different tasks. There is a problem in Block-NeRF caused by the design that blocks overlap with the neighboring blocks. To get better render views, in the evaluation step of Block-NeRF, a weighted average is applied to joint sections. In this work, rays sampled from the current camera will also pass through voxels trained on previous tasks. We design different learning rates for the newly expanded voxels and the existing voxels from previous tasks in the training step. In experiments, we set the learning rate of newly added voxels to twice the learning rate of existing voxels.

V. Experiments

A. Datasets

In this paper, we mainly evaluate the proposed method on three datasets: **Replica** [38], **Synthetic-NeRF** [10], and **Tanks&Temples** [39]. Replica is a synthetic indoor dataset. Synthetic-NeRF contains eight indoor scenes with the trace

TABLE I

Quantitive results of **Q1** on different datasets, results of PackNet, NeRF-Incre are from MEIL-NeRF

	Method	Tanks and Temples						Replica						
		Barn	Caterpillar	Family	Ignatius	Truck	avg	Office0	Office3	Office4	Room0	Room1	Room2	avg
MS-SSIM	MEIL-NeRF [35]	**0.836**	**0.909**	**0.972**	**0.953**	**0.925**	**0.919**	0.948	**0.908**	**0.960**	0.831	0.914	**0.896**	0.910
	PackNet [34]	0.423	0.414	0.877	0.761	0.549	0.605	0.775	0.558	0.716	0.487	0.677	0.631	0.641
	NeRF-Incre [10]	0.394	0.657	0.890	0.747	0.649	0.667	0.679	0.508	0.649	0.552	0.558	0.499	0.574
	ours	0.804	0.889	0.946	0.933	0.886	0.892	**0.962**	0.853	0.916	**0.966**	**0.925**	0.874	**0.916**
	NeRF-Joint [10]	0.907	0.944	0.992	0.966	0.959	0.953	0.987	0.972	0.984	0.910	0.972	0.904	0.955
PSNR	MEIL-NeRF [35]	22.14	23.41	29.95	25.01	24.31	25.37	34.35	26.52	32.72	25.73	29.47	28.23	29.50
	PackNet [34]	15.67	12.77	22.91	19.99	16.76	17.62	28.68	20.36	24.54	19.28	24.53	22.55	23.32
	NeRF-Incre [10]	14.17	15.90	23.38	17.00	16.57	17.40	24.44	16.28	19.55	19.18	17.44	19.10	19.33
	ours	**25.27**	**24.66**	**31.34**	**27.06**	**25.85**	**26.84**	**36.87**	**30.25**	**35.60**	**28.32**	**32.56**	**28.63**	**32.03**
	NeRF-Joint [10]	24.51	25.91	33.89	26.59	27.12	27.60	39.99	32.85	37.67	29.35	34.36	28.79	33.84

of the camera on the sphere and the views inward-facing. Tanks&Temples is an outdoor real-world dataset containing five scenes. For each scene, the camera's trajectory surrounds the objects, and the camera's view remains oriented toward the scene. The trajectory is not on a specific circle because we have different radii. We set thresholds for different coordinates to split the scene into several blocks as in Block-NeRF [4] for different tasks.

B. Implementation details

This paper proposes two experiments that center around problems stated in Sec: IV-A. The works of MEIL-NeRF and iMAP are related to answering **Q1**, where the training data is partitioned into multiple parts to fulfill the demand for instantaneous reconstruction. Conversely, Block-NeRF and Switch-NeRF address the same issue as **Q2**, which involves the need to split a dataset because the representational capacity of a single model is limited.

To simulate problem **Q1**, we partition the dataset into 20 tasks, following the approach used in MEIL-NeRF. Although a single NeRF's ability can cover a larger scene, these datasets still allow us to assess the incremental performance of various architectures. We propose setting the voxel number to 102400 for the initial task and maintaining a constant voxel size for subsequent tasks, expanding voxel grid expansion without altering existing voxels. The training process for the fine stage can be divided into two parts: the coarse-to-fine process and the incremental build process. During the coarse-to-fine phase, we transfer the voxel grid size and density from the coarse phase. We update the parameters of the fine-stage MLP using both training data and knowledge distilled from previous tasks. If model expansion exceeds the existing parameters' representation ability, we can deepen the MLP.

In **Q2**, we evaluate our proposed method's performance by splitting each scene's camera pose into two halves based on the x-axis midpoint of the effective voxel grid (as explained in Keyframe Selection). This method of splitting the dataset with a plane is similar to Block-NeRF. We first train an arbitrary half of the data as the initial task, followed by training the second task with the remaining data and distilled information from task one. When estimating views in a joint region, only the model from task two is utilized, contrasting with Block-NeRF, where both blocks are used for joint estimation.

As Block-NeRF's code is currently unavailable, we implement a simplified version named Block-NeRF* using standard NeRF [10]. Our aim is to compare Block-NeRF's representation of joint regions with that of our proposed model, disregarding their representation power within a single block.

C. Results

The numerical results of incremental reconstruction on the Synthetic-NeRF and Tanks&Temples datasets are presented in Tab. I. The 'NeRF-Joint' case, which involves training the entire scene as one task, serves as an upper bound for **NeRF-based methods** like MEIL-NeRF. Conversely, the 'NeRF-Incre' case, which involves training multiple NeRFs without auxiliary modules to retain prior knowledge, serves as a lower bound. Our proposed method surpasses MEIL-NeRF in terms of PSNR across all scenes of Tanks&Temples and outperforms MEIL-NeRF in terms of PSNR and SSIM in most scenes of TUM-RGBD. In scene 'Ignatius', our method even exceeds NeRF-Joint. MEIL-NeRF surpasses our method in SSIM on Tanks&Temples due to our choice of a relatively low-resolution voxel grid. We refrain from enhancing the voxel size as the PSNR is already competitive. Other continual learning methods mentioned in MEIL-NeRF [35] (e.g., iMAP and EWC [31]) are less competitive than MEIL-NeRF, as reported in MEIL-NeRF, hence their results are not included.

The comparison of rendering quality in the joint areas between two adjacent tasks using the weighted average method from Block-NeRF and our proposed hybrid-incremental reconstruction method is depicted in Fig. 2. We train a standard NeRF for each block (or task) and render the views of overlayed sections using the weighted average of the results from both blocks as in Block-NeRF. For our proposed method, we treat the first block as Task 1 and incrementally train the entire two blocks only using data from the second block. The results indicate that Block-NeRF*'s weighted average method may result in blurs and over-smoothing, which can

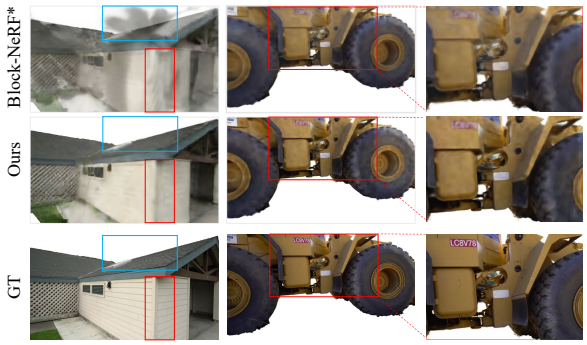

Fig. 2. Visualization results of our proposed method and block-NeRF* on the Tanks&Temple dataset **Q2**. We select views in the joint region of different blocks (also a task in our approach). In order to achieve the requirement of representing the entire scene with different blocks, for each scene, we divide the data into two blocks by the coordinate x. Each block may contain some incomplete parts of an object or scene.

also be found in the official results published on the Block-NeRF website. In our proposed method, we use voxels of varying resolutions for comparison. The numerical results are presented in Fig 3.

D. Ablation studies

This section presents an evaluation of the effectiveness of the different submodules within the proposed pipeline. The results, as depicted in Table II, showcase various cases. The case denoted as 'hybrid+joint' represents the utilization of the hybrid module to process all data, serving as an upper bound for comparison. In contrast, 'hybrid+incre' refers to incremental training of the hybrid model without retaining knowledge from previous tasks. 'Hybrid+dist' explores the inclusion of distillation during the fine-stage, while 'hybrid+kf' involves the preservation of keyframes to retain previous scene information. Finally, 'hybrid+dist+kf' represents our comprehensive method incorporating both distillation and keyframes. The findings from the table indicate the effectiveness of both distillation and keyframes in our method. The 'hybrid+joint' case, utilizing all data for a single task, demonstrates the upper bound performance. On the other hand, 'hybrid+incre' signifies incremental training without retaining knowledge from previous tasks. It is observed that both the distillation method and the use of keyframes contribute to retaining previous scene information and improving rendering quality for novel views.

Comparison of Keyframe Selection Methods. We conducted a comparative analysis involving the proposed maximum-valid-voxel-keyframe method and alternative approaches, including non-keyframe, uniform-selection, and random-selection methods. In the case of 'hybrid+dist', the model is trained without incorporating any keyframes. 'Hybrid+dist+kf-u' involves the uniform sampling of keyframes from the training data at a predetermined rate. Alternatively, 'hybrid+dist+kf-r' randomly selects a specific number of frames as keyframes. Lastly, in 'hybrid+dist+kf', the keyframe is chosen from the coarse stage and specifically projected to the most effective voxel.

TABLE II

PSNR of different implementations on Tanks&Temples dataset.

Method	Scenes				
	Barn	Caterpillar	Family	Ignatius	Truck
hybrid+joint	26.04	25.05	32.32	27.91	26.37
hybrid+incre	22.32	23.21	26.83	25.30	25.38
hybrid+dist	23.94	23.07	29.56	26.40	25.60
hybrid+kf	22.85	24.48	31.11	26.47	25.66
hybrid+dist+kf	**25.83**	**24.69**	**31.41**	**27.13**	25.80
hybrid+dist+kf-u	25.27	24.66	31.34	27.06	25.85
hybrid+dist+kf-r	25.80	24.58	31.37	26.92	**25.91**

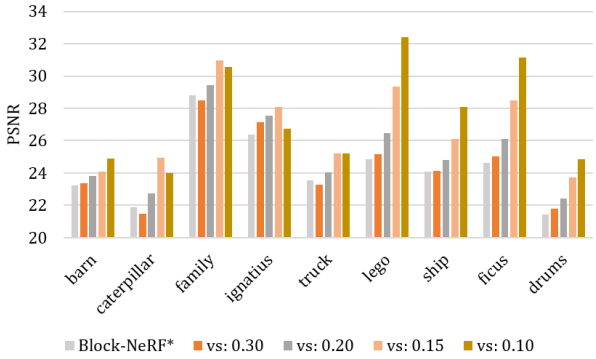

Fig. 3. Numerical comparison of Block-NeRF* and our proposed method implemented with different resolutions of voxel grids. vs here refers to the voxel size. These nine scenes are from Tanks&Temple and Synthetic-NeRF.

Comparison of Different Voxel Grid Resolutions. In order to ensure high rendering quality, we propose a strategy for setting the total number of voxels based on the principle that ray samples originating from the same frame should not traverse the same voxel within the grid. This approach can be considered as the lower bound, as it strives to prevent conflicts in color estimation by avoiding situations where rays from the same frame pass through the same voxel. Fig 3 presents the numerical results supporting the abovementioned findings.

VI. Conclusion

This paper presents a novel hybrid 3D incremental reconstruction method designed to address the challenge of large-scale 3D incremental reconstruction. Our approach combines direct expansion techniques applied to explicit voxel grids with knowledge distillation applied to implicit neural networks. In addition, we propose a new method for selecting keyframes based on voxel grids, and we compare its performance against random selection methods and uniform selections. To evaluate the effectiveness of our method, we conduct experiments comparing its performance with two existing approaches: Block-NeRF, which focuses on large-scale scene reconstruction, and MEIL-NeRF, which addresses instant scene reconstruction. Furthermore, we investigate the relationship between memory consumption and rendering quality by employing voxel grids of varying resolutions.

REFERENCES

[1] S. Thrun, "Simultaneous localization and mapping," in *Robotics and cognitive approaches to spatial mapping*. Springer, 2008, pp. 13–41.

[2] Q. Fu, H. Yu, L. Lai, J. Wang, X. Peng, W. Sun, and M. Sun, "A robust rgb-d slam system with points and lines for low texture indoor environments," *IEEE Sensors Journal*, vol. 19, no. 21, pp. 9908–9920, 2019.

[3] S. Weber, N. Demmel, T. C. Chan, and D. Cremers, "Power bundle adjustment for large-scale 3d reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 281–289.

[4] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar, "Block-nerf: Scalable large scene neural view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8248–8258.

[5] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "imap: Implicit mapping and positioning in real-time," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6229–6238.

[6] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5459–5469.

[7] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, "Building rome in a day," *Communications of the ACM*, vol. 54, no. 10, pp. 105–112, 2011.

[8] S. Zhu, R. Zhang, L. Zhou, T. Shen, T. Fang, P. Tan, and L. Quan, "Very large-scale global sfm by distributed motion averaging," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4568–4577.

[9] M. Pollefeys, D. Nistér, J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S.-J. Kim, P. Merrell *et al.*, "Detailed real-time urban 3d reconstruction from video," *International Journal of Computer Vision*, vol. 78, pp. 143–167, 2008.

[10] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[11] L. Liu, J. Gu, K. Zaw Lin, T.-S. Chua, and C. Theobalt, "Neural sparse voxel fields," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 651–15 663, 2020.

[12] P. E. Debevec, C. J. Taylor, and J. Malik, "Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach," in *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, 2023, pp. 465–474.

[13] J. Thies, M. Zollhöfer, and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures," *Acm Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–12, 2019.

[14] L. Shi, H. Hassanieh, A. Davis, D. Katabi, and F. Durand, "Light field reconstruction using sparsity in the continuous fourier domain," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 1, pp. 1–13, 2014.

[15] T. He, J. Collomosse, H. Jin, and S. Soatto, "Deepvoxels++: Enhancing the fidelity of novel view synthesis from 3d voxel embeddings," in *Proceedings of the Asian Conference on Computer Vision*, 2020.

[16] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh, "Neural volumes: Learning dynamic renderable volumes from images," *arXiv preprint arXiv:1906.07751*, 2019.

[17] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhofer, "Deepvoxels: Learning persistent 3d feature embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2437–2446.

[18] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, "Nerf in the wild: Neural radiance fields for unconstrained photo collections," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7210–7219.

[19] H. Turki, D. Ramanan, and M. Satyanarayanan, "Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 922–12 931.

[20] M. Zhenxing and D. Xu, "Switch-nerf: Learning scene decomposition with mixture of experts for large-scale neural radiance fields," in *The Eleventh International Conference on Learning Representations*, 2022.

[21] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "Nice-slam: Neural implicit scalable encoding for slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 786–12 796.

[22] A. Rosinol, J. J. Leonard, and L. Carlone, "Nerf-slam: Real-time dense monocular slam with neural radiance fields," *arXiv preprint arXiv:2210.13641*, 2022.

[23] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin, "Fastnerf: High-fidelity neural rendering at 200fps," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 346–14 355.

[24] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec, "Baking neural radiance fields for real-time view synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5875–5884.

[25] C. Reiser, S. Peng, Y. Liao, and A. Geiger, "Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 335–14 345.

[26] C. Yang, P. Li, Z. Zhou, S. Yuan, B. Liu, X. Yang, W. Qiu, and W. Shen, "Nerfvs: Neural radiance fields for free view synthesis via geometry scaffolds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 549–16 558.

[27] H. Zhou, L. Song, J. Chen, Y. Zhou, G. Wang, J. Yuan, and Q. Zhang, "Rethinking soft labels for knowledge distillation: A bias-variance tradeoff perspective," *arXiv preprint arXiv:2102.00650*, 2021.

[28] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, pp. 1789–1819, 2021.

[29] J. Yang, B. Martinez, A. Bulat, and G. Tzimiropoulos, "Knowledge distillation via softmax regression representation learning," in *International Conference on Learning Representations*, 2020.

[30] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[31] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[32] Y. Li, R. Zhang, J. Lu, and E. Shechtman, "Few-shot image generation with elastic weight consolidation," *arXiv preprint arXiv:2012.02780*, 2020.

[33] A. Aich, "Elastic weight consolidation (ewc): Nuts and bolts," *arXiv preprint arXiv:2105.04093*, 2021.

[34] A. Mallya and S. Lazebnik, "Packnet: Adding multiple tasks to a single network by iterative pruning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 7765–7773.

[35] J. Chung, K. Lee, S. Baik, and K. M. Lee, "Meil-nerf: Memory-efficient incremental learning of neural radiance fields," *arXiv preprint arXiv:2212.08328*, 2022.

[36] Z. Cai and M. Mueller, "Clnerf: Continual learning meets nerf," *arXiv preprint arXiv:2308.14816*, 2023.

[37] L. Liu, J. Gu, K. Zaw Lin, T.-S. Chua, and C. Theobalt, "Neural sparse voxel fields," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 651–15 663, 2020.

[38] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma *et al.*, "The replica dataset: A digital replica of indoor spaces," *arXiv preprint arXiv:1906.05797*, 2019.

[39] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, pp. 1–13, 2017.