PVD-AL: Progressive Volume Distillation with Active Learning for Efficient Conversion Between Different NeRF Architectures

Shuangkang Fang, Yufeng Wang, Yi Yang, Weixin Xu, Heng Wang, Wenrui Ding, Shuchang Zhou

Abstract—Neural Radiance Fields (NeRF) have been widely adopted as practical and versatile representations for 3D scenes, facilitating various downstream tasks. However, different architectures, including plain Multi-Layer Perceptron (MLP), Tensors, low-rank Tensors, Hashtables, and their compositions, have their trade-offs. For instance, Hashtables-based representations allow for faster rendering but lack clear geometric meaning, making spatial-relation-aware editing challenging. To address this limitation and maximize the potential of each architecture, we propose Progressive Volume Distillation with Active Learning (PVD-AL), a systematic distillation method that enables any-to-any conversions between different architectures. PVD-AL decomposes each structure into two parts and progressively performs distillation from shallower to deeper volume representation, leveraging effective information retrieved from the rendering process. Additionally, a Three-Levels of active learning technique provides continuous feedback during the distillation process, resulting in high-performance results. Empirical evidence is presented to validate our method on multiple benchmark datasets. For example, PVD-AL can distill an MLP-based model from a Hashtables-based model at a $10\times\sim20\times$ faster speed and 0.8dB$\sim$2dB higher PSNR than training the NeRF model from scratch. Moreover, PVD-AL permits the fusion of diverse features among distinct structures, enabling models with multiple editing properties and providing a more efficient model to meet real-time requirements. Project website: *http://sk-fun.fun/PVD-AL*.

✦

1 INTRODUCTION

NOVEL view synthesis (NVS) generates photo realistic 2D images for unknown view-ports of a 3D scene [2], [3], [4], and has wide applications in rendering, localization, and robot arm manipulations [5], [6], [7], especially with the neural modeling capabilities offered by the recently developed Neural Radiance Fields (NeRF) [8]. By exploiting the strong generalization capabilities of Multi-Layer Perceptrons (MLPs), NeRF can significantly improve the quality of NVS. Several following developments incorporate feature tensors as complementary explicit representations to relieve the MLPs from remembering all details of the scene, resulting in faster training speed and more flexible manipulation of geometric structure. The bloated size of the feature tensors in turn spurs works targeting more compact representations, like TensoRF [9] that leverages VM (vector-matrix) decomposition and canonical polyadic decomposition (CPD), Plenoxels [10] that exploits the sparsity of the tensor, and Instant Neural Graphics Primitives (INGP) [11] that utilizes multilevel hash tables for effective compression of feature tensors.

All these schemes have their own advantages and limitations. Generally, with implicit representations, it would be easier to perform texture editing of a scene (such as color, lighting changes, and deformations, etc.), to the extent of artistic stylization and dynamic scene modeling [12], [13], [14], [15], [16]. On the other hand, methods with explicit or hybrid representation usually enjoy faster training due to the shallower representations and cope better with geometric-aware editing [10], [17], [18], like merging and other manipulations of scenes, which is in clear contrast to the case of purely implicit representations.

Due to the diversity of downstream tasks of NVS, there is *no single answer* as to which representation is the best. The particular choice would depend on the specific application scenarios and the available hardware computation capabilities. Researching them independently can not reach their full potential. In this paper, we tackle the problem from another perspective. Instead of focusing

on an ideal alternative representation that embraces the advantages of all variants, we propose a method to achieve arbitrary conversions between known NeRF architectures, including MLPs, sparse Tensors, low-rank Tensors, hash tables, and combinations thereof. Such flexible conversions can obviously bring the following advantages. Firstly, the study would throw insights into the modeling capabilities and limitations of the already rich and ever-growing constellation of architectures of NeRF. Secondly, the possibility of such conversions would free the designer from the burden of pinning down architectures beforehand, as now they can simply adapt a trained model agilely to other architectures to meet the needs of later discovered application scenarios. Last but not least, complementary benefits may be leveraged in cases where the teacher and student are of different attributes. For example, by designing an efficient conversion strategy, students can inherit editing attributes from teachers as well as a wealth of 3D reconstruction knowledge.

In order to convert between various architectures, two issues must be addressed. One is that there are significant differences in training and inference speeds across different architectures. How can we ensure that the slower model does not hinder distillation efficiency? The second is how to condense the most relevant and useful information from a trained model to utilize as a learning guide for the student.

We propose PVD-AL as a solution to address the distillation efficiency and performance noted above. We decompose each architecture into two parts and apply the effective information retrieved from the rendering process to perform the distillation process progressively, stage-by-stage, on different levels of volume representation, from shallower to deeper, thus accomplishing the objective of rapid distillation. In this design, the teacher directs student training with the high-level semantic information of the model's middle layer, the density and color of the spatial points, and the rendered RGB values.

However, we discover that the learning of students is insufficient if only the aforementioned techniques are employed. For example, when sampling rays randomly and uniformly, some rays may pass through blank areas, and these rays can be easily fitted by students, whereas for rays that pass through the surface of the object, students typically require more time to fit. Similar rules apply to camera poses and sample points along a ray.

- *Shuangkang Fang, Yufeng Wang, and Wenrui Ding are with Beihang University, Beijing 100083, China.*
  E-mail: {skfang, wyfeng, ding}@buaa.edu.cn.
- *Yi Yang, Weixin Xu, Heng Wang, Shuchang Zhou are with Megvii Inc, Beijing 100190, China.*
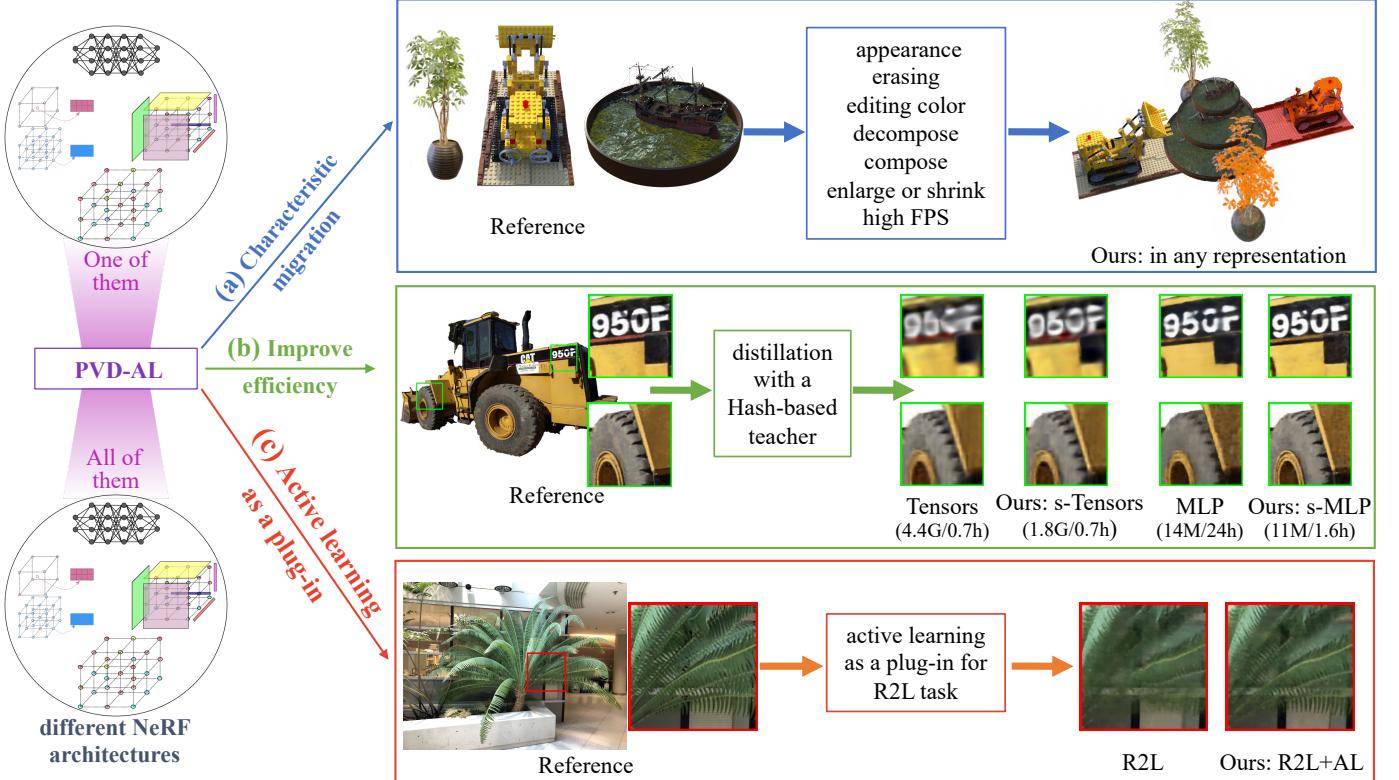 *E-mail: {yangyi, xuweixin02, wangheng, zsc}@megvii.com.*

Fig. 1: PVD-AL facilitates different NeRF architecture conversions, breaking down the barrier of independent research between them. Its efficient and flexible design supports versatile applications. (a) PVD-AL has the ability to migrate any architecture's attributes to other architectures or concentrate them on a single architecture. (b) A model obtained by PVD-AL performs better than by training it from scratch, with smaller parameters and shorter training time. (c) Active learning strategy can be plugged into other NeRF-related distillation tasks (like radiance field to light field: R2L [1]) to enhance final performance. More results can be found in the Section 4.

Therefore, we introduce an active learning strategy in PVD-AL. By including this strategy in the distillation procedure, students would be informed in real-time which camera poses, sample rays, and sample points offer the greatest challenge. Students will pay more attention to these vital pieces of knowledge after obtaining this feedback in the next training step, leading to high-performance distillation results.

To sum up, our contributions are summarized as follows:

- We propose PVD-AL, a distillation framework to accelerate the training procedure based on a unified view that allows conversions between different NeRF architectures, including MLP, sparse Tensors, low-rank Tensors and hashtables. The whole distillation process is data-free. To the best of our knowledge, this is the first systematic attempt at such conversions.
- We propose an active learning strategy so that students can acquire knowledge from teachers to the greatest extent. We continuously evaluate the camera poses, sample rays, and sample points that are difficult to fit for students, which help them actively enhance the learning of crucial knowledge. The three levels of active learning strategies are decoupled, flexible, and highly versatile, thus can be also easily applied as plug-in to other distillation tasks that use NeRF-based model as a teacher or student.
- Using high-performance teachers, such as hashtables and VM-decomposition structures, frequently improves student model synthesis quality while taking less time than training the student from scratch. As a result, Our technique can be employed as a new tool to effectively train a model. Experiments suggest that it can also be used to compress the NeRF family of models, producing more valuable models for applications.
- Our method allows for the fusion of various properties between different structures. For example, we can call PVD-AL multiple times to obtain models with multiple editing properties. It is also possible to convert a scene under a specific model to another model that runs more efficiently

to meet the real-time requirements of downstream tasks.

A few results of this article were published originally in its conference version[1] [19]. And in this longer manuscript, we additionally introduce an active learning strategy to take the distillation performance one step further, which provides a more comprehensive and deeper understanding of distillation between different architectures. Besides, all experiments are based on the more powerful PVD-AL algorithm, and we have also conducted additional in-depth experiments to evaluate both the properties of PVD-AL and its various application scenarios. Please consult Section 4 for additional details.

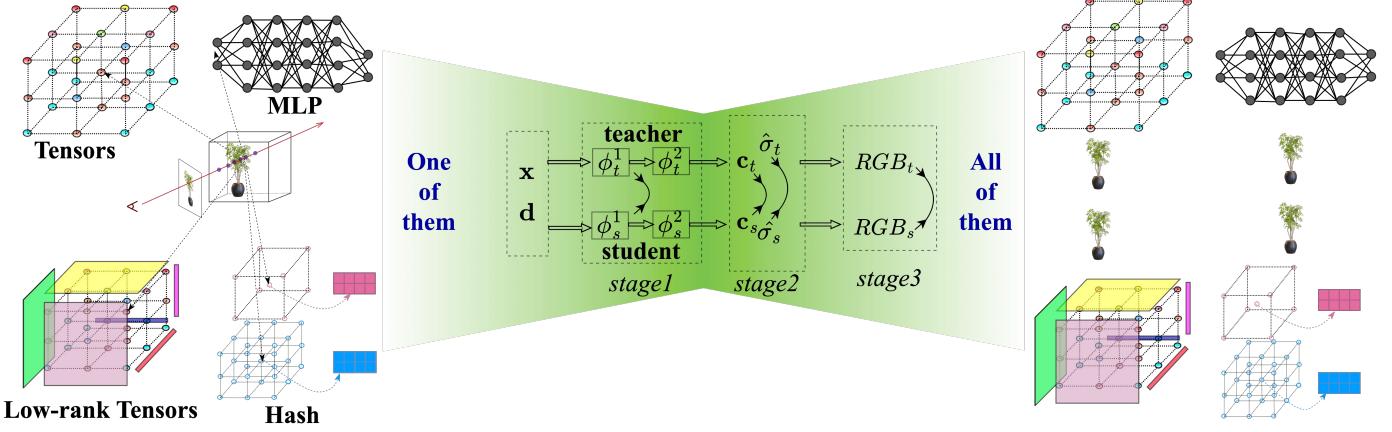## 2 RELATED WORK

2.1 Neural Implicit Representations

Neural implicit representation methods use MLP to construct a 3D scene from coordinate space, as proposed in NeRF [8]. The input of the MLP is a 5D coordinate (spatial location $[x, y, z]$ and viewing direction $[\theta, \phi]$), and the output is the volume density and view-dependent color [20], [21], [22], [23]. The advantage of implicit modeling is that the representation is conducive to controlling or changing texture-like attributes of the scene. For example, DFFs [12] use the pretrained CLIP model [24] to induce editing of NeRF representation of a scene. DNeRF [13] successfully applies NeRF to the rendering of dynamic scenes by mapping time t to implicit space through an MLP. NeRFW [25] realizes the control of scene lighting by adding appearance embedding using an MLP. However, MLP-based model requires on-the-fly dense sampling of spatial points, which leads to multiple queries of the MLP during training and inference, resulting in slower running speed.

2.2 Neural Explicit Representations and Hybrids

With explicit representations, the scene is placed directly on a 3D grid (a huge tensor). Each voxel on the grid stores information on density and color. Plenoxels [10] show that a 3D scene can

Progressive Volume Distillation (PVD)

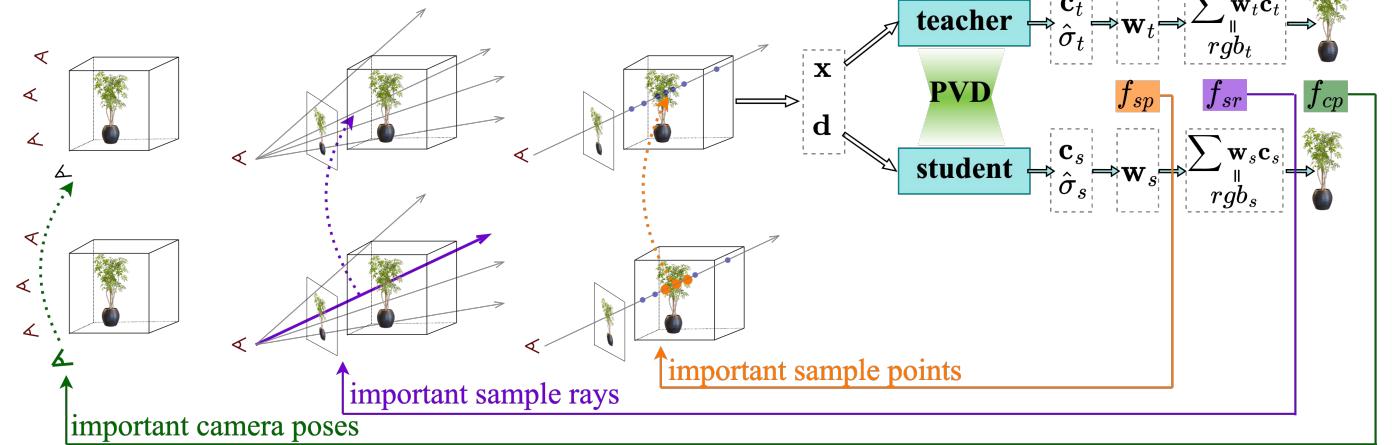## Progressive Volume Distillation with Active Learning (PVD-AL)

Fig. 2: Illustration of PVD-AL. Given one trained NeRF model, different NeRF architectures, like sparse Tensors, MLP, low-rank Tensors, and hash tables can be quickly obtained through PVD-AL. The loss in intermediate volume representations (shown as double arrow symbol in PVD) like output of ϕ_*^1, color, and density are used alongside the final rendered RGB volume to accelerate distillation. In PVD-AL, regular feedback from the teacher will be given to the student. This form of feedback is organized into three tiers based on camera poses, sample rays, and sample points, which prepares the student to proactively learn what they need to be strengthened for the next round of training.

be represented by an explicit grid, and the spherical harmonic coefficients at each voxel can be used to obtain the density and color at arbitrary spatial points by trilinear interpolation. The training and inference speed of Plenoxels is significantly superior to that of MLP-based NeRF.

Recently, motivated by the low-rank tensor approximation algorithm, TensoRF [9] decomposes the explicit tensor into low-rank components, which significantly reduces the model size. PeRF [26] continues to evolve the explicit expression and regard the optimization of grid as a non-linear least squares optimization problem that can be solved more efficiently by the Gauss-Newton method. With explicit representation, it is not as easy to make artistic creations as with implicit representation. Nevertheless, explicit representations facilitate the geometry editing of the scene, including merging of multiple scenes, inpainting and manipulations of objects at specific positions [10], [17], [18].

There are also attempts exploiting a hybrid of the explicit and implicit representations as NeRF architectures [9], [11], [27], [28], [29]. The explicit part usually stores features related to the scene, while the implicit part is typically an MLP that interprets the features to obtain densities and colors. Differences between hybrid representations are mainly exhibited in the explicit part. NSVF [30] uses a spare grid to store features, while Plenoctrees [18] optimizes the 3D grid through an octree. Nex [31] proposes an Implicit-Explicit modeling strategy by storing the coefficient as a learnable parameter to accelerate the training procedure. Recently, INGP [11] proposes the multi-resolution hash encoding (MHE), which maps the given coordinate to a feature via a cascade of hash tables at different scales. Like TensoRF [9], MHE significantly reduces memory footprint and improves inference

speed. However, the compactness of MHE comes at a cost of less straightforward geometric interpretation as there are abundant spatial aliases caused by the hash mechanism.

2.3 Knowledge Distillation

Knowledge distillation commonly refers to training a small model to match the output of a larger model (maybe trained beforehand or on the fly), which is widely used in model optimization and compression [32], [33]. Multiple attempts have been made in the field of NVS. Mip-nerf 360 [34] proposes an online distillation method to improve the quality of rendering. R2L [1] converts a NeRF model into a model based on neural light fields. The most related to our work is KiloNeRF [35], which uses a huge pretrained NeRF (teacher) to guide thousands of small NeRF models (students) for speeding up. However, KiloNeRF only performs distillation between the same MLP architecture, and the distilling process is significantly slowed down by the continuous querying of the huge MLP in the teacher model.

2.4 Active Learning

Active learning is a special case of machine learning in which a learning algorithm can interactively query a teacher to label new data points with the desired outputs [36]. There are situations in which unlabeled data is abundant but manual labeling is expensive. In such a scenario, learning algorithms can actively query the user or teacher for labels. This type of iterative supervised learning is called active learning. Since the learner chooses the examples, the number of examples to learn a concept can often be much lower than the number required in normal supervised learning.

ActiveNeRF [37] proposes to supplement the existing training set with newly captured samples by camera based on an active learning scheme, which is totally different from our PVD-AL.

3 METHODS

Our method aims to achieve mutual conversions between different architectures of Neural Radiance Fields. Since there are an ever-increasing number of such architectures, we will not attempt to achieve these conversions one by one. Rather, we first formulate typical architectures in a unified form and then design a systematic distillation scheme based on the unified view. The architectures we have derived formula include implicit representations like MLP in NeRF, explicit representations like sparse Tensors in Plenoxels, and two hybrid representations: hash tables (in INGP) and low-rank Tensors (VM-decomposition in TensoRF). Once formulated, any-to-any conversion between these architectures and their compositions is possible. We will first cover some preliminary topics before moving on to a detailed description of our method.

3.1 Preliminaries

Neural Radiance Fields. NeRF represents scenes with an implicit function that maps spatial point $\mathbf{x} = (x, y, z)$ and view direction $\mathbf{d} = (\theta, \phi)$ into the density σ and color $\mathbf{c}$. Given a ray $\mathbf{r}$ originating at $\mathbf{o}$ with direction $\mathbf{d}$, the RGB value $\hat{\mathbf{C}}(\mathbf{r})$ of the corresponding pixel is estimated by the numerical quadrature of the color $\mathbf{c}_i$ and density σ_i of the spatial points $\mathbf{x}_i = \mathbf{o} + t_i \mathbf{d}$ sampled along the ray:

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_i^N T_i (1 - \exp(-\sigma_i \delta_i)) \mathbf{c}_i \qquad (1)$$

where $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_i \delta_i)$, and δ_i is the distance between adjacent samples.

Tensors and Low-rank Tensors. The Plenoxels directly represents a 3D scene by an explicit grid (sparse Tensors) [10]. Each grid point stores density and spherical harmonic (SH) coefficients. The color $\mathbf{c}$ is obtained according to the SH and the view direction $\mathbf{d}$ as follows:

$$\mathbf{c} = S \left(\sum_{\ell=0}^{\ell_{\max}} \sum_{m=-\ell}^{\ell} k_\ell^m Y_\ell^m(\mathbf{d}) \right) \qquad (2)$$

where $S : x \mapsto (1 + \exp(-x))^{-1}$, $\mathbf{k} = (k_\ell^m)_{\ell:0 \le \ell < \ell_{\max}}^{m:-\ell \le m \le \ell}$, and k_ℓ^m is a set of coefficients, and l is the degree of the SH function Y_ℓ^m.

The performance of explicit sparse Tensors depends excessively on the spatial resolution of the grid. In order to reduce the memory footprint caused by the enormous size of the tensor, The VM (Vector-Matrix) decomposition [9] factorizes the huge tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$ into low-rank matrices $\mathbf{M}$ and vectors $\mathbf{v}$ as follows:

$$\mathcal{T} = \sum_{r=1}^{R_1} \mathbf{v}_r^1 \circ \mathbf{M}_r^{2,3} + \sum_{r=1}^{R_2} \mathbf{v}_r^2 \circ \mathbf{M}_r^{1,3} + \sum_{r=1}^{R_3} \mathbf{v}_r^3 \circ \mathbf{M}_r^{1,2} \qquad (3)$$

where $\mathbf{v}_r^1 \in \mathbb{R}^I$, $\mathbf{v}_r^2 \in \mathbb{R}^J$, $\mathbf{v}_r^3 \in \mathbb{R}^K$, $\mathbf{M}_r^{2,3} \in \mathbb{R}^{J \times K}$, $\mathbf{M}_r^{1,3} \in \mathbb{R}^{I \times K}$, and $\mathbf{M}_r^{1,2} \in \mathbb{R}^{I \times J}$. And $\circ$ represents the outer product. Unlike Plenoxels, VM decomposition does not store color directly but rather features that can be decoded by an MLP.

Multi-resolution Hash Encoding. INGP [11] maps a series of grids of different scales to the corresponding feature vectors with a fixed size. It uses a hash function as in Equation (4) to map a spatial point in the grid to a hash table with different resolutions that are adopted to details of different levels of these grids.

$$h(\mathbf{x}) = \left(\bigoplus_{i=1}^d x_i \pi_i \right) \mod S \qquad (4)$$

where $\bigoplus$ denotes bit-wise XOR operation. π_i is an unique large prime number. And S is the hash table size. These hash tables store learnable parameters, which are fed to a shallow MLP to interpret densities and colors. INGP effectively reduces the model size by these hash tables and improves the synthesis quality by introducing multi-resolution.

3.2 PVD-AL: Progressive Volume Distillation with Active Learning

We propose PVD-AL to realize the mutual-conversion between different architectures. To speed up the training process, we develop a volume-aligned loss and construct a block-wise distillation method based on a unified perspective of various NeRF architectures in PVD-AL. We also employ a special treatment of the dynamic density volume range by clipping, which improves the training stability and significantly improves the synthesis quality. Furthermore, we have designed three levels of active learning strategies based on the unique characteristics of such distillation, which significantly boost final distillation performance and outperform the training of a model from scratch.

3.2.1 Loss Design

In our method, we not only use the RGB but also the density, color and an additional intermediate feature to calculate loss between different structures. We observed that the implicit and explicit structures in the hybrid representation are naturally separated and correspond to different learning objectives. Therefore, we consider splitting a model into these similar expression forms so that different parts can be aligned during distillation. Specifically, given a model ϕ_*, we represent them as a cascade of two modules as follows:

$$\phi_*(\mathbf{x}, \mathbf{d}) = \phi_*^2(\phi_*^1(\mathbf{x}, \mathbf{d})) \qquad (5)$$

TABLE 1: The division of each typical model under our unified two-level view.

methods	ϕ_*^1	ϕ_*^2
NeRF [8]	first K layers	remaining MLP
Plenoxels [10]	full	identity function
TensoRF [9]	decomposed tensors	MLP decoder
INGP [11]	hash tables	MLP decoder

Here * can be either a teacher or a student. K=4 is used by default in NeRF. For hybrid representations, we directly regard the explicit part as ϕ_*^1, and the implicit part as ϕ_*^2. While for purely implicit representation, we divide the network into two parts with a similar number of layers according to its depth and denote the former part as ϕ_*^1 and the latter part as ϕ_*^2. As for the purely explicit representation Plenoxels, we still formulate it into two parts by letting ϕ_*^2 be the identity, though it can be transformed without splitting. The specific splitting of the model is shown in Table 1. Based on the splitting, we design volume-aligned losses as follows:

$$\mathcal{L}_2^v = \left\| \phi_t^1(\mathbf{x}, \mathbf{d}) - \phi_s^1(\mathbf{x}, \mathbf{d}) \right\|_2 \qquad (6)$$

In essence, the reason for designing this loss is that models in different forms can be mapped to the same space that represents the scene. Our experiments have shown that this volume-aligned loss can accelerate the distillation and improve the quality significantly. The complete loss function during distillation is as follows:

$$\mathcal{L} = \omega_1 \mathcal{L}_2^v + \omega_2 \mathcal{L}_2^\sigma + \omega_3 \mathcal{L}_2^c + \omega_4 \mathcal{L}_2^{rgb} + \omega_5 \mathcal{L}_{reg} \qquad (7)$$

where $\mathcal{L}^\sigma, \mathcal{L}^c, \mathcal{L}^{rgb}$, denote the density loss, color loss and RGB loss respectively. $\mathcal{L}_2$ is the mean-squared error (MSE). The last item $\mathcal{L}_{reg}$ represents the regularization term, which depends on the form of the student model. For Plenoxels and VM-decomposition, we add L1 sparsity loss and total variation (TV) regularization loss. It should be noted that we only perform density, color, RGB and regularization loss on Plenoxels for its explicit representation.

Algorithm 1 PyTorch pseudocode of active learning in PVD-AL.

```
# TCP: important camera poses
# TSR: important sample rays
# TSP: important sample points
initial(TCP, TSR, TSP)
for each epoch:
    #generate random poses and sample important poses
    train_poses=cat(GenRandomPoses(),RandomSelect(TCP))
    all_batch_rays=SampleRays(train_poses)

    for each batch_rays:
        #generate random rays and sample important rays
        train_rays=cat(batch_rays, RandomSelect(TSR))
        train_data=SamplePoints(train_rays)

        #distillation process
        tea_out=tea_model(train_data)
        stu_out=stu_model(train_data)

        #select important points TSP by its weights:PW
        PW=cat(tea_out[PW], stu_out[PW])
        sortedPW= GetPointsIdx(sort(PW))
        TSP=sortedPW[:Nsp]

        #use TSP to calculate loss sigma and loss color
        loss_s=L2(tea_out['s'][TSP], stu_out['s'][TSP])
        loss_c=L2(tea_out['c'][TSP], stu_out['c'][TSP])

        #calculate loss rgb and other losses
        loss_rgb=L2(tea_out['rgb'], stu_out['rgb'])
        loss_other=CalculateOtherLosses()

        #select important rays TSR by loss rgb
        sortedRays=GetRaysIdx(sort(loss_rgb))
        TSR[replace_idx]=sortedRays[:Nsr]

        #record img loss by train_rays
        loss_img[RaysMapToImgIdx(batch_rays)]+=loss_rgb
        backward_option()

    # select important poses TCP by loss img
    sortedImgs=GetPoses(loss_img)
    TCP[repalce_idx]=sortedImgs[:Ncp]
```

3.2.2 Density Range Constrain

We found that the loss of density σ is hardly directly optimized. And we impute this problem to its specific numerical instability. That is, the density reflects the light transmittance of a point in space. When σ is greater than or less than a certain value, its physical meaning is consistent (i.e., completely transparent or completely opaque). Therefore the value range of σ can be too wide for a teacher, but in fact, only one interval of the density values plays a key role. On the basis of this, we limit the numerical range of σ to $[a, b]$. Then the $\mathcal{L}_2^{\sigma}$ is calculated as follow:

$$\mathcal{L}_2^{\sigma} = \|\min(\max(\sigma_t, a), b) - \min(\max(\sigma_s, a), b)\|_2 \quad (8)$$

According to our experiments, this restriction has an inappreciable impact on the performance of teacher and bring a tremendous benefit to the distillation. We also consider directly performing the density loss on the $\exp(-\sigma_i \delta_i)$, but we found it is an inefficiency way since the gradient of $\exp$ is easier to saturate, and it requires computing an exponent that increases the amount of calculation when the block-wise is implemented.

3.2.3 Block-wise Distillation

During volume rendering, most of the computation occurs in MLP forwarding for each sampled point and integrating the output over each ray. Such a heavy process slows down training and distillation significantly. In our PVD-AL, thanks to the design of $\mathcal{L}_2^v$, we can implement the block-wise strategy to get rid of this problem. Specifically, we only forward stage 1 at the beginning of training and then run stages 2 and 3 in turn, as shown in Fig.2. Consequently, the student and the teacher do not need to forward the complete network and render RGB in the early stages of training. In our experiment, the conversion from INGP to NeRF can be completed in tens of minutes, which requires several hours in the past.

3.2.4 Three-Levels of Active Learning

For the purpose of facilitating the most efficient possible transfer of knowledge from teachers to students, we propose an active learning technique in PVD-AL. We continually analyze the camera poses, sample rays, and sample points from coarse to fine which

is tough to suit for students, which helps students actively boost their understanding of these crucial knowledge as shown in 2.

Camera poses. After training a teacher model, given any camera pose, the teacher can render the corresponding image. Therefore, we do not need to use real data but only use the images rendered by the teacher to guide the training of students. In our experiments, we found that students and teachers can show a large performance gap under some poses. Therefore, we consider actively feeding these poses to students during distillation to enhance their performance under these poses. Given several camera poses, the images rendered by the teacher and student are respectively marked as I_t and I_s, and the important camera poses TCP can be obtained by the selection function f_{cp}:

$$TCP = f_{cp}(I_t, I_s, N_{cp}) \quad (9)$$

f_{cp} will select the N_{cp} poses with the largest gap between I_t and I_s.

Sample Rays. Sample rays carry varying amounts of information depending on the location they travel through. For example, some rays will travel through fully opaque object surfaces while others may travel through translucent sections. Each type of ray presents a unique challenge to students. Therefore, we provide students with feedback for the rays where there are substantial performance gaps between students and teachers. Given several sample rays, the RGB values rendered by the teacher and student are respectively marked as RGB_t and RGB_s, and the important sample rays TSR can be obtained by the selection function f_{sr}:

$$TSR = f_{sr}(RGB_t, RGB_s, N_{sr}) \quad (10)$$

f_{sr} will select the N_{sr} rays with the largest gap between RGB_t and RGB_s.

Sample points. When performing the sigma and color loss in Eq.(7), the default is to use all of the rays' sampling points. In fact, different sampling points contribute differently to the final RGB value, with points near the surface of the object being relatively more important, while points across the surface do not contribute anything to the RGB value. When calculating sigma and color losses, the student should zero in on the sample points that have a significant impact on the final RGB value. Given several sample points along a ray, the weight of each point calculated by teacher and student is respectively marked as W_t and W_s, the important sample points TSP can be obtained by the selection function f_{sp}:

$$TSP = f_{sp}([W_t : W_s], N_{sp}) \quad (11)$$

where $[:]$ means concatenate, and W_* is calculated from Eq.(1), that is

$$W_* = \sum_i^N T_i(1 - \exp(-\sigma_i \delta_i)) \quad (12)$$

f_{sr} will select the N_{sp} points with the largest value of W_t and W_s.

The specific execution process of the active learning strategy in PVD-AL is show in Algorithm 1.

4 EXPERIMENTS

On a number of benchmark datasets, we empirically demonstrate the conversion effectiveness of PVD-AL. We primarily evaluate the effectiveness of our method in enhancing performance, expediting training, compressing parameters, and transmitting structure-specific attributes such as editing and fast inference abilities.

4.1 Implementation Details

Datasets Our experiments are conducted mainly on the three datasets: Synthetic-NeRF [8], forward-facing (LLFF) [20] and TanksAndTemple [38]. We solely use the aforementioned datasets to train teacher models. In the distillation phase, we believe it is
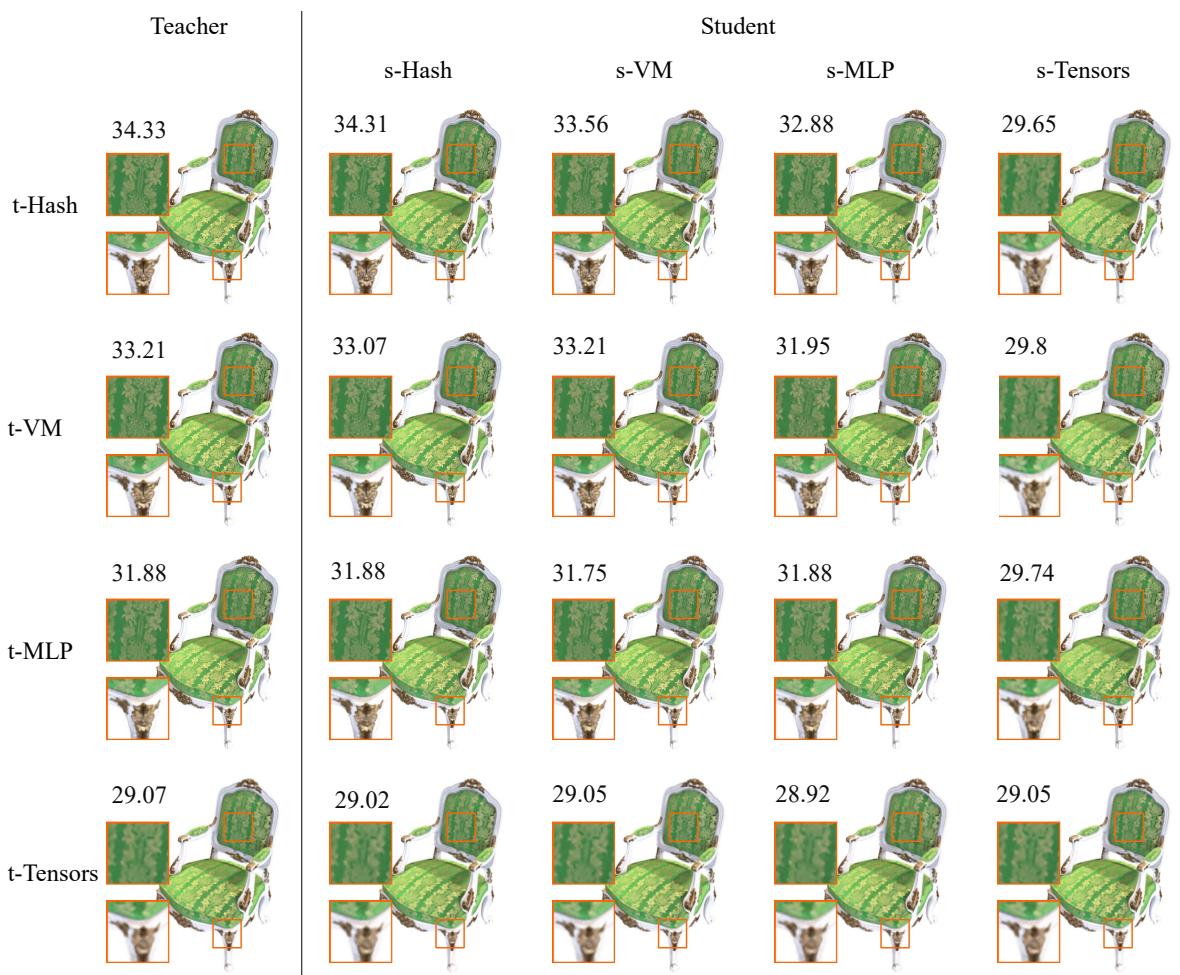
Fig. 3: Results of mutual-conversion by PVD-AL between Hash, VM-decomposition, MLP, and Tensors on the chair scene from Synthetic-NeRF dataset. The numbers mean PSNR. The s- stands for student and the t- for teacher.

TABLE 2: The quantitative results of mutual-conversion between Hash, VM-decomposition, MLP, and Tensors on the Synthetic-NeRF, LLFF, and TanksAndTemple datasets. We first train four teachers with different structures for each scenario in the three benchmark datasets, yielding 84 teacher models in total, and then perform mutual conversion between the four structures for each scene, for a total of 336 distillation experiments.

| | PSNR↑ | | | | SSIM↑ | | | | $LPIPS_{alex}$ ↓ | | | | $LPIPS_{vgg}$ ↓ | | | |
	t-Hash	t-VM	t-MLP	t-Tensors	t-Hash	t-VM	t-MLP	t-Tensors	t-Hash	t-VM	t-MLP	t-Tensors	t-Hash	t-VM	t-MLP	t-Tensors
Synthetic-NeRF																
teacher	32.58	31.52	30.78	27.49	0.96	0.955	0.946	0.917	0.032	0.04	0.049	0.122	0.055	0.061	0.075	0.112
s-Hash	32.58	31.35	30.75	27.44	0.96	0.954	0.947	0.915	0.032	0.042	0.051	0.117	0.055	0.067	0.077	0.116
s-VM	31.83	31.52	30.57	27.48	0.957	0.955	0.945	0.916	0.038	0.04	0.053	0.121	0.06	0.061	0.077	0.114
s-MLP	31.63	30.78	30.78	27.38	0.953	0.948	0.946	0.914	0.043	0.05	0.049	0.119	0.069	0.076	0.075	0.118
s-Tensors	28.19	28.2	28.09	27.49	0.924	0.923	0.921	0.917	0.102	0.103	0.105	0.122	0.105	0.107	0.11	0.112
LLFF																
teacher	26.7	25.27	25	21.33	0.832	0.777	0.748	0.59	0.13	0.196	0.227	0.53	0.231	0.295	0.344	0.512
s-Hash	26.68	25.36	25.09	21.39	0.83	0.782	0.754	0.592	0.133	0.195	0.233	0.526	0.23	0.287	0.334	0.511
s-VM	25.98	25.24	24.85	21.39	0.793	0.774	0.74	0.591	0.19	0.197	0.251	0.529	0.29	0.292	0.346	0.511
s-MLP	25.95	24.89	25.01	21.33	0.786	0.741	0.748	0.592	0.201	0.253	0.227	0.53	0.315	0.361	0.347	0.524
s-Tensors	21.87	21.26	21.34	21.44	0.611	0.59	0.586	0.591	0.512	0.543	0.536	0.527	0.499	0.514	0.517	0.512
TanksAndTemples																
teacher	29.26	27.27	25.96	25	0.915	0.897	0.879	0.865	0.106	0.189	0.201	0.279	0.134	0.182	0.2	0.225
s-Hash	29.24	27.13	25.94	24.92	0.915	0.893	0.88	0.863	0.106	0.184	0.203	0.271	0.134	0.183	0.201	0.229
s-VM	28.3	27.27	25.89	24.96	0.906	0.895	0.879	0.865	0.153	0.188	0.201	0.28	0.165	0.181	0.204	0.227
s-MLP	27.97	26.71	25.93	24.78	0.9	0.887	0.876	0.863	0.152	0.201	0.218	0.273	0.175	0.197	0.201	0.23
s-Tensors	25.43	25.31	24.84	24.98	0.865	0.867	0.863	0.865	0.263	0.262	0.266	0.281	0.228	0.222	0.224	0.225

sufficient to use the teacher to generate bogus images as *pseudo-labeling* without touching the real training data.

Network Architecture. We try to stick as close to the original paper settings as feasible for each structure (Hash / MLP / VM-decomposition / sparse Tensors). For MLP [39], positional encoding is also utilized for coordinates and view directions.

For sparse Tensors [10], we use spherical harmonics of degree 2, and the $128 \times 128 \times 128$ grid for Synthetic-NeRF dataset and TankAndTemple dataset, $512 \times 512 \times 128$ grid for LLFF dataset. For VM-decomposition [9], we take 48 components in total. For Hash [11], we set the coarsest resolution, the finest resolution, levels, hash table size and feature dimensions to 16,
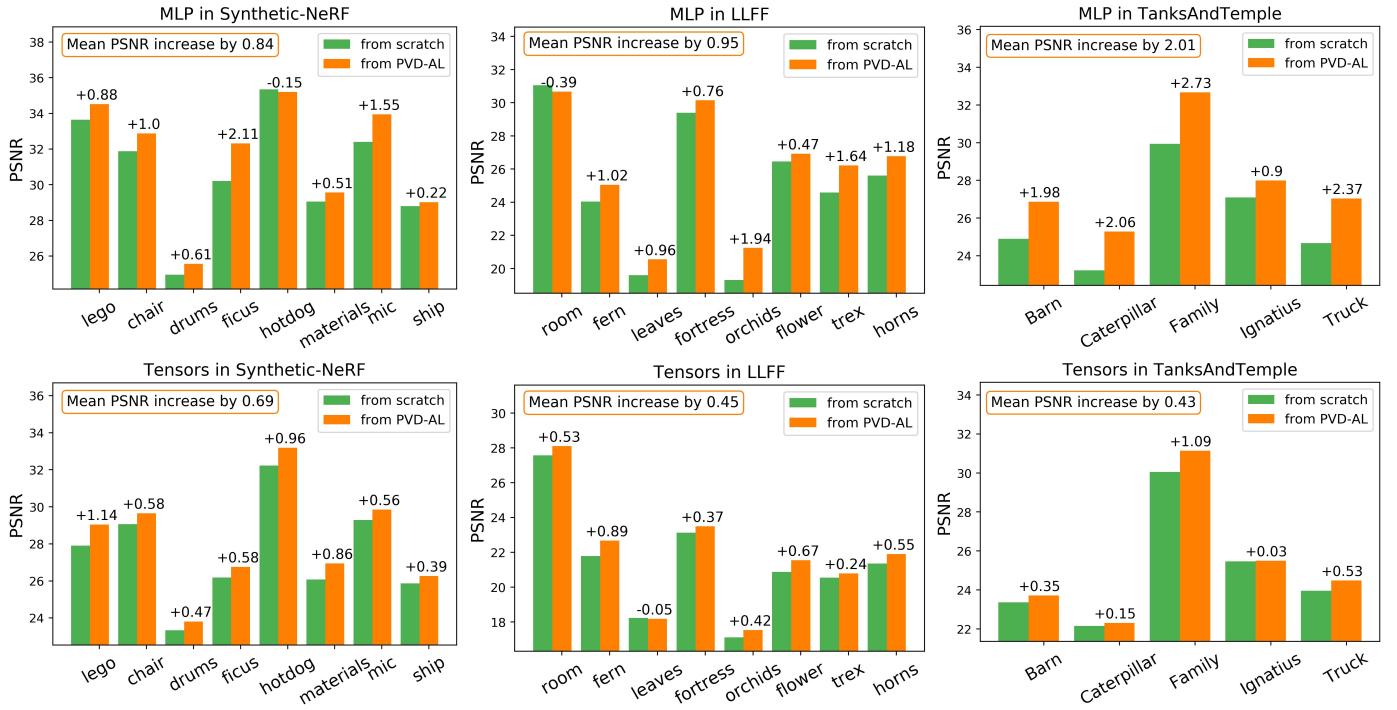
Fig. 4: Quantitative comparison between the model obtained by PVD-AL and the model trained from scratch. The performance of our method in PSNR is typically superior to training a model from scratch.
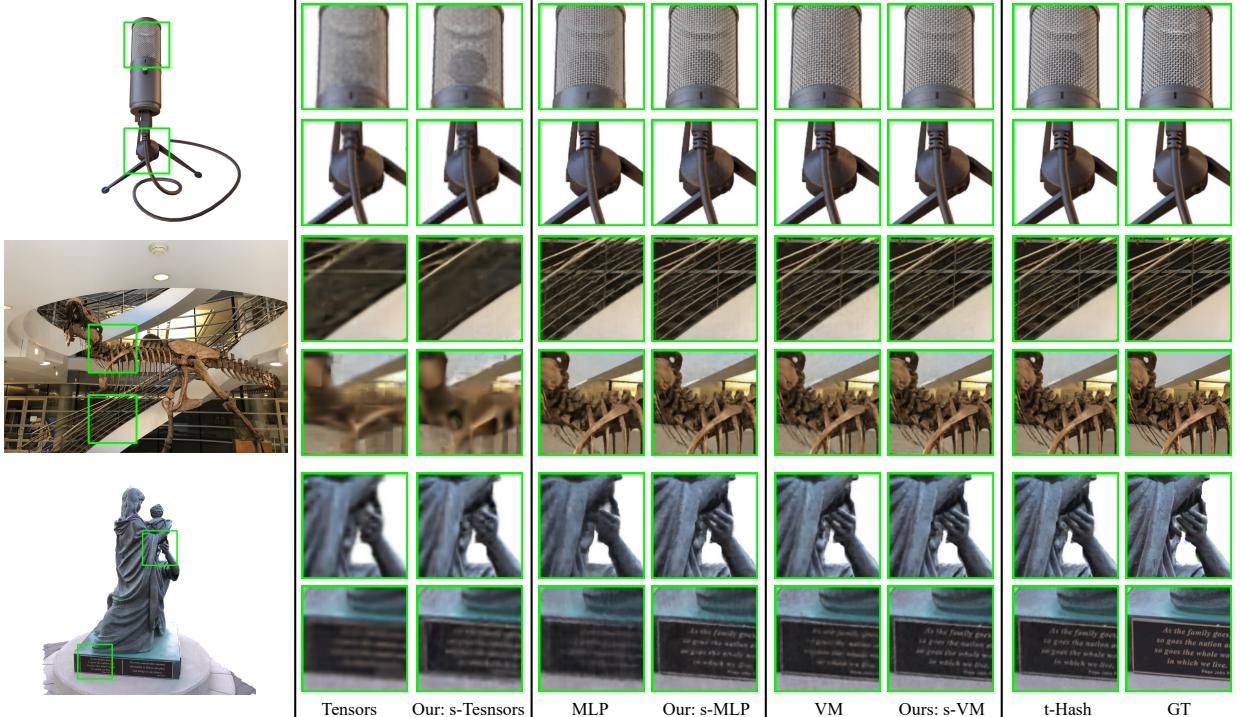

Tensors | Our: s-Tesnsors | MLP | Our: s-MLP | VM | Ours: s-VM | t-Hash | GT

Fig. 5: Qualitative comparison between the model obtained by PVD-AL and the model trained from scratch. The synthetic quality of our method is preferable to building a model from scratch.

$2048 \times$ scene scope, 14, 2^{19}, and 2 respectively.

Training and Distilling Details. We implement our method with the PyTorch framework [40] to train teachers and distill students. We use Adam Optimizer [41] with initial learning rates of 0.02 and run 20k steps with a batch size of 4096 rays. For distilling, we initial the loss rate for volume-aligned, density, color and RGB with 2e-3, 2e-3, 2e-3 and 1 respectively. The first stage consumes 3k steps, the second stage consumes 5k steps, and the third stage will take all the resting steps. Three levels of active learning strategy will be incorporated into the rest training process after the first two phases have been completed. We set 10% of important rays as important rays and 30% sample points as important points in each training iteration, and 10% camera poses as important poses in each training epoch. All the experiments are performed on a single NVIDIA V100 GPU.

4.2 Verification of Efficient Conversion

4.2.1 Quantitative Results

We average the PSNR/SSIM (higher is better) and LPIPS [42] (lower is better) results for mutual conversion experiments, which are shown in Table 2. This Table demonstrates that our method is able to successfully complete the capability of mutual conversion between four different structures, and the students who are the outcome of our method display exceptionally competitive performance. When the structures of the student and the teacher are consistent, for example, the student almost entirely inherits the performance of the teacher. This demonstrates that our method is able to transfer as much of the teacher's extensive knowledge to the student as possible, which exemplifies the efficacy of our method.

One of the most potent features of our method is its capacity

TABLE 3: An ablation study of our method. Metrics are averaged over the 5 scenes from TanksAndTemples dataset in the conversion from VM-decomposition to MLP.

	PSNR	SSIM	Lpips$_{alex}$	Lpips$_{vgg}$
w/o $\mathcal{L}_2^v$	25.87	0.858	0.208	0.202
w/o $\mathcal{L}_2^\sigma$	26.16	0.873	0.205	0.201
w/o $\mathcal{L}_2^c$	26.34	0.873	0.206	0.199
w/o $\mathcal{L}_2^{rgb}$	24.53	0.808	0.225	0.232
w/o sigma-restric	25.19	0.849	0.214	0.207
w/o block-wise	26.13	0.868	0.206	0.203
w/o poses^{AL}	26.34	0.876	0.204	0.200
w/o rays^{AL}	26.03	0.869	0.205	0.207
w/o points^{AL}	26.52	0.881	**0.201**	0.199
w/all	**26.71**	**0.887**	**0.201**	**0.197**

to surpass the upper limit of a model trained from scratch. Using a better-performing teacher to distill a student enables the student's performance to greatly exceed its training from scratch, as seen in Fig.4. For instance, when distilling a Hash-based model to an MLP-based model, the PSNR on the TanksAndTemple dataset increases by 2 compared to training the MLP-based model from scratch, demonstrating the superiority of our distillation scheme.

A closer look reveals that the student's performance is constrained in two ways: by the teacher's performance and by the student's own capacity. To the maximum extent possible, students can mimic teachers' performance when the modeling talents of students inferior or close teachers'. And if a student's modeling capabilities are superior to the teacher's, the student can be further enhanced by finetuning, which will be covered in detail in subsection 4.7.

4.2.2 Qualitative Results

Fig.3 illustrates an example of mutual-conversion visualization results on the chair scene from the Synthetic-NeRF dataset amongst different architectures of Hash, VM-decomposition, MLP, and sparse Tensors. The visual quality of the student is frequently indistinguishable from that of the teacher, illustrating the outstanding features of our method for sustaining synthesis quality.

We also present a visual comparison between the model obtained by our method and the model trained from scratch on three datasets. As shown in Fig.5, the quality of synthesis produced by our s-MLP is superior to that produced by training the MLP from scratch. The advancement is mostly attributable to our distillation method across distinct structures, which makes it possible for an experienced teacher to break the upper limit of a student's capabilities.

4.3 Verification of Each Component of PVD-AL

To assess the necessity of each component of PVD-AL, we designed several groups of ablation experiments. Table 3 shows how much our method's various components affect students' performance. It is evident that the loss that we developed boosts PSNR by roughly 0.4 to 2.2 dB. Additionally, it is apparent that performance will suffer if the value of density is not restricted. We also try the distillation without using the block-wise technique and discover that its outcome is inferior while having the same amount of time allotted for training.

Furthermore, Table 3 demonstrates the significance of active learning at each level. In each training session, our method actively records and assesses three tiers of samples that are most beneficial for the distillation process as described in subsection 3.2.4. This strategy is analogous to the teacher drawing vital knowledge for the students during the distillation process, with the end goal of facilitating the students' acquisition of knowledge in a manner that is both more efficient and effective.

In general, picking a proper K in Table 1 is striking a balance between performance and training time. We conducted an ablation study of distilling Hash into MLP by PVD on the Synthetic-NeRF

TABLE 4: An ablation study of the division position for MLP. Metrics are averaged over the 8 scenes on Synthetic-NeRF dataset in the conversion from Hash to MLP.

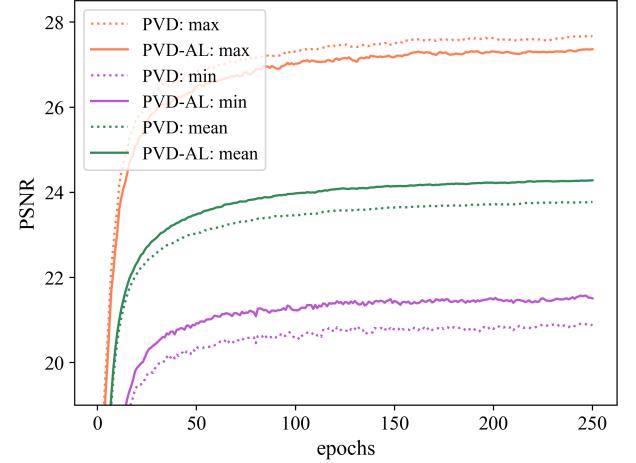|	K=2	K=3	K=4	K=5	K=6
PSNR	31.35	31.55	31.63	31.69	31.70
Time	1.35h	1.39h	1.42h	1.47h	1.53h

Fig. 6: Trend of maximum, minimum and average PSNR on validation set when distilling VM to Tensors in Truck scene from TanksAndTemples dataset.

dataset and the average PSNR and training time are as shown in Table 4. Here we get a higher PSNR with a larger K, which implies using more layers to fit hash tables can improve performance. In contrast, having a smaller K reduces the training time due to our blockwise distillation strategy. In this case, K=4 would be a Pareto optimum.

4.4 Properties of Active Learning
4.4.1 The role of active learning

On the truck scene from the TanksAndTemples dataset, we observe the trend of PSNR on the validation set for our method with (PVD-AL) or without (PVD) active learning during the training phase. Fig.6 depicts the changes in the maximum, minimum, and mean values of PSNR for every image in the validation set. It can be seen from Fig.6 that the maximum PSNR of PVD-AL is slightly lower and the minimum PSNR is significantly higher compared to PVD. This may indicate that the active learning technique can increase students' performance on challenging samples, resulting in a more balanced performance and a higher average PSNR.

4.4.2 visualization of active learning

On the Family scene from the TanksAndTemple dataset, we extract the important camera poses, the pixel positions corresponding to important rays, and the weights of sample points found by our active learning strategy during the last epoch of training. We depict them in Fig.7.

It can be seen from Fig.7(a) that the important camera poses do not exist randomly but are rather concentrated in several specific areas. Typically, students only struggle with a small number of camera poses, and increasing attention to these poses during training can significantly boost the student's performance.

For the display of important rays in Fig.7(b), it is evident that, during the distillation process, our active learning strategy concentrates mostly on the high-frequency information of the images, which is consistent with the results of Fig.5 that the synthetic quality in high-frequency areas is improved. In addition, it is essential to observe that there are almost no important rays in the spatially empty zones, indicating that students may readily adapt to these regions. Therefore, If a scene is fitted indiscriminately, the student is actually doing meaningless work on these well-learned regions. While our method liberates students from this

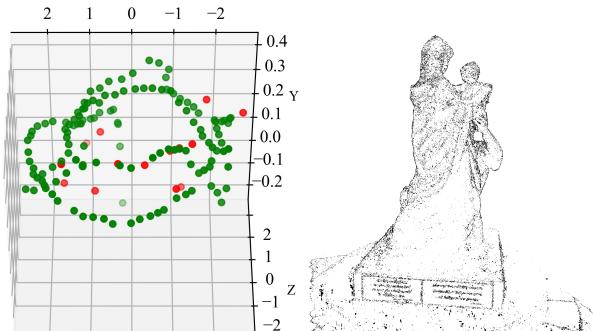| Family scene | (a) important poses | (b) important rays | (c) important points |

Fig. 7: Visualization of important camera poses, sample rays and sample points when distilling a student by PVD-AL on the Family scene from TanksAndTemple dataset.

futile "effort" and focuses their learning potential on the most important regions in a scene.

In Fig.7(c), we choose two important rays and displayed the weights of the sample points along the two rays. An approximate impulse distribution can be seen in these weights. The RGB value that corresponds to a ray is more significantly affected by these points with high weights. By informing students in advance of these significance points, students can prioritize fitting the key points when computing the sigma and color loss on the sample points.

TABLE 5: Using our active learning strategy as a plug-in to improve the performance of other NeRF-based distillation tasks. "+AL" indicates the active learning is used.

	Synthetic-NeRF		LLFF		TanksAndTemples	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
teacher	30.42	0.946	27.54	0.896	-	-
R2L [1]	30.02	0.938	27.19	0.89	-	-
R2L+AL	**30.35**	**0.94**	**27.27**	**0.891**	-	-
teacher	31.14	0.952	-	-	28.26	0.901
KiloNeRF [35]	28.02	0.921	-	-	26.18	0.793
KiloNeRF+AL	**29.21**	**0.943**	-	-	**27.41**	**0.847**

4.4.3 As Plug-in to Improve Other Methods

Our active learning strategy is not only available for PVD-AL, but it can also be simply applied to other distillation tasks and improve the performance of those tasks as well. For instance, we apply the active learning strategy proposed in this paper to R2L method [1] which distills a radiation field model into a light field model and KiloNeRF [35] which distills a high-capacity MLP to thousands of small MLPs. We only apply a two-level active learning strategy of sample rays and camera poses to the above models for ease of implementation. The visualization results are displayed in Fig.1(c), and the metric results are shown in Table 5. It can be seen that the performance of R2L and KiloNeRF has vastly improved since our active learning strategy was implemented, which indicates the generalizability of our method, i.e., our active learning strategy can be viewed as a plug-in to improve students' performance in other NeRF-related distillation tasks. Besides, it is worth noting that the design of the three levels of the active learning strategy in this study is decoupled, making it very adaptable when applied to other methods.

4.5 Editing Ability Conversion.

In addition to the various advantages mentioned above, one of the most important features of our approach is the implementation of property migration between different structures, which is the key motivation of this paper, i.e., *there is no single "best" architecture for neural rendering*. For example, we find that hash-based architecture is fast and produces high-quality modeling but lacks clear geometric interpretation due to the spatial aliasing caused by its hashtables. Hence, it is difficult to perform geometric editing like

erasing or combining. In addition, for MLP, its implicit space can be easily embedded with other features to achieve artistic-style rendering, but its geometry structure is not explicit. The Tensors structure, on the other hand, has an explicit geometric structure, but it is difficult to achieve an artistic design like MLP.

As one concrete example, we first run an editing experiment by distilling between MLP and Tensors. We train an MLP-based model with appearance [25] in Lego scene as shown in Fig.8(a)(2), a Tensors-based model with erasing bucket as shown in Fig.8(a)(3). Then we can transfer the appearance ability from MLP to Tensors by distilling the appearance-MLP to Tensors, and simply erase the bucket in the scene by Tensors' clear geometric. Then, we obtain a scene with both appearance and geometry changes represented by Tensors. Likewise, we can also transfer the erasing ability from Tensors to MLP to empower MLP with two editing capabilities as shown in Fig.8(a)(4).

The decomposition and composition of scenes is another illustration of the migration property between implicit MLP and explicit Tensors. The hidden space vector of the MLP can be employed as a retrievable feature to achieve semantic-level deconstruction [12], as shown in Fig.8(b)(2). The unambiguous spatial geometric relationship of explicit Tensors can be used to combine various scenes [17], as shown in Fig.8(b)(3). Our method may simultaneously realize decomposition and composition and describe it as any architecture like in Fig.8(b)(4).

The third illustration is the aesthetic editing of a specific object in a scene utilizing implicit MLP-based models [12], as seen in Fig.8(c)(2). Nevertheless, MLP-based models typically suffer from sluggish inference speed and low FPS, reducing their usefulness in practical applications. In this instance, we can transfer the MLP scenes with artistic effects to other fast models (such as VM and Hash) by our PVD-AL, as depicted in Fig.8(c)(3) and Fig.8(c)(4). It can be seen that the FPS of the Hash-based model is approximately $120\times$ greater than that of the MLP-based model.

It should be emphasized that our method does not place a cap on the number of characteristics that can be migrated. Hence, all the characteristics shown in Fig.8 can be incorporated into a single model. By focusing on the appearance, erasing, editing color, decomposition, composition, enlarge or shrink, high-FPS and high-performance properties from many structures on one structure, as seen in Fig.1(a), we have successfully merged the benefits of various structures, which was something that was previously difficult to accomplish on a single structure.

4.6 Compress Model Size and Training Time

Our method is able to produce better outcomes with a more concise parameter representation for students with inferior modeling capabilities since it can help students study more thoroughly. In order to confirm this, we attempt to compress the parameters of the NeRF [39] and Plenoxels [10], and compare the performance between the model obtained by PVD-AL and the model trained from scratch with the same number of iterations. The results are displayed in Table 7. It can be observed that the model produced
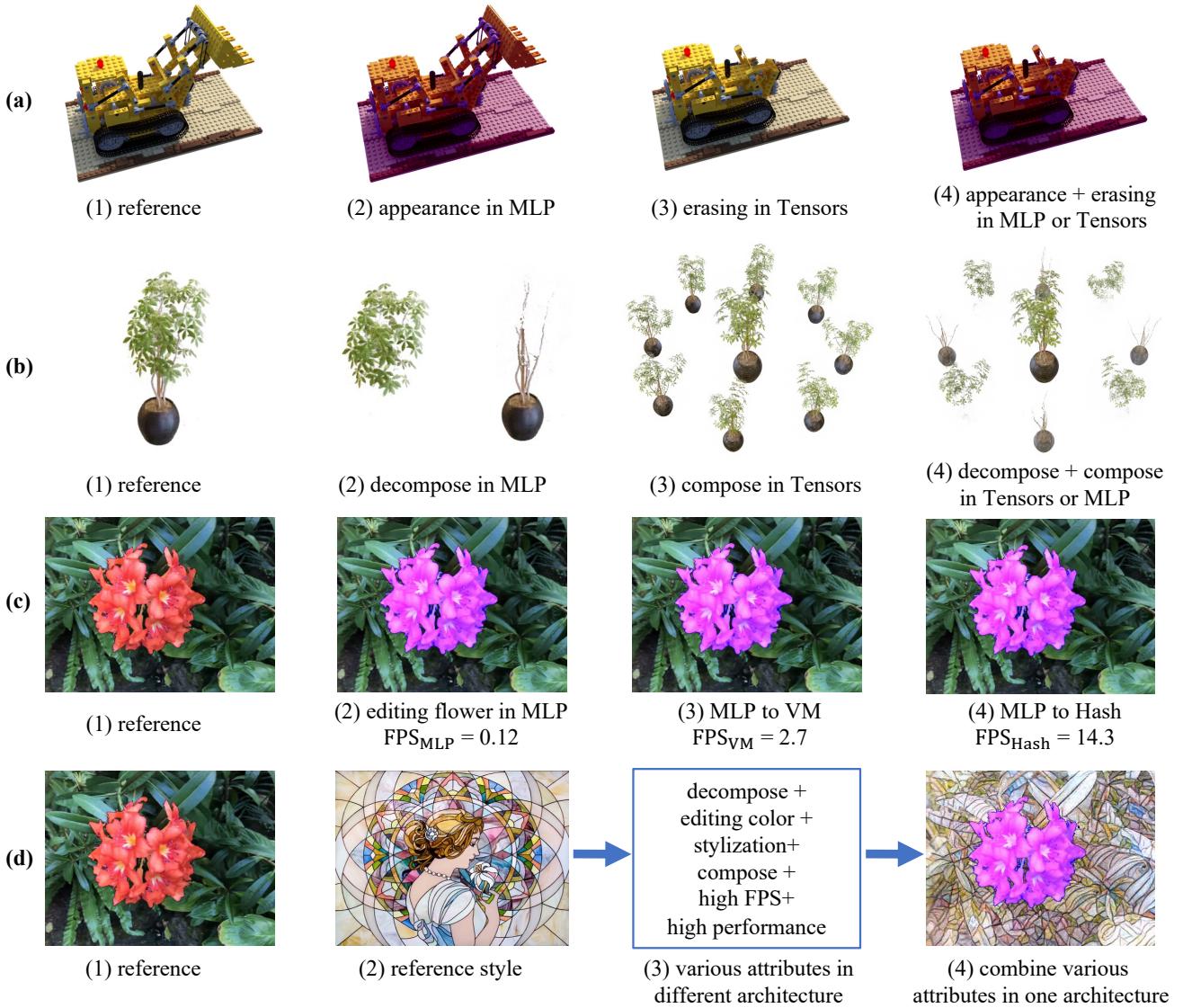
Fig. 8: Transferring of different characteristics between diverse architectures. Our PVD-AL enables the superposition of distinct editing skills and the integration of editing capabilities and high-speed benefits.

TABLE 6: Finetuning results after PVD-AL. Metrics are averaged over the scenes on the Synthetic-NeRF, LLFF, and TanksAndTemple datasets. Case 1: The teacher is superior in modeling capabilities. Case 2:The student is superior in modeling capabilities.

	Synthetic-NeRF						LLFF						TanksAndTemples					
	t-Hash (Case 1)			t-Tensors (Case 2)			t-Hash (Case 1)			t-Tensors (Case 2)			t-Hash (Case 1)			t-Tensors (Case 2)		
	PSNR	SSIM	$LPIPS_{alex}$	PSNR	SSIM	$LPIPS_{alex}$	PSNR	SSIM	$LPIPS_{alex}$	PSNR	SSIM	$LPIPS_{alex}$	PSNR	SSIM	$LPIPS_{alex}$	PSNR	SSIM	$LPIPS_{alex}$
	32.58	0.96	0.032	27.49	0.917	0.122	26.7	0.832	0.13	21.33	0.59	0.53	29.26	0.915	0.106	25	0.865	0.279
s-VM	**31.83**	**0.957**	0.038	27.48	0.916	0.121	25.98	0.793	**0.19**	21.39	0.591	0.529	**28.3**	0.906	**0.153**	24.96	0.865	0.28
s-VM$_{ft}$	31.72	0.956	**0.037**	**31.5**	**0.955**	**0.042**	**26.11**	**0.795**	0.192	**25.25**	**0.77**	**0.194**	28.22	0.906	0.156	**27.2**	**0.892**	**0.186**
s-MLP	**31.63**	**0.953**	0.043	27.38	0.914	0.119	**25.95**	0.786	0.201	21.33	0.592	0.53	**27.97**	**0.9**	**0.152**	24.78	0.863	0.273
s-MLP$_{ft}$	31.46	0.95	0.046	**30.81**	**0.948**	**0.045**	25.84	**0.788**	0.201	**25.04**	**0.75**	**0.22**	27.93	0.896	0.155	**25.89**	**0.877**	**0.205**

TABLE 7: Ability of PVD-AL to compress model parameters and reduce training time. The teacher is based on the representation of Hash.

| | Synthetic-NeRF | | | LLFF | | | TanksAndTemples | | |
Method	PSNR	Memory	Time	PSNR	Memory	Time	PSNR	Memory	Time
Plenoxles [10]	27.74	560M	**0.34h**	21.72	2.2G	**0.76h**	**25.75**	4.4G	**0.62h**
s-Tensors	**28.19**	**233M**	0.42h	**21.87**	**1.8G**	0.91h	25.43	**1.8G**	0.78h
NeRF [39]	**31.21**	14M	19.5h	25.2	14M	25.7h	26.02	14M	24.7h
s-MLP	31.14	**11M**	**1.4h**	**25.61**	**11M**	2.02h	**27.28**	**11M**	1.87h

by our method still performs fairly or even better with fewer parameters than training the model from scratch. Consequently, our method can be utilized as a compression tool, which is advantageous in situations with limited computing resources, such as terminals.

4.7 Finetuning Effects

As shown in Table 6, we divide the finetuning effects into two cases. Case 1: The teacher is superior in modeling capabilities. Finetuning has few benefits in this case. The main reason is that a superior teacher can provide sufficient pseudo datasets to train students adequately, therefore, the final performance boost from using real data is limited. Case 2: The student is superior in modeling capabilities. In this case, the performance of the student is improved after finetuning because the student's performance is capped by the teacher when distilling.

It should be noted that the primary role of our method is to exploit different properties of different structures as described in subsection 4.5. Hence it still makes sense to distill to a student architecture of inferior modeling capabilities. Nevertheless, common tricks like increasing model parameter numbers can be applied to better match the capabilities of teachers and students to avoid unnecessary loss of information.

5 CONCLUSION

In this work, we present PVD-AL, a systematic distillation method that allows conversions between different NeRF architectures, including MLP, sparse Tensors, low-rank Tensors, and hash tables, while maintaining high synthesis quality. Empirical experiments solidly demonstrate the efficiency of our approach, on both synthetic and real-world datasets, both measured in quantitative metrics and under visual inspection. Central to the success of PVD-AL is the careful design of loss functions, progressive distilling schemes utilizing intermediate volume representations, and special treatment of density values, which makes rapid work of the distillation process. Based on the modeling characteristics of the NeRF family of methods, we have also developed three tiers of active learning methodologies. Our approach continuously evaluates and updates the camera pose, sample rays, and sample points that are challenging for students to grasp, allowing the student to improve their understanding of these essential pieces of knowledge. The three layers of active learning strategies are decoupled, flexible in use, and highly versatile, and they can be easily applied as plug-in to other distillation tasks that use NeRF-based model as a teacher or student.

By breaking through the barriers between different architectures, PVD-AL allows downstream applications to optimally adapt the neural representation for the task at hand in a post hoc fashion. First of all our method can be used as a tool to train a model efficiently. Compared to training a model from scratch, our method is able to obtain a model in a faster and better-performing way. Secondly, thanks to the excellent performance of our method, it can be used as a tool to compress the NeRF family of models, resulting in more valuable models for applications. Last but not the least, our approach allows for the fusion of various properties between different structures. For example, we can call PVD-AL multiple times to obtain models with multiple editing properties. It is also possible to convert a scene under a specific model to another model that runs more efficiently to meet the real-time requirements of downstream tasks.

In view of certain limitations existing in the method, we consider possible corresponding solutions. For example, during distillation, the performance of the student will be limited by the teacher. In this case, it can be solved by finetuning the student or increasing the modeling ability of the teacher. In addition, both teacher and student models will be run during distillation, so out-of-memory may occur. At this time, one solution is to run our method in serial mode, that is, to use the teacher to generate a certain amount of data before distilling the student. Finally, we have not yet validated our approach to other NeRF-related technologies, such as the distillation of SDF, which is one of our future research expectations.

ACKNOWLEDGMENTS

REFERENCES

[1] H. Wang, J. Ren, Z. Huang, K. Olszewski, M. Chai, Y. Fu, and S. Tulyakov, "R2l: Distilling neural radiance field to neural light field for efficient novel view synthesis," *arXiv preprint arXiv:2203.17261*, 2022. 2, 3, 9

[2] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely, "Stereo magnification: Learning view synthesis using multiplane images," *arXiv preprint arXiv:1805.09817*, 2018. 1

[3] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein, "pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 5799–5809. 1

[4] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3d-structure-aware neural scene representations," *Advances in Neural Information Processing Systems*, vol. 32, 2019. 1

[5] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager, "Vision-only robot navigation in a neural radiance world," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4606–4613, 2022. 1

[6] A. Moreau, N. Piasco, D. Tsishkou, B. Stanciulescu, and A. de La Fortelle, "Lens: Localization enhanced by nerf synthesis," in *Conference on Robot Learning*. PMLR, 2022, pp. 1347–1356. 1

[7] S. Peng, Z. He, H. Zhang, R. Yan, C. Wang, Q. Zhu, and X. Liu, "Megloc: A robust and accurate visual localization pipeline," *arXiv preprint arXiv:2111.13063*, 2021. 1

[8] 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," in *European conference on computer vision*. Springer, 2020, pp. 405–421. 1, 2, 4, 5

[9] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, "Tensorf: Tensorial radiance fields," *arXiv preprint arXiv:2203.09517*, 2022. 1, 3, 4, 6

[10] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5501–5510. 1, 2, 3, 4, 6, 9, 10

[11] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *arXiv preprint arXiv:2201.05989*, 2022. 1, 3, 4, 6

[12] S. Kobayashi, E. Matsumoto, and V. Sitzmann, "Decomposing nerf for editing via feature field distillation," *arXiv preprint arXiv:2205.15585*, 2022. 1, 2, 9

[13] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer, "D-nerf: Neural radiance fields for dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 318–10 327. 1, 2

[14] J. Gu, L. Liu, P. Wang, and C. Theobalt, "Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis," *arXiv preprint arXiv:2110.08985*, 2021. 1

[15] F. Zhan, Y. Yu, R. Wu, J. Zhang, and S. Lu, "Multimodal image synthesis and editing: A survey," *arXiv preprint arXiv:2112.13592*, 2021. 1

[16] Y. Li, Z.-H. Lin, D. Forsyth, J.-B. Huang, and S. Wang, "Climatenerf: Physically-based neural rendering for extreme climate synthesis," *arXiv e-prints*, pp. arXiv–2211, 2022. 1

[17] J. Tang, X. Chen, J. Wang, and G. Zeng, "Compressible-composable nerf via rank-residual decomposition," *arXiv preprint arXiv:2205.14870*, 2022. 1, 3, 9

[18] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, "Plenoctrees for real-time rendering of neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5752–5761. 1, 3

[19] S. Fang, W. Xu, H. Wang, Y. Yang, Y. Wang, and S. Zhou, "One is all: Bridging the gap between neural radiance fields architectures with progressive volume distillation," *arXiv preprint arXiv:2211.15977*, 2022. 2

[20] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, "Local light field fusion: Practical view synthesis with prescriptive sampling guidelines," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–14, 2019. 2, 5

[21] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3d-structure-aware neural scene representations," *Advances in Neural Information Processing Systems*, vol. 32, 2019. 2

[22] 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. 2

[23] S. Bi, Z. Xu, K. Sunkavalli, M. Hašan, Y. Hold-Geoffroy, D. Kriegman, and R. Ramamoorthi, "Deep reflectance volumes: Relightable reconstructions from multi-view photometric images," in *European Conference on Computer Vision*. Springer, 2020, pp. 294–311. 2

[24] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8748–8763. 2

[25] 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. 2, 9

[26] S. Rasmuson, E. Sintorn, and U. Assarsson, "Perf: performant, explicit radiance fields," *Frontiers in Computer Science*, vol. 4, p. 871808, 2022. 3

[27] M. Usvyatsov, R. Ballester-Rippoll, L. Bashaeva, K. Schindler, G. Ferrer, and I. Oseledets, "T4dt: Tensorizing time for learning temporal 3d visual data," *arXiv preprint arXiv:2208.01421*, 2022. 3

[28] 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. 3

[29] L. Wu, J. Y. Lee, A. Bhattad, Y.-X. Wang, and D. Forsyth, "Diver: Real-time and accurate neural radiance fields with deterministic integration for volume rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 200–16 209. 3

[30] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt, "Neural sparse voxel fields," *NeurIPS*, 2020. 3

[31] S. Wizadwongsa, P. Phongthawee, J. Yenphraphai, and S. Suwajanakorn, "Nex: Real-time view synthesis with neural basis expansion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8534–8543. 3

[32] G. Hinton, O. Vinyals, J. Dean *et al.*, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, vol. 2, no. 7, 2015. 3

[33] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021. 3

[34] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-nerf 360: Unbounded anti-aliased neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5470–5479. 3

[35] 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. 3, 9

[36] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, "A survey of deep active learning," *ACM computing surveys (CSUR)*, vol. 54, no. 9, pp. 1–40, 2021. 3

[37] X. Pan, Z. Lai, S. Song, and G. Huang, "Activenerf: Learning where to see with uncertainty estimation," in *European Conference on Computer Vision*. Springer, 2022, pp. 230–246. 4

[38] 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. 5

[39] L. Yen-Chen, "Nerf-pytorch," https://github.com/yenchenlin/nerf-pytorch/, 2020. 6, 9, 10

[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019. 7

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 7

[42] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595. 7