One Is All: Bridging the Gap between Neural Radiance Fields Architectures with Progressive Volume Distillation

Shuangkang Fang[1]*, Weixin Xu[2], Heng Wang[2], Yi Yang[2], Yufeng Wang[1]†, Shuchang Zhou[2]†

[1] Beihang University
[2] Megvii Research
{skfang, wyfeng}@buaa.edu.cn, {xuweixin02, wangheng, yangyi, zsc}@megvii.com

Abstract

Neural Radiance Fields (NeRF) methods have proved effective as compact, high-quality and versatile representations for 3D scenes, and enable downstream tasks such as editing, retrieval, navigation, etc. Various neural architectures are vying for the core structure of NeRF, including the plain Multi-Layer Perceptron (MLP), sparse tensors, low-rank tensors, hashtables and their compositions. Each of these representations has its particular set of trade-offs. For example, the hashtable-based representations admit faster training and rendering but their lack of clear geometric meaning hampers downstream tasks like spatial-relation-aware editing. In this paper, we propose Progressive Volume Distillation (PVD), a systematic distillation method that allows any-to-any conversions between different architectures, including MLP, sparse or low-rank tensors, hashtables and their compositions. PVD consequently empowers downstream applications to optimally adapt the neural representations for the task at hand in a post hoc fashion. The conversions are fast, as distillation is progressively performed on different levels of volume representations, from shallower to deeper. We also employ special treatment of density to deal with its specific numerical instability problem. Empirical evidence is presented to validate our method on the NeRF-Synthetic, LLFF and TanksAndTemples datasets. For example, with PVD, an MLP-based NeRF model can be distilled from a hashtable-based Instant-NGP model at a 10×∼20× faster speed than being trained the original NeRF from scratch, while achieving a superior level of synthesis quality. Code is available at *https://github.com/megvii-research/AAAI2023-PVD*.

Introduction

Novel view synthesis (NVS) generates photo realistic 2D images for unknown view-ports of a 3D scene (Zhou et al. 2018; Chan et al. 2021; Sitzmann, Zollhöfer, and Wetzstein 2019a), and has wide applications in rendering, localization, and robot arm manipulations (Adamkiewicz et al. 2022; Moreau et al. 2022; Peng et al. 2021), especially with the neural modeling capabilities offered by the recently developed Neural Radiance Fields (NeRF). 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 (Chen et al. 2022) that leverages VM (vector-matrix) decomposition and canonical polyadic decomposition (CPD), Fridovich-Keil et al. that exploits the sparsity of the tensor, and Instant Neural Graphics Primitives (INGP) (Müller et al. 2022) that utilizes multilevel hash tables for effective compression of feature tensors.

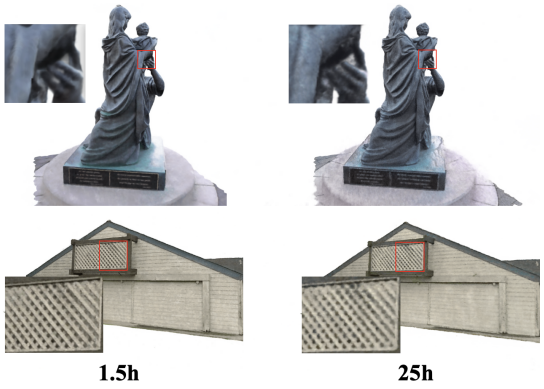

Figure 1: Comparison of two models trained in the Family and Barn scene from TanksAndTemples dataset. The left is the results of a NeRF model distilled by PVD from an INGP teacher within 1.5 hours. The right is the results of NeRF trained from scratch using 25 hours. PVD improves synthesis quality and reduces training time.

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 (Tang et al. 2022; Kobayashi, Matsumoto, and Sitzmann 2022; Pumarola et al. 2021; Gu et al. 2021; Zhan et al. 2021). 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,

*Work done during an internship at Megvii.
†Corresponding authors.

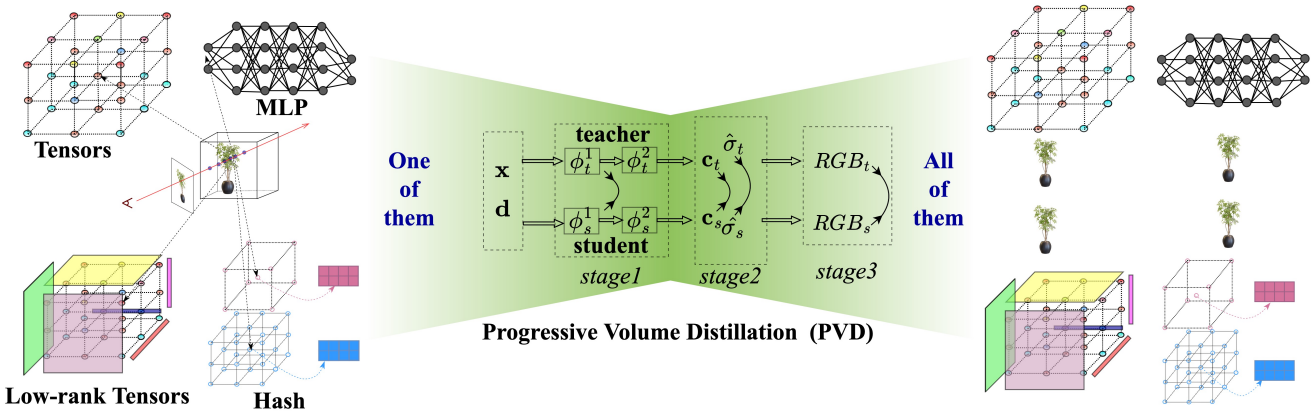

Figure 2: With PVD, given one trained NeRF model, different NeRF achitecutres, like sparse tensors, MLP, low-rank tensors and hash tables can be obtained quickly through distillation. The loss in intermediate volume representations (shown as double arrow symbol) like output of ϕ_*^1, color and density are used alongside the final rendered RGB volume to accelerate distillation.

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. 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 teacher and student are of different attributes. For example, when a teacher model with hash table is used to distill a student model of explicit representation, it is now possible to benefit from the faster training speed of the teacher while still producing a student model with clear geometric structures.

The way we realize conversions between different NeRF architectures is PVD, a progressive volume distillation method that operates on different levels of volume representations, from shallower to deeper, with special treatment of the density volume for better numerical stability. In contrast to previous methods proposed for distillation between models of the same architecture, PVD offers any-to-any conversion between possibly heterogeneous NeRF architectures, by first constructing a unified view of them, and then employing a systematic progressive distillation in multiple stages. Our contributions are summarized as follows.

- We propose PVD, a distillation framework that allows conversions between different NeRF architectures, including the MLP, sparse tensor, low-rank tensor and hash table architectures. To the best of our knowledge, this is the first systematic attempt at such conversions. An array of any-to-any conversion results is presented in Fig. 3.
- In PVD, we build a block-wise distillation strategy to accelerate the training procedure based on a unified view of different NeRF architectures. 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.
- As concrete examples, we find that distillation from hashtable and VM-decomposition structures often either helps boost student model synthesis quality and consumes less time than training from scratch. A particular beneficial case, where a NeRF student model is distilled from an INGP teacher, is presented in Fig. 1.

Related Work

Neural Implicit Representations

Neural implicit representation methods use MLP to construct a 3D scene from coordinate space, as proposed in NeRF (Mildenhall et al. 2020). 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 (Mildenhall et al. 2019; Sitzmann, Zollhöfer, and Wetzstein 2019b; Lombardi et al. 2019; Bi et al. 2020). The advantage of implicit modeling is that the representation is conducive to controlling or changing texture-like attributes of the scene. For example, Kobayashi, Matsumoto, and Sitzmann use the pretrained CLIP model (Radford et al. 2021) to induce editing of NeRF representation of a scene. Pumarola et al. successfully apply NeRF to the rendering of dynamic scenes by mapping time t to implicit space through an MLP. Martin-Brualla et al. realize the

control of scene lighting by adding appearance embedding. However, MLP-based NeRF 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.

Neural Explicit Representations and Hybrids

With the explicit representations, the scene is placed directly on a 3D grid (a huge tensor). Each voxel on the grid stores the information of density and color. Fridovich-Keil et al. first show that a 3D scene can 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 point 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 (Tang et al. 2022) decomposes the explicit tensor into low-rank components, which significantly reduces the model size. Rasmuson, Sintorn, and Assarsson continue 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 Gauss-Newton method. With explicit representation, it is not as easy to make artistic creations as with implicit representation. Nevertheless, explicit representations facility the geometry editing of the scene, including merging of multiple scenes, inpainting and manipulations of objects at specific positions. There are also attempts exploiting a hybrid of the explicit and implicit representations as NeRF architectures (Usvyatsov et al. 2022; Garbin et al. 2021; Müller et al. 2022; Chen et al. 2022; Wu et al. 2022). The explicit part usually stores features related to the scene, while the implicit part is typically an MLP that interpret the features to obtain densities and colors. Differences between hybrid representations are mainly exhibited in the explicit part. Liu et al. use a spare grid to store features, while Yu et al. optimize the 3D grid through an octree. Wizadwongsa et al. propose an Implicit-Explicit modeling strategy by storing the coefficient as a learnable parameter to accelerate training procedure. Recently, Müller et al. propose the multi-resolution hash encoding (MHE), which maps the given coordinate to feature via a cascade of hash tables at different scales. Like TensoRF (Chen et al. 2022), MHE significantly reduces memory footprint and improve inference speed. However, the compactness of MHE comes at a cost of less straightforward geometric interpretation as there are abundant spatial aliasings caused by the hash mechanism.

Knowledge Distillation

Knowledge distillation commonly refers to training a small model to match the output of a larger model (may be trained beforehand or on-the-fly) (Xu et al. 2021), which is widely used in model optimization and compression (Hinton et al. 2015; Gou et al. 2021). Multiple attempts have been made in the field of NVS. Barron et al. propose an online distillation method to improve the quality of rendering. Wang et al. distill a NeRF model into a model based on neural light fields. The most related to our work is KiloNeRF (Reiser et al. 2021), 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.

Method

Our method aims to achieve mutual conversions between different architectures of Neural Radiance Fields. Since there is 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 preliminaries before moving to a detailed description of our method.

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:

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

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 (tensor) (Fridovich-Keil et al. 2022). Each grid point stores density and spherical harmonic (SH) coefficients. The color c is obtained according to the SH and the view direction $\mathbf{d}$ as follows:

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

where $S : x \mapsto (1 + \exp(-x))^{-1}$, $\mathbf{k} = (k_\ell^m)_{\ell:0\le\ell\le\ell_{\max}}^{m:-\ell\le m\le\ell}$, 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) (Chen et al. 2022) decomposition 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:

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

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 density and color directly but features that can be decoded by an MLP.

Multi-resolution Hash Encoding INGP (Müller et al. 2022) maps a series of grids of different scales to the corresponding feature vectors with fixed size. INGP uses a hash function as in Equation (4) to map a spatial point in the grid to a hash table with different resolution that is adopted to details of different levels of these grids.

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

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.

PVD: Progressive Volume Distillation

Next we outline the details of PVD. Given a trained model, our task is to distill it into other models, possibly with different architectures. In PVD, we design a volume-aligned loss and build a blockwise distillation strategy to accelerate the training procedure based on a unified view of different NeRF architectures. 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. The illustration of our method is shown in Fig. 2.

Loss Design In our method, we not only use the RGB, but also use 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 this 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:

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

methods	ϕ_*^1	ϕ_*^2
NeRF	first K layers	remaining MLP
INGP	hash tables	MLP decoder
TensoRF	decomposed tensors	MLP decoder
Plenoxels	full	identity function

Table 1: The division of each architecture under our unified two-level view. Regarding NeRF, K=4 is used by default in this paper.

Here * can be either a teacher or a student. 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 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:

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

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. Our complete loss function during distillation is as follows:

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

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. Please refer to supplementary materials for more details.

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 the 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 play a key role (a more detailed analysis is in the supplementary material). 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:

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

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

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 the training and distillation significantly. While in our PVD, thanks to the designed of $\mathcal{L}_2^v$, we can implement the block-wise strategy to get rid of this problem. Specifically, we only forward stage1 at the beginning of training, and then run stage2 and stage3 in turn as

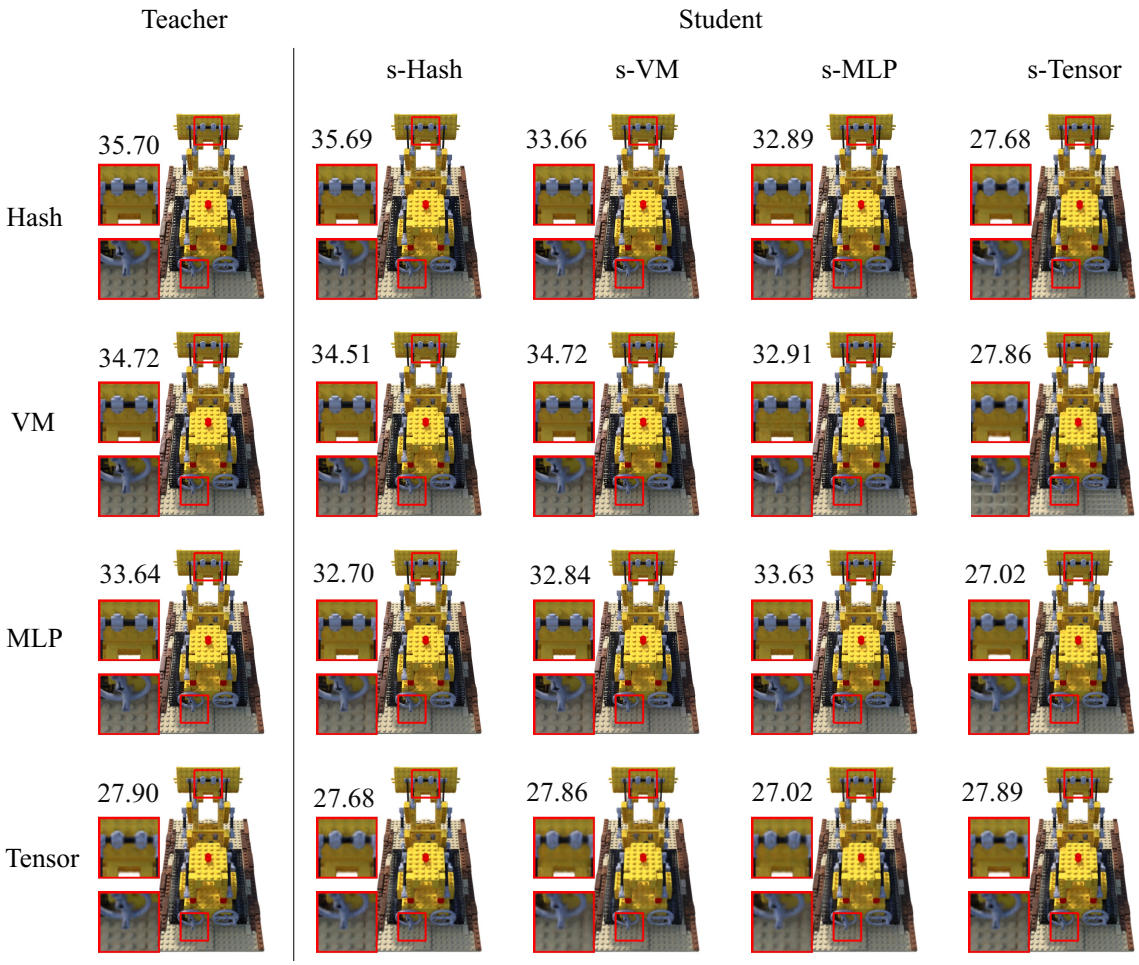

Figure 3: Quantitative and qualitative results of mutual-conversion between Hash / VM-decomposition / MLP / sparse tensors on the Lego scene from the NeRF-Synthetic dataset. We first train a teacher model for each structure, then use them to distill the student models. The numbers indicate PSNR of the quality of the synthesis. See the supplementary material for more results.

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.

Experiments

Implementation Details

Dataset. Our experiments are mainly carried out on the following three datasets: NeRF-Synthetic dataset (Mildenhall et al. 2020), forward-facing dataset (LLFF) (Mildenhall et al. 2019) and TanksAndTemple dataset (Knapitsch et al. 2017). We only use the above datasets for the training of teacher models. In the distillation stage, we find it sufficient to utilize the teacher to generate fake data as in *pseudo-labeling*, and not touch any of the training data.

Network Architecture. For each structure (Hash / MLP / VM-decomposition / sparse tensors), we keep consistent with their original settings as much as possible. For MLP (Yen-Chen 2020), we also use positional encoding for coordinates and view directions. For sparse tensors (Fridovich-Keil et al. 2022), we use spherical harmonics of degree 2, and the $128 \times 128 \times 128$ grid for NeRF-Synthetic dataset and TankAndTemple dataset, $512 \times 512 \times 128$ grid for LLFF dataset. For VM-decomposition (Chen et al. 2022), we take 48 components totally. For Hash (Müller et al. 2022), we set the coarsest resolution, the finest resolution, levels, hash table size and feature dimensions to 16, $2048 \times$ scene size, 14, 2^{19}, and 2 respectively.

Training and Distilling Details. We implement our method with the PyTorch framework (Paszke et al. 2019) to train teachers and distill students. We use Adam Optimizer (Kingma and Ba 2014) with initial learning rates of 0.02 and run 20k steps with batchsize 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 rest steps. All the experiments are performed on a single NVIDIA V100 GPU.

student	Teacher											
	PSNR↑				SSIM↑				LPIPS$_{Alex}$ ↓			
	Hash	VM	MLP	Tensors	Hash	VM	MLP	Tensor	Hash	VM	MLP	Tensor
	32.58	31.52	30.78	27.49	0.960	0.955	0.946	0.917	0.032	0.040	0.049	0.122
s-Hash	32.58	30.96	30.52	27.32	0.960	0.949	0.944	0.913	0.032	0.047	0.053	0.119
s-VM	31.33	31.52	30.29	27.46	0.954	0.955	0.944	0.916	0.042	0.040	0.056	0.121
s-MLP	30.76	30.49	30.78	26.87	0.946	0.945	0.946	0.906	0.056	0.055	0.049	0.127
s-Tensors	27.85	27.72	27.44	27.49	0.921	0.921	0.918	0.917	0.100	0.099	0.098	0.122

Table 2: The qualitative results(PSNR / SSIM / LPIPS$_{Alex}$) of mutual-conversion between Hash / VM-decomposition / MLP / sparse tensors representations on NeRF-Synthetic dataset. The top number of each column represents the metric of the teacher, and the four numbers below represent the metric of the student obtained by distillation from the teacher. The s- means distillation.

method	TanksAndTemple				LLFF			
	PSNR	SSIM	LPIPS$_{Alex}$	LPIPS$_{VGG}$	PSNR	SSIM	LPIPS$_{Alex}$	LPIPS$_{VGG}$
Teacher-Hash	29.26	0.915	0.134	0.106	26.70	0.832	0.231	0.130
TensoRF-VM	**28.06**	**0.909**	**0.145**	**0.155**	**26.51**	**0.832**	**0.217**	**0.135**
Ours: s-VM	27.86	0.899	0.176	0.181	25.73	0.793	**0.195**	0.269
NeRF	25.78	0.864	-	-	**26.50**	**0.811**	0.250	-
Ours: s-MLP	**27.50**	**0.891**	**0.194**	0.190	25.77	0.784	**0.213**	0.310
Plenoxels	25.18	0.865	**0.219**	0.261	**21.69**	**0.607**	**0.527**	0.527
Ours: s-Tensors	**25.31**	**0.866**	0.263	**0.220**	21.36	0.600	0.561	**0.524**

Table 3: Comparison of the qualitative results of models (s-VM, s-MLP, s-Tensors) obtained by our distillation method with the models (TensoRF-VM, NeRF, Plenoxels) trained from scratch on LLFF and TanksAndTemples datasets.

Please check the supplementary materials for more details.

Performance and Efficiency

It should be noted that this is the first time to propose a conversion method between different representations, so we do not have any comparable baseline. Our experiments mainly focus on whether the conversion between different models can maintain the performance of the teacher or its own upper limit. And we also expect to get some benefits from the distillation between different structures.

Quantitative Results For four representations (Hash / VM-decomposition / MLP / sparse tensors), we first train the models of each representation from scratch in 8 scenes on the NeRF-Synthetic dataset, and a total of 32 models are obtained as teachers. Then using the PVD proposed in this article to convert these teachers into the students with different structures. At the same time, we also consider the conversion between the same structures. We count the average metrics in Table 2 after the conversion is complete. It can be seen that our method is very effective for the conversion. When a model is transformed into another forms, its performance has little difference with the result of training the model from scratch or the result of the teacher, which fully shows that the common representations based on radiance fields can be converted into each other. In addition, our PVD shows excellent nearly nondestructive performance in distillation between the same structures.

In Fig.4, we can see that the value of $\max(diff_1, diff_2)$ is very close to 0, which means that the model obtained by distillation can be close to the teacher or training it from scratch. The performance of students is mainly limited by two aspects, one is the performance of teachers, and the other is the fitting ability of the student itself. Fig.4 shows strong evidence that our method has migrated knowledge from teacher to student to the maximum extent.

We further verify our method in Table 3 on the LLFF and TanksAndTemples datasets. We use INGP as a teacher to distill NeRF, VM-decomposition and Plenoxels, and we compare them with the results obtained by training from scratch of these students. It can be seen from Table 3 that our method is also effective on these two datasets. It is gratifying that the NeRF model obtained by our distillation performs better than its original implementation on TanksAndTemples dataset. This is mainly due to the fact that our PVD method provides more prior information to students, making training more efficient and fully improving the expression limit of the student. In addition to the possibility of improving the performance of the model, we also show another benefit from our method in Table 4. It can be clearly seen that our method obtains a NeRF model significantly faster than

method	Lego	Orchids	Truck
NeRF	32.54/30h	20.36/35h	25.36/35h
s-MLP	31.83/30min	20.61/100min	23.98/30min
s-MLP	**32.70**/1.5h	**21.25**/3h	**26.69**/1.7h

Table 4: Comparison of running time. The teacher is based on the representation of VM-decomposition. We calculate the PSNR at different times for student and NeRF trained form scratch.

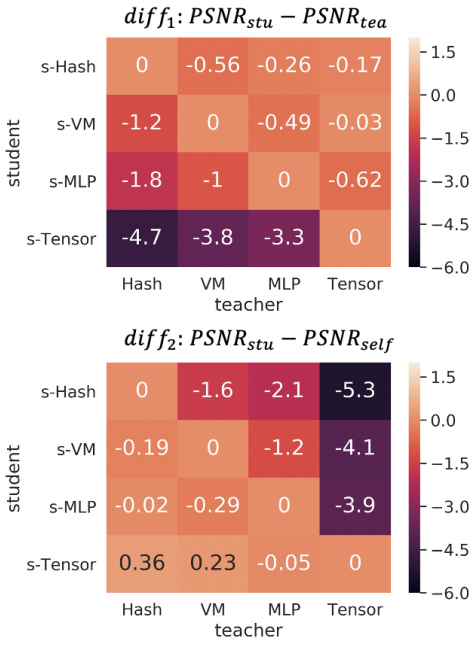

Figure 4: Gaps in PSNR of mutual-conversion in Synthetic-NeRF dataset. $PSNR_{stu}$ indicates the PSNR of student obtained by distillation. $PSNR_{self}$ represents the PSNR of student obtained by training it from scratch. $PSNR_{tea}$ is the PSNR of the teacher.

training the model from scratch. As we mentioned earlier, distilling from a large NeRF model to a smaller one is typically inefficient due to the need to frequently query the large model. While our distillation between heterogeneous forms can achieve a more efficient distillation.

Qualitative Results Fig. 3 shows the qualitative results of mutual-conversion between Hash, VM-decomposition, MLP, and sparse tensors on NeRF-Synthetic dataset. We can see the excellent properties of PVD in maintaining the synthesis quality, as the visual quality of the student is often indistinguishably close to either the teacher or trained from scratch. We also show the result on TanksAndTemples dataset in Fig.1. Our s-MLP achieves a better synthesis quality than NeRF training from scratch. The improvement is mainly due to the distillation between different structures. A powerful teacher can let the student approach its upper limit of expression capability. In addition, Fig.5 shows that our method not only maintains the synthesis quality but also maintains the accuracy of the depth information of the scene.

Ablation Studies and Limitations

Our ablation studies demonstrate the degree of influence of each component in our method on the performance. We implement the conversion from VM-decomposition to MLP on the Synthetic-Nerf dataset as in Table 5. It can be seen the intermediate feature loss we designed brings about 0.9dB PSNR improvement. It can also be seen that the performance will drop sharply without the restriction on the value of density. We also take the distillation without using block-wise

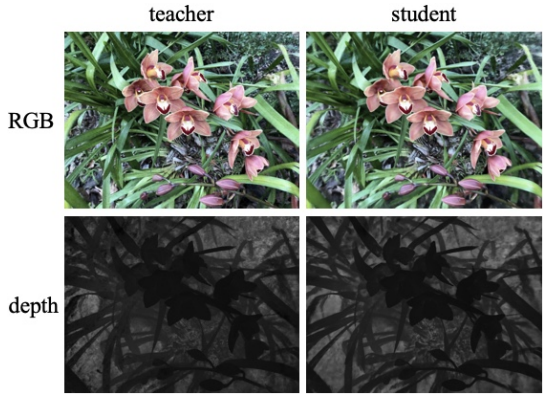

Figure 5: Qualitative comparison of depth in Orchids scene from LLFF dataset. The teacher is INGP and the student is s-MLP.

	PSNR↑	SSIM↑	LPIPS$_{Alex}$ ↓
w/o $\mathcal{L}_2^v$	29.63	0.937	0.065
w/o $\mathcal{L}_2^\sigma$	30.01	0.939	0.063
w/o $\mathcal{L}_2^c$	29.95	0.938	0.063
w/o $\mathcal{L}_2^{rgb}$	27.07	0.908	0.945
w/o sigma-constrain	28.45	0.929	0.074
w/o block-wise	29.62	0.941	0.060
w/all	**30.49**	**0.945**	**0.055**

Table 5: An ablation study of our method. Metrics are averaged over the 8 scenes from NeRF-Synthetic dataset in the conversion from VM-decomposition to s-MLP.

strategy, and we find that it attains poor performance under the same budget of training time.

Our method also has some limitations inherited from the distillation. For example, the performance of student models is generally upper-bounded by the performances of teacher models, and in those cases further finetuning may be beneficial. Similarly, the modeling ability of the student model may limit its final performance. In addition, as both teacher and student models need be active during training, memory and computation cost will be duly increased.

Conclusions

In this work, we present PVD, a systematic distillation method that allows conversions between different NeRF architectures, including MLP, sparse tensor, low-rank tensor, and hash tables, while maintaining high sysnthesis quality. Central to the success of PVD is careful design of loss functions, a progressive distilling schemes utilizing intermediate volume representations, and special treatment of density values. By breaking through the barriers between different architectures, PVD allows downstream applications to optimally adapt the neural representation for the task at hand in a post hoc fashion. Empirical experiments solidly demonstrate the efficiency of our approach, on both synthetic and realworld datasets, both measured in quantitative PSNR and under visual inspection.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (U20B2042, 62076019), Science and Technology Innovation 2030-Key Project of "New Generation Artificial Intelligence"(2020AAA0108201).

References

Adamkiewicz, M.; Chen, T.; Caccavale, A.; Gardner, R.; Culbertson, P.; Bohg, J.; and Schwager, M. 2022. Vision-only robot navigation in a neural radiance world. *IEEE Robotics and Automation Letters*, 7(2): 4606–4613.

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

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

Chan, E. R.; Monteiro, M.; Kellnhofer, P.; Wu, J.; and Wetzstein, G. 2021. 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*, 5799–5809.

Chen, A.; Xu, Z.; Geiger, A.; Yu, J.; and Su, H. 2022. TensoRF: Tensorial Radiance Fields. *arXiv preprint arXiv:2203.09517*.

Fridovich-Keil, S.; Yu, A.; Tancik, M.; Chen, Q.; Recht, B.; and Kanazawa, A. 2022. Plenoxels: Radiance Fields Without Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5501–5510.

Garbin, S. J.; Kowalski, M.; Johnson, M.; Shotton, J.; and Valentin, J. 2021. Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 14346–14355.

Gou, J.; Yu, B.; Maybank, S. J.; and Tao, D. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6): 1789–1819.

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

Hinton, G.; Vinyals, O.; Dean, J.; et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Knapitsch, A.; Park, J.; Zhou, Q.-Y.; and Koltun, V. 2017. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4): 1–13.

Kobayashi, S.; Matsumoto, E.; and Sitzmann, V. 2022. Decomposing NeRF for Editing via Feature Field Distillation. *arXiv preprint arXiv:2205.15585*.

Liu, L.; Gu, J.; Lin, K. Z.; Chua, T.-S.; and Theobalt, C. 2020. Neural Sparse Voxel Fields. *NeurIPS*.

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

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

Mildenhall, B.; Srinivasan, P. P.; Ortiz-Cayon, R.; Kalantari, N. K.; Ramamoorthi, R.; Ng, R.; and Kar, A. 2019. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4): 1–14.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, 405–421. Springer.

Moreau, A.; Piasco, N.; Tsishkou, D.; Stanciulescu, B.; and de La Fortelle, A. 2022. LENS: Localization enhanced by NeRF synthesis. In *Conference on Robot Learning*, 1347–1356. PMLR.

Müller, T.; Evans, A.; Schied, C.; and Keller, A. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*.

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

Peng, S.; He, Z.; Zhang, H.; Yan, R.; Wang, C.; Zhu, Q.; and Liu, X. 2021. MegLoc: A Robust and Accurate Visual Localization Pipeline. *arXiv preprint arXiv:2111.13063*.

Pumarola, A.; Corona, E.; Pons-Moll, G.; and Moreno-Noguer, F. 2021. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10318–10327.

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

Rasmuson, S.; Sintorn, E.; and Assarsson, U. 2022. PERF: performant, explicit radiance fields. *Frontiers in Computer Science*, 4: 871808.

Reiser, C.; Peng, S.; Liao, Y.; and Geiger, A. 2021. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 14335–14345.

Sitzmann, V.; Zollhöfer, M.; and Wetzstein, G. 2019a. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems*, 32.

Sitzmann, V.; Zollhöfer, M.; and Wetzstein, G. 2019b. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems*, 32.

Tang, J.; Chen, X.; Wang, J.; and Zeng, G. 2022. Compressible-composable NeRF via Rank-residual Decomposition. *arXiv preprint arXiv:2205.14870*.

Usvyatsov, M.; Ballester-Rippoll, R.; Bashaeva, L.; Schindler, K.; Ferrer, G.; and Oseledets, I. 2022. T4DT: Tensorizing Time for Learning Temporal 3D Visual Data. *arXiv preprint arXiv:2208.01421*.

Wang, H.; Ren, J.; Huang, Z.; Olszewski, K.; Chai, M.; Fu, Y.; and Tulyakov, S. 2022. R2L: Distilling Neural Radiance Field to Neural Light Field for Efficient Novel View Synthesis. *arXiv preprint arXiv:2203.17261*.

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

Wu, L.; Lee, J. Y.; Bhattad, A.; Wang, Y.-X.; and Forsyth, D. 2022. 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*, 16200–16209.

Xu, W.; Feng, Z.; Fang, S.; Yuan, S.; Yang, Y.; and Zhou, S. 2021. Arch-Net: Model Distillation for Architecture Agnostic Model Deployment. *arXiv preprint arXiv:2111.01135*.

Yen-Chen, L. 2020. NeRF-pytorch. https://github.com/yenchenlin/nerf-pytorch/.

Yu, A.; Li, R.; Tancik, M.; Li, H.; Ng, R.; and Kanazawa, A. 2021. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5752–5761.

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

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