Fast and Explicit Neural View Synthesis

Pengsheng Guo Miguel Angel Bautista Alex Colburn Liang Yang
Daniel Ulbricht Joshua M. Susskind Qi Shan
Apple
{pengsheng_guo,mbautistamartin,alexcolburn,ericyang,dulbricht,jsusskind,qshan}@apple.com

Abstract

We study the problem of novel view synthesis from sparse source observations of a scene comprised of 3D objects. We propose a simple yet effective approach that is neither continuous nor implicit, challenging recent trends on view synthesis. Our approach explicitly encodes observations into a volumetric representation that enables amortized rendering. We demonstrate that although continuous radiance field representations have gained a lot of attention due to their expressive power, our simple approach obtains comparable or even better novel view reconstruction quality comparing with state-of-the-art baselines [49] while increasing rendering speed by over 400x. Our model is trained in a category-agnostic manner and does not require scene-specific optimization. Therefore, it is able to generalize novel view synthesis to object categories not seen during training. In addition, we show that with our simple formulation, we can use view synthesis as a self-supervision signal for efficient learning of 3D geometry without explicit 3D supervision.

1. Introduction

In order to understand the 3D world, an intelligent agent must be able to perform quick inferences about a scene's appearance and shape from unseen viewpoints given few observations. Being able to synthesize images at target camera viewpoints efficiently given sparse source views serves a fundamental purpose in building intelligent visual behaviour [10, 30, 40]. The problem of learning to synthesize novel views has been widely studied in literature, with approaches ranging from traditional small-baseline view synthesis relying on multi-plane imaging [6, 36, 51, 21], flow estimation [52, 37], to explicitly modeling 3D geometry via point-clouds [1], meshes [18], and voxels [7].

A recent wave of approaches for view synthesis have adopted continuous radiance field representations [49, 35, 22, 39, 29], where scenes are represented as a continuous function that shares its domain with the signal being fitted

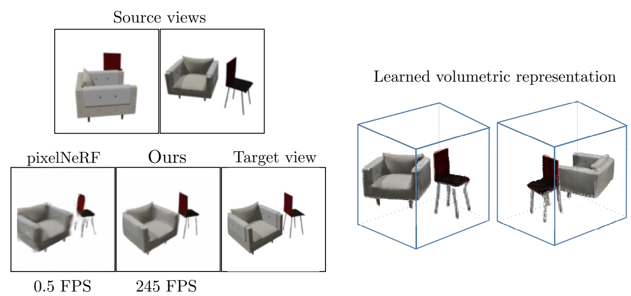


Figure 1: Our model performs scene-agnostic and category-agnostic novel view synthesis in real time. A complete 3D geometry of the scene is estimated through a single forward pass. Our model consistently produces higher quality results comparing to the state-of-the-art view synthesis approach pixelNeRF [49], while being over 400x faster in rendering time.

(*e.g.* a function that takes points in $\mathbb{R}^3$ as input, to model a 3D signal), as opposed to discrete representations where the 3D signals are encoded in a discrete geometric structure like a volume [7] or a mesh [18]. Although continuous radiance field representations enjoy the benefits of being resolution-free or modeling view-dependent effects, they are not efficient for real-world use cases that require real-time performance. Typically, radiance field representations have the following disadvantages. First, being computationally costly to obtain when implicitly modeled [22, 29, 31], *e.g.* the model parameters are optimized via gradient descent for each object or scene, usually taking tens of hours on commodity hardware. Second, requiring to densely capture observations of the scene being modeled [22, 29, 31] for optimization. Third, not being able to amortize the rendering cost across views, since radiance fields are evaluated independently for every pixel being rendered [49, 39]. This dramatically impacts the practicality of radiance field, since rendering an image can take seconds on modern GPUs.

What makes an approach for novel view synthesis useful? While photo-realistic results have been obtained with continuous/implicit representations, these approaches are severely impacted by capture, optimization and rendering time, hindering their practicality for deployment in systems that require real-time performance, *e.g*. the ability to infer views of unseen objects in real-time from sparse observations. Our approach enjoys the following benefits: *(i)* the scene representation is fast to obtain, as it does not require gradient-based optimization for new scenes and can be obtained from sparse observations, and *(ii)* it is efficient to render, since it models the complete 3D geometry and appearance in a single forward pass, allowing for amortized rendering. Our experiments show that despite the simplicity of our method, our performance notably matches or beats recent state-of-the-art baselines based on few-shot continuous scene representations across different metrics and settings, producing accurate novel view reconstruction, while rendering objects over 400x faster than the state-of-the-art, pixelNeRF [49]. In addition, we find that the 3D geometry learned by our model in an unsupervised manner (i.e. without the need to train with 3D geometry supervision) is extremely compelling and very efficient to obtain, requiring only a single forward pass of the model.

2. Related Work

Learning to synthesize novel views of an object or a scene given one or more sparse observation has been widely studied in the literature [6, 36, 51, 21, 4, 52, 3, 37, 7, 27, 1, 35]. A unifying problem definition for this set of approaches is to predict a target view given a source view/s, conditioned on a relative camera transformation. One set of approaches focuses on small and/or wide baseline view synthesis where the goal is to synthesize a parallax effect by using multi-plane imaging [6, 36, 51], local light-field fusion [21] or cost volume estimation [4]. Another set of approaches focuses on learning a free-form 2D flow field that takes pixels from a single [52] or multiple source views [37] and reconstructs a target view given the relative camera transformation between source/s and target.

In addition, there is an extensive literature on tackling view synthesis with voxel grid 3D representations [7, 27, 34, 53, 24, 11, 19, 41]. Although our approach uses a voxel grid 3D representation, it differs from existing work in the following. As opposed to [7, 27, 41] where convolutional 2D decoders are used to generate an image, our approach uses volumetric rendering to directly render an image from the explicit voxel grid representation. In contrast to [34, 19], our approach can generalize to multiple objects without per scene training/optimization. Moreover, our approach is trained in a category-agnostic way as opposed to [53, 24, 11], and it is trained on a large set of object categories (as opposed to 4 object categories in [41]) which can be generalized to unseen object categories (cf. Sect. 4).

In order to deal with the limitations of voxel grids, implicit representations that model continuous radiance fields for view interpolation [22, 29, 31, 32] have been proposed. These approaches learn a radiance field for every scene or object by fitting the parameters of a model (using gradient descent) to a dense set of views of a scene and then interpolating between those views. Note that this setting is different from the novel view synthesis setting where the problem is to predict a target view given a sparse source view and a relative camera transformations. However, recent approaches have applied continuous radiance fields to the novel view synthesis problem [49, 39], showing that it is possible to model multiple objects or scenes within a single model and extrapolating to object categories unseen during training. We can group recent approaches to novel view synthesis with implicit and continuous radiance field representations into two mutually exclusive categories. In the first category we find approaches that provide an efficient approach to explicitly encode source views into a continuous representation but are inefficient during rendering due not being able to amortize the rendering process across views [49, 39, 43] (see Sect. 3.3 for details). In the second category, we find recent approaches that enable efficient rendering through amortized rendering [48, 8, 45] but where their continuous representation is implicit, and must be fitted via gradient descent for every new object or scene (typically taking days on commodity hardware).

In this paper we present a simple yet powerful approach for novel view synthesis which explicitly encodes sources views into a volumetric representation that enables amortized rendering. Thus combining the best of both types of recent approaches for novel view synthesis.

3. Methodology

The novel view synthesis problem is defined as follows. Given a set $\mathcal{S} = \{(\mathbf{I}_i, \mathbf{P}_i)\}_{i=0}^{n}$ of one or more source views, where a view is defined as an image $\mathbf{I}_i \in \mathbb{R}^{3\times h\times w}$ together with the camera pose $\mathbf{P}_i \in SO(3)$, we want to learn a model f_θ that can reconstruct a ground-truth target image $\mathbf{I}_t$ conditioned on its pose $\mathbf{P}_t$, where the predicted target image is obtained as $\hat{\mathbf{I}}_t = f_\theta(\mathcal{S}, \mathbf{P}_t)$.

We design f_θ as a simple fully convolutional model that allows amortized rendering. Our model processes a source view through a 2D U-Net encoder [33] to produce a feature map that is projected onto a latent volumetric representation via an inverse projection step. This volumetric representation is further processed with a 3D U-Net model to learn an RGBα volume[1] to which the relative pose transformation between source and target views is applied, and finally rendered into the predicted target view. We illustrate our pipeline in Fig. 2.

[1]Note that we do not supervise training with an RGBα volume, the model is forced to learn the RGBα volume through the rendering process.

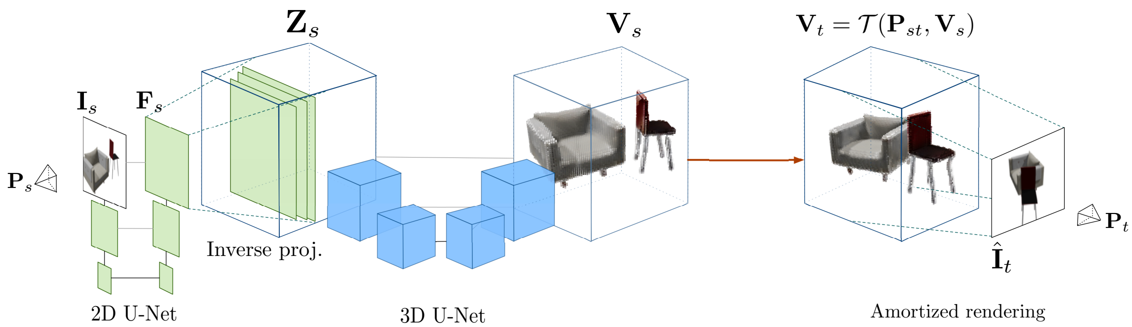


Figure 2: Our model is comprised of three main components: a) a 2D U-Net image encoder, b) a 3D U-Net scene encoder, and c) an amortized rendering process. The 2D U-Net encoder generates a 2D feature map $\mathbf{F}_s$ from the input image $\mathbf{I}_s$. The feature map is then projected into a latent volume $\mathbf{Z}_s$ via an inverse project step. A 3D U-Net network maps $\mathbf{Z}_s$ into an RGBα volume $\mathbf{V}_s$. This RGBα volume is applied a relative pose transformation $\mathbf{P}_{st}$ to match the target view pose $\mathbf{P}_t$, and the resulting image $\hat{\mathbf{I}}_t$ is created by rendering the RGBα using a simple volume rendering process that is amortized across views.

3.1. Encoding

The initial step of our model is to encode the source $\mathbf{I}_s \in \mathbb{R}^{3\times h\times w}$ with a fully convolutional U-Net encoder that produces a feature map $\mathbf{F}_s \in \mathbb{R}^{c\times h\times w}$ that preserves the spatial resolution of the source image. Once a feature map $\mathbf{F}_s$ is obtained, we cast the features along rays into a latent volumetric tensor using the perspective camera matrix. In practice we perform an inverse projection step to back-project $\mathbf{F}_s$ into a latent volumetric tensor $\mathbf{Z}_s \in \mathbb{R}^{c\times d_s\times h_s\times w_s}$, where d_s, h_s, w_s are depth, height and width for the volumetric representation [2]. Instead of reshaping 2D feature maps into a 3D volumetric representation [7, 27], we found that using an inverse projection step is beneficial to preserve the 3D geometry and texture information (cf. Sect. 4 for empirical evidence).

3.2. Learning a Renderable Volume

After the inverse projection step we simply process $\mathbf{Z}_s$ with a 3D U-Net [23] model and predict a final volume $\mathbf{V}_s \in \mathbb{R}^{4\times d_s\times h\times w}$. At this stage $\mathbf{V}_s$ encodes an RGBα volume of the object or scene that can be efficiently rendered. Similar to [49, 7] we apply the relative transformation $\mathbf{P}_{st} = \mathbf{P}_t\mathbf{P}_s^{-1}$ between source and target camera poses to the volumetric representation $\mathbf{V}_s$ to obtain a transformed volumetric representation $\mathbf{V}_t$ that is aligned with the target view. We define this transformation operation as a function $\mathcal{T}(\mathbf{P}, \mathbf{V})$ that takes as input a rigid transformation $\mathbf{P} \in SO(3)$ and a volume $\mathbf{V}$ and applies the rigid transformation to the volume. Note that we define $\mathbf{P} \in SO(3)$, however, our formulation naturally extends to other transformations groups (*e.g*. non-rigid or free-form deformations).

[2]We used intrinsic camera parameters for the inverse projection step, which we assume to be constant.

3.3. Amortized Rendering

We now turn to the task of rendering an RGBα volume $\mathbf{V}$ into an image. Recent work on modeling scenes with continuous neural radiance fields [22] has shown great results by using the rendering equation [13] in order to model pixels. At rendering time, [22] propose to obtain a pixel value by tracing the camera ray $\mathbf{r}$ from the near plane t_n to the far plane t_f, and the expected color of a 2D pixel can be calculated as follows (see [22] for details):

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt \tag{1}$$

where $T(t) = \exp(-\int_{t_n}^{t_f} \sigma(\mathbf{r}(s))ds)$ denotes the accumulated transmittance between the near plane and the current point $\mathbf{r}(t)$ along the ray. In practice, numerical quadrature and stratified sampling strategies are adopted to discretize the continuous integral and make the computation viable.

However, a critical problem of the sampling process in NeRF [22] is that it prevents the rendering process to be *amortized* across views. This is because each ray integral in Eq. 1 is independent and points sampled to approximate one ray integral are not reusable for other ray integrals in the scene. Our approach side-steps the need to perform sampling by modelling the scenes complete geometry and appearance as an RGBα volume $\mathbf{V} \in \mathbb{R}^{4\times d_s\times h\times w}$. This allows us to amortize rendering across views (since all rendered images of a scene share the same RGBα volume) obtaining dramatic rendering speed improvements without sacrificing reconstruction accuracy with respect to recent baselines [49].

Before rendering our RGBα volume $\mathbf{V}$, we apply a perspective deformation (using intrinsic camera parameters) on the viewing frustum using inverse warping and trilinear sampling [12] (see Appendix for details). For a given pixel location (i, j), the expected color $\hat{C}$ is calculated as:

$$\hat{C}_{i,j} = \sum_{k=1}^{d_s} T_{i,j}^k \alpha_{i,j}^k \mathbf{c}_{i,j}^k, \text{ where } T_{i,j}^k = \prod_{m=1}^{k-1} (1 - \alpha_{i,j}^m) \quad (2)$$

where $\mathbf{c}_{i,j}^k$ is the color value encoded in the first 3 channels of $\mathbf{V}_t$ and $\alpha_{i,j}^k$ is the value at the last channel.

3.4. Multiple View Aggregation

Our model can take an arbitrary number of source views in $\mathcal{S}$ as input. In order to do so, we first obtain latent volumes $\mathbf{Z}_i$ for each source view i using the same encoding process as in Sect. 3.1. Next, we take an arbitrary source view i^* in the set of source views as the origin of the coordinate system. Latent volumes $\mathbf{Z}_i$ are then aligned to this origin using the relative transformation $\mathbf{P}_{i,i^*}$ between corresponding pose $\mathbf{P}_i$ and origin pose $\mathbf{P}_{i^*}$. After that, we pool the aligned volumes by taking the mean across views:

$$\bar{\mathbf{Z}} = \frac{1}{n} \sum_{i \in n} \mathcal{T}(\mathbf{P}_{i,i^*}, \mathbf{Z}_i) \quad (3)$$

Finally, the pooled volumetric latent $\bar{\mathbf{Z}}$ is fed to our 3D U-Net to generate an RGBα volume $\mathbf{V}$ which can be efficiently rendered as outlined in Sect. 3.3.

3.5. Training

Similar to [7, 49], we sample tuples of source and target views together with their relative transformation $(\mathbf{I}_s, \mathbf{I}_t, \mathbf{P}_{st})$ during training. We use the model f_θ to predict the target from source $\hat{\mathbf{I}}_t = f_\theta(\mathbf{I}_s, \mathbf{P}_{st})$ and minimize a rendering loss. The rendering loss is a weighted sum of ℓ_2 loss and SSIM [44] loss, defined as

$$\begin{aligned} \mathcal{L}_{\text{render}} = \sum_t & \|f_\theta(\mathbf{I}_s, \mathbf{P}_{st}) - \mathbf{I}_t\|_2^2 \\ & + \lambda \mathcal{L}_{\text{ssim}}(f_\theta(\mathbf{I}_s, \mathbf{P}_{st}), \mathbf{I}_t) \end{aligned} \quad (4)$$

One advantage of our formulation is that it supports the use of structural losses like SSIM [44] during training. The SSIM loss has been previously proved useful for view synthesis [7], and are not directly applicable to NeRF-like methods [49], as they randomly sample sparse rays from each image during training due to the computational constraints.

4. Experiments

Our model is evaluated on a series of well established ShapeNet [3] [2] benchmarks where it achieves similar or better visual quality compared to the state-of-the-art method pixelNeRF [49] and other recent baselines [7, 35], while rendering objects in real time. We also evaluate the 3D reconstruction capabilities of our model, where it outperforms baseline unsupervised 3D reconstruction models. The following sections detail evaluations on category-specific view synthesis for scenes with single and multiple objects, as well as category-agnostic, multi-category, and unseen-category objects. 3D reconstruction is evaluated in Section 4.2, and the design and effectiveness of different components of our model are discussed in Section 4.3.

[3] licensed for non-commercial research purposes

4.1. Novel View Synthesis

In the novel view synthesis experiments we compare our approach with several state-of-the-art techniques: ENR[7], pixelNeRF [49], DVR [26], and SRN [35]. Our SSIM, PSNR, and LPIPS [50] scores demonstrates that we produce comparable or better rendering quality than pixelNeRF [49] with an explicit volumetric scene representation, while increasing the rendering speed $100\times$ per view, allowing us to render scenes in real-time as show in Table 3.

4.1.1 Category-Specific View Synthesis of Single Objects

We evaluate our model on the ShapeNet chairs and cars categories in single-view and two-view settings, following the same experimental protocol as baseline methods [35, 7, 49]. These category-specific datasets contain $6,591$ different chairs and $3,514$ different cars. Each object has 50 views sampled uniformly on the full sphere, rendering images resolution 128×128 pixels.

Following pixelNeRF, we train a single model for both the single-view and two-view settings. During training, we randomly choose either one or two source views to predict the target view. For evaluation, we use either one or two source views of an unseen object and predict 250 target views. Additionally, we also report the rendering time comparison between pixelNeRF and our method.

Despite its simplicity, our model obtains very competitive results compared to pixelNeRF, as shown in Table 1. In general, we don't observe obvious mistakes made by our model when visually inspecting results. Fig. 3 shows a random subset of source and predicted targets.

In Table 3 we show the average inference and rendering time of both pixelNeRF and our approach. The inference time is defined as the interval of time required to generate scene information (2D feature maps for pixelNeRF and 3D feature maps for our model) from the source views. The rendering time is the time required to render a target view given scene information. We compute per-view rendering time and per-object rendering time, where per-object rendering time is accumulated by rendering a total of 250 views. To conduct a fair comparison, we equate the effective image batch size between pixelNeRF and our model. All the run times are reported on an NVIDIA Tesla V100 GPU. As shown in Table 3, our per-view rendering time is $0.0178s$, 100x faster than pixelNeRF, taking $1.9047s$ to render an image. In other words, our model achieves a rendering speed of 56 FPS, which enables a real-time rendering experience. By amortizing the rendering step across multiple views, our model renders 250 views in $1.022s$ (245 FPS), while pixelNeRF renders the same 250 views in $474.8606s$ (0.5 FPS). This translates to over a 400x speedup. In addition, we test the generalization capabilities of our model on real world data. We use the model trained with ShapeNet cars categories and perform novel view synthesis on real car images from [17]. We found our model can generate plausible novel views with less artifacts and blurry effects compared to pixelNeRF [49]. The complete experiments protocol and qualitative visualizations can be found in the appendix.

Table 1: Results on category-specific novel view synthesis for ShapeNet chairs and cars. Our method achieves competitive results compared to state-of-the-art approaches.

Data	Methods	1-view		2-view	
		PSNR↑	SSIM↑	PSNR↑	SSIM↑
Chairs	ENR	22.83	-	-	-
	SRN	22.89	0.89	24.48	0.92
	pixelNeRF	**23.72**	0.91	**26.20**	**0.94**
	Ours	23.21	**0.92**	25.25	**0.94**
Cars	ENR	22.26	-	-	-
	SRN	22.25	0.89	24.84	0.92
	pixelNeRF	**23.17**	0.90	**25.66**	**0.94**
	Ours	22.83	**0.91**	24.64	0.93

Table 2: Results on category-specific novel view synthesis for multiple chairs. Compared to pixelNeRF, our method predicts much more coherent synthesis results, and it beats pixelNeRF by a significant margin on all three metrics.

Methods	2-view		
	PSNR↑	SSIM↑	LPIPS ↓
SRN	14.67	0.664	0.431
pixelNeRF	23.40	0.832	0.207
Ours	**24.13**	**0.907**	**0.098**

4.1.2 Category-Specific View Synthesis of Multiple Objects

We further extend the category-specific evaluation to the multiple-chair dataset proposed by pixelNeRF. This dataset consists of images rendered with two randomly located and oriented chairs. The dataset is designed so that the model cannot simply rely on certain semantic cues such as the symmetric property of a chair to perform geometry completion. The learned model should be flexible and robust enough to represent scenes instead of a single object. All images are rendered with a resolution of 128×128.

Table 3: Inference and rendering time (in seconds) analysis between pixelNeRF and our method. We show our model can achieve over 100x faster per-frame and over 400x faster per-object rendering speed.

	pixelNeRF		Ours	
	Inference	Rendering	Inference	Rendering
Per-view	0.0053	1.8994	0.0146	0.0032
Per-object	0.0053	474.8553	0.0146	1.0074

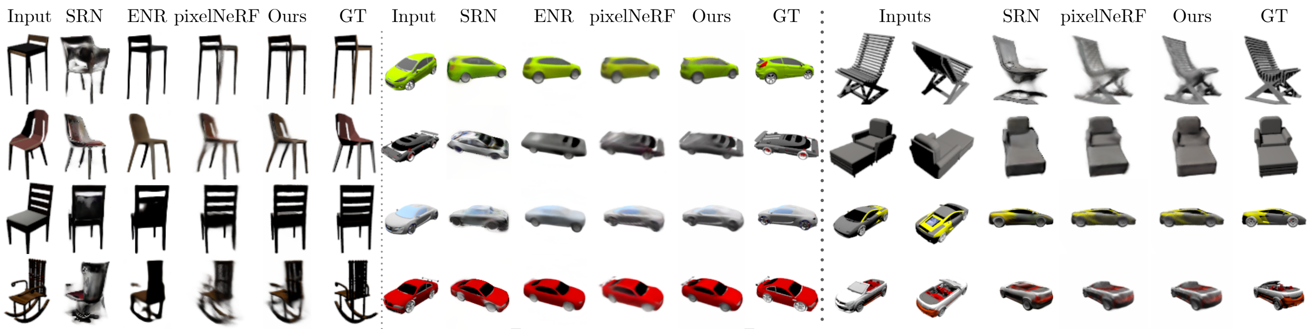
We report reconstruction quality metrics in Table 2. Despite the increased complexity of this setting, our simple model outperforms pixelNeRF across metrics, and exceeds the object-centric method SRN [35] by a large margin. Fig. 4 shows randomly sampled qualitative results. We observe that the views rendered by our model have cleaner geometry than pixelNeRF, which fails to predict a reasonable geometry at certain angles and suffers from ghosting artifacts.

4.1.3 Category-agnostic View Synthesis

The category-agnostic setting is much more challenging than the category-specific one, because the model needs capacity to jointly learn objects across a range of completely different categories. To evaluate our model in the category-agnostic setting, we follow the same training protocol as baseline method [16] and evaluate on 13 different categories. Each object was rendered in 24 different views with a resolution of 64×64. We summarize our results in Table 4. Our model beats all baseline methods in every metric. The qualitative visualization in Fig. 5 indicates our model can generate more clean geometry compared to pixelNeRF, which is corroborates the results obtained by our method in the multi-chair dataset.

4.1.4 Unseen-Category View Synthesis

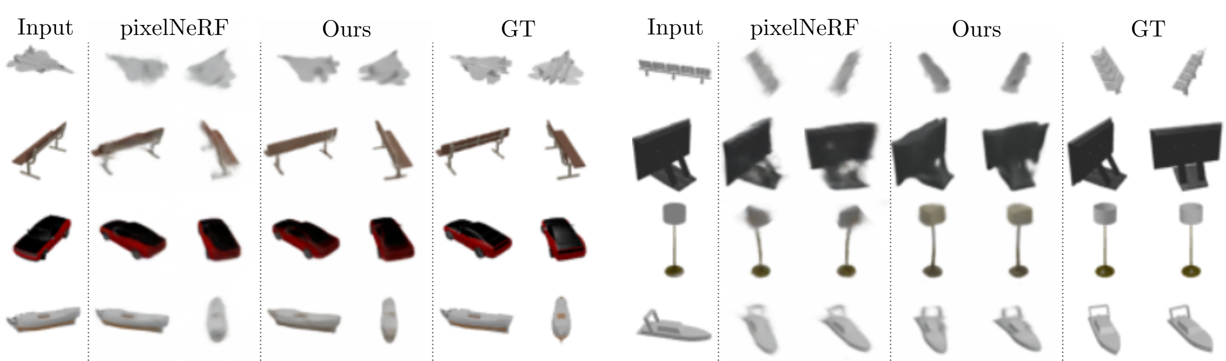
In order to evaluate how our model generalizes to categories not seen during training, we follow the settings in pixelNeRF, and use only three object categories for training, namely airplane, car, and chair. We then evaluate on 10 unseen object categories. Table 5 compares the performance of our method with several baselines. We achieve state-of-the-art performance in SSIM and LPIPS, while performing slightly worse than pixelNeRF [49] on PSNR. Fig. 5 indicates that our method is able to learn a good object



(a) Single-View Novel View Synthesis on Chairs & Cars (b) Two-View Novel View Synthesis on Chairs & Cars

Figure 3: **Qualitative results on category-specific single chair & single car.** The model can either take (a) single view or (b) two views as input to synthesis novel views. We find similar rendering quality comparing to pixelNeRF [49] and better geometry prediction comparing to ENR [7] and SRN [35].

Figure 4: **Qualitative results on category-specific multiple chairs.** The models take two-view images as input. Compared to pixelNeRF, our model renders a cleaner appearance and more complete geometry for chairs with complex shapes.



(a) Novel View Synthesis on Multiple Categories (b) Novel View Synthesis on Unseen Categories

Figure 5: **Qualitative results on (a) category-agnostic and (b) unseen-category datasets.** We test the capacity of our model by training it across different categories in a single-view setting. We evaluate the performance on both seen an unseen categories. We consistently observe cleaner views predicted by our model compared to the baseline.

Table 4: **Quantitative results on category-agnostic view synthesis.** Our model beats all baselines with a noticeable margin in terms of the mean metrics. The LPIPS score for our mode is significant better compared to state-of-the-art methods in all categories.

Metrics	Methods	plane	bench	cbnt.	car	chair	disp.	lamp	spkr.	rifle	sofa	table	phone	boat	mean
PSNR↑	DVR	25.29	22.64	24.47	23.95	19.91	20.86	23.27	20.78	23.44	23.35	21.53	24.18	25.09	22.70
	SRN	26.62	22.20	23.42	24.40	21.85	19.07	22.17	21.04	24.95	23.65	22.45	20.87	25.86	23.28
	pixelNeRF	29.76	26.35	27.72	27.58	23.84	**24.22**	**28.58**	24.44	**30.60**	26.94	25.59	27.13	**29.18**	26.80
	Ours	**30.15**	**27.01**	**28.77**	**27.74**	**24.13**	24.13	28.19	**24.85**	30.23	**27.32**	**26.18**	**27.25**	28.91	**27.08**
SSIM↑	DVR	0.905	0.866	0.877	0.909	0.787	0.814	0.849	0.798	0.916	0.868	0.840	0.892	0.902	0.860
	SRN	0.901	0.837	0.831	0.897	0.814	0.744	0.801	0.779	0.913	0.851	0.828	0.811	0.898	0.849
	pixelNeRF	0.956	0.928	0.924	0.946	0.876	**0.871**	0.914	**0.869**	**0.970**	0.919	0.913	0.925	0.940	0.910
	Ours	**0.957**	**0.930**	**0.925**	**0.948**	**0.877**	**0.871**	**0.916**	**0.869**	**0.970**	**0.920**	**0.914**	**0.926**	**0.941**	**0.920**
LPIPS↓	DVR	0.095	0.129	0.125	0.098	0.173	0.150	0.172	0.170	0.094	0.119	0.139	0.110	0.116	0.130
	SRN	0.111	0.150	0.147	0.115	0.152	0.197	0.210	0.178	0.111	0.129	0.135	0.165	0.134	0.139
	pixelNeRF	0.084	0.116	0.105	0.095	0.146	0.129	0.114	0.141	0.066	0.116	0.098	0.097	0.111	0.108
	Ours	**0.061**	**0.080**	**0.076**	**0.085**	**0.103**	**0.105**	**0.091**	**0.116**	**0.048**	**0.081**	**0.071**	**0.080**	**0.094**	**0.082**

prior, allowing it to generate feasible geometry for unseen categories such as benches and sofas. We also observe that the novel view images predicted by pixelNeRF are consistently more blurry, which explains its better performance on PSNR. On contrary, our model predicts sharp images that are more favorable by human perception, resulting in better metrics like LPIPS.

4.2. 3D Reconstruction

We now turn to the task of evaluating the 3D geometry learned by our approach in a self-supervised manner by minimizing a novel view synthesis objective. In this setting, we evaluate 3D reconstruction by taking the mean intersection-over-union (mIoU) over the predicted α volume (the last channel of $\mathbf{V}_s$) and the corresponding ground truth occupancy volume. We compare our model to several unsupervised 3D reconstruction methods: PrGAN [15], PlatonicGAN/3D [11], Multi.-View [47], and 3DGAN[46]. PlatonicGAN and PrGAN adopt a adversarial approach to learn 3D reconstruction given a single image with a canonical view. For this evaluation, we utilize the model trained with category-agnostic supervision and report results on the *airplane* class as introduced in [11]. The predicted alpha volume is binarized using a threshold $\tau = 0.05$. The ground truth data is obtained from the ShapeNet voxelized volumes [5] and upsampled from 32^3 to 64^3 via nearest-neighbor interpolation. We then calculate the mIoU score and report in Table 6. Results of other models are directly taken from PlatonicGAN [11].

As shown in Table 6, our model predicts accurate 3D reconstruction, outperforming the best baseline by 10% in mIoU. We attribute this boost in performance to the fact that our model can easlily tap large quantities of data in a category-agnostic manner. Whereas in GAN approaches like PlatonicGAN category-agnostic training has traditionally been a very an extremely difficult problem, preventing these approaches to tap large quantities of data for view synthesis. Fig. 6 shows qualitative 3D reconstruction results where we observe that our model produces accurate 3D models of objects. Furthermore, we extend the 3D reconstruction evaluation by including two **supervised** baselines V-LSMs[14] and 3D-R2N2 [5]. Our model obtains a mIoU of 63.25% averaged across categories, while V-LSMs achieves 61.5% and 3D-R2N2 achieves 55.1%. Complete comparison details can be found in the appendix.

4.3. Ablation Studies

To better understand the benefits of each component of our model, we perform ablation studies by excluding one of each of the following components: inverse projection, 2D U-Net, 3D U-Net, or halved 3D voxel resolution. We use a $1/4$ training split of the ShapeNet chairs dataset and evaluate the performance on the full test split. Table 7 summarizes our findings. Starting from the right-most column, we sequentially remove and replace the components with their simplified variants and measure the model performance using the PSNR metric. It turns out that each component contributes *[0.1, 0.5]* metric improvements. The inverse projection component is essential in terms of preserving the implicit geometric and texture information, in comparison to naively reshaping 2D feature volume into 3D [7, 27]. 2D/3D U-Nets are useful to synthesize abstract geometry while preserving texture with skip connections, in comparison to single-path ResNet network structure. The halved 3D resolution is beneficial in reducing the tensor memory footprint and increasing the receptive field.

5. Conclusion

We have presented a simple yet effective approach to perform novel view synthesis of objects without explicit 3D supervision. Contrary to recent developments using radiance

Table 5: **Quantitative results on unseen-category view synthesis.** Our model obtains slightly worse PSNR, similar SSIM and better LPIPS metrics when compared to pixelNeRF.

Metrics	Methods	bench	cbnt.	disp.	lamp	spkr.	rifle	sofa	table	phone	boat	mean
PSNR↑	DVR	18.37	17.19	14.33	18.48	16.09	20.28	18.62	16.20	16.84	22.43	17.72
	SRN	18.71	17.04	15.06	19.26	17.06	23.12	18.76	17.35	15.66	24.97	18.71
	pixelNeRF	**23.79**	**22.85**	**18.09**	**22.76**	**21.22**	**23.68**	**24.62**	**21.65**	**21.05**	**26.55**	**22.71**
	Ours	23.10	22.27	17.01	22.15	20.76	23.22	24.20	20.54	19.59	25.77	21.90
SSIM↑	DVR	0.754	0.686	0.601	0.749	0.657	0.858	0.755	0.644	0.731	0.857	0.716
	SRN	0.702	0.626	0.577	0.685	0.633	0.875	0.702	0.617	0.635	0.875	0.684
	pixelNeRF	0.863	0.814	**0.687**	0.818	0.778	0.899	0.866	**0.798**	**0.801**	0.896	**0.825**
	Ours	**0.865**	**0.819**	0.686	**0.822**	**0.785**	**0.902**	**0.872**	0.792	0.796	**0.898**	**0.825**
LPIPS↓	DVR	0.219	0.257	0.306	0.259	0.266	0.158	0.196	0.280	0.245	0.152	0.240
	SRN	0.282	0.314	0.333	0.321	0.289	0.175	0.248	0.315	0.324	0.163	0.280
	pixelNeRF	0.164	0.186	0.271	0.208	0.203	0.141	0.157	0.188	0.207	0.148	0.182
	Ours	**0.135**	**0.156**	**0.237**	**0.175**	**0.173**	**0.117**	**0.123**	**0.152**	**0.176**	**0.128**	**0.150**

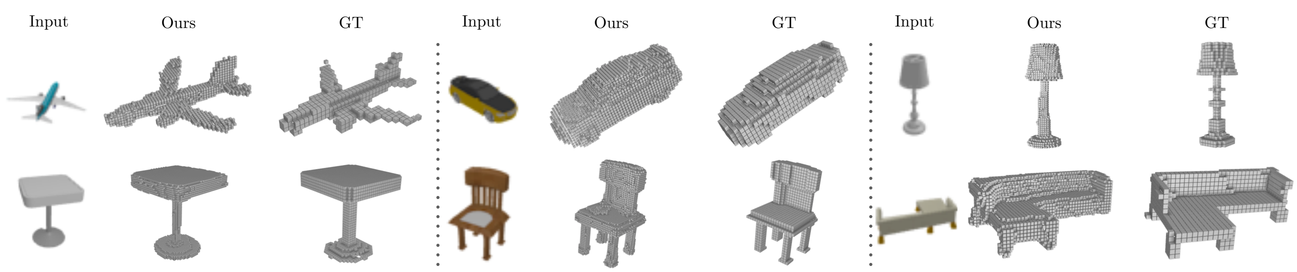


Figure 6: **Qualitative results for 3D geometry reconstruction.** We visualize the predicted α volume with its raw resolution 64^3 and the ground truth volume with its raw resolution 32^3. With a single-forward pass, our model can perform 3D geometry reconstruction given a single view of objects from 13 different categories. Our model trained with only 2D supervision consistently predicts meaningful and full geometry.

Table 6: **Quantitative results for 3D geometry reconstruction on Airplanes class.** Our model outperforms all baseline models in terms of single-view 3D reconstruction.

	PrGAN	PlatonicGAN	Multi.-View	3DGAN	PlatonicGAN 3D	Ours
mIoU↑	0.11	0.20	0.36	0.46	0.44	**0.58**

Table 7: **Ablation studies on different model components.** We show the effectiveness of various model components, trained with a $1/4$ size of the ShapeNet chairs dataset.

	- inv projection	- 2D U-Net	- 3D U-Net	- half 3D resolution	Full
PSNR	20.62	21.10	21.45	21.82	21.94

fields for view synthesis, our approach is neither continuous nor implicit. Despite the simplicity of our approach, we demonstrate that our model obtains comparable or even better performance than recent state-of-the-art approaches for few shot view synthesis using radiance fields [49], while rendering objects at over 400x speed up. In addition, our model learns accurate 3D geometry in a self-supervised manner, relaxing the need of a large amount of 3D geometry data, and surpassing recent baselines for unsupervised learning of 3D geometry.

As a future work (appendix), we plan to investigate the use of explicit sparse space representations such as octrees [38, 42, 48], mixture of volumetric primitives [20], and scene graphs [28, 25] to increase our geometric capacity. Our current model cannot produce view-dependent lighting effects. This limitation can be addressed with a more informative material representation and a shading model that incorporates view direction, lighting, and surface information. We can also utilize techniques such as spherical harmonics [9] or a learned multilayer perceptron (MLP) to synthesize the color with view-dependent specular effects. By doing so during rendering time, we can leverage more advanced rendering techniques such as deferred rendering to better estimate the radiance field that captures both incoming light and material properties.

References

[1] Kara-Ali Aliev, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. *arXiv preprint arXiv:1906.08240*, 2(3):4, 2019.

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[3] Xu Chen, Jie Song, and Otmar Hilliges. Monocular neural image based rendering with continuous view control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4090–4100, 2019.

[4] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7781–7790, 2019.

[5] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[6] Peng Dai, Yinda Zhang, Zhuwen Li, Shuaicheng Liu, and Bing Zeng. Neural point cloud rendering via multi-plane projection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7830–7839, 2020.

[7] Emilien Dupont, Miguel Angel Bautista, Alex Colburn, Aditya Sankar, Carlos Guestrin, Josh Susskind, and Qi Shan. Equivariant neural rendering. *arXiv preprint arXiv:2006.07630*, 2020.

[8] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. *arXiv preprint arXiv:2103.10380*, 2021.

[9] Robin Green. Spherical harmonic lighting: The gritty details. In *Archives of the game developers conference*, volume 56, page 4, 2003.

[10] Adam W Harley, Shrinidhi Kowshika Lakshmikanth, Paul Schydlo, and Katerina Fragkiadaki. Tracking emerges by looking around static scenes, with neural 3d mapping. In *European Conference on Computer Vision*, pages 598–614. Springer, 2020.

[11] Philipp Henzler, Niloy J Mitra, and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9984–9993, 2019.

[12] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *arXiv preprint arXiv:1506.02025*, 2015.

[13] James T Kajiya and Brian P Von Herzen. Ray tracing volume densities. *ACM SIGGRAPH computer graphics*, 18(3):165–174, 1984.

[14] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine, 2017.

[15] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[16] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer, 2017.

[17] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[18] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7708–7717, 2019.

[19] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019.

[20] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *arXiv preprint arXiv:2103.01954*, 2021.

[21] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019.

[22] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *arXiv preprint arXiv:2003.08934*, 2020.

[23] Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-to-end 3d scene reconstruction from posed images. *arXiv preprint arXiv:2003.10432*, 2020.

[24] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7588–7597, 2019.

[25] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. *arXiv preprint arXiv:2011.12100*, 2020.

[26] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3504–3515, 2020.

[27] Kyle Olszewski, Sergey Tulyakov, Oliver Woodford, Hao Li, and Linjie Luo. Transformable bottleneck networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7648–7657, 2019.

[28] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2856–2865, 2021.

[29] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo-Martin Brualla. Deformable neural radiance fields. *arXiv preprint arXiv:2011.12948*, 2020.

[30] Mihir Prabhudesai, Hsiao-Yu Fish Tung, Syed Ashar Javed, Maximilian Sieb, Adam W Harley, and Katerina Fragkiadaki. Embodied language grounding with 3d visual feature representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2220–2229, 2020.

[31] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields. *arXiv preprint arXiv:2011.12490*, 2020.

[32] Konstantinos Rematas, Ricardo Martin-Brualla, and Vittorio Ferrari. Sharf: Shape-conditioned radiance fields from a single view. *arXiv preprint arXiv:2102.08860*, 2021.

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[34] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019.

[35] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, pages 1121–1132, 2019.

[36] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 175–184, 2019.

[37] Shao-Hua Sun, Minyoung Huh, Yuan-Hong Liao, Ning Zhang, and Joseph J Lim. Multi-view to novel view: Synthesizing novel views with self-learned confidence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 155–171, 2018.

[38] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2088–2096, 2017.

[39] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d scene representation and rendering. *arXiv preprint arXiv:2010.04595*, 2020.

[40] Hsiao-Yu Fish Tung, Ricson Cheng, and Katerina Fragkiadaki. Learning spatial common sense with geometry-aware recurrent networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2595–2603, 2019.

[41] Hsiao-Yu Fish Tung, Ricson Cheng, and Katerina Fragkiadaki. Learning spatial common sense with geometry-aware recurrent networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2595–2603, 2019.

[42] Peng-Shuai Wang, Yang Liu, and Xin Tong. Deep octree-based cnns with output-guided skip connections for 3d shape and scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 266–267, 2020.

[43] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. *arXiv preprint arXiv:2102.13090*, 2021.

[44] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[45] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. *arXiv preprint arXiv:2103.05606*, 2021.

[46] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *arXiv preprint arXiv:1610.07584*, 2016.

[47] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. *arXiv preprint arXiv:1612.00814*, 2016.

[48] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. *arXiv preprint arXiv:2103.14024*, 2021.

[49] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. *arXiv preprint arXiv:2012.02190*, 2020.

[50] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[51] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018.

[52] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *European conference on computer vision*, pages 286–301. Springer, 2016.

[53] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Joshua B Tenenbaum, and William T Freeman. Visual object networks: Image generation with disentangled 3d representation. *arXiv preprint arXiv:1812.02725*, 2018.