Generalizable Patch-Based Neural Rendering

Mohammed Suhail[1], Carlos Esteves[4], Leonid Sigal[1,2,3], and Ameesh Makadia[4]

[1] University of British Columbia {suhail33,lsigal}@cs.ubc.ca
[2] Vector Institute for AI
[3] Canada CIFAR AI Chair
[4] Google {machc,makadia}@google.com

Abstract. Neural rendering has received tremendous attention since the advent of Neural Radiance Fields (NeRF), and has pushed the state-of-the-art on novel-view synthesis considerably. The recent focus has been on models that overfit to a single scene, and the few attempts to learn models that can synthesize novel views of unseen scenes mostly consist of combining deep convolutional features with a NeRF-like model. We propose a different paradigm, where no deep visual features and no NeRF-like volume rendering are needed. Our method is capable of predicting the color of a target ray on a novel scene directly, just from a collection of patches sampled from the scene. We first leverage epipolar geometry to extract patches along the epipolar lines of each reference view. Each patch is linearly projected into a 1D feature vector and a sequence of transformers process the collection. For positional encoding, we parameterize rays as in a light field representation, with the crucial difference that the coordinates are canonicalized with respect to the target ray, which makes our method independent of the reference frame and improves generalization. We show that our approach outperforms the state-of-the-art on novel view synthesis of unseen scenes even when being trained with considerably less data than prior work.

1 Introduction

Synthesizing novel views of a scene from a set of images obtained from different viewpoints is a long-standing problem in computer graphics and computer vision. Recent advances in this problem [64] employ neural networks to learn scene representations (neural scene representations), combined with classical volume rendering to produce a novel view from any desired viewpoint, an idea spearheaded by NeRF [37]. Most of these methods are trained by overfitting to a single scene in order to produce arbitrary novel views of that same scene. While capable of producing high-quality photorealistic images, the need for retraining on each new scene limits their practical application.

In this paper, we consider the more difficult task of training a single model that is capable of generating novel views of unseen scenes. There are a few notable efforts in this direction [8,69,78]. One key idea of these methods is to augment

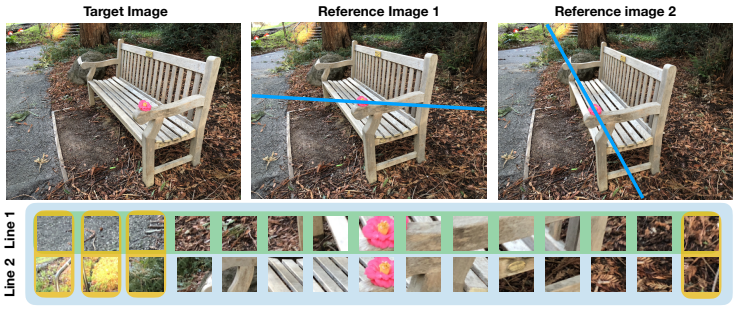

Fig. 1. Sample scene. Our goal is to predict the color of a target ray, given only the reference images and camera poses. Consider the patches along each epipolar line, which correspond to samples of increasing depth along the target ray. If there are many matching patches at some depth, there is a high chance that the patch around the target ray also matches. In this example, the matching patches contain the flower, which is where the target ray hits. This motivates our three-stage architecture, that first exchanges information along views at each depth (yellow), then aggregates information along depths for each view (green), and finally aggregates information among reference views to predict the ray color (blue). The figure shows only 2 reference views with 15 sampled patches each, but in practice we use a larger number of views and samples.

NeRF inputs with deep convolutional features, which include both local and global context. However, these methods still rely on scene-specific inputs such as 3D positions and directions, which are not reliable on unseen scenes. We also hypothesize that using feature extractors that have large receptive fields such as UNet [50] or Feature Pyramid Networks [28] is harmful when generalizing to scenes visually far from the training distribution.

We propose a different approach that takes only local linear patch embeddings as input, eschewing deep convolutional networks. Moreover, our method does not require the ubiquitous volume rendering from NeRF; it produces the color of a target pixel directly from a set of reference view patches.

We are inspired by both classical and recent works. Classical computer vision tasks such as optical flow and image feature matching for 3D reconstruction were historically dominated by techniques operating on local patches [19,32,34,57]. In fact, for some tasks the classical methods still outperform modern deep learning ones [53]. Another example is COLMAP [54,55] which is a widely popular method for 3D reconstruction and typically used to generate camera poses (and sometimes depth maps) that are inputs to modern neural rendering.

Our decision to focus on local patches and to avoid convolutional features is supported by the recent success of the Vision Transformers (ViT) [14], which we employ. A second reason to use transformers [67] is that our input is effectively a set of patches, and self-attention is a powerful mechanism to learn from sets without making any assumption about the order of the elements. We show that transformers can effectively replace both the convolutional features and the volume rendering typically employed in the tasks we consider.

Our key contribution is to leverage the structure of the patch collection to build a multiview representation that is further refined along epipolar lines and reference views to predict the final color. Figure 1 explains the idea. Another unique aspect of our method is the canonicalized positional encoding of rays, depths, and camera poses, which is independent of the frame of reference, enabling superior generalization performance.

Contributions: Our contributions can be summarized as follows,

- We introduce a model that renders target rays in unseen scenes directly from a collection of patches sampled along epipolar lines of reference views.
- To exploit the structure of the patch collection, we design an architecture with stacked transformers operating over different subsets of the collection such that features are learned, combined and aggregated in principled ways.
- To improve generalization to unseen scenes, we introduce canonicalized positional encodings of rays, depths, and camera poses such that all inputs to the model are independent of the scene's frame of reference.
- Our model outperforms the baselines in multiple train and evaluation datasets, while using as little as 11% of the amount of training data in certain cases.

2 Related Work

2.1 Neural scene representations

Our method is in the broad category known as neural rendering, where neural networks are used to represent a scene and/or directly render views [64]. Neural fields [72] are also closely related. The majority of recent methods employ neural scene representations coupled with classical rendering methods, as popularized by NeRF [37]. These works can be broadly classified into models that represent the scene using a *surface* or *volumetric* representation [64]. Surface representation methods either explicitly represent the scene as point clouds [1,26,44,51,70,75], meshes [4,22,65], or implicitly using signed distance function [11,25,43,63,74]. Volumetric representations on the other hand typically use voxel grids [40,61], octrees [29,77], multi-plane [71,79], implicitly using a neural network [17,30,41] or a coordinate-based network as in NeRF [37] and its variants [3,35,38]. Recently, works such as Vol-SDF [73], NeuS [68] and UNISURF [42] propose to use volumetric rendering methods to extract a surface representation.

Our method differs from these because there is no structured neural scene representation and no volume rendering – the target pixel color is obtained directly by learning weights to blend reference pixels, taking only a set of patches around the reference pixels as input. Thus, our approach fits into the category of image-based rendering.

Moreover, our model can be trained once on a set of scenes and applied to novel scenes, which can be more efficient than re-training scene-specific models for every new scene as is common. Concurrent work has achieved impressive results on accelerating NeRFs [76,39], providing a reasonable alternative when efficiency is important and re-training for every scene is not a hindrance.

2.2 Image-based rendering

Image-based rendering methods [58,59] typically construct novel views of a scene by warping and compositing a set of reference images. Shum and Kang [58] classified most of these works into categories that use *no* geometry, *explicit* geometry or *implicit* geometry. Methods that do not model the geometry rely on the characterization of the plenoptic function. Light field rendering [27] is one such method that used 4D light field plenoptic function to render novel views by interpolating a set of input samples. Light field rendering, however, requires a dense sampling of input views to be accurate. Follow-up works such as Lumigraph [18] incorporate approximate geometry to overcome the dense sampling requirement. Explicit geometry based methods [20,21,47,48] generate a geometric reconstruction of the scene in the form of a 3D mesh. However, explicit 3D reconstruction without 3D supervision is a hard learning problem, and undesirable artifacts in the reconstructed geometry impact rendering quality. Implicit geometry methods [10,56] rely on aggregating multiple input views to synthesize a novel view. Recently, LFNR [62] proposed to use epipolar geometry in conjunction with light field ray representations to model view-dependent effects. Other works [2,15,60] similarly have explored neural representations for light field rendering. Part of our architecture is similar to LFNR; however, our method is aimed at generalizing to unseen scenes as opposed to overfitting on a single scene, which avoids expensive retraining for each new scene.

Stereo Radiance Fields [13] has a focus on efficiency and was one of the first methods tackling generalization to novel scenes. PixelNeRF [78] conditions a NeRF [37] on deep convolutional visual features of the reference views, enabling generalization to new scenes; however, it uses absolute positions and directions as inputs to the NeRF, which generalize poorly across scenes. Similarly, IBRNet [69] also uses deep features and NeRF-like volume rendering, but it learns to blend colors from neighboring views for each point along a ray. IBRNet uses the difference between view directions as MLP inputs; while this is superior to absolute coordinates, the relative view directions still depend on a global reference frame which is scene-specific. MVSNeRF [8] constructs a cost volume from deep visual features. The voxel features are then concatenated to the usual NeRF inputs including absolute positions and directions for rendering novel views.

In contrast with these works, our method 1) does not require deep convolutional features, operating directly on linear projections of local patches, similarly to ViT [14]; 2) does not require volume rendering, producing the final colors directly from a reference set of patches; and 3) is independent of the input frame of reference, leveraging canonicalized ray, point and camera representations, which improves its generalization ability. Concurrent work on neural rendering generalizable to unseen scenes include GeoNeRF [24] and NeuRay [31], but both require at least partial depth maps during training.

2.3 Transformers in vision

Popularized by Vaswani *et al.* [67], transformers are sequence-to-sequence models that use an attention mechanism to incorporate contextual information from

relevant parts of the input. Initially developed for NLP tasks [12], transformer-based models have also achieved state-of-the-art on a variety of vision problems [14,6,33,7].

Recently, Robin *et al.* [49] proposed the use of transformers to generate novel views from a single image without explicit geometric modeling. Scene representation transformers [52] similarly presented a model for novel view synthesis using self-supervision from images. Their experiments, however, are limited to low resolution images (maximum size of 178×128 pixels). Slightly more related to our approach are IBRNet [69], which employs a ray-transformer module to estimate densities via self-attention over samples along the ray, and NerFormer [45], which alternates self-attention over views and rays, but is object-based and aims to generalize only to new instances of the same object category.

Our use of transformers differs greatly from such works because (i) we use transformers in all stages, from the patch embedding to final target ray color prediction, not requiring deep convolutional features nor volume rendering, and (ii) we design a unique architecture with three different transformers operating along and collapsing different dimensions.

3 Approach

Given a set of scenes with a collection of images and their corresponding camera poses, we aim to learn a generic rendering model that is capable of rendering novel views of a scene without training on it. At the core of our model is a reference-frame-agnostic rendering network that relies only on local patches observed from nearby reference cameras. Figure 2 provides a visual overview. We present our approach in the following order: first we introduce light field representations; then we discuss the construction and embedding of reference patches; and finally we detail our transformer-based rendering network that maps a target light field and reference patches to radiance.

3.1 Light field representation

The light field characterizes the radiance through points in space. It can be described by a five-dimensional function on $\mathbb{R}^3 \times S^2$, mapping each direction through each point to its radiance. In free space, the radiance along a ray remains constant, thus allowing to parametrize the light field as a $4D$ function [27].

Depending on the camera configuration, different light field representations can be used. For example, for a scene with forward-facing camera configuration, the rays can be parametrized by their intersections with two planes perpendicular to the forward direction, a representation known as the light slab [27]. The entries of the $4D$ vector are the coordinates of the intersections on each plane's $2D$ coordinate system. An alternative representation suitable for bounded scenes observed from all directions is known as the two-sphere [5], and represents rays by their two intersections with a sphere bounding the scene. Prior works such as LFNR [62] exploit the camera configuration information of the scene to decide

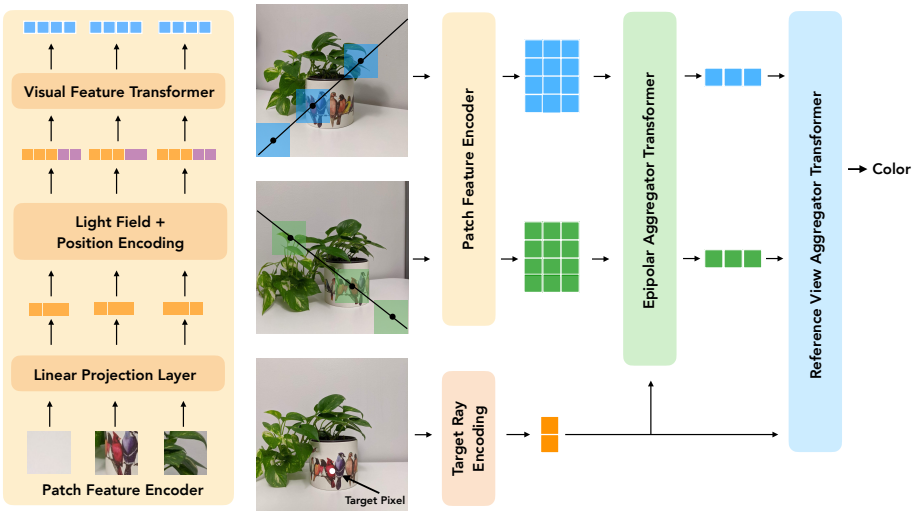

Fig. 2. Model Overview. Our model consists of three stages, with a different transformer per stage. First, patches along epipolar lines are extracted, linearly projected, and arranged in a grid of K reference views by M sampled depths. The first transformer takes a sequence of views and is repeated for each depth, returning another $K \times M$ grid. The second transformer takes a sequence of depths and is repeated for each view; it collapses features along the depth dimension, returning K view features. The third transformer aggregates the K view features. Attention weights extracted from the second and third transformers are used to blend colors over views and epipolar lines and make the final prediction. A canonicalized positional encoding of rays, depths and cameras is appended to the transformer inputs.

the underlying parametrization. They use light slab parametrization for forward-facing-scenes and two-sphere parametrization for 360° scenes.

In this work, the ray representations are used as positional encoding in the transformers. Since we wish to generalize to new scenes and therefore cannot make assumptions about the camera configurations, we use Plücker coordinates as the choice of parametrization. Given a ray through a point o (the ray origin) with direction v, the Plücker coordinates can be obtained as $r = (v, o \times v)$. The representation is six-dimensional, however it has only four degrees of freedom since it is defined up to a scale factor and the two vectors that compose it must be orthogonal. The Light Field Networks [60] use the same parametrization but in a different context.

3.2 Patch extraction

Given a target viewpoint, our method relies on eliciting "local" light field patches to produce the ouput images. To extract such patches, we first identify a set of reference images that serve as 2D slices of the plenoptic function observed from

neighboring viewpoints. While our model is agnostic to the number of reference images, we use a subset of the available input images for patch extraction. Specifically, for a target camera we take a subset of the N closest views. We randomly sub-sample K views from this subset during training, and use the closest K views for inference.

Given the set of reference images $\mathcal{I} = \{I_1, I_2, ..., I_K\}$, the next step is to fragment them into patches. Dosovitskiy *et al.* [14] split the entire image into fixed-size non-overlapping patches. While this partition is useful for global reasoning (e.g. image classification), for view synthesis the relevant regions in the image can isolated by exploiting the epipolar geometry between views. For a given image in the reference set $\mathcal{I}$, we compute the epipolar line corresponding to the target pixel. We sample M points along this epipolar line such that their $3D$ re-projections on the target ray are spaced linearly in depth. We then extract square patches around each of the M points, and this process is repeated for all reference images. The resulting reference patch set is indexed by view and depth: $\mathcal{P} = \{P_k^m \mid 1 \leq k \leq K, 1 \leq m \leq M\}$.

3.3 Patch embedding and positional encoding

We use a linear projection layer to flatten the patches into $1D$ feature vectors, which are input to transformers. The patch features for the m-th sample along the epipolar line on view k is denoted p_k^m.

Since transformers are agnostic to the position of each element in the input sequence, typically a positional encoding is added to the features to represent the spatial relationship between elements. Unlike prior works [14], since the location and source of patches do not remain the same across batches, we cannot include a learnable embedding into the sequence. Instead, we extract the geometric information associated with each patch and append them to the flattened patch feature vectors.

We use three forms of positional encoding:

1. To retain the reference patch position in space, we use the light field encoding of the rays emanating from the reference camera as described in Section 3.1. We represent the m-th ray along the epipolar line of view k by r_k^m.
2. To retain the position of the patch in the sequence of patches along the epipolar line, we encode the distance along the target ray corresponding to the patch center using a sinusoidal positional encoding that follows NeRF [37]. The encoded distance for the m-th sample is represented by d^m.
3. To retain geometry between target and reference cameras, we also append the relative camera pose as a flattened rotation matrix and a 3D translation, which is shared among all patches associated to the same camera and denoted by c_k for camera k.

3.4 Canonicalized ray representation

Structure-from-Motion (SfM) methods that are used to estimate camera extrinsics can only reconstruct scenes up to an arbitrary similarity transformation –

rotation, translation and scaling. Prior works [8,69,78] use such estimations to compute scene specific coordinates such as view directions and $3D$ coordinates of points.

We hypothesize that for best generalization to unseen scenes, the inputs to the model should be invariant to similarity transformations. This means that model should produce the same result upon a change of reference frame or rescaling of the input camera poses. IBRNet [69] takes a step towards this idea by using the difference between reference and target direction vectors instead of absolute directions, but the difference is still a 3D vector that is not independent of the frame of reference, and so are the 3D positions of points along the target ray.

The positional encoding of relative camera poses and distance values, as described in Section 3.3, are made invariant to similarities by simply scaling the camera positions by the maximum depth of the scene output by SfM.

The encoding of rays in the light field, however, need to be canonicalized. Our key idea is to define a local frame centered on each ray (not camera). For a target pixel $x \in \mathbf{RP}^2$ in the target camera with extrinsics $[R \mid t]$ and intrinsics C, we first obtain the corresponding ray direction $v = R^\top C^{-1} x$. We use v and the camera y axis to determine the local frame. Specifically, we use the Gram-Schmidt orthonormalization process. Let $v' = v / \|v\|$ and $y' = y - (y \cdot v')v'$. The canonicalizing transformation is then

$$R_c = \begin{bmatrix} \frac{y'}{\|y'\|} \times v' & \frac{y'}{\|y'\|} & v' \end{bmatrix} \tag{1}$$

$$T = \begin{bmatrix} R_c^\top \mid -R_c^\top t \end{bmatrix} \tag{2}$$

where $T \in \mathbf{SE}(3)$. We apply T to every camera pose, which results in the target ray having origin $(0, 0, 0)$ and direction $(0, 0, 1)$, and all other ray representations computed from the canonicalized camera poses will be invariant to similarities. We show the benefit of such canonicalization in Section 4.2.

3.5 Rendering network

Given the patch embeddings and positional encodings of a target ray, as described in Sections 3.3 and 3.4, our rendering network predicts the ray color.

We argue that predicting the target ray color is deeply related to finding correspondences to the target ray in the reference images. Take, for example, LFNR [62]. Its first stage aggregates features along each epipolar line, which is essentially finding correspondences to the target ray. Since LFNR overfits to a single scene, the model can learn the structure of the scene and use it to estimate correspondences based only on ray coordinates.

However, the LFNR [62] approach cannot generalize to novel scenes, since, given just an epipolar line, it is impossible to know which point corresponds to a target ray without knowing the structure of the scene.

Our main contribution is to provide visual features for a similar epipolar transformer, such that the correspondence is solved visually (see Fig. 1 for illustration), which is advantageous because the visual features can be extracted

from novel scenes in a single forward step starting from small local patches. Crucially, such features cannot come from a single epipolar line. It is the combination of visual features from different epipolar lines cast by the same target ray that allows correspondences to be established. To learn this combination, we propose to use a transformer.

Thus, our model consists of three transformers. The first, which we call "Visual Feature Transformer", learns visual features by combining information from patches along different reference views. The second and third are similar to the ones in LFNR [62], with the major differences that the positional encodings of rays, depths and cameras are canonicalized as described in Section 3.4 and that the final color is predicted by directly blending pixel colors from reference views, instead of using learned features; both changes greatly improve the generalization performance.

Each transformer follows the ViT [14] architecture, which uses residual connections to interleave layer normalization (LN), self-attention (SA), and multi-layer perceptron (MLP). Each layer consists of LN $\rightarrow$ SA $\rightarrow$ LN $\rightarrow$ MLP.

Visual Feature Transformer. This stage exchanges visual information between potentially corresponding patches on different reference images, leading to visual features with multi-view awareness. The input to this stage is the set of patch linear embeddings and positional encoding vectors p_k^m, r_k^m, d^m, c_k, indexed by the view k and the m-th sampled depth, as described in Section 3.3. We first define the feature concatenation at layer zero (the input) as

$$f_0^{k,m} = [p_k^m \parallel r_k^m \parallel d^m \parallel c_k]. \tag{3}$$

This stage is repeated for each depth sample, therefore it operates on sequences of K views. Formally, it repeats

$$f_1^m = T_1\left(\left\{f_0^{k,m} \mid 1 \leq k \leq K\right\}\right) \tag{4}$$

for $1 \leq m \leq M$, where T_1 is a transformer written as a set to set map. This stage takes a (K, M, C_0) tensor of C_0-dimensional features of K views sampled at M depths, and returns a (K, M, C_1) tensor of C_1-dimensional features.

Epipolar Aggregator Transformer. This stage aggregates information along each epipolar line, resulting in per reference view features. The input to this stage is the set $f_1 = \{f_1^m \mid 1 \leq m \leq M\}$, concatenated with positional encodings. We refer to the features corresponding to view k in the set f_1^m as $f_1^{k,m}$. The transformer is repeated for each view, therefore operating along the sequence of M epipolar line samples. Formally, we first compute

$$f_2^k = T_2\left(\left\{r^0\right\} \bigcup \left\{\left[f_1^{k,m} \,\middle\|\, r_k^m \,\middle\|\, d^m \,\middle\|\, c_k\right] \,\middle|\, 1 \leq m \leq M\right\}\right), \tag{5}$$

for $1 \leq k \leq K$, where r^0 is a special token to represent the target ray. We then apply a learned weighted sum along the M epipolar line samples as follows,

$$\alpha_k^m = \frac{\exp\left(W_1 \left[f_2^{k,0} \,\middle\|\, f_2^{k,m}\right]\right)}{\sum_{m'=1}^{M} \exp\left(W_1 \left[f_2^{k,0} \,\middle\|\, f_2^{k,m'}\right]\right)}, \tag{6}$$

$$f_{2'}^k = \sum_{m=1}^{M} \alpha_k^m f_2^{k,m}, \tag{7}$$

for $1 \leq k \leq K$, resulting in a feature vector per view k, where W_1 are learnable weights and $f_2^{k,0}$ is the output corresponding to the target ray token.

Dimension-wise, this stage takes a (K, M, C_1) tensor and returns a (K, C_2) tensor of C_2-dimensional features per reference view.

Reference View Aggregator Transformer. This final transformer aggregates the features over reference views and predicts the color of the target ray. Its input is the set of per reference view features $f_{2'} = \{f_{2'}^k \mid 1 \leq k \leq K\}$, concatenated with the camera relative positional encoding. Formally, we compute

$$f_3 = T_3\left(\left\{r^0\right\} \bigcup \left\{\left[f_{2'}^k \,\middle\|\, c_k\right] \;\middle|\; 1 \leq k \leq K\right\}\right). \tag{8}$$

Similarly to the previous stage, we compute the blending weights

$$\beta_k = \frac{\exp\left(W_2 \left[f_3^0 \,\middle\|\, f_3^k\right]\right)}{\sum_{k'=1}^{K} \exp\left(W_2 \left[f_3^0 \,\middle\|\, f_3^k\right]\right)}, \tag{9}$$

which are used in conjunction with the weights from the previous stage to estimate the color of the target ray by blending colors along each epipolar line sample at each reference view,

$$\mathfrak{c} = \sum_{k=1}^{K} \beta_k \left(\sum_{m=1}^{M} \alpha_k^m \mathfrak{c}_k^m\right), \tag{10}$$

where $\mathfrak{c}_k^m$ is the pixel color at the m-th sample along the epipolar line of view k. Our approach here differs from the last stage of LFNR, which does the aggregation on feature space using only the weights β_k, and linearly projects the resulting feature to predict the color. We argue that using the input pixel values from reference views instead helps generalization, which we confirm experimentally (see supplementary material). This is possible by using the two sets of attention weights α_k^m (Eq. (6)) and β_k (Eq. (9)), which allow blending colors from all epipolar line samples and all reference views.

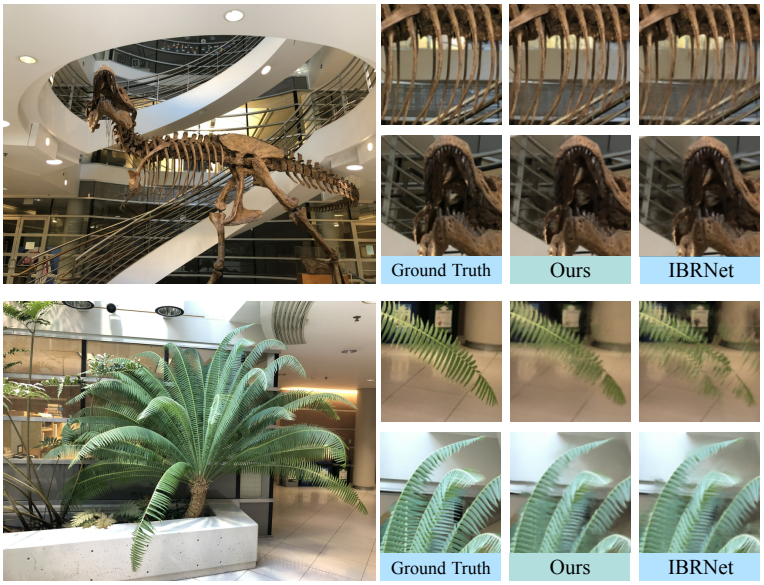

Fig. 3. Qualitative results on RFF (setting 1). We show our method and the baseline on the *Trex* and *Fern* scenes from the real forward-facing dataset. Compared with IBRNet [69], our method produce sharper details and less blurring at boundaries. For example, the top row in the *Fern* scene shows that the baseline methods either fail to reconstruct the leaves or produce inconsistent shapes. Our method is able to retain the shape boundaries accurately along with majority of the texture details.

4 Experiments

4.1 Implementation Details

Each of the three transformers in our model consist of 8 blocks each with a feature dimension of 256. We select reference views using $K = 10$ and $N = 20$ (see Section 3.2). We use a batch size of 4096 rays and train for $250k$ iterations with a Adam optimizer and initial learning rate of $3 \cdot 10^{-4}$. We use a linear learning rate warm-up for $5k$ iterations and cosine decay afterwards. Training our model takes $\sim$24 hours on 32 TPUs. We report the average PSNR (peak signal-to-noise ratio), SSIM (structural similarity index measure) and LPIPS (learned perceptual image patch similarity) for all our experiments.

4.2 Results

There is no standard training and evaluation procedure for generalizable neural rendering. IBRNet [69] trains on the LLFF dataset [36], renderings of Google scanned objects [46], Spaces dataset [16], RealEstate10K dataset [79] and on their own scenes. They evaluate on the real forward-facing (RFF) dataset [37], which comprises held-out LLFF scenes, Blender (consisting of 360° scenes) [37],

Method	Real Forward-Facing			Shiny-6			Blender		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
LLFF [36]	24.13	0.798	0.212	-	-	-	24.88	0.911	0.114
IBRNet [69]	25.13	0.817	0.205	23.60	0.785	0.180	25.49	0.916	0.100
GeoNeRF [24]	25.44	0.839	0.180	-	-	-	28.33	0.938	0.087
IBRNet*	24.33	0.801	0.213	23.37	0.784	0.181	21.32	0.888	0.131
Ours	25.72	0.880	0.175	24.12	0.860	0.170	26.48	0.944	0.091

Table 1. Results for setting 1. Our model outperforms the baselines even when training with strictly less data. IBRNet uses three datasets that are not part of our training set, while GeoNeRF uses one extra dataset and also leverages input depth maps during training. IBRNet* was trained using the same training set as our method; in this fair comparison, our advantage in accuracy widens.

and Diffuse Synthetic 360° [61]. Contrastingly, MVSNeRF [9] trains on DTU [23] and tests on held out DTU scenes, real forward-facing dataset (RFF) [37], and Blender [37]. Various other works [24,31,66] have explored different experimental setups. In this work, in an attempt to fairly evaluate against prior works, we use two experimental settings.

Setting 1. In the first setting, we train on a strict subset of the IBRNet training set, comprised of 37 LLFF scenes and 131 IBRNet collected scenes (amounting to 11% of the training set used by IBRNet). We then evaluate on the real forward-facing, Shiny [71] and Blender datasets. On Shiny, we compute the results for IBRNet using their publicly available pretrained weights. Table 1 reports quantitative while Figs. 3 and 4 show qualitative results. IBRNet and GeoNeRF (a concurrent work) use a larger training set than ours, and GeoNeRF uses depth maps during training, but our method shows the best performance in most metrics regardless. Additionally, IBRNet is trained on 360° scenes whereas our method is trained only on forward-facing scenes. Nonetheless, our model achieves superior performance on Blender as compared to IBRNet.

Setting 2. Here, we train our model on DTU, following the MVSNeRF [8] procedure, and evaluate on the held-out DTU scenes and the Blender dataset. For training on DTU, we follow the same split as PixelNeRF [78] and MVSNeRF. We partition the dataset into 88 scenes for training and 16 scenes for testing, each containing images of resolution 512×640. Table 2 shows quantitative results. MVSNeRF is trained with 3 reference views while our method performs best with 10. We evaluated MVSNeRF with 10 views, which did not improve their performance; Table 2, thus, compares the best number of views for each model. Our model consistently outperforms across all three metrics.

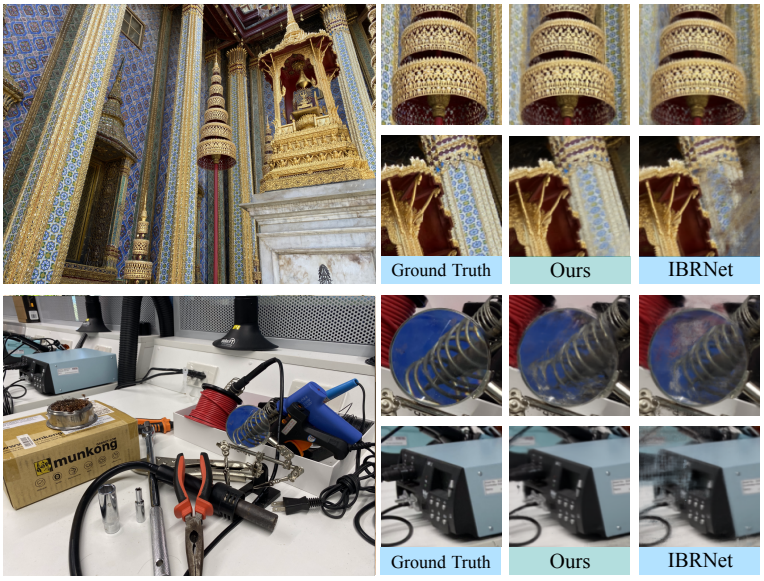

Fig. 4. Qualitative results on Shiny [71] (setting 1). While still consisting of forward facing scenes, Shiny scenes have scale and view density that differ from the usual in setting 1, which makes it more challenging than LLFF. IBRNet [71] produces noticeable artifacts that are not present in our method's renderings.

Ablation. To investigate the effectiveness of our contributions, we perform various ablations experiments. We train the model on 504×378 resolution images of LLFF and IBRNet scenes and test on the real forward-facing dataset at the same resolution. We start with a "base model" that does not use the visual feature transformer or the coordinate canonicalization. We then incrementally add components of our proposed approach. Table 3 reports the ablation results. We observe that the "base" model generalized poorly to unseen scenes. Incorporating the visual feature transformer improves the performance significantly. For the canonicalization ablation, we split the component into two, (i) ray canonicalization, where the light field ray representation is computed independent of the frame of reference, and (ii) coordinate canonicalization where the $3D$ samples along the target ray are canonicalized. We observe that both forms of canonicalization help improving the accuracy.

5 Limitations

One limitation of our model is that since it operates on small local patches to aid generalization, it relies on a large number of views to produce meaningful features. In the comparison against MVSNeRF [8] in Table 2, while our method is more accurate by significant margins, it also requires 10 reference views while MVSNeRF only uses 3. Rendering novel scenes with our approach is fast since it

Method	DTU			Blender		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
PixelNeRF [78]	19.31	0.789	0.671	7.39	0.658	0.411
IBRNet [69]	26.04	0.917	0.190	22.44	0.874	0.195
MVSNeRF [9]	26.63	0.931	0.168	23.62	0.897	0.176
Ours	28.50	0.932	0.167	24.10	0.933	0.097

Table 2. Results for setting 2. All models are trained on DTU and evaluated on either the DTU held-out set or Blender. Our approach outperforms the baselines.

Visual Transformer	Ray Canonicalization	Coordinate Canonicalization	PSNR	SSIM	LPIPS
✗	✗	✗	22.62	0.763	0.313
✓	✗	✗	25.42	0.879	0.154
✓	✓	✗	25.86	0.885	0.142
✓	✓	✓	26.42	0.896	0.129

Table 3. Ablations. Ablation study for model trained on LLFF and IBRNet scenes and tested on RFF with a resolution of 504×378. Results show that our main contributions – the visual feature transformer and the canonicalized positional encoding – lead to superior generalization performance.

consists only of forward steps, but training is slow, comparable with LFNR [62]. The appendix shows a quantitative timing evaluation.

6 Conclusion

This paper introduced a method to generate novel views from unseen scenes that predicts the color of an arbitrary ray directly from a collection of small local patches sampled from reference views according to epipolar constraints. Our model departs from the common combination of using deep visual features and NeRF-like volume rendering for this task. We introduced a three-stage transformer architecture, coupled with canonicalized positional encodings, which operates on local patches – all these properties aid on generalizing to unseen scenes. This is demonstrated by our outperforming of the current state-of-the-art while using only 11% of the amount of training data.

We include more details and results in the supplementary material, including more ablation experiments, timing evaluation, other combinations of train and evaluation sets, and more qualitative results.

References

1. Aliev, K.A., Sevastopolsky, A., Kolos, M., Ulyanov, D., Lempitsky, V.: Neural point-based graphics. In: European Conference on Computer Vision (ECCV). pp. 696–712 (2020) 3
2. Attal, B., Huang, J.B., Zollhöfer, M., Kopf, J., Kim, C.: Learning neural light fields with ray-space embedding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19819–19829 (2022) 4
3. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5855–5864 (2021) 3
4. Burov, A., Nießner, M., Thies, J.: Dynamic surface function networks for clothed human bodies. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 10754–10764 (2021) 3
5. Camahort, E., Lerios, A., Fussell, D.: Uniformly sampled light fields. In: Eurographics Workshop on Rendering Techniques. pp. 117–130 (1998) 5
6. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European Conference on Computer Vision (ECCV). pp. 213–229 (2020) 5
7. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: Masked generative image transformer. arXiv preprint arXiv:2202.04200 (2022) 5
8. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 14124–14133 (2021) 1, 4, 8, 12, 13
9. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In: IEEE/CVF International Conference on Computer Vision (CVPR). pp. 14124–14133 (2021) 12, 14
10. Chen, S.E., Williams, L.: View interpolation for image synthesis. In: Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques. p. 279–288. SIGGRAPH '93, Association for Computing Machinery (1993) 4
11. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5939–5948 (2019) 3
12. Chernyavskiy, A., Ilvovsky, D., Nakov, P.: Transformers:" the end of history" for nlp? arXiv preprint arXiv:2105.00813 (2021) 5
13. Chibane, J., Bansal, A., Lazova, V., Pons-Moll, G.: Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7911–7920 (2021) 4
14. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020) 2, 4, 5, 7, 9
15. Feng, B.Y., Varshney, A.: Signet: Efficient neural representation for light fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14224–14233 (2021) 4
16. Flynn, J., Broxton, M., Debevec, P., DuVall, M., Fyffe, G., Overbeck, R., Snavely, N., Tucker, R.: Deepview: View synthesis with learned gradient descent. In:

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2367–2376 (2019) 11
17. Genova, K., Cole, F., Vlasic, D., Sarna, A., Freeman, W.T., Funkhouser, T.: Learning shape templates with structured implicit functions. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 7154–7164 (2019) 3
18. Gortler, S.J., Grzeszczuk, R., Szeliski, R., Cohen, M.F.: The lumigraph. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. pp. 43–54 (1996) 4
19. Harris, C., Stephens, M.: A combined corner and edge detector. In: In Proc. of Fourth Alvey Vision Conference. pp. 147–151 (1988) 2
20. Hedman, P., Alsisan, S., Szeliski, R., Kopf, J.: Casual 3d photography. ACM Transactions on Graphics (TOG) **36**(6), 1–15 (2017) 4
21. Hedman, P., Kopf, J.: Instant 3d photography. ACM Transactions on Graphics (TOG) **37**(4), 1–12 (2018) 4
22. Hu, R., Ravi, N., Berg, A.C., Pathak, D.: Worldsheet: Wrapping the world in a 3d sheet for view synthesis from a single image. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 12528–12537 (2021) 3
23. Jensen, R., Dahl, A., Vogiatzis, G., Tola, E., Aanæs, H.: Large scale multi-view stereopsis evaluation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 406–413 (2014) 12
24. Johari, M.M., Lepoittevin, Y., Fleuret, F.: Geonerf: Generalizing nerf with geometry priors. arXiv preprint arXiv:2111.13539 (2021) 4, 12
25. Kellnhofer, P., Jebe, L.C., Jones, A., Spicer, R., Pulli, K., Wetzstein, G.: Neural lumigraph rendering. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4287–4297 (2021) 3
26. Lassner, C., Zollhofer, M.: Pulsar: Efficient sphere-based neural rendering. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1440–1449 (2021) 3
27. Levoy, M., Hanrahan, P.: Light field rendering. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. pp. 31–42 (1996) 4, 5
28. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2117–2125 (2017) 2
29. Liu, L., Gu, J., Zaw Lin, K., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. Advances in Neural Information Processing Systems **33**, 15651–15663 (2020) 3
30. Liu, S., Zhang, Y., Peng, S., Shi, B., Pollefeys, M., Cui, Z.: Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2019–2028 (2020) 3
31. Liu, Y., Peng, S., Liu, L., Wang, Q., Wang, P., Theobalt, C., Zhou, X., Wang, W.: Neural rays for occlusion-aware image-based rendering. arXiv preprint arXiv:2107.13421 (2021) 4, 12
32. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004) 2
33. Lu, J., Batra, D., Parikh, D., Lee, S.: Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. Advances in neural information processing systems **32** (2019) 5
34. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th International Joint Conference

on Artificial Intelligence - Volume 2. p. 674–679. IJCAI'81, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1981) 2
35. Mildenhall, B., Hedman, P., Martin-Brualla, R., Srinivasan, P., Barron, J.T.: Nerf in the dark: High dynamic range view synthesis from noisy raw images. arXiv preprint arXiv:2111.13679 (2021) 3
36. Mildenhall, B., Srinivasan, P.P., Ortiz-Cayon, R., Kalantari, N.K., Ramamoorthi, R., Ng, R., Kar, A.: Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (TOG) **38**(4), 1–14 (2019) 11, 12
37. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European Conference on Computer Vision (ECCV). pp. 405–421 (2020) 1, 3, 4, 7, 11, 12
38. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. arXiv preprint arXiv:2201.05989 (2022) 3
39. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. CoRR (2022), http://arxiv.org/abs/2201.05989v1 3
40. Nguyen-Phuoc, T., Li, C., Theis, L., Richardt, C., Yang, Y.L.: Hologan: Unsupervised learning of 3d representations from natural images. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 7588–7597 (2019) 3
41. Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3504–3515 (2020) 3
42. Oechsle, M., Peng, S., Geiger, A.: Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5589–5599 (2021) 3
43. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 165–174 (2019) 3
44. Pfister, H., Zwicker, M., Van Baar, J., Gross, M.: Surfels: Surface elements as rendering primitives. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques. pp. 335–342 (2000) 3
45. Reizenstein, J., Shapovalov, R., Henzler, P., Sbordone, L., Labatut, P., Novotny, D.: Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In: International Conference on Computer Vision (2021) 5
46. Research, G.: Google scanned objects, https://app.ignitionrobotics.org/GoogleResearch/fuel/collections/GoogleScannedObjects 11
47. Riegler, G., Koltun, V.: Free view synthesis. In: European Conference on Computer Vision. pp. 623–640. Springer (2020) 4
48. Riegler, G., Koltun, V.: Stable view synthesis. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12216–12225 (2021) 4
49. Rombach, R., Esser, P., Ommer, B.: Geometry-free view synthesis: Transformers and no 3d priors. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 14356–14366 (2021) 5
50. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015) 2

51. Rückert, D., Franke, L., Stamminger, M.: Adop: Approximate differentiable one-pixel point rendering. arXiv preprint arXiv:2110.06635 (2021) 3
52. Sajjadi, M.S., Meyer, H., Pot, E., Bergmann, U., Greff, K., Radwan, N., Vora, S., Lucic, M., Duckworth, D., Dosovitskiy, A., et al.: Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. arXiv preprint arXiv:2111.13152 (2021) 5
53. Schönberger, J.L., Hardmeier, H., Sattler, T., Pollefeys, M.: Comparative evaluation of hand-crafted and learned local features. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6959–6968 (2017) 2
54. Schönberger, J.L., Frahm, J.M.: Structure-from-Motion Revisited. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2016) 2
55. Schönberger, J.L., Zheng, E., Pollefeys, M., Frahm, J.M.: Pixelwise View Selection for Unstructured Multi-View Stereo. In: European Conference on Computer Vision (ECCV) (2016) 2
56. Seitz, S.M., Dyer, C.R.: View morphing. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. p. 21–30. SIGGRAPH '96, Association for Computing Machinery, New York, NY, USA (1996). https://doi.org/10.1145/237170.237196, https://doi.org/10.1145/237170.237196 4
57. Shi, J., Tomasi: Good features to track. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 593–600 (1994) 2
58. Shum, H., Kang, S.B.: Review of image-based rendering techniques. In: Visual Communications and Image Processing 2000. vol. 4067, pp. 2–13. SPIE (2000) 4
59. Shum, H.Y., Chan, S.C., Kang, S.B.: Image-based rendering. Springer Science & Business Media (2008) 4
60. Sitzmann, V., Rezchikov, S., Freeman, W.T., Tenenbaum, J.B., Durand, F.: Light field networks: Neural scene representations with single-evaluation rendering. Advances in Neural Information Processing Systems (NeurIPS) (2021) 4, 6
61. Sitzmann, V., Thies, J., Heide, F., Nießner, M., Wetzstein, G., Zollhofer, M.: Deepvoxels: Learning persistent 3d feature embeddings. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2437–2446 (2019) 3, 12
62. Suhail, M., Esteves, C., Sigal, L., Makadia, A.: Light field neural rendering. CoRR (2021), http://arxiv.org/abs/2112.09687v1 4, 5, 8, 9, 14
63. Takikawa, T., Litalien, J., Yin, K., Kreis, K., Loop, C., Nowrouzezahrai, D., Jacobson, A., McGuire, M., Fidler, S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11358–11367 (2021) 3
64. Tewari, A., Thies, J., Mildenhall, B., Srinivasan, P., Tretschk, E., Wang, Y., Lassner, C., Sitzmann, V., Martin-Brualla, R., Lombardi, S., et al.: Advances in neural rendering. arXiv preprint arXiv:2111.05849 (2021) 1, 3
65. Thies, J., Zollhöfer, M., Nießner, M.: Deferred neural rendering: Image synthesis using neural textures. ACM Transactions on Graphics (TOG) **38**(4), 1–12 (2019) 3
66. Trevithick, A., Yang, B.: Grf: Learning a general radiance field for 3d representation and rendering. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 15182–15192 (2021) 12
67. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017) 2, 4

68. Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W.: Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. arXiv preprint arXiv:2106.10689 (2021) 3
69. Wang, Q., Wang, Z., Genova, K., Srinivasan, P.P., Zhou, H., Barron, J.T., Martin-Brualla, R., Snavely, N., Funkhouser, T.: Ibrnet: Learning multi-view image-based rendering. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4690–4699 (2021) 1, 4, 5, 8, 11, 12, 14
70. Wiles, O., Gkioxari, G., Szeliski, R., Johnson, J.: Synsin: End-to-end view synthesis from a single image. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020) 3
71. Wizadwongsa, S., Phongthawee, P., Yenphraphai, J., Suwajanakorn, S.: Nex: Real-time view synthesis with neural basis expansion. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8534–8543 (2021) 3, 12, 13
72. Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., Sridhar, S.: Neural fields in visual computing and beyond (2021), https://neuralfields.cs.brown.edu/ 3
73. Yariv, L., Gu, J., Kasten, Y., Lipman, Y.: Volume rendering of neural implicit surfaces. Advances in Neural Information Processing Systems **34** (2021) 3
74. Yenamandra, T., Tewari, A., Bernard, F., Seidel, H.P., Elgharib, M., Cremers, D., Theobalt, C.: i3dmm: Deep implicit 3d morphable model of human heads. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12803–12813 (2021) 3
75. Yifan, W., Serena, F., Wu, S., Öztireli, C., Sorkine-Hornung, O.: Differentiable surface splatting for point-based geometry processing. ACM Transactions on Graphics (TOG) **38**(6), 1–14 (2019) 3
76. Yu, A., Fridovich-Keil, S., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. CoRR (2021), http://arxiv.org/abs/2112.05131v1 3
77. Yu, A., Li, R., Tancik, M., Li, H., Ng, R., Kanazawa, A.: Plenoctrees for real-time rendering of neural radiance fields. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5752–5761 (2021) 3
78. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelnerf: Neural radiance fields from one or few images. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4578–4587 (2021) 1, 4, 8, 12, 14
79. Zhou, T., Tucker, R., Flynn, J., Fyffe, G., Snavely, N.: Stereo magnification: Learning view synthesis using multiplane images. arXiv preprint arXiv:1805.09817 (2018) 3, 11