Neural Plenoptic Sampling: Learning Light-field from Thousands of Imaginary Eyes

Junxuan Li, Yujiao Shi, and Hongdong Li

Australian National University
{junxuan.li,yujiao.shi,hongdong.li}@anu.edu.au

Abstract. The Plenoptic function describes light rays observed from any given position in every viewing direction. It is often parameterized as a 5-D function $L(x, y, z, \theta, \phi)$ for a static scene. Capturing all the plenoptic functions in the space of interest is paramount for Image-Based Rendering (IBR) and Novel View Synthesis (NVS). It encodes a complete light-field (*i.e.*, lumigraph) therefore allows one to freely roam in the space and view the scene from any location in any direction. However, achieving this goal by conventional light-field capture technique is expensive, requiring densely sampling the ray space using arrays of cameras or lenses. This paper proposes a much simpler solution to address this challenge by using only a small number of sparsely configured camera views as input. Specifically, we adopt a simple Multi-Layer Perceptron (MLP) network as a universal function approximator to learn the plenoptic function at every position in the space of interest. By placing virtual viewpoints (dubbed 'imaginary eyes') at thousands of randomly sampled locations and leveraging multi-view geometric relationship, we train the MLP to regress the plenoptic function for the space. Our network is trained on a per-scene basis, and the training time is relatively short (in the order of tens of minutes). When the model is converged, we can freely render novel images. Extensive experiments demonstrate that our method well approximates the complete plenoptic function and generates high-quality results.

1 Introduction

Image-Based Rendering (IBR) for view synthesis is a long-standing problem in the field of computer vision and graphics. It has a wide range of important applications, *e.g.*, robot navigation, film industry, AR/VR applications. The plenoptic function, introduced by Adelson *et al.* [1], offers an ultimate solution to this novel view synthesis problem. The plenoptic function captures the visual appearances of a scene viewed from any viewing direction (θ, ϕ) and at any location (x, y, z). Once a complete plenoptic function (i.e. the light-field) for the entire space is available, one can roam around the space and synthesize free-viewpoint images simply by sub-sampling the plenoptic light-field.

To model the plenoptic function, the best-known methods in the literature are the light field rendering and the lumigraph [25, 20]. These approaches interpolate

rays instead of scene points to synthesize novel views. However, they require the given camera positions to be densely or regularly sampled or restrict the target image to be a linear combination of source images. Unstructured light-field/lumigraph methods [5, 13] were proposed to address this limitation; they do so by incorporating geometric reconstruction with light ray interpolation.

This paper introduces a novel solution for plenoptic field sampling from a few and often sparse and unstructured multi-view input images. Since a plenoptic function is often parameterized by a 5D function map, we use a simple Multi-Layer Perceptron (MLP) network to learn such functional map: the MLP takes a 5D vector as input and outputs an RGB color measurement, *i.e.*, $\mathbb{R}^5 \rightarrow \mathbb{R}^3$. However, capturing the complete plenoptic function for a scene remains a major challenge in practice. It requires densely placing many physical cameras or moving a camera (or even a commercial light-camera) to scan *every point and in every direction*.

To address this challenge, this paper uses an MLP to *approximate* the plenoptic function (*i.e.*, the entire light field), by placing thousands of virtual cameras (*i.e.*, imaginary eyes) during the network training. We use the available physical camera views, however a few and sparsely organized, to provide multi-view geometry constraints as the self-supervision signal to supervise the training of the MLP networks. We introduce *proxy-depth* as a bridge to ensure that the multi-view geometry relationship is well respected during the training process. Those "imaginary eyes" is sampled randomly throughout the space following a uniform distribution. We use proxy-depth to describe the estimated depth by the visual similarity among input images. Once the proxy-depth of a virtual ray is determined, we can retrieve candidate colors from input images and pass them to a color blending network to determine the real color.

2 Related Work

Conventional view synthesis. Novel view synthesis is a long-standing problem in the field of computer vision and graphics [9, 14, 46]. Conventional methods use image colors or handcrafted features to construct correspondences between the views [16, 42]. With the advance of deep networks, recent approaches employ neural networks to learn the transformation between input and target views implicitly [15, 38, 40, 57, 70]. In order to explicitly encode the geometry guidance, several specific scene representations are proposed, such as Multi-Plane Images (MPI) [69, 35, 17, 55, 60], and Layered Depth Images (LDI) [48, 50, 61]. Some Image-Based Rendering (IBR) techniques [11, 21, 42, 44, 59, 45, 49, 7] warp input view images to a target viewpoint according to the estimated proxy geometry, and then blend the warped pixels to synthesize a novel view.

Panorama synthesis. Zheng *et al.* [68] propose a layered depth panorama (LDP) to create a layered representation with a full field of view from a sparse set of images taken by a hand-held camera. Bertel *et al.* [4] investigate two blending methods for interpolating novel views from two nearby views, one is

a linear blending, and the other is a view-dependent flow-based blending. Serrano *et al.* [47] propose to synthesize new views from a fixed viewpoint 360° video. Huang *et al.* [23] employ a typical depth-warp-refine procedure in synthesizing new views. They estimate the depth map for each input image and reconstruct the 3D point cloud by finding correspondences between input images using handcrafted features. They then synthesize new views from the reconstructed point cloud. With panorama synthesis, scene roaming and lighting estimation [19, 26] is possible for AR/VR applications.

Plenoptic modeling. Early light-field/lumigraph methods [25, 20] reduce the 5D representation (position (x, y, z) and direction (θ, ϕ)) of the plenoptic function to 4D ((u, v, s, t), intersection between two image planes). They do not require scene geometry information, but either require the camera grid is densely and regularly sampled, or the target viewing ray is a linear combination of the source views [6, 30]. For unstructured settings, a proxy 3D scene geometry is required to be combined with light-field/lumigraph methods for view synthesis [5, 13]. Recent methods [65, 24, 56, 64] applied learning methods to improve light field rendering.

Neural rendering. Deep networks have also demonstrated their capability of modeling specific scenes as implicit functions [36, 39, 52, 53, 58, 67, 32, 31, 66, 54, 41]. They encapsulate both the geometry and appearance of a scene as network parameters. They take input as sampled points along viewing rays and output the corresponding color and density values during the inference stage. The target image is then rendered from the sampled points by volume rendering techniques [33]. The denser the sampled points, the higher quality of rendered images. However, densely sampling points along viewing rays would significantly increase the rendering time, prohibiting interactive applications to real-world scenarios.

Neural Radiance Fields. Our idea of using MLP to learn light-field is similar Neural radiance fields [36, 22, 29, 10], which estimate the radiance emitted at 3D location by a network; a recent work, DoNeRF [37] speed up the rendering process by only querying a few samples around the estimated depth; but they requires ground-truth depth map for each viewing ray during training; FastNeRF [18] is proposed to accelerate the rendering speed during inference by caching pre-sampled scene points. IBRNet [63] is another recent NeRF-based work that utilizes neighboring features and a transformer for image rendering. MVSNeRF [8] constructed a cost volume from nearby views and use these features to help regressing the neural networks. LFNs [51] proposed a neural implicit representation for 3D light field. However, their method is limited to simple objects and geometries.

There are key differences between ours and previous works. Previous methods focused on estimating the lights emitted at every location, in any direction, within a bounded volumetric region, often enclosing the 3D scene or 3D object of interest. In contrast, our method focuses on estimating all the light rays observed at any point in space, coming in any direction. In essence, our formulation is not only close to, but precisely is, the plenoptic function that Adelson and Bergen had

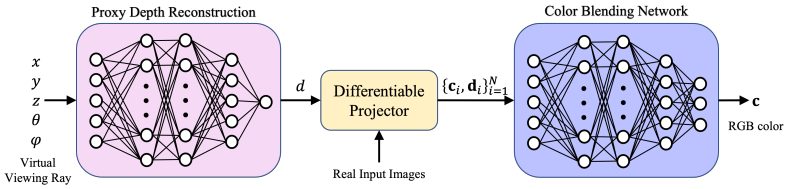


Fig. 1: The overall pipeline of the proposed framework. Our framework includes a proxy depth reconstruction (PDR) model to determine the depth of a virtual viewing ray, a differentiable ray tracer to retrieve corresponding colors from real input images, and a color blending network (CBNet) to recover the RGB color information.

contrived. In fact, in principle, our formulation can be extended to the original 7D plenoptic function by adding time and wavelength as new dimensions [28, 3, 27]. Our method also offers a computational advantage over NeRF. Namely, when the model has been well-approximated, we can directly display the network output as rendered images without sampling points along viewing rays and then rendering them in a back-to-front order. Our model will significantly accelerate the rendering speed and facilitate interactive applications.

3 Neural Plenoptic Sampling

A complete plenoptic function corresponds to the holographic representation of the visual world. It is originally defined as a 7D function $L(x, y, z, \theta, \phi, \lambda, t)$ which allows reconstruction of every possible view (θ, ϕ) from any position (x, y, z), at any time t and every wavelength λ. McMillan and Bishop [34] reduce its the dimensionality from 7D to 5D by ignoring the time and wavelength for the purpose of static scene view synthesis. By restricting the viewpoints or the object inside a box, light field [25] and lumigraph [20] approaches reduce the dimensionality to four. Without loss of generality, this paper uses original 5D representations $L(x, y, z, \theta, \phi)$ for plenoptic function and focuses on the scene representation at a fixed time.

We model the plenoptic function by an MLP. However, a brute-force training of a network mapping from *viewing* position and direction to RGB colors is infeasible. The observed images only have a partial coverage of the input space. By using the above training method, the model may fit well on the observed viewpoints, but also generates highly-blurred images on the non-observed regions. Our experiments in Fig. 5 demonstrate this situation.

To address this problem, we introduce an Imaginary Eye Sampling (IES) method to fully sample the target domain. We evaluate a proxy depth to provide self-supervision by leveraging photo-consistency among input images. Our method firstly outputs a proxy depth for a virtual viewing ray from the imaginary eye we randomly placed in the scene. Then, we retrieve colors from input views by a differentiable projector using this depth. Lastly, the colors pass through a color blending network to generate the real color. Figure. 1 depicts the overall pipeline of our framework.

3.1 Proxy depth reconstruction

We model the Proxy Depth Reconstruction (PDR) network by an MLP network F_Θ. It takes input as a camera position $\mathbf{x} = [x, y, z]^T \in \mathbb{R}^3$ and a camera viewing ray $\mathbf{v} = [\theta, \phi]^T \in \mathbb{R}^2$. The network estimates the distance value $d \in \mathbb{R}_+$ between the location $\mathbf{x}$ of the virtual camera and its observing scene in viewing direction,

$$d = F_\Theta(\mathbf{x}, \mathbf{v}), \tag{1}$$

where Θ represents the trainable network parameters.

We use a similar MLP structure from NeRF [36] to parameterize the neural plenoptic modeling. The difference is that NeRF approximates the emitting colors and transparency on the scene objects location, while our PDR model estimates the distance between the scene objects and observing cameras along the viewing direction.

3.2 Imaginary eye sampling

Since our purpose is to move around the scene and synthesize new views continuously, we need to sample the input space for the network training densely. However, in general, the camera locations of input images are sparsely sampled. The observed images only cover partial regions of such an input space.

To address this problem, we propose an Imaginary Eye Sampling (IES) strategy. We place thousands of imaginary eyes (virtual cameras) in the space of interest. Those imaginary eyes are randomly generated in the space to allow dense sampling of the plenoptic input space. By doing so, we are able to approximate a whole complete plenoptic function.

Here, note that we do not have ground-truth depths for supervision, even for real-observed images (viewing rays). In order to provide training signals, we propose a self-supervision method by leveraging photo-consistency among real input images.

3.3 Self-supervision via photo-Consistency

Given a virtual camera at a random location $\mathbf{x}$ and a viewing direction $\mathbf{v}$, our network predicts a depth d between the observed scene point and the input camera location. The world coordinate $\mathbf{w}$ of the scene point is then computed as

$$\mathbf{w} = \mathbf{x} + d\mathbf{v}. \tag{2}$$

When the estimated depth d is at the correct value, the colors of its projected pixels on real observed images should be consistent with each other. Hence, we then use a differentiable projector $T(\cdot)$ to find the projected pixel of this scene point at real camera image planes. Denote $\mathbf{P}_i$ as the projection matrix of real camera i. The projected image coordinate of a 3D point $\mathbf{w}$ is computed as $[u_i, v_i, 1]^T = \mathbf{P}_i\mathbf{w}$. Our projector then uses bilinear interpolation to fetch information (*e.g.*, color) from the corresponding real images.

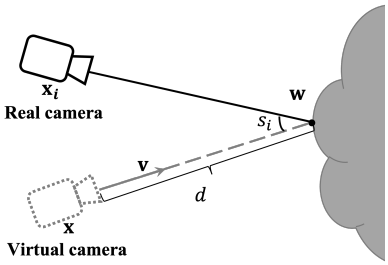


Fig. 2: For a virtual camera position $\mathbf{x}$ and viewing direction $\mathbf{v}$, we estimate a depth d between the scene point $\mathbf{w}$ and the camera location $\mathbf{x}$. By reprojecting the scene point to real cameras, we retrieve the color $\mathbf{c}_i$ and high-level feature $\mathbf{f}_i$ from the observed images. The cosine distance (angle) s_i between the virtual viewing direction and real viewing direction determine the influence of corresponding real cameras when calculating the photometric consistency.

By computing the photo-consistency (similarity) among the retrieved colors, we can measure the correctness of the estimated depth. In practice, we argue that only using the colors of the retrieved pixels is not accurate enough for this measurement because it cannot handle textureless and reflective regions. To increase the representative and discriminative ability, we propose to retrieve colors as well as high-level features from real input images for the photo-consistency measure.

Denote $\mathbf{f}_i$ and $\mathbf{c}_i$ as the retrieved features and colors from input camera i, respectively. The photo-consistency among all input cameras is defined as

$$\mathcal{L}_d = \sum_{i=1}^{N} s_i \left(\|\mathbf{c}^{\text{top}_k} - \mathbf{c}_i\|_1 + \lambda \|\mathbf{f}^{\text{top}_k} - \mathbf{f}_i\|_1 \right), \tag{3}$$

where $\|\cdot\|_1$ denotes the L_1 distance, N is the number of real input cameras, λ is the balance of the influence between color difference and the feature difference, s_i is the normalized weight assigned to each real camera i, and it is determined by the angle difference (cosine distance) between the virtual camera viewing ray $(\mathbf{w} - \mathbf{x})$ and the real camera viewing ray $(\mathbf{w} - \mathbf{x_i})$. Figure. 2 illustrates the situation. Mathematically, it is expressed as

$$s_i = \frac{\cos\left(\mathbf{w} - \mathbf{x}, \mathbf{w} - \mathbf{x_i}\right)}{\sum_{j=1}^{N} \cos\left(\mathbf{w} - \mathbf{x}, \mathbf{w} - \mathbf{x_j}\right)}, \tag{4}$$

where $\cos(\cdot, \cdot)$ is the cosine of the angle spanned by the two vectors. The smaller of the angle between the virtual camera viewing ray and the real camera viewing ray, the larger s_i is. Given the weight for each input camera, the reference color $\mathbf{c}^{\text{top}_k}$ and feature $\mathbf{f}^{\text{top}_k}$ in Eq. 3 is computed as the average of top k retrieved colors and features

$$\mathbf{c}^{\text{top}_k} = \sum_{i \in \text{top}_k} \mathbf{c}_i / k, \qquad \mathbf{f}^{\text{top}_k} = \sum_{i \in \text{top}_k} \mathbf{f}_i / k. \tag{5}$$

We use Eq. 3 as the supervision for our PDR model and the network is trained to minimize this objective function.

3.4 Color blending for view synthesis

Given an estimated depth d for a virtual viewing ray, we can retrieve colors from real input images for the virtual camera view synthesis. However, a naive aggregation of the retrieved colors would cause severe tearing or ghosting artifacts in the synthesized images. Hence, we propose a Color Blending Network (CBNet) to blend the retrieved colors and tolerate the errors caused by inaccurate depths to synthesize realistic images.

In order to provide sufficient clues, we feed the direction differences between the reprojected real viewing rays (solid line in Fig. 2) and the virtual (target) viewing ray (dash line in Fig. 2) along with the retrieved colors to the color blending network. Formally, our CBNet is expressed as

$$\mathbf{c} = F_{\Phi}\left(\{\mathbf{c}_i, \mathbf{d}_i\}_{i=1}^{N}\right), \tag{6}$$

where Φ is the trainable parameter of the CBNet, $\mathbf{c}_i$ is the RGB color retrieved from the real camera i and $\mathbf{d}_i$ is the projection of the real viewing ray on the target virtual viewing ray, $\mathbf{c}$ is the estimated color of the virtual viewing ray. We employ a Pointnet network architecture for our CBNet. The supervision of our CBNet is the colors observed from real cameras, denoted as

$$\mathcal{L}_{\mathbf{c}} = \|\mathbf{c}^* - \mathbf{c}\|_1, \tag{7}$$

where $\mathbf{c}^*$ is the ground truth colors.

Unlike our PDR model, the CBNet is trained only on the observed images (viewing rays) since it needs the ground-truth color as supervision. Instead of remembering the color of each training ray, the CBNet is trained to learn a sensible rule for blending retrieved colors from real input views. Thus it is able to be generalized to unseen viewing rays. The PDR and the CBNet in our framework are trained separately. During the training of CBNet, we fix the model parameters of PDR to not destroy the learned patterns for the whole plenoptic space. For inference, a query viewing ray first passes through our PDR model to compute a depth value; its corresponding colors on real input views are then retrieved and fed into the CBNet to estimate the color information. Since it is a single feed-forward pass through the network, the rendering speed is rapid (less than one second when rendering a 1024×512 image).

4 Experiments

In this section, we conduct comprehensive experiments to demonstrate the effectiveness of the proposed algorithm. We use 360° panoramas captured by an omnidirectional camera for the plenoptic modeling, since its representation well

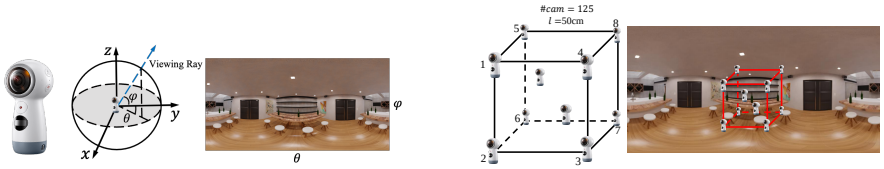


Fig. 3: (a) An illustration of an omni-directional camera and its captured light-field and a sample image. (b) An illustration of our camera arrangement for dataset generation. For each scene, we capture 125 omnidirectional images at different locations for evaluation. The cameras are positioned in a $50 \times 50 \times 50$ centimeter volume (roughly) at the center of each scene.

Table 1: Quantitative comparison of our method and others given eight input views. Here, **bold** indicates the best results and underline denotes the second best results.

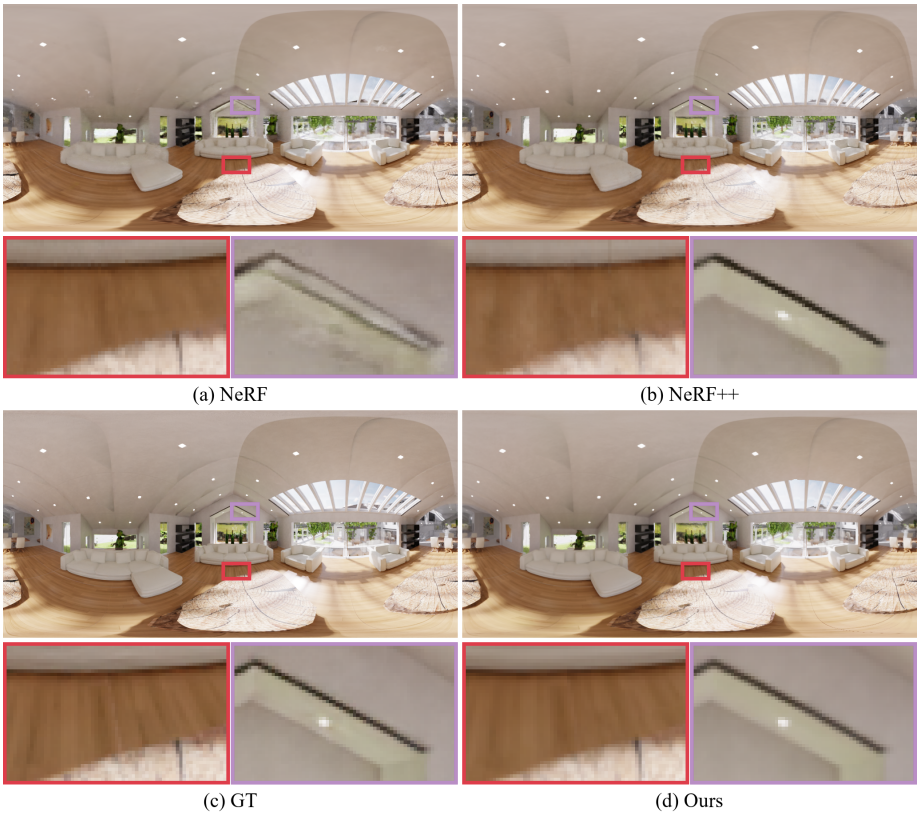
	Diningroom		Bar		Livingroom		Lounge		Average	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
FVS [44]	26.09	0.770	24.54	0.800	25.61	0.780	21.23	0.690	24.37	0.760
NeRF [36]	37.54	0.938	33.95	0.941	33.62	0.936	31.96	0.939	34.27	0.939
NeRF++ [67]	36.29	0.931	32.87	0.936	33.72	0.929	34.05	0.947	34.23	0.936
IBRNet [63]	**37.83**	0.953	**34.12**	0.959	33.39	0.941	32.35	0.953	34.42	0.952
Ours w/o Imaginary Eye	32.32	0.929	32.93	0.950	32.57	0.948	30.50	0.932	32.08	0.940
Ours w/o Feature	36.03	0.957	33.47	0.954	33.97	0.957	32.17	0.960	33.91	0.957
Ours w/o weighting	32.69	0.931	29.57	0.903	30.81	0.919	29.18	0.925	30.56	0.920
Ours	36.62	**0.959**	33.86	**0.961**	**34.33**	**0.965**	**34.31**	**0.968**	**34.78**	**0.963**

aligns with the plenoptic function. We show an omnidirectional camera, its imaging geometry, and an example image in Fig. 3. The pixel coordinates of a 360° panorama correspond to the azimuth angle θ and the elevation angle ϕ of the viewing rays.

4.1 Datasets and evaluations

Synthetic dataset When the plenoptic function has been correctly (approximately) modeled, we want to freely move across the space to synthesize new views. For the purpose of performance evaluation, we need to sample evaluation viewpoints densely in the space and their corresponding ground truth data. Hence, we propose to synthesize a dataset for our evaluation. Following recent novel view synthesis methods, we use SSIM and PSNR for the performance evaluation.

We use Blender [12] to synthesize images with freely moving camera viewpoints. Figure. 3 shows the camera setting. Specifically, we randomly sample 125 points in a $50 \times 50 \times 50$ cm^3 volume within the space and synthesize corresponding omnidirectional images. The images are generated from four scenes, *i.e.*, "Bar", "Livingroom", "Lounge" and "Diningroom". We refer the readers to our qualitative comparisons for the visualization of sampled images from the four scenes. This evaluation set is adopted for all the experiments in this paper, although the input views and training methods might be changed.



(a) NeRF (b) NeRF++

(c) GT (d) Ours

Fig. 4: Qualitative comparison with NeRF and NeRF++ on our generated scenes "Lounge". Our method generates sharper results than the comparison algorithms.

Real dataset To fully demonstrate the effectiveness of the proposed method, we also conduct experiments on real-world data. The real-world data we use are from [62], which sparsely captured two images per scene. We only provide qualitative results for visual evaluation, and interested readers are suggested to watch our supplementary video for more results.

4.2 Training details

We train a separate plenoptic function for each scene. To approximate the sharp edge of real world objects and textures, our plenoptic function model usually has high frequency output in both the viewing rays and camera position. We encode the 5D input into Fourier features as the positional encoding [36] before feeding it into the proxy-depth reconstruction network. The PDR network F_Θ is designed following the structure of NeRF. It consists of 8 fully-connected (fc) layers with 256 hidden channels, and a ReLU activation layer follows each fc

Table 2: Quantitative comparison with volume-based method on two input views.

	Diningroom		Bar		Livingroom		Lounge	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
360SD-Net [62]	24.76	0.746	23.38	0.781	23.30	0.747	21.10	0.700
Ours (Vertical)	**27.54**	**0.910**	**27.29**	**0.918**	**28.20**	**0.907**	**26.13**	**0.888**
MatryODShka [2]	20.43	0.673	27.26	0.864	23.85	0.766	22.19	0.765
Ours (Horizontal)	**30.50**	**0.921**	**28.20**	**0.918**	**29.07**	**0.907**	**27.68**	**0.898**

Table 3: Training and testing time comparison with NeRF and NeRF++ given eight unstructured input views. The testing time is for rendering images with resolution of 512×1024.

	Training (hours)	Testing (seconds)
NeRF [36]	10	30
NeRF++ [67]	20	110
Ours	2	0.14

layer. We use the pretrained model from Shi *et al.* [49] as our feature extractor to compute photo consistency.

When training the MLP, we randomly sample a virtual camera at location $\mathbf{x}$ and draw an arbitrary viewing direction $\mathbf{v}$. Given this 5D input, the MLP estimates a proxy-depth d in the output, which is then self-supervised by the photometric consistency loss $\mathcal{L}_d$. The above network is end-to-end differentiable. Once we have sampled and trained the virtual camera domain thoroughly, the MLP for proxy-depth reconstruction is then frozen for the training of the color blending network later.

The CBNet takes a series color and direction $(\mathbf{c}_i, \mathbf{d}_i)$ to inference the output color of the plenoptic function. Its design follows the structure of the PointNet [43]. The observations from real cameras are firstly processed separately by three fc layers. Next, a max-pooling layer is applied to select the most salient features from them. We then employ a prediction layer to generate the color values $\mathbf{c}$.

In our experiments, we use 200k rays per iteration for the PDR network training, and 100k rays for the CBNet training. Our model is trained from scratch with an Adam optimizer. The learning rate is set to 5×10^{-4}. The PDR network takes around 30k iterations to converge, while the CBNet only takes 10k iterations. The complete model takes around one hour to converge in a NVIDIA RTX 3090 GPU.

4.3 Comparison with the state-of-the-art

Comparison with NeRF variants. We conduct experiments to compare with NeRF and its variants NeRF++ [67], IBRNet [63]. In this comparison, all of

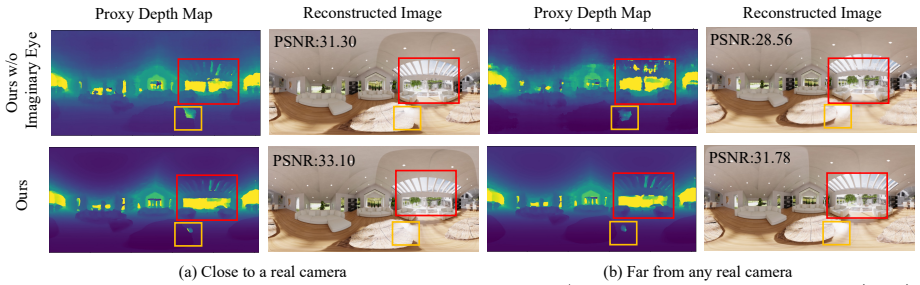


Fig. 5: Qualitative comparisons of our method w or w/o Imaginary Eye Sampling (IES). Without using IES, the image synthesized at a position far from any real camera (top right) suffers much lower quality compared to the one closer to a real camera (top left) (2.74dB drop). When the IES is applied, the quality of both images (bottom left and bottom right) improves, and the PSNR gap decreases (1.32dB).

the methods take eight views as input. The quantitative evaluation results are presented in Table 1. Visual comparison is presented in Fig. 4. It can be seen that our method achieves better performance than NeRF and NeRF++ in most of the scenarios. Other NeRF variants aim to estimate the radiance emitted by scene points at any position and direction, while our method is designed to recover the irradiance perceived by an observer from any point and direction. Since NeRF and NeRF++ need to sample points along viewing rays and render them in a back-to-front order, they require hundreds of network calls when synthesizing an image. Thus their rendering time is very long. In contrast, our method directly outputs the color information given a viewing ray. Thus, our training and testing time are relatively shorter, as shown in Table 3.

Comparison with color blending approaches. To demonstrate the effectiveness of our CBNet. We compare our CBNet with another image-based warping method FVS [44]. FVS is a neural network based color blending approach for view synthesis. We retrain their network on our datasets. The results are presented in the first row of Table 1. It is evident that our method achieves significantly better performance.

Comparison with panorama synthesis approaches. We employ a deep-based method 360SD-Net [62] to estimate depth maps for input images. We then build a point cloud from the depth map and input images. The point cloud are warped and refined for novel view synthesis. Since 360SD-Net only takes two vertically aligned panoramas as input, we take the same vertical inputs in this comparison, denoted as "Our (Vertical)" in Tab. 2. We further compare with a multi-sphere-images-based method MatryODShka [2] on view synthesis. Note that MatryODShka only takes two horizontally aligned panoramas as input. For fair comparison, we take the same input and denoted as "Our (Horizontal)" in Tab. 2. The numerical evaluations in Tab. 2 demonstrate that our method significantly outperforms the conventional depth-warp-refine and multi-sphere-images procedure in synthesizing new views. Besides, the competing methods both re-

Table 4: Quantitative comparison of different imaginary eye sampling (IES) regions (large or small). Larger imaginary eye sampling space contributes to higher image quality.

Scene	Lounge				Livingroom			
N	2		4		2		4	
IES Range	PSNR↑	SSIM↑	PSNR	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
Small	24.82	0.8484	27.98	0.8995	26.77	0.8690	29.93	0.9144
Large	**26.13**	**0.8883**	**29.13**	**0.9193**	**28.20**	**0.9068**	**31.27**	**0.9383**

quires a structured input (horizontally or vertically aligned). This limitation does not apply to our method.

4.4 Effectiveness of Imaginary Eye Sampling

We demonstrate the necessity and effectiveness of the imaginary eye sampling strategy. In doing so, we train our network only using real camera locations and directions, without any imaginary eye sampling, denoted as "Ours w/o Imaginary Eye". The quantitative results and qualitative evaluations are presented in Tab. 1 and Fig. 5 respectively. For better comparison, we select two images for visualization. One is close to a real camera, and the other is far from input cameras.

As illustrated by the results, the performance of "Ours w/o Imaginary Eye" is inferior to our whole pipeline. More importantly, the performance gap between images that are near and far from the real camera is significant. There is 2.75dB difference in terms of PSNR metric. This demonstrates that the model learns better for training data while does not have the ability to interpolate similar-quality test data.

A network is usually good at learning a continuous representation from discrete but uniformly distributed samples in a general case. In our plenoptic modeling, the values of the input parameters (x, y, z, θ, ϕ) are continuous and always span in a large range, while the input images only cover small and sparsely sampled regions in the whole space. Hence, it is not surprising that the model can fit well in training data while interpolating low-quality images at camera locations far from real cameras. Using our imaginary eye sampling strategy, the performance gap between the two cases is reduced (1.32dB in terms of PSNR). Furthermore, the quality of synthesized images on the location that is near to a real camera is further improved. This is owed to our photometric consistency self-supervision loss for the virtual eye training. It helps the learned model to encode the geometry constraints across different viewpoint images.

We also conduct comparison experiments on the imaginary eye sampling area (large or small). The results are shown in Tab. 4. We found that sampling on a larger region will allow more freedom on the moving space of rendering cameras, while the downside is that it requires longer training time.

Table 5: Quantitative evaluations on different input view numbers.

N	2		4		8		25	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
Lounge	26.13	0.8883	29.13	0.9193	34.31	0.9684	37.27	0.9775
Livingroom	28.20	0.9068	31.27	0.9383	34.33	0.9648	36.72	0.9746

4.5 Proxy-depth from colors and features

In what follows, we conduct experiments to demonstrate why the features are required in our photometric consistency loss. In doing so, we remove the feature item in Eq. 3 and train our model again, denoted as "Ours w/o Feature". The quantitative results are presented in the third last row of Tab. 1.

Compared to pixel-wise RGB colors, features have a larger reception field that makes textureless regions discriminative, and the encoded higher level information is more robust to illumination changes and other noises. Thus, the reconstructed proxy depths from both RGB colors and features are more accurate than those purely from RGB colors. Consequently, the quality of synthesized images is facilitated. We present the qualitative illustrations in the supplementary material.

4.6 With or without view-direction weighting

We also ablate the necessity of the view-direction based weighting in Eq. 3. In this experiment, we set the weighting term s_i to one, denoted as "Ours w/o weighting". The results are presented in the second last row of Tab. 1. Not surprisingly, the performance drops. This demonstrates the effectiveness of our weighting strategy.

4.7 Different Number of Input Views

Below, we conduct experiments on a different number of input views. For this experiment setting, we aim to investigate the performance difference when the input views are located on a line, a flat plane, and a cube, corresponding to 2, 4, 8, and 25 input views, respectively. Tab. 5 and Fig. 6 provide the quantitative and qualitative results, respectively. As shown by the results, when the input view number is reduced to 2, our method still generates acceptable quality novel view images. As the number of input views increases, the quality of the view synthesis improves rapidly. For 8 and 25 input views in this experiment, the input cameras are randomly sampled within the region (cube) of interest. This demonstrates that our method is not limited to structured settings and can synthesize free-viewpoint images from unstructured input images.

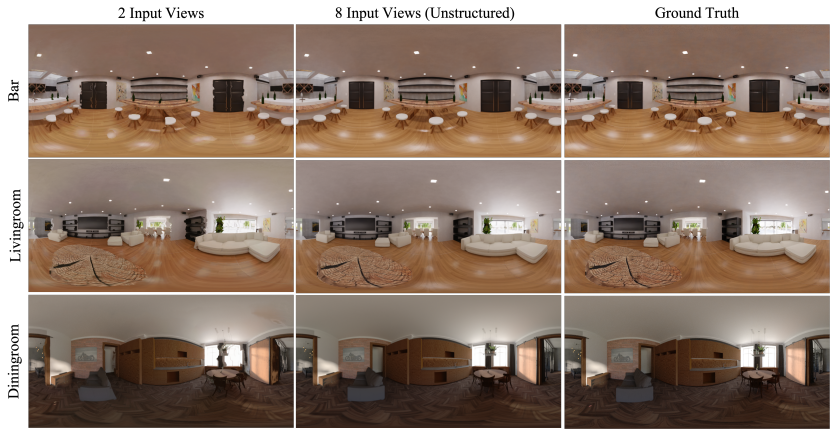


Fig. 6: Qualitative visualization of synthesized images by different view number and camera configurations. The three images are from our generated scenes “Bar”, “Livingroom” and “Diningroom” respectively.

5 Conclusion

Capturing a complete and dense plenoptic function from every point and angle within a space has been the “holy grail" for IBR-based view synthesis applications. There is always a tension between how densely one samples the space using many real cameras and the total efforts and cost that one has to bear in doing this task. This paper proposes a simple yet effective solution to this challenge. By placing thousands of imaginary eyes (virtual cameras) at randomly sampled positions in the space of interest, this paper proposes a new neural-network-based method to learn (or to approximate) the underlying 5D plenoptic function. Real images captured by physical cameras are used as a teacher to train our neural network. Although those randomly placed imaginary eyes themselves do not provide new information, they are critical to the success of our method, as they provide a bridge to leverage the existing multi-view geometry relationship among all the views (of both real and virtual). Our experiments also validate this claim positively and convincingly. Our method produces accurate and high-quality novel views (on the validation set) and compelling visual results (on unseen testing images). We will release the code and models in this paper.

Limitations and future works: The training of the PDR network uses photo-consistency constraint. The proposed method may not work well when such constraint is violated, such as transparent and large areas of occlusion. While the followed CBNet can learn how to resolve these challenging scenarios to some extent, this is not a principled rule. We expect that more sophisticated solutions can be proposed in the future.

Acknowledgments This research is funded in part by ARC-Discovery grants (DP190102261 and DP220100800).

References

1. Adelson, E.H., Bergen, J.R., et al.: The plenoptic function and the elements of early vision, vol. 2. Vision and Modeling Group, Media Laboratory, Massachusetts Institute of ... (1991)
2. Attal, B., Ling, S., Gokaslan, A., Richardt, C., Tompkin, J.: Matryodshka: Real-time 6dof video view synthesis using multi-sphere images. In: European Conference on Computer Vision. pp. 441–459. Springer (2020)
3. Bemana, M., Myszkowski, K., Seidel, H.P., Ritschel, T.: X-fields: implicit neural view-, light-and time-image interpolation. ACM Transactions on Graphics (TOG) **39**(6), 1–15 (2020)
4. Bertel, T., Campbell, N.D., Richardt, C.: Megaparallax: Casual 360° panoramas with motion parallax. IEEE transactions on visualization and computer graphics **25**(5), 1828–1835 (2019)
5. Buehler, C., Bosse, M., McMillan, L., Gortler, S., Cohen, M.: Unstructured lumigraph rendering. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques. pp. 425–432 (2001)
6. Chai, J.X., Tong, X., Chan, S.C., Shum, H.Y.: Plenoptic sampling. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques. pp. 307–318 (2000)
7. Chaurasia, G., Duchene, S., Sorkine-Hornung, O., Drettakis, G.: Depth synthesis and local warps for plausible image-based navigation. ACM Transactions on Graphics (TOG) **32**(3), 1–12 (2013)
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: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14124–14133 (2021)
9. Chen, S.E., Williams, L.: View interpolation for image synthesis. In: Proceedings of the 20th annual conference on Computer graphics and interactive techniques. pp. 279–288 (1993)
10. Chibane, J., Bansal, A., Lazova, V., Pons-Moll, G.: Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7911–7920 (2021)
11. Choi, I., Gallo, O., Troccoli, A., Kim, M.H., Kautz, J.: Extreme view synthesis. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 7781–7790 (2019)
12. Community, B.O.: Blender - a 3d modelling and rendering package (2020), `http://www.blender.org`
13. Davis, A., Levoy, M., Durand, F.: Unstructured light fields. In: Computer Graphics Forum. vol. 31, pp. 305–314. Wiley Online Library (2012)
14. Debevec, P.E., Taylor, C.J., Malik, J.: Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. pp. 11–20 (1996)
15. Eslami, S.A., Rezende, D.J., Besse, F., Viola, F., Morcos, A.S., Garnelo, M., Ruderman, A., Rusu, A.A., Danihelka, I., Gregor, K., et al.: Neural scene representation and rendering. Science **360**(6394), 1204–1210 (2018)
16. Fitzgibbon, A., Wexler, Y., Zisserman, A.: Image-based rendering using image-based priors. International Journal of Computer Vision **63**(2), 141–151 (2005)

17. Flynn, J., Broxton, M., Debevec, P., DuVall, M., Fyffe, G., Overbeck, R., Snavely, N., Tucker, R.: Deepview: View synthesis with learned gradient descent. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2367–2376 (2019)
18. Garbin, S.J., Kowalski, M., Johnson, M., Shotton, J., Valentin, J.: Fastnerf: High-fidelity neural rendering at 200fps. arXiv preprint arXiv:2103.10380 (2021)
19. Gera, P., Dastjerdi, M.R.K., Renaud, C., Narayanan, P., Lalonde, J.F.: Casual indoor hdr radiance capture from omnidirectional images. arXiv preprint arXiv:2208.07903 (2022)
20. 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)
21. Hedman, P., Philip, J., Price, T., Frahm, J.M., Drettakis, G., Brostow, G.: Deep blending for free-viewpoint image-based rendering. ACM Transactions on Graphics (TOG) **37**(6), 1–15 (2018)
22. Hedman, P., Srinivasan, P.P., Mildenhall, B., Barron, J.T., Debevec, P.: Baking neural radiance fields for real-time view synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5875–5884 (2021)
23. Huang, J., Chen, Z., Ceylan, D., Jin, H.: 6-dof vr videos with a single 360-camera. In: 2017 IEEE Virtual Reality (VR). pp. 37–44. IEEE (2017)
24. Kalantari, N.K., Wang, T.C., Ramamoorthi, R.: Learning-based view synthesis for light field cameras. ACM Transactions on Graphics (TOG) **35**(6), 1–10 (2016)
25. Levoy, M., Hanrahan, P.: Light field rendering. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. pp. 31–42 (1996)
26. Li, J., Li, H., Matsushita, Y.: Lighting, reflectance and geometry estimation from 360 panoramic stereo. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10586–10595. IEEE (2021)
27. Li, T., Slavcheva, M., Zollhoefer, M., Green, S., Lassner, C., Kim, C., Schmidt, T., Lovegrove, S., Goesele, M., Lv, Z.: Neural 3d video synthesis. arXiv preprint arXiv:2103.02597 (2021)
28. Li, Z., Xian, W., Davis, A., Snavely, N.: Crowdsampling the plenoptic function. In: European Conference on Computer Vision. pp. 178–196. Springer (2020)
29. Lin, K.E., Yen-Chen, L., Lai, W.S., Lin, T.Y., Shih, Y.C., Ramamoorthi, R.: Vision transformer for nerf-based view synthesis from a single input image. arXiv preprint arXiv:2207.05736 (2022)
30. Lin, Z., Shum, H.Y.: On the number of samples needed in light field rendering with constant-depth assumption. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662). vol. 1, pp. 588–595. IEEE (2000)
31. Liu, L., Gu, J., Lin, K.Z., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. arXiv preprint arXiv:2007.11571 (2020)
32. Martin-Brualla, R., Radwan, N., Sajjadi, M.S., Barron, J.T., Dosovitskiy, A., Duckworth, D.: Nerf in the wild: Neural radiance fields for unconstrained photo collections. arXiv preprint arXiv:2008.02268 (2020)
33. Max, N.: Optical models for direct volume rendering. IEEE Transactions on Visualization and Computer Graphics **1**(2), 99–108 (1995)
34. McMillan, L., Bishop, G.: Plenoptic modeling: An image-based rendering system. In: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques. pp. 39–46 (1995)

35. 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)
36. 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. pp. 405–421. Springer (2020)
37. Neff, T., Stadlbauer, P., Parger, M., Kurz, A., Chaitanya, C.R.A., Kaplanyan, A., Steinberger, M.: Donerf: Towards real-time rendering of neural radiance fields using depth oracle networks. arXiv preprint arXiv:2103.03231 (2021)
38. Nguyen-Ha, P., Huynh, L., Rahtu, E., Heikkila, J.: Sequential neural rendering with transformer. arXiv preprint arXiv:2004.04548 (2020)
39. Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3504–3515 (2020)
40. Park, E., Yang, J., Yumer, E., Ceylan, D., Berg, A.C.: Transformation-grounded image generation network for novel 3d view synthesis. In: Proceedings of the ieee conference on computer vision and pattern recognition. pp. 3500–3509 (2017)
41. Park, K., Sinha, U., Barron, J.T., Bouaziz, S., Goldman, D.B., Seitz, S.M., Brualla, R.M.: Deformable neural radiance fields. arXiv preprint arXiv:2011.12948 (2020)
42. Penner, E., Zhang, L.: Soft 3d reconstruction for view synthesis. ACM Transactions on Graphics (TOG) **36**(6), 1–11 (2017)
43. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)
44. Riegler, G., Koltun, V.: Free view synthesis. In: European Conference on Computer Vision. pp. 623–640. Springer (2020)
45. Riegler, G., Koltun, V.: Stable view synthesis. arXiv preprint arXiv:2011.07233 (2020)
46. Seitz, S.M., Dyer, C.R.: View morphing. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. pp. 21–30 (1996)
47. Serrano, A., Kim, I., Chen, Z., DiVerdi, S., Gutierrez, D., Hertzmann, A., Masia, B.: Motion parallax for 360 rgbd video. IEEE Transactions on Visualization and Computer Graphics **25**(5), 1817–1827 (2019)
48. Shade, J., Gortler, S., He, L.w., Szeliski, R.: Layered depth images. In: Proceedings of the 25th annual conference on Computer graphics and interactive techniques. pp. 231–242 (1998)
49. Shi, Y., Li, H., Yu, X.: Self-supervised visibility learning for novel view synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9675–9684 (2021)
50. Shih, M.L., Su, S.Y., Kopf, J., Huang, J.B.: 3d photography using context-aware layered depth inpainting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8028–8038 (2020)
51. Sitzmann, V., Rezchikov, S., Freeman, W.T., Tenenbaum, J.B., Durand, F.: Light field networks: Neural scene representations with single-evaluation rendering. arXiv preprint arXiv:2106.02634 (2021)
52. Sitzmann, V., Thies, J., Heide, F., Nießner, M., Wetzstein, G., Zollhofer, M.: Deepvoxels: Learning persistent 3d feature embeddings. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2437–2446 (2019)

53. Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene representation networks: Continuous 3d-structure-aware neural scene representations. In: Advances in Neural Information Processing Systems. pp. 1121–1132 (2019)
54. Srinivasan, P.P., Deng, B., Zhang, X., Tancik, M., Mildenhall, B., Barron, J.T.: Nerv: Neural reflectance and visibility fields for relighting and view synthesis. arXiv preprint arXiv:2012.03927 (2020)
55. Srinivasan, P.P., Tucker, R., Barron, J.T., Ramamoorthi, R., Ng, R., Snavely, N.: Pushing the boundaries of view extrapolation with multiplane images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 175–184 (2019)
56. Srinivasan, P.P., Wang, T., Sreelal, A., Ramamoorthi, R., Ng, R.: Learning to synthesize a 4d rgbd light field from a single image. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2243–2251 (2017)
57. Sun, S.H., Huh, M., Liao, Y.H., Zhang, N., Lim, J.J.: Multi-view to novel view: Synthesizing novel views with self-learned confidence. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 155–171 (2018)
58. Thies, J., Zollhöfer, M., Nießner, M.: Deferred neural rendering: Image synthesis using neural textures. ACM Transactions on Graphics (TOG) **38**(4), 1–12 (2019)
59. Thies, J., Zollhöfer, M., Theobalt, C., Stamminger, M., Nießner, M.: Image-guided neural object rendering. In: International Conference on Learning Representations (2019)
60. Tucker, R., Snavely, N.: Single-view view synthesis with multiplane images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 551–560 (2020)
61. Tulsiani, S., Tucker, R., Snavely, N.: Layer-structured 3d scene inference via view synthesis. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 302–317 (2018)
62. Wang, N.H., Solarte, B., Tsai, Y.H., Chiu, W.C., Sun, M.: 360sd-net: 360° stereo depth estimation with learnable cost volume. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 582–588. IEEE (2020)
63. 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: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4690–4699 (2021)
64. Wu, G., Zhao, M., Wang, L., Dai, Q., Chai, T., Liu, Y.: Light field reconstruction using deep convolutional network on epi. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6319–6327 (2017)
65. Yoon, Y., Jeon, H.G., Yoo, D., Lee, J.Y., So Kweon, I.: Learning a deep convolutional network for light-field image super-resolution. In: Proceedings of the IEEE international conference on computer vision workshops. pp. 24–32 (2015)
66. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelnerf: Neural radiance fields from one or few images. arXiv preprint arXiv:2012.02190 (2020)
67. Zhang, K., Riegler, G., Snavely, N., Koltun, V.: Nerf++: Analyzing and improving neural radiance fields (2020)
68. Zheng, K.C., Kang, S.B., Cohen, M.F., Szeliski, R.: Layered depth panoramas. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8. IEEE (2007)
69. Zhou, T., Tucker, R., Flynn, J., Fyffe, G., Snavely, N.: Stereo magnification: Learning view synthesis using multiplane images. arXiv preprint arXiv:1805.09817 (2018)

70. Zhou, T., Tulsiani, S., Sun, W., Malik, J., Efros, A.A.: View synthesis by appearance flow. In: European conference on computer vision. pp. 286–301. Springer (2016)