Light Field Networks: Neural Scene Representations with Single-Evaluation Rendering

Anonymous Author(s)
Affiliation
Address
email

Abstract

Inferring representations of 3D scenes from 2D observations is a fundamental problem of computer graphics, computer vision, and artificial intelligence. Emerging 3D-structured neural scene representations are a promising approach to 3D scene understanding. In this work, we propose a novel neural scene representation, Light Field Networks or LFNs, which represent both geometry and appearance of the underlying 3D scene in a 360-degree, four-dimensional light field parameterized via a neural implicit representation. Rendering a ray from an LFN requires only a *single* network evaluation, as opposed to hundreds of evaluations per ray for ray-marching or volumetric based renderers in 3D-structured neural scene representations. In the setting of simple scenes, we leverage meta-learning to learn a prior over LFNs that enables multi-view consistent light field reconstruction from as little as a single image observation. This results in dramatic reductions in time and memory complexity, and enables real-time rendering. The cost of storing a 360-degree light field via an LFN is two orders of magnitude lower than conventional methods such as the Lumigraph. Utilizing the analytical differentiability of neural implicit representations and a novel parameterization of light space, we further demonstrate the extraction of sparse depth maps from LFNs.

1 Introduction

A fundamental problem across computer graphics, computer vision, and artificial intelligence is to infer a representation of a scene's 3D shape and appearance given impoverished observations such as 2D images of the scene. Recent contributions have advanced the state of the art for this problem significantly. First, neural implicit representations have enabled efficient representation of local 3D scene properties by mapping a 3D coordinate to local properties of the 3D scene at that coordinate [1–6]. Second, differentiable neural renderers allow for the inference of these representations given only 2D image observations [3, 4]. Finally, leveraging meta-learning approaches such as hypernetworks or gradient-based meta-learning has enabled the learning of distributions of 3D scenes, and therefore reconstruction given only a single image observation [3]. This has enabled a number of applications, such as novel view synthesis [7, 3, 6], 3D reconstruction [5, 3] semantic segmentation [8, 9], and SLAM [10]. However, 3D-structured neural scene representations come with a major limitation: Their rendering is prohibitively expensive, on the order of *tens of seconds for a single* 256×256 *image* for state-of-the-art approaches. In particular, parameterizing the scene in 3D space necessitates the discovery of surfaces along camera rays during rendering. This can be solved either by encoding geometry as a level set of an occupancy or signed distance function, or via volumetric rendering, which solves an alpha-compositing problem along each ray. Either approach, however, requires tens or even hundreds of evaluations of the 3D neural scene representation in order to render a single camera ray.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

We propose a novel neural scene representation, dubbed Light Field Networks or LFNs. Instead of encoding a scene in 3D space, Light Field Networks encode a scene by directly mapping an oriented camera ray in the four dimensional space of light rays to the radiance observed by that ray. This obviates the need to query opacity and RGB at 3D locations along a ray or to ray-march towards the level set of a signed distance function, speeding up rendering by *three orders of magnitude* compared to volumetric methods. In addition to directly encoding appearance, we demonstrate that LFNs encode information about scene geometry in their derivatives. Utilizing the unique flexibility of neural implicit representations, we introduce the use of Plücker coordinates to parameterize 360-degree light fields, which allow for storage of a-priori unbounded scenes and admit a simple expression for the depth as an analytical function of an LFN. Using this relationship, we demonstrate the computation of geometry in the form of sparse depth maps. While 3D-structured neural scene representations are multi-view consistent *by design*, parameterizing a scene in light space does not come with this guarantee: the additional degree of freedom enables rays that view the same 3D point to change appearance across viewpoints. For the setting of simple scenes, we demonstrate that this challenge can be overcome by learning a prior over 4D light fields in a meta-learning framework. We benchmark with current state-of-the-art approaches for single-shot novel view synthesis, and demonstrate that LFNs compare favorably with globally conditioned 3D-structured representations, while accelerating rendering and reducing memory consumption by orders of magnitude.

In summary, we make the following contributions:

1. We propose Light Field Networks (LFNs), a novel neural scene representation that directly parameterizes the light field of a 3D scene via a neural implicit representation, enabling real-time rendering and vast reduction in memory utilization.
2. We demonstrate that we may leverage 6-dimensional Plücker coordinates as a parameterization of light fields, despite their apparent overparameterization of the 4D space of rays, thereby enabling continuous, 360-degree light fields.
3. By embedding LFNs in a meta-learning framework, we demonstrate light field reconstruction and novel view synthesis of simple scenes from sparse 2D image supervision only.
4. We propose an efficient algorithm to extract depth maps from 360-degree light fields leveraging the analytical differentiability of neural implicit representations, and demonstrate that inferred LFNs encode both appearance and geometry of the underlying 3D scenes.

Scope. The proposed method is currently constrained to the reconstruction of simple scenes, such as single objects and simple room-scale scenes, in line with recent work on learning generative models in this regime [3, 11].

2 Related Work

Neural Scene Representations and Neural Rendering. A large body of work addresses the question of inferring feature representations of 3D scenes useful to downstream tasks across graphics, vision, and machine learning. Models without 3D structure suffer from poor data efficiency [12, 13]. Voxel grids [14–20] offer 3D structure, but scale poorly with spatial resolution. Inspired by neural implicit representations of 3D geometry [1, 2], recent work has proposed to encode properties of 3D scenes in the weights of a neural network that maps 3D coordinates to local properties of the 3D scene at these coordinates. Using differentiable rendering, these models can be learned from image observations only [3, 4, 21, 11]. Reconstruction from sparse observations can be achieved by learning priors over the space of neural implicit representations [3, 5, 11, 22, 23] or by conditioning of the implicit representation on local features [6, 24, 25]. Differentiable rendering of 3D-structured neural scene representations is exceptionally computationally intensive, requiring hundreds of evaluations of the neural representation per ray, with tens of thousands to millions of rays per image. Some recent work seeks to accelerate test-time rendering, but either does not admit generalization [26, 27], or does not alleviate the cost of rendering at training/inference time [28–30]. With Light Field Networks, we propose to leverage 360-degree light fields as neural scene representations. We introduce a novel neural implicit parameterization of 360-degree light fields, infer light fields via meta-learning from as few as a single 2D image observation, and demonstrate that LFNs encode both scene geometry and appearance.

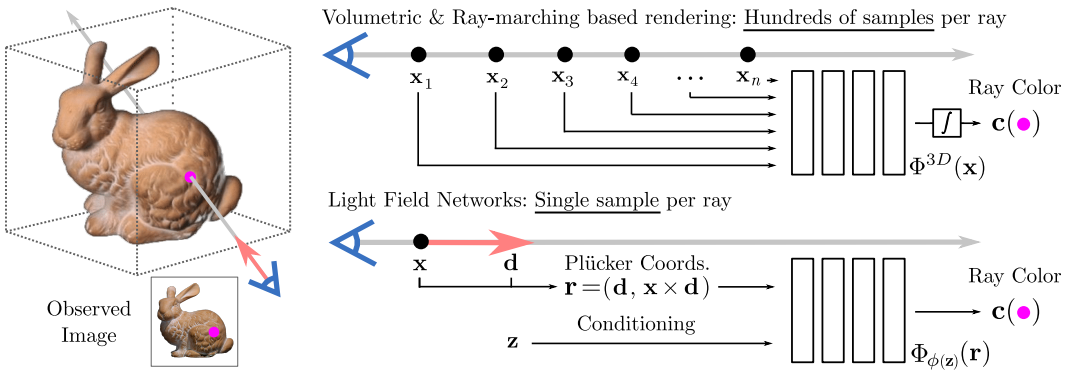


Figure 1: **Overview.** We propose Light Field Networks or LFNs, which encode the full 360-degree light field of a 3D scene in the weights of a fully connected neural network Φ_ϕ (with weights ϕ conditioned on a latent code $\mathbf{z}$) that maps an oriented ray $\mathbf{r}$ to the radiance $\mathbf{c}$ observed by that ray. Rendering an LFN Φ_ϕ only requires evaluating the underlying MLP once per ray, in contrast to 3D-structured neural scene representations Φ_{3D} such as SRNs [3], NeRF [4], or DVR [5] that require hundreds of evaluations per ray. We leverage meta-learning to learn a multi-view consistent space of LFNs. Once trained, this enables reconstruction of a 360-degree light field and subsequent real-time novel view synthesis of simple scenes from only a single observation.

Light fields and their reconstruction. Light fields have a rich history as a scene representation in both computer vision and computer graphics. Adelson et al. [31] introduced the 5D plenoptic function as a unified representation of information in the early visual system [32]. Levoy et al. [33] and, concurrently, Gortler et al. [34] introduced light fields in computer graphics as a 4D sampled scene representation for fast image-based rendering. Light fields have since enjoyed popularity as a representation for novel view synthesis [35] and computational photography, e.g. [36]. Light fields enable direct rendering of novel views by simply extracting a 2D slice of the 4D light field. However, they tend to incur significant storage cost, and since they rely on two-plane parameterizations, they make it hard to achieve a full 360-degree representation without concatenating multiple light fields. A significant amount of prior work addresses reconstruction of fronto-parallel light fields via hand-crafted priors, such as sparsity in the Fourier or shearlet domains [37–39]. With the advent of deep learning, approaches to light field reconstruction that leverage convolutional neural networks to in-paint or extrapolate light fields from sparse views have been proposed [40, 7, 41], but similarly only support fronto-parallel novel view synthesis. We are instead interested in light fields as a representation of 3D appearance and geometry that enables efficient inference of and reasoning about the properties of the full underlying scene.

3 Background: 3D-structured Neural Scene Representations

Recent progress in neural scene representation and rendering has been driven by two key innovations. The first are neural implicit (or coordinate-based) scene representations Φ^{3D} [3, 4], which model a scene as a continuous function, parameterized as an MLP which maps a 3D coordinate to a representation $\mathbf{v}$ of whatever is at that 3D coordinate:

$$\Phi^{3D} : \mathbb{R}^3 \to \mathbb{R}^n, \quad \mathbf{x} \mapsto \Phi^{3D}(\mathbf{x}) = \mathbf{v}. \tag{1}$$

The second is a differentiable renderer $\mathbf{m}$, which, given a ray $\mathbf{r}$ in $\mathbb{R}^3$, and the representation Φ^{3D}, computes the value of the color $\mathbf{c}$ of the scene when viewed along $\mathbf{r}$:

$$\mathbf{m}(\mathbf{r}, \Phi^{3D}) = \mathbf{c}(\mathbf{r}) \in \mathbb{R}^3. \tag{2}$$

Existing rendering methods broadly fall into two categories: ray-marching-based renderers [3, 42, 5, 43] and volumetric renderers [19, 4]. These methods require on the order of tens or hundreds of evaluations of the values of Φ^{3D} along a ray $\mathbf{r}$ to compute $\mathbf{c}(\mathbf{r})$. This leads to extraordinarily large memory and time complexity of rendering. As training requires error backpropagation through the renderer, this impacts both training and test time.

4 The Light Field Network Scene Representation

We propose to represent a scene as a 360-degree *neural light field*, a function parameterized by an MLP Φ_ϕ with parameters ϕ that directly maps the 4D space $\mathcal{L}$ of oriented rays to their observed radiance:

$$\Phi_\phi : \mathcal{L} \to \mathbb{R}^3, \mathbf{r} \mapsto \Phi_\phi(\mathbf{r}) = \mathbf{c}(\mathbf{r}). \tag{3}$$

A light field completely characterizes the flow of light through unobstructed space in a static scene with fixed illumination. Light fields have the unique property that rendering is achieved by a *single evaluation* of Φ per light ray, i.e., *no ray-casting is required*. Moreoever, while the light field only encodes appearance explicitly, its derivatives encode geometry information about the underlying 3D scene [44, 31, 32]. This makes many methods to extract 3D geometry from light fields possible [45–48], and we demonstrate efficient recovery of sparse depth maps from LFNs below.

4.1 Implicit representations for 360 degree light fields

To fully represent a 3D scene requires a parameterization of all light rays in space. Conventional light field methods are constrained to leverage minimal parameterizations of the 4D space of rays, due to the high memory requirements of discretely sampled high-dimensional spaces. In contrast, our use of neural implicit representations allows us to freely choose a continuous parameterization that is mathematically convenient. In particular, we propose to leverage the 6D Plücker parameterization of the space of light rays $\mathcal{L}$ for LFNs. The Plücker coordinates (see [49] for an excellent overview) of a ray ℓ through a point $\mathbf{p}$ in a normalized direction $\mathbf{d}$ are

$$\mathbf{r} = (\mathbf{d}, \mathbf{m}) \in \mathbb{R}^6 \text{ where } \mathbf{m} = \mathbf{p} \times \mathbf{d}, \text{ for } \quad \mathbf{d} \in \mathbb{S}^2, \mathbf{p} \in \mathbb{R}^3. \tag{4}$$

where $\times$ denotes the cross product. While Plücker coordinates are a-priori 6-tuples of real numbers, the coordinates of any ray lie on a 4-dimensional subspace $\mathcal{L}$. Plücker coordinates uniformly represent all oriented rays in space without singular directions or special cases. This is in contrast to conventional light field parameterizations: Fronto-parallel two-plane or cylindrical parameterizations cannot represent the full 360-degree light field of a scene [33]. Cubical two-plane arrangements [34, 35] are not continuous, complicating the parameterization via a neural implicit representation. In contrast to the two-sphere parameterization [50], Plücker coordinates do not require that scenes are bounded in size and do not require spherical trigonometry.

The parameterization via a neural implicit representation enables compact storage of a 4D light field that can be sampled at arbitrary resolutions, while non-neural representations are resolution-limited. Neural implicit representations further allow the analytical computation of derivatives. This enables the efficient computation of sparse depth maps, where prior representations of light fields require finite-differences approximations of the gradient [45–47].

Rendering LFNs. To render an image given an LFN, one computes the Plücker coordinates $\mathbf{r}_{u,v}$ of the camera rays at each u, v pixel coordinate in the image according to Equation 4. Specifically, given the extrinsic $\mathbf{E} = \left[\mathbf{R}|\mathbf{t}\right] \in SE(3)$ and intrinsic $\mathbf{K} \in \mathbb{R}^{3\times3}$ camera matrices [51] of a camera, one may retrieve the Plücker coordinates of the ray $\mathbf{r}_{u,v}$ at pixel coordinate u, v as:

$$\mathbf{r}_{u,v} = (\mathbf{d}_{u,v}, \mathbf{t} \times \mathbf{d}_{u,v})/\|\mathbf{d}_{u,v}\|, \text{ where } \mathbf{d}_{u,v} = \mathbf{R}^T(\mathbf{K}^{-1}\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} - \mathbf{t}) \tag{5}$$

Rendering then amounts to a *single* evaluation of the LFN Φ for each ray, $\mathbf{c}_{u,v} = \Phi(\mathbf{r}_{u,v})$. For notational convenience, we introduce a rendering function

$$\Theta^{\Phi}_{\mathbf{E},\mathbf{K}} : \mathbb{R}^\ell \to \mathbb{R}^{H\times W\times 3} \tag{6}$$

which renders an LFN Φ_ϕ with parameters $\phi \in \mathbb{R}^\ell$ when viewed from a camera with extrinsic and intrinsic parameters $(\mathbf{E}, \mathbf{K})$ into an image.

4.2 The geometry of Light Field Networks

We will now analyze the properties of LFNs representing Lambertian 3D scenes, and illustrate how the geometry of the underlying 3D scene is encoded. We will first derive an expression that establishes a relationship between LFNs and the classic two-plane parameterization of the light field.

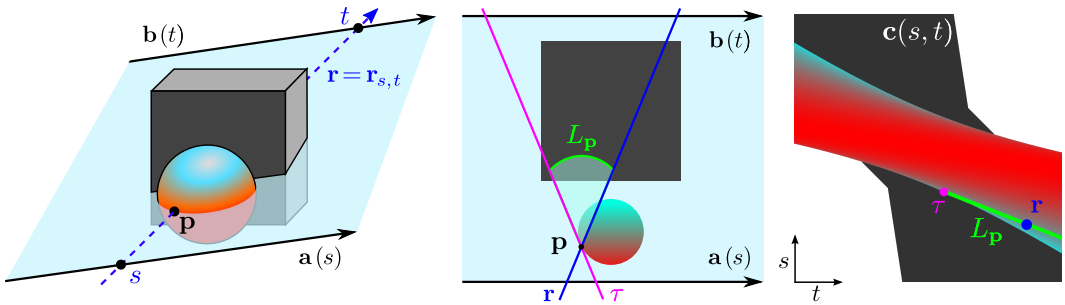


Figure 2: Given a 3D scene (left) and a light ray $\mathbf{r}$ (blue), we can slice the scene along a 2D plane containing the ray (light blue), yielding a 2D scene (center). The light field of all rays in a 2D plane, the *Epipolar Plane Image* (EPI) $\mathbf{c}(s,t)$ (right) can be analytically computed from our 360-degree LFN. See below for further discussion of EPI geometry.

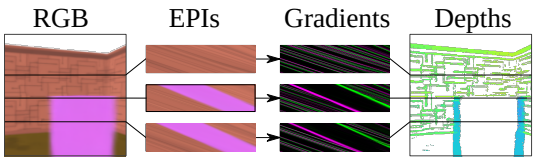
Subsequently, we will derive an expression for the depth of a ray in terms of the local color gradient of the light field, therefore allowing us to efficiently extract sparse depth maps from the light field at any camera pose via analytical differentiation of the neural implicit representation. Please see Figure 2 for an overview.

Locally linear slices of the light field. We derive here a local parametrization that will allow us to work with an LFN as if it were a conventional 2-plane light field. Given a ray $\mathbf{r}$ in Plücker coordinates, we pick two points $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^3$ along this ray. We then find a normalized direction $\mathbf{d} \in \mathbb{S}^2$ orthogonal to the ray direction. We may now parameterize two parallel lines $\mathbf{a}(s) = \mathbf{x} + s\mathbf{d}$ and $\mathbf{b}(t) = \mathbf{x}' + t\mathbf{d}$ that give rise to a local two-plane basis of the light field with ray coordinates s and t. $\mathbf{r}$ intersects these lines at the two-plane coordinates $(s,t) = (0,0)$. This choice of local basis now assigns the two-plane coordinates (s,t) to the ray $\mathbf{r}$ from $\mathbf{a}(s)$ to $\mathbf{b}(t)$. In Figure 2, we illustrate this process on a simple 2D scene.

Epipolar Plane Images and their geometry. Plücker coordinates enable us to extract a 2D slice from an LFN for any viewpoint of the light field by varying (s,t) and sampling Φ:

$$\mathbf{c}(s,t) = \Phi\left(\frac{\mathbf{b}-\mathbf{a}}{\|\mathbf{b}-\mathbf{a}\|}, \frac{\mathbf{a}\times\mathbf{b}}{\|\mathbf{b}-\mathbf{a}\|}\right). \tag{7}$$

The image of this 2D slice $\mathbf{c}(s,t)$ is well-known in the light field literature as an *Epipolar Plane Image* (EPI) [44]. EPIs carry rich information about the geometry of the underlying 3D scene. To see this, consider a point $\mathbf{p}$ on the surface of an object in the scene; please see Figure 2 for a diagram. A point $\mathbf{p} \in \mathbb{R}^2$ has a 1-dimensional family of rays going through the point, which correspond to a (green) line $L_\mathbf{p}$ in the EPI. In a Lambertian scene, all rays that meet in this point and that are not occluded by other objects must observe the same color. Therefore, the light field is constant along this line. As one travels along $L_\mathbf{p}$, rotating through the family of rays through $\mathbf{p}$, one eventually reaches a (magenta) *tangent ray* τ to the object. At a tangent ray, the value of the EPI ceases to be constant, and the light field changes its color to whatever is disoccluded by the object at this tangent ray. Because objects of different depth undergo differing amounts of parallax, the *slope* of the segment of $L_\mathbf{p}$ along which $\mathbf{c}$ is constant determines the 3D coordinates of $\mathbf{p}$. Finally, by observing that we may extract EPIs from *any* perspective, it is clear that an LFN encodes the full 3D geometry of the underlying scene. Intuitively, this may also be seen by considering that one could render out all possible perspectives of the underlying scene, and solve a classic multi-view stereo problem to retrieve the shape.



Extracting depth maps from LFNs. To extract 3D geometry from a LFN, we utilize the property of the 2-plane parameterization that the light field is constant on segments $L_\mathbf{p}$, the slopes of which determine $\mathbf{p}$. In the supplemental material, we derive that the distance d along $\ell = \overrightarrow{\mathbf{a}(s)\mathbf{b}(t)}$ from

$\mathbf{a}(s)$ to the point $\mathbf{p}$ on the object is

$$d(s,t) = D\frac{\partial_t \mathbf{c}(s,t)}{\partial_s \mathbf{c}(s,t) + \partial_t \mathbf{c}(s,t)}, \text{ and thus } \mathbf{p} = \mathbf{a}(s) + \mathbf{d}(s,t)\frac{\mathbf{b}(t) - \mathbf{a}(s)}{\|\mathbf{b}(t) - \mathbf{a}(s)\|} \tag{8}$$

where $\mathbf{a}(s)$ and $\mathbf{b}(t)$ are as above, $\mathbf{c}(s,t)$ is defined by (7) and D is the distance between the lines $\mathbf{a}(t)$ and $\mathbf{b}(t)$. This formula yields meaningful depth estimates wherever the derivatives of the light fields are nonzero along the ray. This occurs when $\mathbf{r}$ hits the object at a point where the surface color is changing, or when $\mathbf{r}$ is a tangent ray. We note that there is a wealth of prior art that could be used to extend this approach to extract *dense* depth maps [45–48].

4.3 Meta-learning Light Field Networks

We consider a dataset $\mathcal{D}$ consisting of N 3D scenes

$$S_i = \{(\mathbf{I}_j, \mathbf{E}_j, \mathbf{K}_j)\}_{j=1}^K \in \mathbb{R}^{H\times W\times 3} \times SE(3) \times \mathbb{R}^{3\times 3},\ i = 1\dots N$$

with K images $\mathbf{I}_j$ of each scene taken with cameras with extrinsic parameters $\mathbf{E}_j$ and intrinsic parameters $\mathbf{K}_j$ [51]. Each scene is completely described by the parameters $\phi_i \in \mathbb{R}^\ell$ of its corresponding light field MLP $\Phi_i = \Phi_{\phi_i}$.

Meta-learning and multi-view consistency. In the case of 3D-structured neural scene representations, ray-marching or volumetric rendering naturally ensure multi-view consistency of the reconstructed 3D scene representation. In contrast, a general 4D function $\Phi : \mathcal{L} \to \mathbb{R}^3$ is not multi-view consistent, as most such functions are not the light fields of any 3D scene. We propose to overcome this challenge by learning a prior over the space of light fields. As we will demonstrate, this prior can also be used to reconstruct an LFN from a single 2D image observation. In this paradigm, *differentiable ray-casting is a method to force the light field of a scene to be multi-view consistent*, while we instead impose multi-view consistency by learning a prior over light fields.

Meta-learning framework. We propose to represent each 3D scene S_i by its own latent vector $\mathbf{z}_i \in \mathbb{R}^k$. Generalizing to new scenes amounts to learning a prior over the space of light fields that is concentrated on the manifold of multi-view consistent light fields of natural scenes. To represent this latent manifold, we utilize a hypernetwork [52, 3]. The hypernetwork is a function, represented as an MLP

$$\Psi : \mathbb{R}^k \to \mathbb{R}^\ell, \Psi_\psi(\mathbf{z}_i) = \phi_i \tag{9}$$

with parameters ψ which sends the latent code $\mathbf{z}_i$ of the i-th scene to the parameters of the corresponding LFN.

Several reasonable approaches exist to obtain latent codes $\mathbf{z}_i$. One may leverage a convolutional- or transformer-based image encoder, directly inferring the latent from an image [11, 5], or utilize gradient-based meta-learning [22]. Here, we follow an auto-decoder framework [1, 3] to find the latent codes $\mathbf{z}_i$, but note that LFNs are in no way constrained to this approach. We assume that the latent vectors have a Gaussian prior with zero mean and a diagonal covariance matrix. At training time, we jointly optimize the latent parameters $\mathbf{z}_i$ together with the hypernetwork parameters ψ using the objective

$$\underset{\mathbf{z}_i,\psi}{\arg\min} \sum_i \sum_j \|\Theta^\Phi_{\mathbf{E}_j,\mathbf{K}_j}(\Psi_\psi(\mathbf{z}_i)) - \mathbf{I}_j)\|_2^2 + \lambda_{lat}\|\mathbf{z}_j\|_2^2. \tag{10}$$

Here the Θ^Φ is the rendering function (Equation 6), the first term is an ℓ_2 loss penalizing the light fields that disagree with the observed images, and the second term enforces the prior over the latent variables. We solve Equation 10 using gradient descent. At test time, we freeze the parameters of the hypernetwork and reconstruct the light field for a new scene S given a *single observation* of the scene $\{(\mathbf{I}, \mathbf{E}, \mathbf{K})\}$ by optimizing, using gradient descent, the latent variable $\mathbf{z}_S$ of the scene, such that the reconstructed light field $\Phi_{\Psi_\psi(\mathbf{z}_S)}$ best matches the given observation of the scene:

$$\mathbf{z}_S = \underset{\mathbf{z}}{\arg\min} \|\Theta^\Phi_{\mathbf{E},\mathbf{K}}(\Psi_\psi(\mathbf{z})) - \mathbf{I})\|_2^2 + \lambda_{lat}\|\mathbf{z}\|_2^2. \tag{11}$$

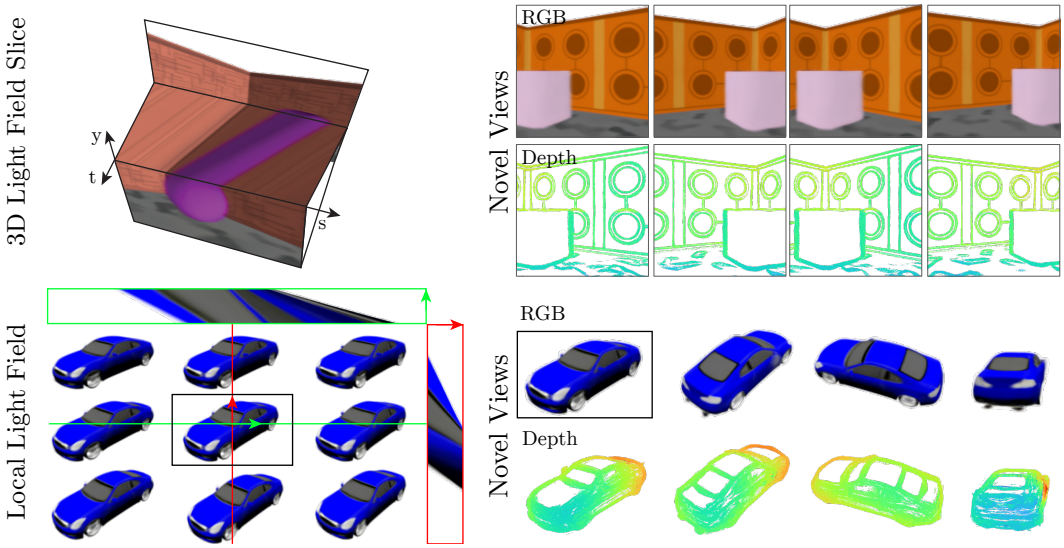


Figure 3: **360-degree light field parameterization.** Top left: 3D slice in the style of the Lumigraph [34] of an LFN of a room-scale scene. Bottom left: nearby views of a car as well as vertical (right, red) and horizontal (top, green) Epipolar Plane Images, sampled from an LFN. Reconstructed from training set, 50 and 15 views, respectively. Right: LFNs enable rendering from arbitrary, 360-degree camera perspectives as well as sparse depth map extraction from only a single sample per ray. Please see the supplemental video for more qualitative results.

Global vs. local conditioning The proposed meta-learning framework globally conditions an LFN on a single latent variable $\mathbf{z}$. Recent work instead leverages *local* conditioning, where a neural implicit representation is conditioned on local features extracted from a context image [24, 6, 25]. In particular, the recently proposed pixelNeRF [6] has achieved impressive results on few-shot novel view synthesis. As we will see, the current formulation of LFNs does *not* outperform pixelNeRF. We note, however, that local conditioning methods solve a *different problem*. Rather than learning a prior over classes of *objects*, local conditioning methods learn priors over *patches*, answering the question "How does this image patch look like from a different perspective?". As a result, this approach does not learn a latent space of neural scene representations. Rather, scene context is required to be available at test time to reason about the underlying 3D scene, and the representation is not *compact*: the size of the conditioning grows with the number of context observations. In contrast, globally conditioned methods [3, 11, 1, 2] first infer a *global* representation that is invariant to the number of context views and subsequently discard the observations. However, local conditioning enables better generalization due to the shift-equivariance of convolutional neural networks. An equivalent to local conditioning in light fields is non-obvious, and an exciting direction for future work.

5 Experiments

We demonstrate the efficacy of LFNs by reconstructing 360-degree light fields of a variety of simple 3D scenes. In all experiments, we parameterize LFNs via a 6-layer ReLU MLP, and the hypernetwork as a 3-layer ReLU MLP, both with layer normalization. We solve all optimization problems using the ADAM solver with a step size of 10^{-4}. Please find more results, precise implementation and dataset details in the supplemental document and video.

Reconstructing appearance and geometry of single-object and room-scale light fields. We demonstrate that LFN can parameterize 360-degree light fields of both single-object ShapeNet [53] objects and simple, room-scale environments. We train LFNs on the ShapeNet "cars" dataset with 50 observations per object from [3], as well as on simple room-scale environments as proposed in [13]. Subsequently, we evaluate the ability of LFNs to generate novel views of the underlying 3D scenes. Please see Figure 3 for qualitative results. LFNs succeed in parameterizing the 360-degree light field, enabling novel view synthesis at *real-time* frame-rates (see supplemental video). We further demonstrate that LFNs encode scene geometry by presenting Epipolar Plane Images and leveraging the relationship derived in Equation 8 to infer sparse depth maps. We highlight that both rendering

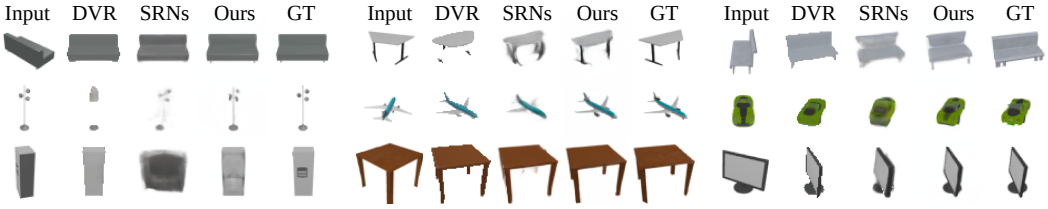


Figure 4: **Category-agnostic single-shot reconstruction.** We train a single model on the 13 largest Shapenet classes. LFNs consistently outperform global conditioning baselines. Baseline results courtesy of the pixelNeRF [6] authors. Please see the supplemental video for more qualitative results.

Table 1: **Single-shot multi-class reconstruction results.** We benchmark LFNs with SRNs [3] and DVR [5] on the task of single-shot multi-class reconstruction of the 13 largest ShapeNet [53] classes. LFNs significantly outperform DVR and SRNs on almost all classes, on average by more than 1dB, while only requiring a *single* network evaluation per ray. Note that DVR requires additional supervision in the form of foreground-background masks.

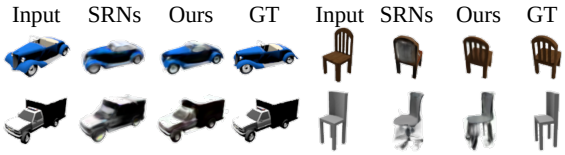
		plane	bench	cbnt.	car	chair	disp.	lamp	spkr.	rifle	sofa	table	phone	boat	mean
↑ PSNR	DVR [5]	25.29	22.64	24.47	23.95	19.91	**20.86**	23.27	20.78	23.44	23.35	21.53	**24.18**	25.09	22.70
	SRN [3]	26.62	22.20	23.42	24.40	21.85	19.07	22.17	21.04	24.95	23.65	22.45	20.87	25.86	23.28
	LFN	**29.95**	**23.21**	**25.91**	**28.04**	**22.94**	20.64	**24.56**	**22.54**	**27.50**	**25.15**	**24.58**	22.21	**27.16**	**24.95**
↑ SSIM	DVR [5]	0.905	**0.866**	**0.877**	0.909	0.787	**0.814**	**0.849**	0.798	0.916	0.868	0.840	0.892	**0.902**	0.860
	SRN [3]	0.901	0.837	0.831	0.897	0.814	0.744	0.801	0.779	0.913	0.851	0.828	0.811	0.898	0.849
	LFN	**0.932**	0.855	0.871	**0.943**	**0.835**	0.786	0.844	**0.808**	**0.939**	**0.874**	**0.868**	0.844	**0.907**	**0.870**

and depth map extraction *do not require ray-casting*, with only a single evaluation of the network or the network and its gradient respectively.

Multi-class single-view reconstruction. Following [5, 6], we benchmark LFNs with recent *global conditioning methods* on the task of single-view reconstruction and novel view synthesis of the 13 largest ShapeNet categories. We follow the same evaluation protocol as [54] and train a single model across all categories. See Figure 4 for qualitative and Table 1 for quantitative baseline comparisons. We significantly outperform both Differentiable Volumetric Rendering (DVR) [5] and Scene Representation Networks (SRNs) [3] on all but two classes by an average of 1dB, while requiring more than an order of magnitude fewer network evaluations per ray. Qualitatively, we find that the reconstructions from LFNs are often crisper than those of either Scene Representation Networks or DVR. Note that DVR requires additional ground-truth foreground-background segmentation masks.

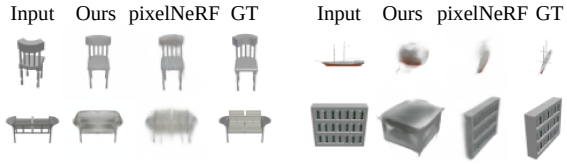
Class-specific single-view reconstruction. We benchmark LFNs on single-shot reconstruction on the Shapenet "cars" and "chairs" classes as proposed in SRNs [3]. See Figure 4 for qualitative and quantitative results. We strictly outperform SRNs on the "cars" class, while performing worse in PSNR but better in terms of SSIM on the "chairs" class, while requiring an order of magnitude fewer network evaluations and rendering in real-time. We attribute the drop in performance compared to multi-class reconstruction to the smaller dataset size, causing multi-view inconsistency.

Global vs. local conditioning and comparison to pixelNeRF [6]. We investigate the role of global conditioning, where a single latent is inferred to describe the whole scene [3], to local conditioning, where latents are inferred *per-pixel* in a 2D image and leveraged to locally condition a neural implicit representation [24, 25, 6]. We benchmark with the recently proposed pixelNeRF [6]. As



	Chairs	Cars
SRNs [3]	**22.89** / 0.89	22.25 / 0.89
LFN	22.26 / **0.90**	**22.42 / 0.89**

Figure 5: **Class-specific single-shot reconstruction.** LFN performs approximately on par with SRNs [3] in the single-class single-shot reconstruction case, while requiring an order of magnitude fewer network evaluations, memory, and rendering time. Quantitative results report (PSNR, SSIM).



	Class-specific	Multi-Class
LFN	22.34 / 0.90	24.95 / 0.87
pixelNeRF	**23.45 / 0.91**	**26.80 / 0.91**

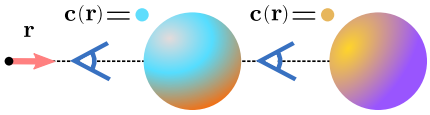
Figure 6: **Global vs. local conditioning.** The locally conditioned pixelNeRF outperforms LFNs in single-shot reconstruction, though LFNs require *three orders of magnitude less rendering time and memory*. Qualitatively, for many objects LFNs are on par with pixelNERF (left), but confuse the object class on others (right). Quantitative results report (PSNR, SSIM).

Table 2: **Comparison of rendering complexity.** LFNs require *three orders of magnitude* less compute than volumetric rendering based approaches, and admit real-time rendering.

	LFNs	SRNs [3]	pixelNeRF [6]
# evaluations per ray	1	11	192
clock time for 256×256 image (ms)	2.1	120	$30e3$

noted above (see Section 4.3), local conditioning does *not* infer a compact neural scene representation of the scene. Nevertheless, we provide the comparison here for completeness. See Figure 6 for qualitative and quantitative results. On average, LFNs perform 1dB worse than pixelNeRF in the single-class case, and 2dB worse in the multi-class setting.

Real-time rendering and storage cost. See Table 2 for a quantitative comparison of the rendering complexity of LFN compared with that of volumetric and ray-marching based neural renderers [3, 42, 19, 4, 6]. All clock times were collected for rendering 256×256 images on an NVIDIA RTX 6000 GPU. We further compare the cost of storing a single LFN with the cost of storing a conventional light field. With approximately 400k parameters, a single LFN requires around 1.6 MB of storage, compared to 146 MB required for storing a 360-degree light field at a resolution of $256 \times 256 \times 17 \times 17$ in the six-plane Lumigraph configuration.



Limitations. First, as every existing light field approach, LFNs store only one color per oriented ray, which makes rendering views from cameras placed in between occluding objects more difficult. Second, though we outperform globally-conditioned methods, we currently do not outperform the locally conditioned pixelNeRF. Finally, as opposed to 3D-structured representations, LFNs do not enforce strict multi-view consistency, and may be inconsistent in the case of small datasets.

6 Discussion and Conclusion

We have proposed Light Field Networks, a novel neural scene representation that directly parameterizes the full 360-degree, 4D light field of a 3D scene via a neural implicit representation. This enables both real-time neural rendering with a *single* evaluation of the neural scene representation per ray, as well as sparse depth map extraction without ray-casting. Light Field Networks outperform globally conditioned baselines in single-shot novel view synthesis, while being three orders of magnitude faster and less memory-intensive than current volumetric rendering approaches. Exciting avenues for future work include combining LFNs with local conditioning, which would enable stronger out-of-distribution generalization, studying non-Lambertian scenes, and enabling camera placement in obstructed 3D space. With this work, we make important contributions to the emerging fields of neural rendering and neural scene representations, with exciting applications across computer vision, computer graphics, and robotics.

Societal Impacts. Potential improvements extending our work on few-observation novel view synthesis could enable abuse by decreasing the cost of non-consensual impersonations. We refer the reader to a recent review of neural rendering [21] for an in-depth discussion of this topic.

References

[1] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proc. CVPR*, 2019.

[2] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proc. CVPR*, 2019.

[3] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Proc. NeurIPS 2019*, 2019.

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

[5] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. CVPR*, 2020.

[6] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. *Proc. CVPR*, 2020.

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

[8] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. *arXiv preprint arXiv:2103.15875*, 2021.

[9] A. Kohli, V. Sitzmann, and G. Wetzstein. Semantic Implicit Neural Scene Representations with Semi-supervised Training. In *Proc. 3DV*, 2020.

[10] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. *arXiv preprint arXiv:2103.12352*, 2021.

[11] Adam R Kosiorek, Heiko Strathmann, Daniel Zoran, Pol Moreno, Rosalia Schneider, Soňa Mokrá, and Danilo J Rezende. Nerf-vae: A geometry aware 3d scene generative model. *arXiv preprint arXiv:2104.00587*, 2021.

[12] Daniel E Worrall, Stephan J Garbin, Daniyar Turmukhambetov, and Gabriel J Brostow. Interpretable transformations with encoder-decoder networks. In *Proc. ICCV*, volume 4, 2017.

[13] SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[14] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proc. CVPR*, 2019.

[15] Hsiao-Yu Fish Tung, Ricson Cheng, and Katerina Fragkiadaki. Learning spatial common sense with geometry-aware recurrent networks. *Proc. CVPR*, 2019.

[16] Thu H Nguyen-Phuoc, Chuan Li, Stephen Balaban, and Yongliang Yang. Rendernet: A deep convolutional network for differentiable rendering from 3d shapes. In *Proc. NIPS*. 2018.

[17] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Josh Tenenbaum, and Bill Freeman. Visual object networks: image generation with disentangled 3d representations. In *Proc. NIPS*, pages 118–129, 2018.

[18] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proc. ICCV*, 2019.

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

[20] Danilo Jimenez Rezende, S. M. Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. In *Proc. NIPS*. 2016.

[21] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. In *Computer Graphics Forum*, volume 39, pages 701–727. Wiley Online Library, 2020.

[22] Vincent Sitzmann, Eric R Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *Proc. NeurIPS*, 2020.

[23] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020.

[24] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proc. ICCV*, pages 2304–2314, 2019.

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

[26] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. *arXiv preprint arXiv:2103.13744*, 2021.

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

[28] David B Lindell, Julien NP Martel, and Gordon Wetzstein. Autoint: Automatic integration for fast neural volume rendering. *Proc. CVPR*, 2020.

[29] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Chakravarty R Alla Chaitanya, Anton Kaplanyan, and Markus Steinberger. Donerf: Towards real-time rendering of neural radiance fields using depth oracle networks. *arXiv e-prints*, pages arXiv–2103, 2021.

[30] Petr Kellnhofer, Lars Jebe, Andrew Jones, Ryan Spicer, Kari Pulli, and Gordon Wetzstein. Neural lumigraph rendering. *Proc. CVPR*, 2021.

[31] Edward H Adelson, James R Bergen, et al. *The plenoptic function and the elements of early vision*, volume 2. Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, 1991.

[32] Edward H Adelson and John YA Wang. Single lens stereo with a plenoptic camera. *IEEE transactions on pattern analysis and machine intelligence*, 14(2):99–106, 1992.

[33] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proc. SIGGRAPH*, 1996.

[34] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proc. SIGGRAPH*, 1996.

[35] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432, 2001.

[36] Ren Ng et al. *Digital light field photography*, volume 7. stanford university Stanford, 2006.

[37] Suren Vagharshakyan, Robert Bregovic, and Atanas Gotchev. Light field reconstruction using shearlet transform. *Proc. PAMI*, 40(1):133–147, 2017.

[38] Lixin Shi, Haitham Hassanieh, Abe Davis, Dina Katabi, and Fredo Durand. Light field reconstruction using sparsity in the continuous fourier domain. *ACM Transactions on Graphics (TOG)*, 34(1):1–13, 2014.

[39] Anat Levin and Fredo Durand. Linear view synthesis using a dimensionality gap light field prior. In *Proc. CVPR*, pages 1831–1838. IEEE, 2010.

[40] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Trans. Graph. (SIGGRAPH Asia)*, 35(6):193, 2016.

[41] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light- and time-image interpolation. *Proc. SIGGRAPH Asia 2020*, 39(6), 2020.

[42] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Proc. NeurIPS*, 2020.

[43] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomír Mech, and Ulrich Neumann. DISN: deep implicit surface network for high-quality single-view 3d reconstruction. In *Proc. NIPS*, 2019.

[44] Robert C Bolles, H Harlyn Baker, and David H Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *Proc. IJCV*, 1(1):7–55, 1987.

[45] Don Dansereau and Len Bruton. Gradient-based depth estimation from 4d light fields. In *Proc. IEEE International Symposium on Circuits and Systems*, volume 3, pages III–549. IEEE, 2004.

[46] Ivana Tosic and Kathrin Berkner. Light field scale-depth space transform for dense depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 435–442, 2014.

[47] Changil Kim, Henning Zimmer, Yael Pritch, Alexander Sorkine-Hornung, and Markus H Gross. Scene reconstruction from high spatio-angular resolution light fields. *ACM Transactions on Graphics (TOG)*, 32(4):73–1, 2013.

[48] Lipeng Si and Qing Wang. Dense depth-map estimation and geometry inference from light fields via global optimization. In *Asian Conference on Computer Vision*, pages 83–98. Springer, 2016.

[49] Yan-Bin Jia. Plücker coordinates for lines in the space. *Problem Solver Techniques for Applied Computer Science, Com-S-477/577 Course Handout. Iowa State University. http://web. cs. iastate.edu/~ cs577/handouts/plucker-coordinates.pdf*, 2020.

[50] Insung Ihm, Sanghoon Park, and Rae Kyoung Lee. Rendering of spherical light fields. In *Proceedings The Fifth Pacific Conference on Computer Graphics and Applications*, pages 59–68, 1997.

[51] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2003.

[52] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In *Proc. ICLR*, 2017.

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

[54] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proc. CVPR*, pages 3907–3916, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code and dataset will be made available upon publication.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] In the supplemental material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Due to the high computational cost, this is unfortunately infeasible.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] In the supplemental material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] All datasets are public.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] Our data does not contain anyone's personal information.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]