Neural Implicit Relightable Participating Media

Anonymous Author(s)
Affiliation
Address
email

Abstract

Learning neural radiance fields of a scene has recently allowed realistic novel view synthesis of the scene, but they are limited to synthesize images under the original fixed lighting condition. Therefore, they are not flexible for the eagerly desired tasks like relighting, scene editing and scene composition. To tackle this problem, several recent methods propose to disentangle reflectance and illumination from the radiance field. These methods can cope with solid objects with opaque surfaces but participating media are neglected. Also, they take into account only direct illumination or at most one-bounce indirect illumination, thus suffer from energy loss due to ignoring the high-order indirect illumination. We propose to learn implicit neural representations for participating media with a complete simulation of global illumination. We estimate direct illumination via ray tracing and compute indirect illumination with spherical harmonics. Our approach avoids computing the lengthy indirect bounces and does not suffer from energy loss. Our experiments on multiple scenes show that our approach achieves superior visual quality and numerical performance compared to state-of-the-art methods, and it can generalize to deal with solid objects with opaque surfaces as well.

1 Introduction

From natural phenomenons like fog and cloud to ornaments like jade artworks or wax figures, participating media objects are pervasive in both real life and virtual content like movies or games. Inferring the bounding geometry and scattering properties of participating media objects from observed images is a long-standing problem in both computer vision and graphics. Traditional methods addressed the problem by exploiting specialized structured lighting patterns [1, 2, 3] or using discrete representations [4]. These methods, however, require the bounding geometry of participating media objects to be known and explicitly reconstruct the volume density.

Recent neural rendering techniques have achieved remarkable progress on this problem by learning neural scene representations [5, 6, 7] and are able to optimize the representations with the assistance of a differentiable ray marching process. However, these methods are mostly designed for novel view synthesis and have baked in materials and lighting into the radiance fields or surface color. Therefore, they can hardly support downstream tasks like relighting and scene editing. Recent work [8, 9] has taken initial steps to disentangle the lighting and materials from radiance. For material, their methods are primarily designed for solid objects with opaque surfaces, thus they assume an underlying surface at each point with a normal and a BRDF. The assumed prior, however, does not apply to non-opaque participating media which has no surfaces inside. For lighting, neural reflectance field [8] simulates direct illumination from a single point light, whereas NeRV [9] handles direct illumination and one-bounce indirect illumination. They generally suffer from the energy loss issue due to ignoring the high-order indirect illumination. However, indirect lighting from multiple scattering plays a significant role in the final appearance [10] of participating media objects.

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

In this paper, we propose a novel neural implicit representation for learning relightable participating media objects. Our method takes as input a set of posed images with varying but known lighting and uses NeRF-like neural networks to learn a disentangled representation for the participating media with volume density and scattering albedo. To synthesize images, we embed a differentiable physically-based ray marching process in the framework. In addition, we propose to handle global illumination by embedding the single scattering and multiple scattering estimation into the ray marching process, where single scattering is simulated by Monte Carlo ray tracing and the incident radiance from multiple scattering is approximated by spherical harmonics (SH). Without supervising the training with ground-truth lighting decomposition, our method is able to learn a decomposition of direct lighting and indirect lighting in an unsupervised manner. Our comprehensive experiments demonstrate that our method achieves better visual quality and higher numerical performance compared to state-of-the-art methods. Meanwhile, our method can generalize to handle solid objects with opaque surfaces. We also demonstrate that our learned neural representations of participating media allow relighting, scene editing and inserting the learned representations into another virtual environment.

To summarize, our approach has the following contributions:

1. We propose a novel method to learn relightable participating media objects from posed images and this method is able to generalize to solid objects.
2. Our method learns a disentangled neural representation for participating media, which separates out volume density, scattering properties and lighting.
3. Our method deals with both single scattering and multiple scattering and enables the unsupervised decomposition of direction illumination and indirect illumination.
4. We demonstrate the learned representation of participating media are flexible for relighting, scene editing and scene compositions.

2 Related Work

Neural scene representations. Recent studies on *neural rendering* [11, 12] have shown promising progress on synthesizing realistic images, where neural scene representations are important building blocks for the recent advances. Neural scene representations aim to infer the 3D context from images rather than learning an embedding manifold from 2D images [13]. Classic explicit 3D representations, such as voxels [14, 15, 6], multiplane images [16, 17] and proxy geometry [18] are exploited to combine with neural networks to learn scene representations. These explicit representations generally suffer from the intrinsic resolution limitation. To sidestep the limitation, most approaches shift towards implicit representations, like signed distance fields [19, 20], volumetric occupancy fields [21, 22, 23, 24], or coordinate-based neural networks [25, 7, 26, 5]. By embedding a differentiable rendering process like ray marching [6, 5], or sphere tracing [7, 27] into these implicit representations. These methods are capable of optimizing the scene representations from observed images and synthesizing novel views after training. While they generally show progress in quality compared to interpolation based novel view synthesis methods [28, 29, 30], the learned representations are usually texture colors and radiance, without separating lighting and materials. By contrast, we propose to learn an neural implicit representation with disentangled volume density, scattering properties and lighting, which allow the usages for tasks like scene editing and relighting.

Volume geometry and properties capture. Acquiring geometry and scattering properties of participating media has long been the interest of the community. Early methods utilize sophisticated scanning and recording devices [31, 2] and specialized lighting patterns [3, 1] to capture volume density. Computational imaging methods [32, 33] frame the inference of scattering properties from images as an inverse problem, but they assume that the geometries of objects are known. Based on the inverse rendering fashion [34], the inverse transport method [35] incorporates a differentiable light transport [36] module within an analysis-by-synthesis pipeline to infer scattering properties, but it aims for known geometry with homogeneous media. In contrast, our method learns the geometry and scattering properties of media objects simultaneously and our method can deal with both homogeneous and heterogeneous media.

Relighting. Neural Radiance Field (NeRF) [5] and its later enhancement [37, 38] encode a scene's geometry and radiance into MLPs and leverage ray marching to synthesize new views. While

they achieve high quality in synthesized new images, they are limited to synthesize views under the same fixed lighting conditions as training data. To remove the limitation, appearance latent code [39, 40] can be learned to condition on the view synthesis, without considering the underlying light transport. By utilizing the physics of light transport, recent approaches [8, 9, 41] employ explicit decomposition of materials and lighting by assuming an opaque surface at each point in the space, but this is not appropriate for participating media which have no surfaces inside. Instead, our method models participating media as an implicit field of particles that scatter and absorb light, which are in accordance with the nature of participating media. Neural reflectance field [42, 8] requires the collocated camera-light setting during training and they simulate only direction illumination. NeRV [9] and OSF [43] consider indirect illumination, but they simulate merely one-bounce indirect light transport because of the prohibitive computation cost for long paths. However, ignoring the high-order indirect illumination leads to the potential problem of energy loss. By contrast, we use Monte Carlo ray tracing to compute direct lighting and propose to learn a spherical harmonic field for estimating the complete indirect lighting. The recent PlenOctree [44] uses spherical harmonic to represent the outgoing radiance field like NeRF, but it does not allow relighting. With all illumination components properly estimated, our method enables a principled disentanglement of volume density, scattering properties, and lighting in an unsupervised way.

3 Background

Volume rendering. The radiance carried by a ray after its interaction with participating media can be computed based on the radiative transfer equation [45]

$$L_o\left(\boldsymbol{r}_0, \boldsymbol{r}_d\right) = \int_0^{\infty} \tau\left(\boldsymbol{r}(t)\right) \sigma\left(\boldsymbol{r}(t)\right) L\left(\boldsymbol{r}(t), -\boldsymbol{r}_d\right) dt. \tag{1}$$

Here, $\boldsymbol{r}$ is a ray starting from $\boldsymbol{r}_0$ along the direction $\boldsymbol{r}_d$ and $\boldsymbol{r}(t) = \boldsymbol{r}_o + t \cdot r_d$ denotes a point along the ray at the parametric distance[1] t. L_o is the received radiance at $\boldsymbol{r}_0$ along $\boldsymbol{r}_d$. σ denotes the extinction coefficient that is also referred to as volume density. The $\tau\left(\boldsymbol{r}(t)\right)$ is the transmittance between $\boldsymbol{r}(t)$ and $\boldsymbol{r}_0$ and it can be computed by $\exp\left(-\int_0^t \sigma(\boldsymbol{r}(s))ds\right)$. The $L(\cdot)$ inside the integral stands for the in-scattered radiance towards $\boldsymbol{r}_0$ along $-\boldsymbol{r}_d$. NeRF [5] models the in-scattered radiance L (Eq. 1) as a view dependent color c, but it glosses the underlying scattering and incident illumination. The learned radiance field of NeRF bakes in the lighting and materials, so it allows merely view synthesis under the same fixed lighting, without the supports for relighting.

Ray marching. The integral in Equation 1 can be solved with the numerical integration method, *ray marching* [46, 47]. This is generally done by casting rays into the volume and taking point samples along each ray to collect volume density and color values [6, 5]. The predicted color of a ray is computed by $L_o\left(\boldsymbol{r}_0, -\boldsymbol{r}_d\right) = \sum_j \tau_j \cdot \alpha_j \cdot L\left(\boldsymbol{r}(t_j), -\boldsymbol{r}_d\right)$, where $\alpha_j = 1 - \exp\left(-\sigma_j \cdot \delta_j\right)$, $\delta_j = \|t_{j+1} - t_j\|$ and $\tau_j = \prod_{i=1}^{j-1}(1 - \alpha_i)$.

4 Neural Relightable Participating Media

In this work, we aim to learn neural representations of participating media with disentangled volume density and scattering properties. We model the participating media as a field of particles that absorb and scatter light-carrying rays. Below we firstly describe our disentangled neural representation based on a decomposition of single scattering and multiple scattering. Then, we depict our neural network design, followed by details of a volume rendering process for synthesizing images and the details of training.

4.1 Lighting Decomposed Formulation

We firstly write the in-scatted radiance $L\left(\boldsymbol{r}(t), -\boldsymbol{r}_d\right)$ in Equation 1 as an integral of light-carrying spherical directions over a 4π steradian range

$$L\left(\boldsymbol{r}(t), -\boldsymbol{r}_d\right) = \int_{\Omega_{4\pi}} S\left(\boldsymbol{r}(t), -\boldsymbol{r}_d, \omega_i\right) L_{in}(\boldsymbol{r}(t), \omega_i) d\omega_i, \tag{2}$$

[1]While the integration accounts for a t going to ∞, t covers only the range with participating media in practice.

where L_{in} is the incident radiance at $\boldsymbol{r}(t)$ from direction ω_i. $S(\cdot)$ is a scattering function which determines the portion of lighting that is deflected towards the $-\boldsymbol{r}_d$. Previous methods [8, 9, 41] assume a surface prior with a normal at every point and account for 2π hemispherical incident directions. Accordingly, they define the scattering function S as a BRDF function. This assumption, however, can hardly match the participating media objects which have no internal surfaces and normals also do no apply. By contrast, we deal with light-carrying directions over the full 4π steradian range Specifically, we define $S = a(\boldsymbol{r}(t)) \cdot \rho(-\boldsymbol{r}_d, \omega_i)$ to account for the scattering over the full spherical directions. Here, $a(\boldsymbol{r}(t))$ is the scattering albedo and ρ is the phase function that decides the scattering directionality. We use the Henyey-Greenstein (HG) phase function [48]. Then, we can rewrite the radiance of ray $\boldsymbol{r}$ to get the disentangled form:

$$L_o\left(\boldsymbol{r}_0, \boldsymbol{r}_d\right) = \int_0^{\infty} \tau\left(\boldsymbol{r}(t)\right) \sigma\left(\boldsymbol{r}(t)\right) \int_{\Omega_{4\pi}} a(\boldsymbol{r}(t)) \rho(-\boldsymbol{r}_d, \omega_i) L_{in}(\boldsymbol{r}(t), \omega_i) d\omega_i dt, \tag{3}$$

from which the volume density σ implicitly decides the geometry, and the albedo a along with the phase function ρ control the scattering of rays. We use MLPs to represent the $\sigma\left(\boldsymbol{r}(t)\right)$ and $a\left(\boldsymbol{r}(t)\right)$.

To compute L_o, we additionally need to estimate L_{in} (Eq. 3). This can be conducted by recursively substituting the L_o into L_{in} and expressing the radiance for pixel k as an integral over all paths $P_k = \int_{\Psi} g_k(\bar{x}) d\mu(\bar{x})$, where $\bar{x}$ is a light-carrying path, g_k denotes a measurement contribution function, μ is the measure for the path space, and Ψ is the space of paths with all possible lengths. P_k is then computed by a summation of the contributions of all paths. The contribution of a path with length i can be written as

$$P_{k,i} = \int_{l_1} \int_{\Omega} \cdots \int_{l_{i-1}} \int_{\Omega} L_e\left(x(t_{i-1})\right) V\left(x(t_{i-1}), \omega_{i-1}\right) \mathcal{T}(P_{k,i}) dt_1 d\omega_1 \cdots dt_{i-1} d\omega_{i-1}, \tag{4}$$

where $x(t_{i-1})$ is a point along the $(i-1)$-th ray segment with the length l_{i-1}, ω_{i-1} is the scattering direction in the space Ω and $\omega_0 = \boldsymbol{r}_d$, L_e denotes the emitted radiance towards $x(t_{i-1})$ from a light source, V accounts for the transmittance between $x(t_{i-1})$ and the light source, $\mathcal{T}(P_{k,i}) = \prod_{j=1}^{i-1} V(x(t_j), \omega_j) \sigma(x(t_j)) \cdot \prod_{j=1}^{i-1} \rho(-\omega_{j-1}, \omega_j) a(x(t_j))$ is called the path throughput. Volumetric path tracing [49] utilizes Monte Carlo method to estimate the integral but its computational cost increases quickly when using high sampling rates of paths with many bounces. To reduce the cost, NeRV [9] truncates the path space and only considers up to one indirect bounce, namely $i = 3$ in Eq. 4. This, however, leads to energy loss since high-order indirect illumination are neglected.

Instead of extending the integral with many long paths, we propose to decompose the outgoing radiance L in Eq. 1 as $L = L_s + L_m$, where L_s is the single-scattering contribution and L_m denotes the multiple-scattering contribution. Therefore, L_o can be split into two integrals: $L_{o,s} = \int_0^{\infty} \tau\left(\boldsymbol{r}(t)\right) \sigma\left(\boldsymbol{r}(t)\right) L_s dt$ and $L_{o,m} = \int_0^{\infty} \tau\left(\boldsymbol{r}(t)\right) \sigma\left(\boldsymbol{r}(t)\right) L_m dt$, that can be evaluated separately.

Single scattering. To compute the $L_{o,s}$, we need to evaluate the following integral at each sample point $\boldsymbol{r}(t)$ along the camera ray $\boldsymbol{r}$

$$L_s = \int_{\Omega_{4\pi}} a(\boldsymbol{r}(t)) \rho(-\boldsymbol{r}_d, \omega_i) L_e(\boldsymbol{r}(t), \omega_i) V(\boldsymbol{r}(t), \omega_i) d\omega_i. \tag{5}$$

Here, L_e is the emitted radiance from a light source to the point $\boldsymbol{r}(t)$ and V is the transmittance between $\boldsymbol{r}(t)$ and a light source. The transmittance can be computed with another integral of volume density as described in Section 3, but computing the integral for all points leads to high computation cost during training and inference. Therefore, we train a visibility neural network to regress the transmittance value as done in [9].

Multiple scattering. For $L_{o,m}$, we evaluate the $L_m = \int_{\Omega_{4\pi}} a(\boldsymbol{r}(t)) \rho(-\boldsymbol{r}_d, \omega_i) L_{in}(\boldsymbol{r}(t), \omega_i) d\omega_i$. L_m aggregates the incoming radiance of rays that have been scattered at least once in media. Since the distribution of incident radiance from multiple scattering is generally smooth, we propose to represent the incident radiance L_{in} as a spherical harmonics expansion: $L_{in}(\omega_i) = \sum_{l=0}^{l_{max}} \sum_{m=-l}^{l} c_l^m Y_l^m(\omega_i)$, where l_{max} is the maximum spherical harmonic *band*, $c_l^m \in \mathbb{R}^3$ are spherical harmonic coefficients for the RGB spectrum and Y_l^m are spherical harmonic basis functions. Therefore, we can compute the multiple-scattering contribution with

$$L_m = \int_{\Omega_{4\pi}} a(\boldsymbol{r}(t)) \rho(-\boldsymbol{r}_d, \omega_i) \cdot \left(\sum_{l=0}^{l_{max}} \sum_{m=-l}^{l} c_l^m Y_l^m(\omega_i) \right) d\omega_i. \tag{6}$$

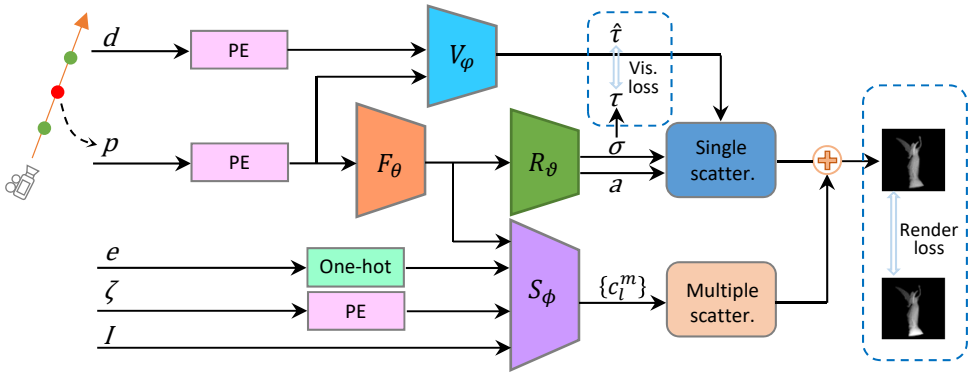


Figure 2: Our overall architecture for learning neural participating media. "PE" denotes the positional encoding and "one-hot" denotes the one-hot encoding. The rendering loss and the visibility loss correspond to the summands in Eq. 9.

We employ an MLP to learn these spherical harmonic coefficients. By using spherical harmonics for the incident radiance from multiple scattering, we sidestep the costly evaluation of the path integral (Eq. 4). Also, our formulation does not truncate the path space, so it does not suffer from energy loss. Since we represent the incident radiance at the primary rays, we also sidestep the explosion of rays. Figure 1 visualizes the explosion of rays when computing indirect illumination under an environment lighting. Multiple shadow rays are needed to account for the directional emission from the light source. The brute-force ray splitting approach leads to explosion of rays and is impractical. NeRV reduces the ray count by tracing up to one indirect bounce, but it has a complexity of $\mathcal{O}(M \cdot N)$, where M is the number of first indirect bounces (red) and N is the number of shadow rays (blue). Our method uses spherical harmonics to handle indirect illumination as a whole and its complexity is $\mathcal{O}(k \cdot M)$, where k is the sampling rates along the primary rays.

Network architectures. Figure 2 presents our overall architecture. Our neural networks are based on the coordinate-based MLP fashion, and we use frequency-based positional encoding [5, 50] E to map an input coordinate p to a higher dimensional vector $E(p)$ before sending it to the neural networks. Specifically, we use the MLP R_ϑ to predict volume density σ (1D) and the scattering albedo a (3D). Meanwhile, we employ the MLP S_ϕ to learn the spherical harmonic coefficients c_l^m. Here, S_ϕ is conditioned on the point light location ζ, the point light intensity I, and the 2-bit one-hot code e which indicates the existence of environment lighting. Since both the density-albedo network R_ϑ and the SH network S_ϕ takes as input the 3D coordinate, we introduce a feature MLP F_θ to predict shared features for downstream MLPs. To get the visibility for shadow rays, we train another MLP V_φ, which takes in the ray direction d in addition to the coordinate p, to learn a visibility fields for the estimation of single scattering. In summary, we have

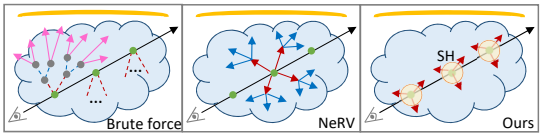


Figure 1: Visualizations of path bounces.

$$R_\vartheta : F_\theta(E(p)) \to (\sigma, a) \quad S_\phi : (e, \zeta, I, F_\theta(E(p))) \to \{c_l^m\} \quad V_\varphi : (\boldsymbol{r}(t), E(d)) \to \tau. \tag{7}$$

Note that the HG phase function (Eq. 3) has an *asymmetry parameter* g. It is specific to a participating media, so we do not predict it with coordinate-based MLPs. Instead, we directly optimize it during the training process.

4.2 Volume Rendering

Based on the above decomposition, we employ *ray marching* (Sec. 3) to numerically estimate $L_{o,s}$ and $L_{o,m}$. Hence, the final radiance L_o of the camera ray $\boldsymbol{r}$ in Eq. 1 can be computed as:

$$L_o(\boldsymbol{r}) = \Sigma_{j=1}^{N} \tau(\boldsymbol{r}(t)) \left(1 - \exp\left(-\sigma(\boldsymbol{r}(t)) \cdot \delta t_j\right)\right) (L_s + L_m), \tag{8}$$

where we stratified sample $N = 64$ query points along the ray $\boldsymbol{r}$ and $\delta t_j = \|\boldsymbol{r}(t_{j+1}) - \boldsymbol{r}(t_j)\|$ is the step size. For a point sample, we can query the MLPs to obtain its volume density σ, albedo a and SH coefficients $\{c_l^m\}$ for computing single scattering and multiple scattering.

We compute single scattering at a point based on Eq. 5. We shoot shadow rays towards light sources to get the emitted radiance L_e according to the light types. For environment lighting, we sample 64 directions stratified over a sphere around the point to obtain incident radiance. For a point light, we directly connect the query point to it. To account for the attenuation of light radiance, we query V_φ to get the visibility to the light source from the query point.

For multiple scattering, we uniformly sample $K = 64$ random incident directions over the sphere around each query point, evaluate the incident radiance L_{in} along each direction using the learned spherical harmonic field, and estimate the integral in Eq. 6 with a Monte Carlo integration $L_m = 1/K \sum_{i=1}^{K} a\rho(\omega_i)L_{in}(\omega_i)$. For brevity, we omit the $\boldsymbol{r}$ notation. Note that the visibility towards the light source is not needed in the computation.

4.3 End-to-end Learning

Based on our fully differentiable pipeline, we can end-to-end train a neural implicit representation for each scene. The learning requires a set of posed RGB images and their lighting conditions. During each training iteration, we trace primary rays through the media. Along each primary ray, we estimate single scattering using shadow rays and compute multiple scattering contribution via the learned spherical harmonic field as described in Sec. 4.2. We optimize the parameters of F_θ, R_ϑ, and S_ϕ by minimizing a rendering loss between the predicted radiance $L_o(\boldsymbol{r})$ from *ray marching* and the radiance $\hat{L}_o(\boldsymbol{r})$ from input images. To train the visibility network V_φ, we use the transmittance $\hat{V}_\vartheta$ computed from the learned volume density as the ground truth and minimize the visibility loss between the prediction V_φ and the ground truth. Therefore, our loss function includes two parts:

$$\mathcal{L} = \sum_{\boldsymbol{r}\in\mathcal{R}} \|\Gamma(L_o(\boldsymbol{r})) - \Gamma(\hat{L}_o(\boldsymbol{r}))\|_2^2 + \mu \cdot \sum_{\boldsymbol{r}\in\mathcal{R},t} \|V_\varphi(\boldsymbol{r}(t), \boldsymbol{r}_d) - \hat{V}_\vartheta(\boldsymbol{r}(t), \boldsymbol{r}_d)\|_2^2, \tag{9}$$

where $\Gamma(L) = L/(1 + L)$ is a tone mapping function, $\mathcal{R}$ is a batch of camera rays and $\mu = 0.1$ is a hyperparameter to weight the visibility loss.

4.4 Implementation Details

Our feature MLP F_θ has 8 fully-connected ReLU (FC-ReLU) layers with 256 channels per layer. The downstream S_ϕ consists of 8 FC-ReLU layers with 128 channels per layer, whereas the R_ϑ uses one such layer with 128 channels. The visibility MLP V_φ has 4 FC-ReLU layers with 256 channels per layer. We set the maximum positional encoding frequency to 2^8 for coordinates p, 2^1 for directions d, and 2^2 for the 3D location of the point light.

We train all neural networks together to minimize the loss (Eq. 9). We use the Adam [51] optimizer with its default hyperparameters and schedule the learning rate to decay from 1×10^{-4} to 1×10^{-5} over 200K iterations. For each iteration, we trace a batch of 1200 primary rays. Note we stop the gradients from the visibility loss to the density-albedo MLP and the feature MLP so that they do not compromise the learning to match the visibility network.

5 Experiments

We firstly evaluate our method by comparing it with state-of-the-art methods on simultaneous relighting and view synthesis. Then, we demonstrate that our learned implicit representations allow flexible scene editing and scene compositions, followed by ablation studies of the method. Please refer to the Appendix and the supplemental video for additional results.

5.1 Experiment Settings

Compared methods. We compare our method with state-of-the-art relighting and acquisition baselines [8, 9]. They are designed for scenes with solid objects and do not trivially extend to handle participating media objects, so we implement them with the new function to handle participating media. Please refer to the Appendix for our implementation details of the two methods.

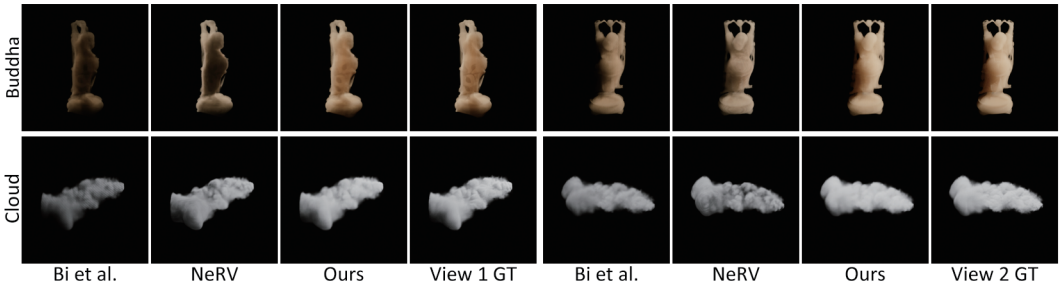


Figure 3: Qualitative comparisons of simultaneous novel view synthesis and relighting. The training illumination is "point" which contains a single point light.

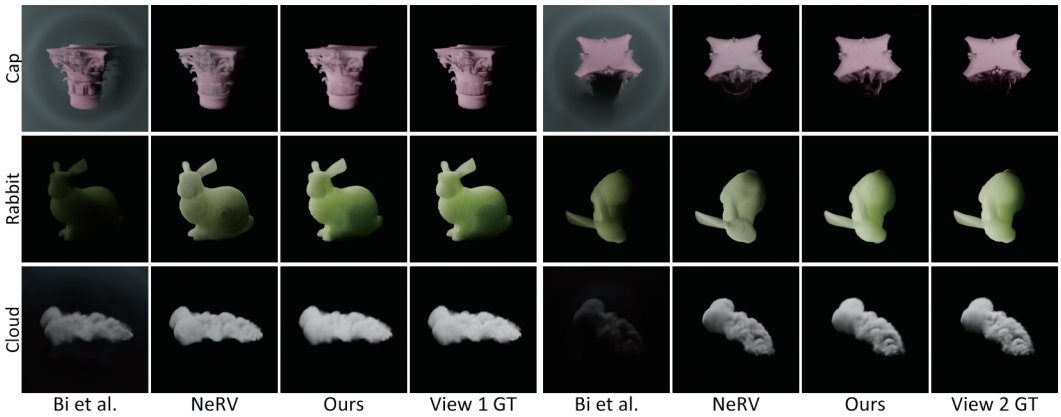


Figure 4: Qualitative comparisons. The training illumination of the three scenes is "env + point".

Datasets. We produce ten datasets from five synthetic participating media scenes. The *Cloud* scene is heterogeneous media and the others (*Buddha*, *Cap*, *Rabbit*, *Lucy*) are homogeneous media. Each scene is individually illuminated with two lighting conditions. the first one "point" has a white point light with varying intensities sampled within $50 \sim 1200$ and its location is randomly sampled on a sphere over the scene; The second one "env + point" contains a fixed environment lighting and a randomly sampled white point light. Each dataset contains 160 images, from which we use 150 images for training and the remaining for validation. In addition, we prepare a test set with 30 images for each scene to test the trained models. Each test image is rendered with a new camera pose and a new white point light. Since Bi's method [8] requires camera-light collocated data, we additionally generated such datasets for it. For the "env + point" datasets, we randomize the usage of environment lighting across images and record a binary indicator for each image.

5.2 Results

Relighting comparisons. We show the qualitative comparisons of simultaneous relighting and view synthesis on the test data in Fig. 3 and Fig. 4, where we present two views for each scene. Bi's method shows artifacts in each case as it has no mechanisms to simulate the multiple scattering that significantly affects the appearance of participating media. Also, it bakes the environment

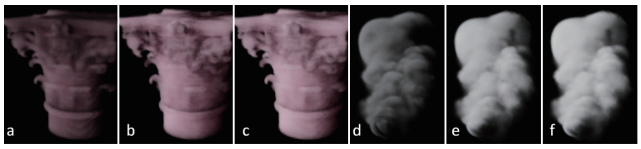
Table 1: Quantitative comparisons on the test data. We measure image qualities with PSNR (↑), SSIM (↑) and ELPIPS (↓) [52]. ELPIPS values below have a scale of $\times 10^{-2}$. Note the tabulated values are the mean values over all images of a test set.

	Single point						Ambient + point light								
	Buddha			Cloud			Cap			Rabbit			Cloud		
Method	PSNR	SSIM	ELPIPS	PSNR	SSIM	ELPIPS	PSNR	SSIM	ELPIPS	PSNR	SSIM	ELPIPS	PSNR	SSIM	ELPIPS
Bi et al.	23.62	0.951	0.518	22.75	0.946	0.543	11.03	0.137	6.655	19.04	0.881	1.123	16.32	0.249	2.051
NeRV	25.47	0.959	0.432	25.81	0.956	0.469	29.52	0.947	0.561	25.31	0.951	0.678	26.83	0.961	0.461
Ours	**32.87**	**0.974**	**0.154**	**33.29**	**0.980**	**0.198**	**31.12**	**0.956**	**0.516**	**31.38**	**0.968**	**0.206**	**32.71**	**0.973**	**0.236**

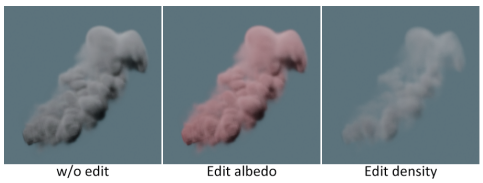
illumination into its representation, so it renders the faint environment illumination though the test set does not contain environment illumination (Fig. 4). NeRV handles the environment illumination properly but shows artifacts on the media objects. Our method achieves realistic results on all test sets with either homogeneous media or heterogeneous media. Also, we encourage readers to watch our supplemental for the dynamic relighting effects. Table 1 presents the corresponding quantitative measurements, where our method overtakes the compared methods significantly on each test set.

Using the same batch size, our training with 200K iterations on a Nvidia Quadro RTX 8000 GPU takes one day, whereas Bi's method and NeRV takes 22h and 46h. For a 400×400 image, our average inference time is 7.9s, while Bi's method and NeRV takes 32.3s and 21.9s, respectively.

Learned lighting decomposition. Merely based on the loss (Eq. 9), our method is able to learn the decomposition of lighting in an unsupervised way without using any ground-truth lighting decomposition data. The figure on the right presents the decomposed results with the single-scattering contribution and the multiple-scattering contribution of the *Cap* and the *Cloud* scenes, where a, d are the single-scattering contribution, b, e are the multiple-scattering contribution, and c, f are the final images.



Scene editing and scene composition. Our method learns implicit representations for the participating media scenes. After training, we can query the neural networks to obtain volume density and albedo data. This allows flexible editing to achieve desired effects or insertion into a new virtual environment for content creation. In addition, we can leverage a standard rendering engine to render these data. The side figure demonstrates an editing of the first channel of the albedo to achieve the red cloud and another editing of the volume density to make the cloud thinner.



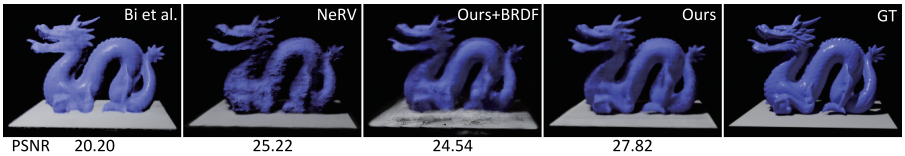
We show in the Fig. 5 that we can compose a scene consisting of our learned rabbit and a gold sculpture represented by traditional meshes and materials. Similarly, we can construct a scene composed entirely of our learned objects. To render the composed scenes, we slice out discrete volumes with a resolution $128 \times 128 \times 128$ from the volume density field and the albedo field and conduct the Monte Carlo rendering in Mitsuba [53].

Figure 5: Scene compositions.

Scene of solid objects. Beyond the scenes with participating media, our method can be used for scenes with solid objects. Figure 6 shows a comparison between our method and the baselines on the *Dragon* scene which contains glossy surfaces. "Our+BRDF" is a variant of our method that still uses SH for indirect illumination, adopts a classical BRDF model [54] and trains the neural network to predict parameters of the BRDF model as done in [8, 9]. Bi's method produces an overexposed appearance and its shadow gets faint. Our method achieves a smooth appearance and high numerical metrics, whereas "Our+BRDF" recovers the highlights on the glossy dragon better.

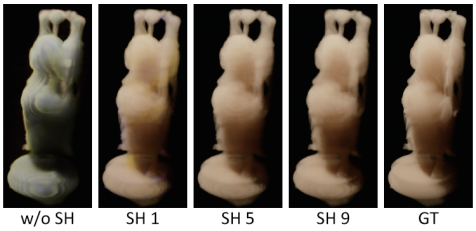
5.3 Ablation Studies

Spherical harmonic bands. We analyze the effect of the maximum spherical harmonics band l_{max} of Eq. 6 based on the *Buddha* scene. Figure 7 compares the same test view of each case on the left and tabulates the average quality measurements over the test set on the right. Removing the spherical harmonics from our method leads to quality drop and we observe color shifts in the test image. Based on the numerical metrics and inference timings, we select the $l_{max} = 5$ for other experiments.



	Bi et al.	NeRV	Ours+BRDF	Ours
PSNR	20.20	25.22	24.54	27.82
SSIM	0.907	0.909	0.885	0.943

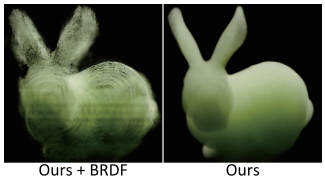
Figure 6: Comparisons on a test view of the solid *Dragon* with glossy surfaces. PSNR and SSIM metrics for this view are listed below images. The training illumination is from a single point light and the test illumination is a new white point light.



	PSNR↑	SSIM↑	ELPIPS↓	Time (s)
w/o SH	25.91	0.9301	0.683	3.91
SH-1	32.53	0.9728	0.175	5.16
SH-3	32.70	0.9739	0.159	5.75
SH-5	**32.87**	**0.9743**	0.154	7.91
SH-7	32.80	0.9740	**0.152**	13.00
SH-9	32.61	0.9739	0.157	16.58

Figure 7: Image quality and mean inference timings of different number of spherical harmonic bands. ELPIPS metrics have a scale of 10^{-2}. The full images, including the SH-3 and SH-7, are documented in the supplemental.

Scattering function. In Fig. 6, we show our method which uses the HG phase function can be applied to a scene with solid objects. In addition, we conduct an ablation by applying the variant "Our+BRDF" to the *Rabbit* scene of participating media. The figure on the right shows that the variant has difficulty in learning the volume density and leads to many cracks, while the propose method by us performs robustly on this scene.



6 Limitations and Future Work

Real-world scenes. In this work, we learn neural implicit representations on synthetic datasets with varying but known lighting conditions. Also, the camera poses are available to our method. It would be interesting to extend this method to handle participating media captured from real-world scenes with unknown lighting conditions and unknown camera poses. In that case, the illumination and camera poses of the scenes need to be estimated in the first place.

Glossy reflections and ringing artifacts. For scenes with glossy solid objects (Fig. 6), our method tends to reproduce a smooth appearance without glossy highlights. In addition, we observe that our variant "Our+BRDF" has "ringing" artifacts [55] (e.g., the dark artifacts on the floor in the 3-rd image of Fig. 6). An avenue for future research would be to develop methods to recover glossy reflections and combat the ringing artifacts for glossy solid objects.

Media within refractive boundaries. Our method achieves realistic results for participating media without refractive boundaries, like cloud, fog, and wax figure. Applying our method to participating media within refractive boundaries, like wine in a glass, entails further work as the refractive boundaries cause ambiguities due to deflecting the camera rays and the light rays.

7 Conclusion

We have proposed a novel method for participating media reconstruction from observed images with varying but known illumination. We propose to simulate direct illumination with Monte Carlo ray tracing and approximate indirect illumination with a learned spherical harmonic field. This enables our approach to learn to decompose the illumination as direct and indirect components in an unsupervised manner. Our method learns a disentangled implicit representation with volume density and scattering albedo for participating media, and we demonstrate its flexible applications in relighting, scene editing and scene compositions.

References

[1] Jinwei Gu, Shree Nayar, Eitan Grinspun, Peter Belhumeur, and Ravi Ramamoorthi. Compressive structured light for recovering inhomogeneous participating media. In *European Conference on Computer Vision*, pages 845–858. Springer, 2008.

[2] Tim Hawkins, Per Einarsson, and Paul Debevec. Acquisition of time-varying participating media. *ACM Transactions on Graphics (ToG)*, 24(3):812–815, 2005.

[3] Christian Fuchs, Tongbo Chen, Michael Goesele, Holger Theisel, and Hans-Peter Seidel. Density estimation for dynamic volumes. *Computers & Graphics*, 31(2):205–211, 2007.

[4] Samuel W Hasinoff and Kiriakos N Kutulakos. Photo-consistent reconstruction of semitransparent scenes by density-sheet decomposition. *IEEE transactions on pattern analysis and machine intelligence*, 29(5): 870–885, 2007.

[5] 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 *European Conference on Computer Vision*, pages 405–421. Springer, 2020.

[6] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: learning dynamic renderable volumes from images. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019.

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

[8] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020.

[9] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. *arXiv preprint arXiv:2012.03927*, 2020.

[10] Yasuhiro Mukaigawa, Yasushi Yagi, and Ramesh Raskar. Analysis of light transport in scattering media. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 153–160. IEEE, 2010.

[11] 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.

[12] Hiroharu Kato, Deniz Beker, Mihai Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon. Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057*, 2020.

[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 Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019.

[15] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. *arXiv preprint arXiv:2003.04618*, 2, 2020.

[16] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2367–2376, 2019.

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

[18] Xiuming Zhang, Sean Fanello, Yun-Ta Tsai, Tiancheng Sun, Tianfan Xue, Rohit Pandey, Sergio Orts-Escolano, Philip Davidson, Christoph Rhemann, Paul Debevec, et al. Neural light transport for relighting and view synthesis. *ACM Transactions on Graphics (TOG)*, 40(1):1–17, 2021.

[19] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019.

[20] Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *European Conference on Computer Vision*, pages 608–625. Springer, 2020.

[21] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019.

[22] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019.

[23] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *arXiv preprint arXiv:2007.11571*, 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2304–2314, 2019.

[25] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *arXiv preprint arXiv:1905.10711*, 2019.

[26] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020.

[27] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2019–2028, 2020.

[28] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG)*, 35(6):1–10, 2016.

[29] Pratul P Srinivasan, Tongzhou Wang, Ashwin Sreelal, Ravi Ramamoorthi, and Ren Ng. Learning to synthesize a 4d rgbd light field from a single image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2243–2251, 2017.

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

[31] Michael Goesele, Hendrik PA Lensch, Jochen Lang, Christian Fuchs, and Hans-Peter Seidel. Disco: acquisition of translucent objects. In *ACM SIGGRAPH 2004 Papers*, pages 835–844. 2004.

[32] Ioannis Gkioulekas, Anat Levin, and Todd Zickler. An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *European Conference on Computer Vision*, pages 685–701. Springer, 2016.

[33] Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. Inverse volume rendering with material dictionaries. *ACM Transactions on Graphics (TOG)*, 32(6):1–13, 2013.

[34] Dejan Azinovic, Tzu-Mao Li, Anton Kaplanyan, and Matthias Nießner. Inverse path tracing for joint material and lighting estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2447–2456, 2019.

[35] Chengqian Che, Fujun Luan, Shuang Zhao, Kavita Bala, and Ioannis Gkioulekas. Towards learning-based inverse subsurface scattering. In *2020 IEEE International Conference on Computational Photography (ICCP)*, pages 1–12. IEEE, 2020.

[36] Cheng Zhang, Lifan Wu, Changxi Zheng, Ioannis Gkioulekas, Ravi Ramamoorthi, and Shuang Zhao. A differential theory of radiative transfer. *ACM Transactions on Graphics (TOG)*, 38(6):1–16, 2019.

[37] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020.

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

[39] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. *arXiv preprint arXiv:2008.02268*, 2020.

[40] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *arXiv preprint arXiv:2007.02442*, 2020.

[41] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. *arXiv preprint arXiv:2012.03918*, 2020.

[42] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Deep reflectance volumes: Relightable reconstructions from multi-view photometric images. *arXiv preprint arXiv:2007.09892*, 2020.

[43] Michelle Guo, Alireza Fathi, Jiajun Wu, and Thomas Funkhouser. Object-centric neural scene rendering. *arXiv preprint arXiv:2012.08503*, 2020.

[44] 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.

[45] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

[46] Joe Kniss, Simon Premoze, Charles Hansen, Peter Shirley, and Allen McPherson. A model for volume lighting and modeling. *IEEE transactions on visualization and computer graphics*, 9(2):150–162, 2003.

[47] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

[48] Louis G Henyey and Jesse Leonard Greenstein. Diffuse radiation in the galaxy. *The Astrophysical Journal*, 93:70–83, 1941.

[49] Jan Novák, Iliyan Georgiev, Johannes Hanika, and Wojciech Jarosz. Monte carlo methods for volumetric light transport simulation. In *Computer Graphics Forum*, volume 37, pages 551–576. Wiley Online Library, 2018.

[50] Matthew Tancik, Pratul P Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739*, 2020.

[51] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[52] Markus Kettunen, Erik Härkönen, and Jaakko Lehtinen. E-lpips: robust perceptual image similarity via random transformation ensembles. *arXiv preprint arXiv:1906.03973*, 2019.

[53] Wenzel Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.

[54] Brian Karis. Real shading in unreal engine 4. *Proc. Physically Based Shading Theory Practice*, 4:3, 2013.

[55] Peter-Pike Sloan. Stupid spherical harmonics (sh) tricks. In *Game developers conference*, volume 9, page 42, 2008.

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] Please refer to Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (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? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
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] The code will be released after publication of this work.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Please refer to Section 4.4 and Section 5.1.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] The training of the models are relatively costly and it is impractical to run the same one for many times.
   (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] Please refer to Section 5.2.
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? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
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]