RNG: Relightable Neural Gaussians

Jiahui Fan[1], Fujun Luan[2], Jian Yang[1*] Miloš Hašan[2], and Beibei Wang[3*]
[1]PCA Lab [†], Nanjing University of Science and Technology
[2]Adobe Research [3]Nanjing University
{fjh, csjyang}@njust.edu.cn, fluan@adobe.com, milos.hasan@gmail.com, beibei.wang@nju.edu.cn

Abstract

3D Gaussian Splatting (3DGS) has shown impressive results for the novel view synthesis task, where lighting is assumed to be fixed. However, creating relightable 3D assets, especially for objects with ill-defined shapes (fur, fabric, etc.), remains a challenging task. The decomposition between light, geometry, and material is ambiguous, especially if either smooth surface assumptions or surface-based analytical shading models do not apply. We propose Relightable Neural Gaussians (RNG), a novel 3DGS-based framework that enables the relighting of objects with both hard surfaces or soft boundaries, while avoiding assumptions on the shading model. We condition the radiance at each point on both view and light directions. We also introduce a shadow cue, as well as a depth refinement network to improve shadow accuracy. Finally, we propose a hybrid forward-deferred fitting strategy to balance geometry and appearance quality. Our method achieves significantly faster training (1.3 hours) and rendering (60 frames per second) compared to a prior method based on neural radiance fields and produces higher-quality shadows than a concurrent 3DGS-based method. Project page: whois-jiahui.fun/project_pages/RNG.

1. Introduction

Creating 3D assets from multi-view captures of the real world is an effective way for content creation, avoiding manual modeling labor. The resulting 3D assets can be objects with well-defined surfaces or ill-defined shapes (e.g., fur, fabric, grass, etc.), as both are important in many applications. Unfortunately, if we want the resulting assets to be *relightable*, the task is still challenging because of the ill-posed nature of the decomposition between light, materials, and geometry. This is especially true for complex non-smooth materials, which raise difficulties in decomposition, as surface-specific constraints or surface-based analytical shading models cannot be leveraged. In this paper, we aim to relight objects with clear surfaces or soft boundaries given multi-view captured images with varying illumination, simultaneously achieving high-quality relighting and shortening training/rendering times.

After ground-breaking view-synthesis work on Neural Radiance Fields (NeRF) [20] and 3D Gaussian Splatting (3DGS) [15], extensive efforts have focused on reconstructing relightable 3D assets [9, 13, 14, 17–19, 24]. However, these methods mostly rely on surface shading models and introduce surface constraints (including the assumption of valid normals), preventing them from reconstructing objects with soft boundaries and/or materials that are not well represented with simple analytic models. Recently, NRHints [27] enabled relightable capture of both smooth surfaces and objects with soft boundaries by using input views with a moving point light and a neural appearance model. Being based on a NeRF framework, NRHints suffers from high training/rendering time costs and some over-smoothing of detail. The concurrent work GS^3 [2] uses the same capture setup but instead uses 3DGS as the underlying framework, which is more efficient in training and rendering, and captures finer details. With the less accurate geometry obtained from 3DGS, GS^3 has relatively lower shadow quality.

In this paper, we propose *Relightable Neural Gaussians* (RNG), a novel 3DGS-based framework for relighting objects with both clear surfaces and soft boundaries. We implicitly model the radiance of objects by learning latent (feature) vectors at each *neural Gaussian*. To interpret neural Gaussians, we use the *neural Gaussian decoder* network, and condition it on the view *and light* directions. Analytical assumptions in shading models and surface constraints are avoided in our neural representation, making it capable of learning appearances that do not fit well into those constraints.

Following prior work [27], we utilize views with a moving point light, to observe many view/light combinations

*Corresponding authors.
†PCA Lab, Key Lab of Intelligent Perception and Systems for High-Dimensional Information, School of Computer Science and Engineering, Nanjing University of Science and Technology, China.

and reduce ambiguities in decomposition. However, point lights produce sharp shadows, which are challenging for neural networks to capture accurately. We present a *shadow cue with depth refinement* to condition the neural Gaussian decoder, improving the shadow quality. We also introduce a two-stage hybrid (forward-deferred) optimization pipeline for better shadow appearance.

In our results, RNG shows not only higher-quality details than the NeRF-based prior method NRHints, but also more accurate shadows than the concurrent 3DGS-based approach GS^3. In terms of performance, RNG takes about 1.3 hours for training and achieves a 60 frame per second (fps) rendering performance on an RTX 4090 GPU, which is competitive with 3DGS and GS^3 and many times faster than NRHints.

To summarize, our main contributions include

- a relightable neural Gaussian framework to render objects with smooth surfaces or soft boundaries, under arbitrary view and light directions, and with no analytic assumptions on the shading model,
- a shadow cue technique and a depth refinement network to enhance the quality of shadows, and
- a hybrid (forward-deferred) optimization strategy, achieving high-quality reconstruction and sharp shadow appearance.

2. Related work

2.1. Inverse rendering

Inverse rendering [1, 19, 29] decomposes the light, material, and geometry with multi-view RGB inputs, and the decomposed assets can be relit under any desired novel lighting. To represent the materials, several methods introduce a standard shading model similar to the Disney Principled BRDF [5] as a physically-based prior, and neural materials [32] can also be utilized for material recovery. With the representation capacity of NeRF, some methods [3, 4, 26, 28] produce high-quality inverse rendering at the cost of high training consumption and slow rendering speed. Jin et al. [14] use grid features to represent the scenes, leading to relatively fast training speed. Zhang et al. [33] adopts the SGGX Microflake model [11] to perform inverse rendering, achieving unique effects for semi-transparent targets. SDFs are also commonly used in the representations [17, 24, 30], leading to smoother surface normals but biasing the method to objects with relatively smooth surfaces.

3DGS brings the rasterization framework into multi-view stereo reconstructions. However, this nature of 3DGS also hurts the quality of its obtained geometric attributes, such as depth and normal, making them noisy and difficult for further use. Some existing methods [6, 7, 10, 12] bring constraints or introduce meshes into the Gaussians, improving the geometry quality. By introducing analytical shading models, several works utilize 3DGS to achieve inverse rendering under unknown environment lighting. Jiang et al. [13] and Shi et al. [23] supervise the normals via the orientation of Gaussians, while Gao et al. [9] and Liang et al. [18] leverage depth to obtain the normal information.

Compared to NeRF-based methods, 3DGS-based methods are more efficient and handle soft-boundary objects better due to their flexible representation. Therefore, we choose to use 3DGS as the underlying framework. Further, previous approaches with surface priors fail to handle soft objects, and the fixed types of analytical models also limit the application. In contrast, our method generalizes across a wider range of scenarios without such assumptions.

2.2. Relighting of ill-defined shapes

Most existing inverse rendering methods cannot simply extend to soft-boundary objects, due to the incompatibility of shading models and the challenging light transport in such scenes. Gao et al. [8] present Deferred Neural Relighting, which leverages learned neural texture on a rough proxy geometry for relighting objects including fluffy shapes. Mullia et al. [21] propose a novel representation that combines explicit geometry with a neural feature grid and an MLP decoder, achieving high-fidelity rendering and relighting with good flexibility and integration. Unlike our method, they only support synthetic inputs and require ground-truth geometry.

Recently, Zeng et al. [27] proposed NRHints, which maintains an implicit neural representation with both SDF and NeRF-style feature grids, and predicts radiance with shadow and highlight hints, achieving high-quality relighting. However, NRHints is computationally heavy in training and rendering, and tends to over-smooth soft objects, especially at boundaries. Our concurrent work GS^3 [2] introduces triple splatting to relight objects. They introduce analytical appearance approximation that is also supplemented by neural networks, enabling high-efficiency relighting for fluffy objects as well. GS^3 still suffers from the inherent lower-quality geometry of Gaussian point cloud, leading to less sharp shadow appearance.

We target the same problem as in NRHints and GS^3, and use point-lit images as input as well. However, we introduce neural Gaussians, avoiding surface constraints and shading model assumptions, gaining more flexibility in representation. We further propose the shadow cue with depth refinement to enhance the shadow quality, and design a hybrid optimization strategy. Overall, RNG achieves faster training/rendering and finer details than NRHints, as well as higher shadow quality under point lights than GS^3.

3. Method

The goal of our work is to reconstruct high-quality relightable assets for objects with both hard surfaces and soft

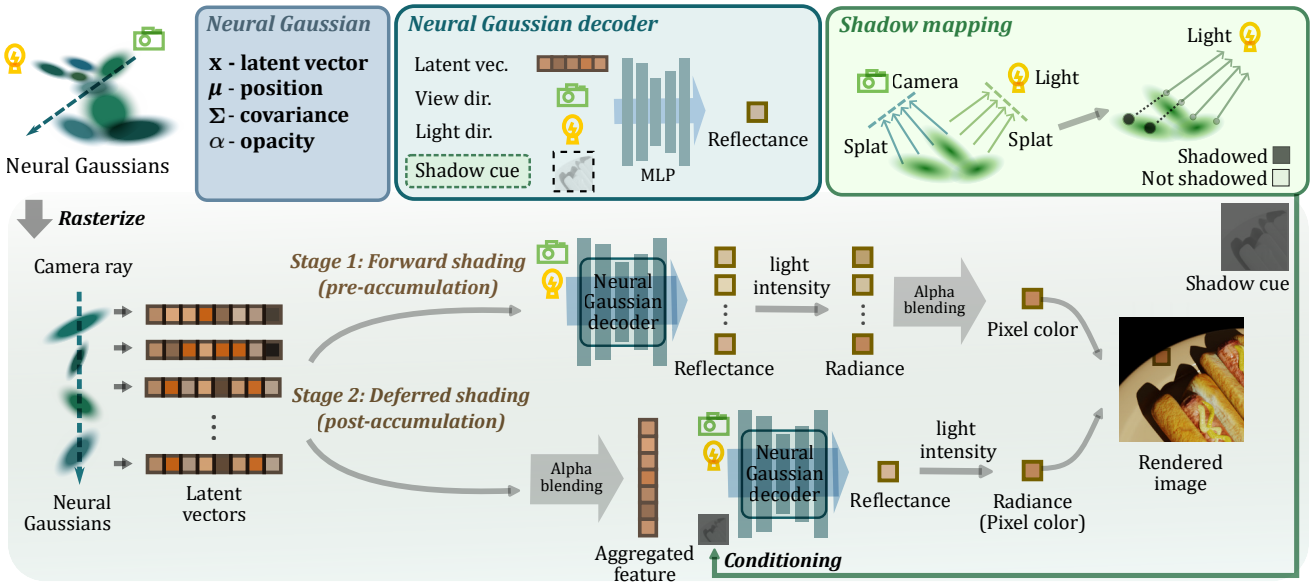


Figure 1. The overview of RNG. Each Gaussian point in the scene contains an extra latent vector that describes the reflectance. The latent values interpreted by an MLP decoder, conditioned on view and light directions. Training has two stages. In the first stage, we employ *forward shading*, where we decode all the latent vectors of Gaussian points into colors, followed by the alpha blending. In the second *deferred shading* stage, we first alpha-blend the neural Gaussian features to get an aggregated feature, and then we feed it to the decoder. We apply shadow mapping to obtain a shadow cue map and use the shadow cue as an extra input for the decoder in the second stage.

boundaries while maintaining fast training and rendering time. We propose relightable neural Gaussians (Sec. 3.2) to implicitly model the reflectance. We also apply a shadow cue with depth refinement (Sec. 3.3) to improve the quality of shadows and design a hybrid forward-deferred optimization strategy (Sec. 3.4) to further improve the shadow appearance while preserving the quality of geometry. Fig. 1 illustrates the overview of our method.

3.1. Background: 3D Gaussian Splatting

3DGS represents a scene with a set of 3D Gaussians, each of which is defined as

$$\text{Gaussian}(x|\mu,\Sigma) = e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}, \tag{1}$$

where x is a position in the scene, μ is the mean of the Gaussian, and Σ denotes the covariance matrix of the 3D Gaussian, which is factorized into a scaling matrix S and a rotation matrix R as $\Sigma = RSS^TR^T$. To render an image, 3DGS projects the 3D Gaussians onto the 2D image plane and employs alpha blending on the sorted Gaussians as

$$C = \sum_{i\in\mathbb{N}} c_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{2}$$

where c_i is the color of each Gaussian, and α_i is given by evaluating a projected 2D Gaussian with covariance Σ' multiplied with a learned per-point opacity. In 3DGS, the alpha blending of color c_i (which depends on the view direction and is represented by spherical harmonics) from every nearby Gaussian point yields the reflectance at position x.

3.2. Relightable neural Gaussians

Existing 3DGS-based relighting methods leverage analytical shading models and/or surface assumptions. Instead, we use a learned latent space to implicitly represent the view- and light-dependent reflectance in the scene. As shown in Fig 1, each Gaussian point carries a latent (feature) vector that models this reflectance. To enable relightability, the reflectance has to be dependent on not only the view directions $\omega_\mathbf{o}$ but also the light directions $\omega_\mathbf{i}$. Therefore, the network can decode and predict the reflectance values at novel light positions. The final reflectance is represented as

$$\rho(\mathbf{x},\omega_\mathbf{o},\omega_\mathbf{i}) = \Theta(\mathbf{x}|\omega_\mathbf{o},\omega_\mathbf{i}), \tag{3}$$

where Θ is the neural Gaussian decoder and $\mathbf{x}$ is the shading point with its corresponding latent vector. This reflectance value is analogous to the BRDF times cosine term from the standard rendering equation, and needs to be multiplied by light intensity and light falloff to obtain final radiance from a point light. When applying novel lighting conditions, the network takes the given point light positions and view directions as conditioning inputs, leading (after combining with incoming light intensity) to a neural implicit relightable

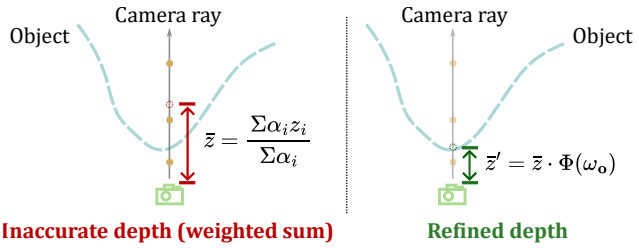


Figure 2. The effect of the depth refinement network. The weighted sum of Gaussian depths is not accurate, resulting in mismatching shadow cues. Therefore, we propose a depth refinement network to correct the depth.

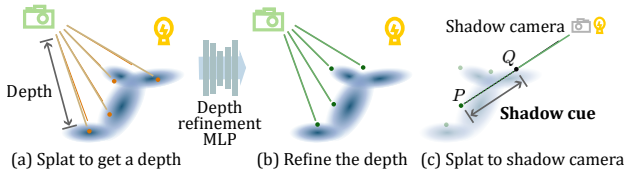


Figure 3. The illustration of shadow cue computation. First, we splat the Gaussians onto the camera to get depth values. Then, we run the depth refinement network to correct them and locate the shading points P. At last, we splat the shading points onto the shadow camera to find the intersections of shadow rays Q, and store the distance $|PQ|$ as the shadow cue.

radiance representation.

3.3. Shadow cue

With our proposed neural Gaussians, the reflectance at positions in the scene can be represented by latent vectors stored in each Gaussian point. However, there are still some potential quality issues. First, the network tends to over-fit all view/light directions in the training set, resulting in blurry or incomplete shadows in unseen predictions or inconsistent shapes in movement. Second, point lights yield sharp shadows, and the MLP is prone to over-smooth such high-frequency signals.

To address the above issues, we introduce a shadow cue to condition the neural Gaussian decoder. The shadow cue is a 1-channel map in the screen space that indicates the visibility to the light of each shading point and will be fed into the MLP together with other inputs described in Eq. 3.

We obtain the shadow cue by performing shadow mapping under the 3DGS framework. Shadow mapping requires the precise locations of shading points. However, since we do not explicitly trace rays, we can instead use the depth value for each pixel for shading point computation. Obtaining the depth values of a Gaussian cloud is not well-defined. Therefore, we introduce a depth refinement network to correct the depth values and help find the valid shading points.

Depth refinement. An intuitive and naive proxy for the depth is the weighted sum of the depth and Gaussian responses along the ray,

$$\bar{z} = \frac{\Sigma \alpha_i z_i}{\Sigma \alpha_i}, \tag{4}$$

where z_i is the depth value of i^{th} Gaussian on the camera ray and α_i is the ray response at each intersections of 3D Gaussians. Usually, this proxy is normalized, thus the background leakage at semi-transparent areas can be reduced. However, sometimes the weighted sum is incorrect, leading to wrongly located shading points and consequently mismatching shadow cues. We discuss and showcase this situation in the supplementary. To address this issue, we propose a depth refinement network to correct the shading point locations by learning a scaling factor, as shown in Fig. 2. We assume the depth correction is linear for each pixel and dependent on view directions ω_o. Therefore, the refined depth value is obtained by

$$\bar{z}' = \bar{z} \cdot \Phi(\omega_o), \tag{5}$$

where Φ is the depth refinement MLP.

Shadow mapping. Conventional shadow mapping marches the ray to get visibility, which can be expensive and difficult to achieve. Instead, under the 3DGS framework, we perform an extra pass of Gaussian splatting to cast shadow mapping. We obtain the shadow cue in the following steps, as shown in Fig. 3. First, we splat Gaussians onto the camera and record the depth values of each pixel. Second, we run the depth refinement network to correct the depth and calculate a shading point P for each pixel based on the pixel depth. After this, we set a virtual shadow camera at the point light position. We splat Gaussians onto the shadow camera for a second pass, recording the depth to find a shading point Q for each pixel in the shadow camera. We project P into the shadow camera frame to find its corresponding Q. Since Q is equivalently the shadow ray intersection, the distance $|PQ|$ is recorded as the shadow cue for this pixel.

Note that for computational efficiency, we omit the depth refinement when we obtain the shadow camera depth. With shadow cues, the neural Gaussian decoder takes multiple inputs, and all of them contribute to the final color of a single Gaussian point. The final reflectance is represented as

$$\rho(\mathbf{x}, \omega_\mathbf{o}, \omega_\mathbf{i}) = \Theta(\mathbf{x} | \omega_\mathbf{o}, \omega_\mathbf{i}, V), \tag{6}$$

where $V = |PQ|$ is the shadow cue. In practice, we use the same resolution as the camera for the shadow camera, and we apply a clamping between zero and the scene units to the shadow cue map for the stability of training.

3.4. Hybrid optimization

With all the components above, we now have the RNG framework, where the scene is represented as a structure of neural Gaussian points, and the reflectance at each Gaussian point is represented as a feature vector that is conditioned on view/light directions and shadow cues. The adjusted 3DGS rasterization operation becomes

$$C_{\text{forward}} = \sum_{i\in\mathbb{N}} \Theta(\mathbf{x_i}|\omega_\mathbf{o}, \omega_\mathbf{i}, S, V)\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \quad (7)$$

where C_{forward} is the color at each pixel. We call this rasterization procedure *forward shading*.

In forward shading, the alpha blending after the reflectance computation blurs the shadow. To address this problem, we introduce the *deferred shading*.

Deferred shading. In deferred shading, we blend the feature vectors of Gaussians first to get an aggregated feature in image space, and then we decode it with the neural Gaussian decoder. In this case, we propose the rasterization of deferred shading as

$$C_{\text{defer}} = \Theta(\sum_{i\in\mathbb{N}} \mathbf{x_i}\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j)|\omega_\mathbf{o}, \omega_\mathbf{i}, V). \quad (8)$$

To our observation, forward shading produces better geometry and worse shadow, while deferred shading improves the shadow appearance but causes floaters. Therefore, we design a two-stage hybrid optimization strategy to benefit from both options. Further investigation into this choice is discussed in the supplementary.

Two-stage strategy. The whole training procedure of RNG consists of two stages. We employ forward shading in the first stage to get Gaussian points and latent vectors and use deferred shading in the second stage. In the second stage, we enable the shadow cue and re-train the neural Gaussian decoder. We keep all learned latent vectors in the first stage as initialization of the second stage, in order to provide more semantic information and accelerate the training of the second stage. Note that in the first stage, we do not enable the shadow cue, because at the early stage, the Gaussians are not well-shaped, and the generated wrong shadow information may hurt the training stability.

4. Results

In this section, we validate the effectiveness and quality of RNG. We first provide the implementation details in Sec. 4.1, and describe about the experiment setups in Sec. 4.2. Then, we evaluate our method with quantitative results in Sec. 4.3 and provide ablation studies in Sec. 4.4.

4.1. Implementation

We implemented our method using the Pytorch [22] framework. The feature vector in each Gaussian is 16-channel, and the neural Gaussian decoder is an MLP with 4 hidden layers and 256 hidden units. We apply frequency encoding to both view/light directions and shadow cues, making them 15 dimensions and 17 dimensions, respectively. We use the Adam optimizer [16] and train it at a learning rate 1.0×10^{-3} for the color decoder MLP, 3.0×10^{-4} for the depth refinement MLP and 2.5×10^{-3} for feature vectors in Gaussian points. The same loss functions from 3DGS [15] are used, which is a combination of L_1 loss and structural similarity index (SSIM) [25]. We train the model for a total 100k steps, and the forward shading stage usually takes the first 30k iterations. In order to improve the computational efficiency, we cache the shadow cue for each training view and only update them every 5 iterations. We run all our results on an RTX 4090 GPU and i9-13900K CPU, powered by a Windows Subsystem Linux 2 (Ubuntu 22.04.5) distribution.

4.2. Experiment setups

Datasets. We validate our relighting quality by comparing it to previous methods on real and synthetic datasets from NRHints [27] and RNA [21]. We run all results on down-sampled datasets with 512×512 resolution and a maximum of 1000 training views. For synthetic data, all backgrounds are colored in black.

Comparing methods. We select four representative NeRF-/GS-based relighting methods for comparison. For validating the relighting quality under point lights, we compare with NRHints [27] and GS^3 [2]. Furthermore, we compare with GS-IR [18] and Relightable 3D Gaussian [9] to validate the relighting under environment lights.

Metrics. We provide the peak signal-to-noise ratio (PSNR), SSIM, and perceptual similarity (LPIPS) [31] values for comparison, to compare both pixel-wise error and visual differences.

4.3. Quality validation

In Fig. 4, we compare our relighting results with NRHints and GS^3 under point lighting on real-world objects and synthetic scenes. We provide renderings under novel views/lights and their difference maps for comparison. Both GS^3 and our method have finer details, especially for furry objects. Both NRHints and our method handle the shadow effects well, producing more solid and sharp shadow regions. We typically have lowest LPIPS and highest/second-highest PSNR and SSIM values, indicating the overall quality of our method. In terms of training time,

Table 1. Comparison of various scenes between NRHints [27], GS^3 [2] and our method. We provide (from left to right) PSNR (↑), SSIM (↑), and LPIPS (↓) for comparison, and the best/second-best results are colored in red / orange, respectively. We produce the best or second-best results on most scenes and with better PSNR, SSIM, and LPIPS values on average, indicating high-fidelity reconstruction and realistic details in our renderings.

Scene	NRHints[27]			GS^3 [2]			Ours		
Cat	28.3712	0.8751	0.1318	26.0850	0.8815	0.1019	28.3869	0.8883	0.0847
CatSmall	35.4472	0.9705	0.0450	34.4018	0.9729	0.0390	34.7511	0.9699	0.0377
Cluttered	32.1470	0.9434	0.0629	30.2874	0.9443	0.0489	30.7970	0.9442	0.0456
CupFabric	38.1833	0.9831	0.0256	37.1364	0.9830	0.0236	38.5429	0.9857	0.0170
Fish	30.2113	0.9000	0.1176	30.8571	0.9180	0.0668	31.0113	0.9195	0.0561
FurBall	26.7098	0.9340	0.0524	26.3552	0.9309	0.0577	27.8211	0.9263	0.0436
HairBlonde	32.4589	0.9497	0.0388	32.9148	0.9715	0.0194	34.7907	0.9731	0.0147
Hotdog	32.8954	0.9728	0.0227	25.4029	0.9489	0.0483	30.3820	0.9603	0.0339
Lego	29.5974	0.9559	0.0300	26.6257	0.9226	0.0514	26.7235	0.9244	0.0506
Pikachu	33.5846	0.9716	0.0248	32.1464	0.9697	0.0294	31.3826	0.9661	0.0289
Pixiu	31.4333	0.9360	0.0751	30.3765	0.9371	0.0640	30.3485	0.9410	0.0540
RedCloth	34.0962	0.9186	0.1002	31.6039	0.9328	0.0489	35.2186	0.9489	0.0282
WhiteFur	23.4099	0.8871	0.1004	32.8326	0.9662	0.0220	33.7007	0.9700	0.0140
Average	31.4266	0.9383	0.0636	30.5404	0.9446	0.0478	31.8352	0.9475	0.0392

Table 2. Comparison of relighting under novel environment lighting with prior GS-based relighting methods [9, 18]. We provide (from left to right) PSNR (↑), SSIM (↑) and LPIPS (↓) for comparison, and the best/second-best results are colored in red / orange, respectively. Our model significantly improves the accuracy in decomposing light and materials, yielding overall prevailing metrics.

Scene	GS-IR[18]			RelightableGS[9]			Ours		
Armadillo	30.4157	0.8726	0.0316	24.4747	0.8765	0.0327	37.0618	0.9062	0.0145
CupPlane	20.7918	0.8577	0.0640	25.5422	0.9083	0.0292	28.7075	0.9189	0.0281
HairBlue	26.6556	0.8154	0.0796	20.7416	0.8187	0.0721	31.3762	0.8731	0.0700
HairYellow	23.2272	0.7996	0.1308	25.3962	0.8266	0.1152	25.6469	0.8527	0.1102
Average	25.2726	0.8363	0.0765	24.0387	0.8575	0.0623	30.6981	0.8877	0.0557

both GS^3 and our method are significantly faster (more than 20×) than NRHints.

In Table 1, we also report the statistics of NRHints, GS^3, and our method on a series of datasets. Our method has overall lower LPIPS and close SSIM values to GS^3, as we have better details. Since we also introduce the shadow cues and hybrid optimization, the shadow quality also increases the realism of our renderings. We are also competitive with NRHints in terms of pixel-wise errors with much lower training/rendering time cost, thanks to the efficiency of 3DGS and the flexibility of neural appearance models.

In Fig. 5, we provide the relighting results under novel environment lighting and compare them with prior GS-based methods [9, 18], and report the quantitative results across different datasets in Table 2. We run their methods with datasets under unknown environment lighting, and run our method with the same amount of training views under point lights, to be as fair as possible. After training all models, we relight them under the same novel environment map for quality comparison. Our method produces closer appearance to the reference and more plausible shadow effects, achieving higher PSNR/SSIM and lower LPIPS values. Our method benefits from two aspects: the point-lit input images and the neural appearance model. Capturing with point lights helps us better decompose the lighting and materials, and the neural appearance model can handle complex light transport such as sub-surface scattering and hair fiber scattering, leading to overall better results.

We also provide additional validation and visualization of our learned geometry and shadows, the relighting quality, and the power of neural appearance in our supplementary. Please refer to them for more details.

4.4. Ablation study

Ablation of all components. In Fig. 6, we show the ablation of our model by gradually removing the depth

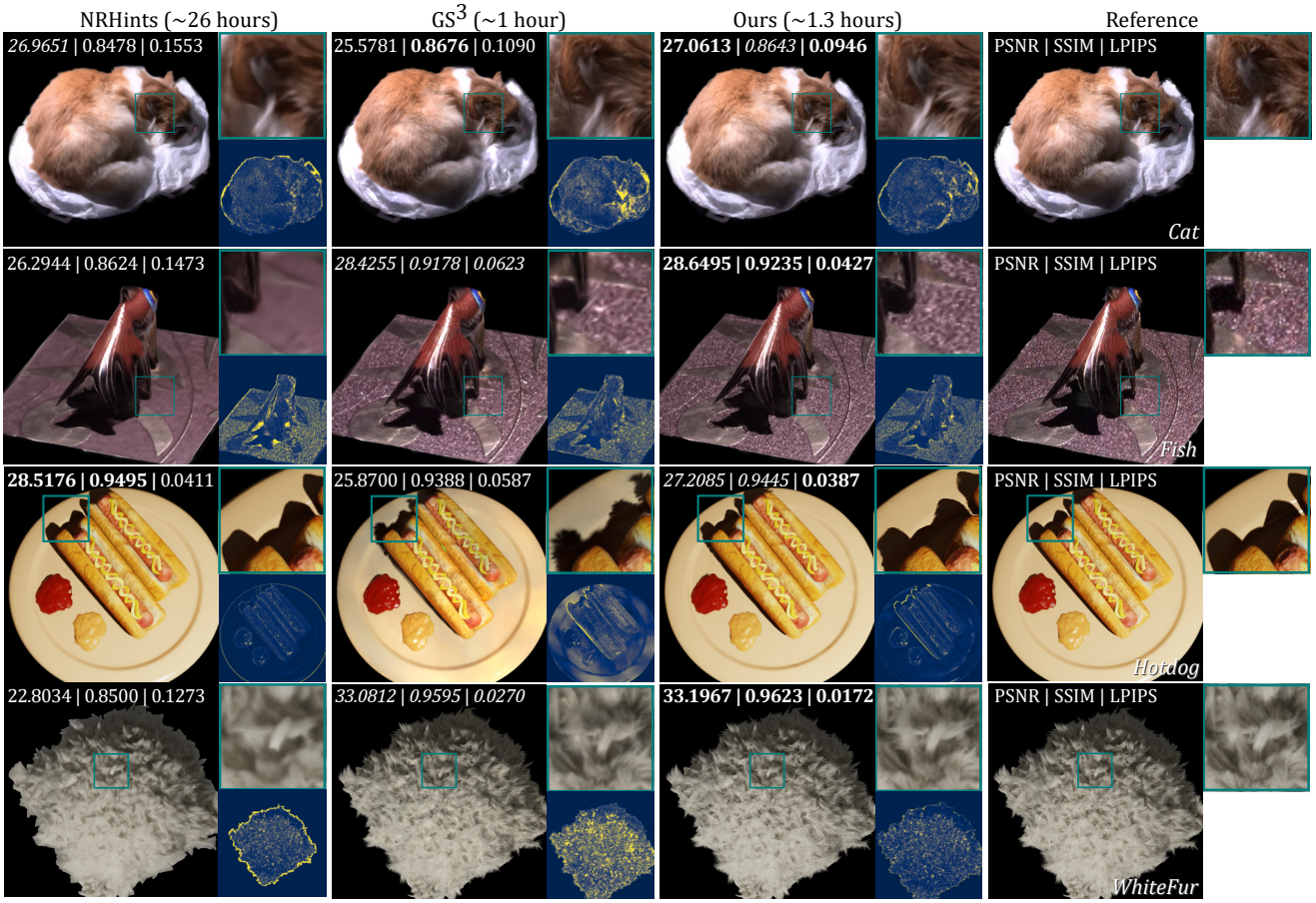


Figure 4. Comparison between NRHints [27], GS^3 [2] and our method on real/synthetic datasets under point lights. The best/second-best results are marked as **bold**/*italic*, respectively. Our method has the lowest LPIPS with the shown images and is also the best or second-best in PSNR and SSIM values. Our method also has better shadow areas than GS^3.

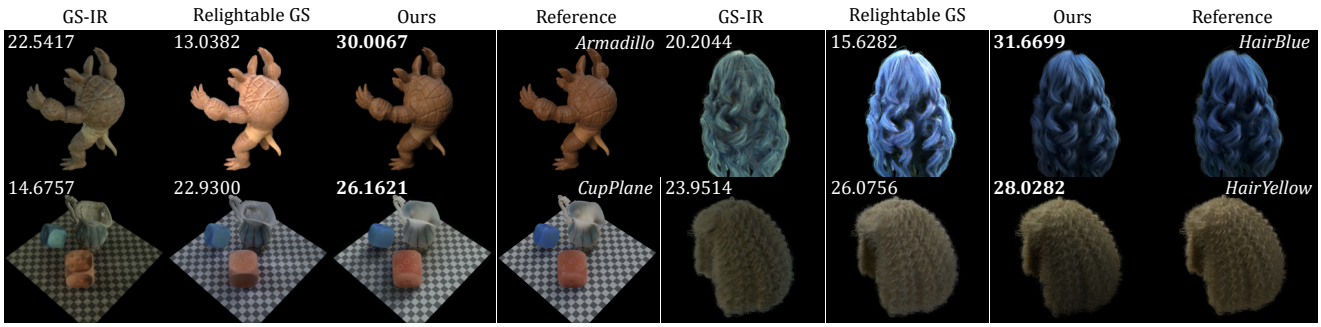


Figure 5. Comparison of relighting results under environment lighting with PSNR values of each image. We compare our relighting results with some previous GS-based methods [9, 18] and the ground truth. Our method decomposes the light and materials better and achieves better relighting, as we utilize point-lit images and the neural appearance model.

refinement MLP, the shadow cue, and the deferred shading. The quality gap indicates the effectiveness and necessity of all the components of RNG. We also provide the ablation study on network sizes in our supplementary.

Effect of depth refinement network. In Fig. 7, we show the significance of the depth refinement network. We compare the shapes and positions of the cast shadows via shadow mapping with/without the depth refinement network. As a result, there is obvious mismatching in

Figure 6. The ablation study of RNG components. We gradually remove them from our full model and show the quality gap between them with PSNR values. The shadow quality is significantly decreased without these components, and the PSNR values also demonstrate their effectiveness and necessity.

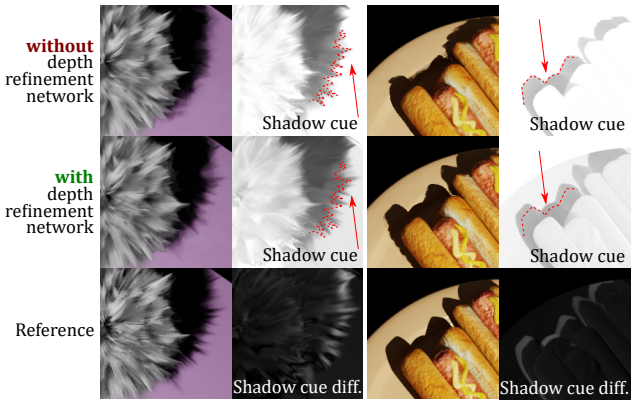


Figure 7. The comparison of results with/without depth refinement MLP and the visualizations of corresponding shadow cues. The corresponding positions of shadow cues in both cases are marked with red arrows and dotted lines. With the depth refinement, the shadow mapping gives more reasonable and matched shadow cues, helping the network to better condition the appearance of the shadow information.

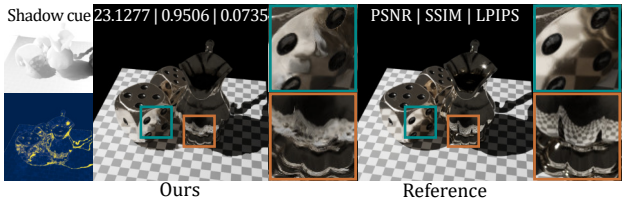


Figure 8. Comparison of our method and the ground truth on highly reflective objects. Our result blurs the reflection, as we did not introduce ray marching into our framework.

positions with the ground truth if we remove the depth refinement network. The model can more accurately locate the shading points and generate more reasonable shadow cues with depth refinement.

4.5. Discussion and limitations

Precision of our representation. RNG shows less accuracy in terms of PSNR values in some scenes than NRHints. The main reason is that NRHints uses an SDF as a powerful prior, which gives very accurate shadows for objects that are clearly surface-like. Furthermore, NRHints uses larger networks than ours; we trade off between quality and computational overhead.

Geometry quality. Although we deploy the shadow cue to help the network predict better shadow appearances and present the depth refinement network to compensate for this inaccuracy, the shadow quality is still limited by the geometry reconstruction precision, since a perfect geometry reconstruction for soft-boundary objects is not trivial. Therefore, we suffer from this disadvantage like most GS-based approaches.

Complex material effects. RNG can handle objects with both hard surfaces and ill-defined shapes. However, since we use rasterization instead of ray marching, it is difficult for our model to handle highly reflective appearances. We also show a failure case in Fig. 8. The reflection of the checkerboard is blurry on the dice, and the reflected shadow is incomplete.

5. Conclusion

In this paper, we have proposed RNG for relighting both surface-based and soft-boundary objects under the 3DGS framework. The proposed neural Gaussian framework avoids assumptions on shading models, and the shadow cue helps produce sharp shadows, together with our hybrid optimization strategy. RNG can render high-fidelity details and high-quality shadow effects, achieving real-time rendering with a significantly improved training (1.3 hours) and rendering (60 fps) performance, compared to prior work.

There are still many potential future research directions. For example, introducing reflection and refraction into the existing framework may be promising. Loosening the requirement on the lighting conditions and supporting more flexible capture setups would be valuable but also challenging. Another potential direction is to explore a more accurate definition of depths in a Gaussian splatting framework, further improving the shadow quality.

Acknowledgements

We thank the reviewers for their valuable comments. This work is supported by the National Science Fund of China under Grant Nos. U24A20330, 62361166670, and 62172220.

References

[1] 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. *ACM TOG*, 2020. 2

[2] Zoubin Bi, Yixin Zeng, Chong Zeng, Fan Pei, Xiang Feng, Kun Zhou, and Hongzhi Wu. Gs^3: Efficient relighting with triple gaussian splatting. In *SIGGRAPH Asia 2024 Conference Papers*, 2024. 1, 2, 5, 6, 7

[3] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2021. 2

[4] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan Barron, and Hendrik Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. *Advances in Neural Information Processing Systems*, 34: 10691–10704, 2021. 2

[5] Brent Burley and Walt Disney Animation Studios. Physically-based shading at disney. In *Acm Siggraph*, pages 1–7. vol. 2012, 2012. 2

[6] Hanlin Chen, Chen Li, and Gim Hee Lee. Neusg: Neural implicit surface reconstruction with 3d gaussian splatting guidance. *arXiv preprint arXiv:2312.00846*, 2023. 2

[7] Pinxuan Dai, Jiamin Xu, Wenxiang Xie, Xinguo Liu, Huamin Wang, and Weiwei Xu. High-quality surface reconstruction using gaussian surfels. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2

[8] Duan Gao, Guojun Chen, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. Deferred neural lighting: free-viewpoint relighting from unstructured photographs. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020. 2

[9] Jian Gao, Chun Gu, Youtian Lin, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. *arXiv preprint arXiv:2311.16043*, 2023. 1, 2, 5, 6, 7

[10] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024. 2

[11] Eric Heitz, Jonathan Dupuy, Cyril Crassin, and Carsten Dachsbacher. The sggx microflake distribution. *ACM Transactions on Graphics (TOG)*, 34(4):1–11, 2015. 2

[12] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2

[13] Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5322–5332, 2024. 1, 2

[14] Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. Tensoir: Tensorial inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2023. 1, 2

[15] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 5

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2014. 5

[17] Jia Li, Lu Wang, Lei Zhang, and Beibei Wang. Tensosdf: Roughness-aware tensorial representation for robust geometry and material reconstruction. *ACM Transactions on Graphics (TOG)*, 43(4):1–13, 2024. 1, 2

[18] Zhihao Liang, Qi Zhang, Ying Feng, Ying Shan, and Kui Jia. Gs-ir: 3d gaussian splatting for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21644–21653, 2024. 2, 5, 6, 7

[19] Yuan Liu, Peng Wang, Cheng Lin, Xiaoxiao Long, Jiepeng Wang, Lingjie Liu, Taku Komura, and Wenping Wang. Nero: Neural geometry and brdf reconstruction of reflective objects from multiview images. *ACM Transactions on Graphics (TOG)*, 42(4):1–22, 2023. 1, 2

[20] 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. *Communications of the ACM*, 65(1):99–106, 2021. 1

[21] Krishna Mullia, Fujun Luan, Xin Sun, and Miloš Hašan. Rna: Relightable neural assets. *ACM Transactions on Graphics*, 2024. 2, 5

[22] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 5

[23] Yahao Shi, Yanmin Wu, Chenming Wu, Xing Liu, Chen Zhao, Haocheng Feng, Jingtuo Liu, Liangjun Zhang, Jian Zhang, Bin Zhou, et al. Gir: 3d gaussian inverse rendering for relightable scene factorization. *arXiv preprint arXiv:2312.05133*, 2023. 2

[24] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 1, 2

[25] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE TIP*, 2004. 5

[26] Ziyi Yang, Yanzhen Chen, Xinyu Gao, Yazhen Yuan, Yu Wu, Xiaowei Zhou, and Xiaogang Jin. Sire-ir: Inverse rendering for brdf reconstruction with shadow and illumination removal in high-illuminance scenes. *arXiv preprint arXiv:2310.13030*, 2023. 2

[27] Chong Zeng, Guojun Chen, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. Relighting neural radiance fields with shadow and highlight hints. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023. 1, 2, 5, 6, 7

[28] Jingyang Zhang, Yao Yao, Shiwei Li, Jingbo Liu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. Neilf++: Inter-reflectable light fields for geometry and material estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3601–3610, 2023. 2

[29] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5453–5462, 2021. 2

[30] Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *CVPR*, 2022. 2

[31] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5

[32] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (ToG)*, 40(6):1–18, 2021. 2

[33] Youjia Zhang, Teng Xu, Junqing Yu, Yuteng Ye, Yanqing Jing, Junle Wang, Jingyi Yu, and Wei Yang. Nemf: Inverse volume rendering with neural microflake field. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22919–22929, 2023. 2

RNG: Relightable Neural Gaussians

Supplementary Material

We propose RNG, a novel relightable asset with neural Gaussians. Without assumptions in the shading model and geometry types, we enable the relighting for both fluffy objects and surface-like scenes. In this supplementary material, we provide extra quality validation and metrics in Sec. 6, and discuss the choice in hybrid optimization and the network size in Sec. 7.

6. Additional validation

In Fig. 9, we visualize the obtained depth maps and shadow cues of our model of both real and synthetic objects. The depth map and shadow cues match well with our renderings and the ground truth. Our forward-deferred optimization strategy provides us with high-quality geometries, and together with reasonable shadow information, our model predicts close results to the reference.

In Table 3, we compare our neural appearance model to the vanilla 3DGS with SHs. We run both methods on datasets rendered with environment lighting and compare the NVS quality. Since the shadow cue and depth refinement MLP are disabled in our method, we only run forward shading for our method. Overall, our neural radiance representation provides more capacity and power in various scenes.

In Fig. 13, we demonstrate the effectiveness of our shadow cue by showcasing an example where the shadow quality dominates. We observe a significant quality improvement in the shadow by adding shadow cues into our model.

In Fig. 10, we move the light source towards and away from the object, showing the different lighting effects. Thanks to the shadow cue, our model shows robustness under different light conditions and can produce reasonable light effects.

In Fig. 15, we show the relighting results of RNG under moving point lights. Each column in the figure shows a different light direction. Our model can render scenes under novel lights with realistic appearance and high-quality details, and can properly model the self-shadowing effects. We suggest the reader refer to the supplementary video for more validation.

7. Additional discussion

The choice in hybrid optimization. To improve the shadow quality and avoid blurry artifacts, we suggest a deferred shading process, regularizing the appearance of shadows in the image space. As shown in Fig. 11, the appearance of shadows is not obtained by blending Gaus-

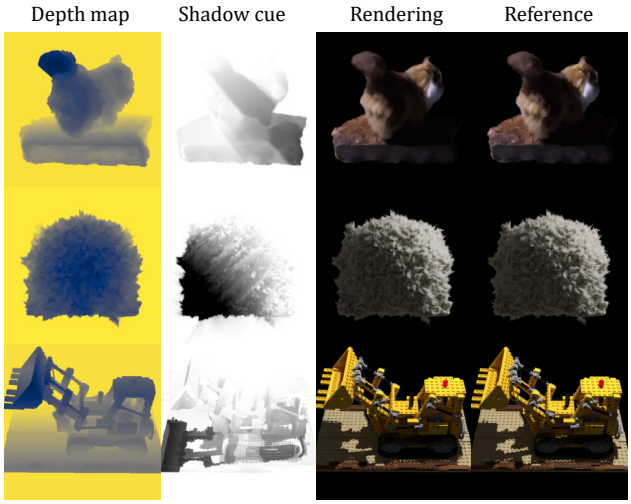


Figure 9. The visualization of depth maps and shadow cue maps of our model for different objects. The two-stage hybrid optimization strategy provides clear and accurate geometries, and the shadow cue also correctly reflects the visibility information.

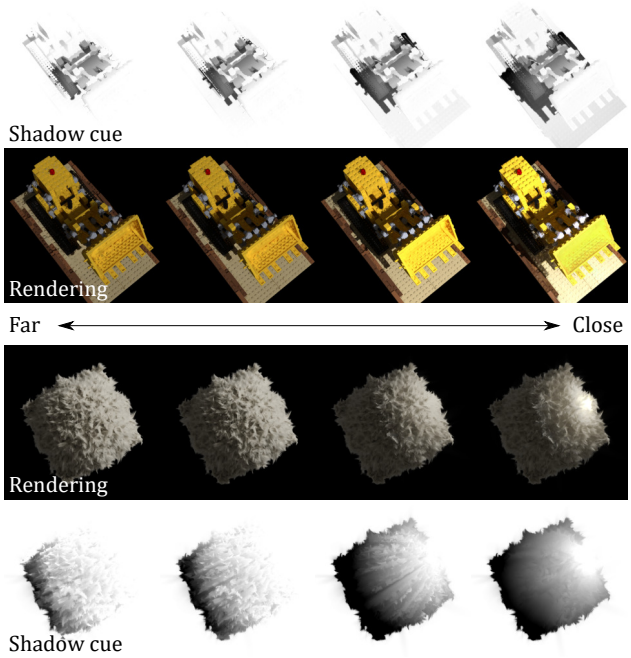


Figure 10. The renderings and visualization of shadow cues when moving the point light towards and away in the scene. The shadow cues correctly reflect the movement of the light source, and our model produces plausible renderings.

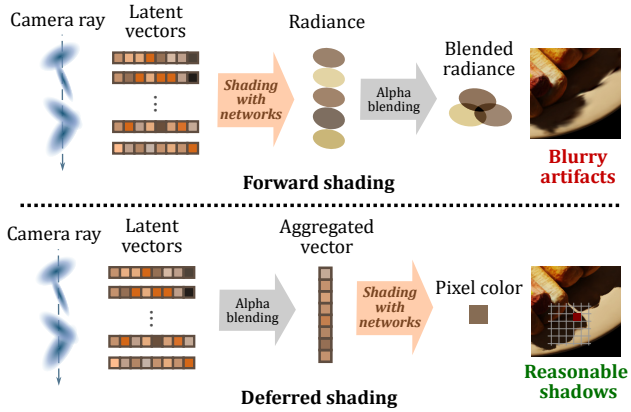
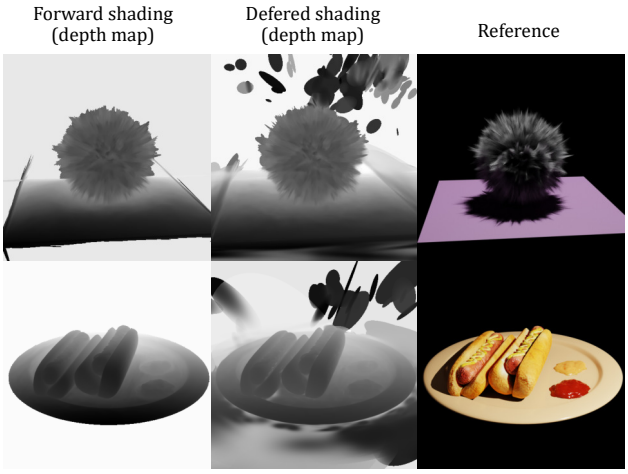


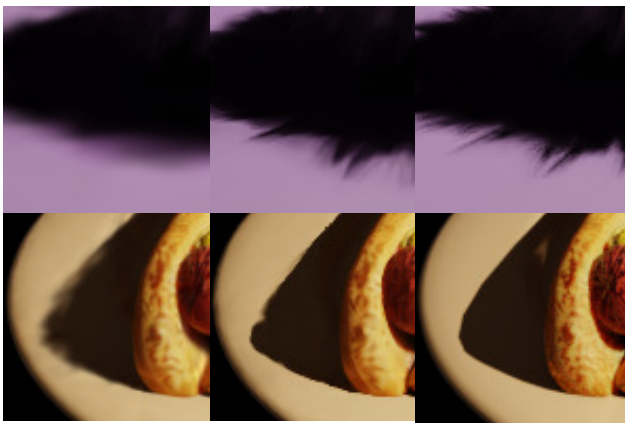
Figure 11. The difference in producing shadows between forward shading and deferred shading. For comparison, alpha blending usually leads to wrong and blurry shadows, while deferred shading provides more capability and flexibility to produce plausible colors for the blended image space features.

sians, but directly decided in the image space, avoiding the artifacts brought by the blending operation. However, according to our observation, forward shading produces better geometry, while deferred shading leads to outliers and floaters. We show such observations in Fig. 12. Therefore, we suggest a two-stage hybrid optimization strategy in the end, preserving both the geometry and shadow qualities.

Network sizes. In Fig. 14, we show the prediction accuracy with varying sizes of the neural Gaussian decoder. We use FURBALL for example, since this scene includes both complex appearance and shadows. All metrics are normalized so that higher values indicate better performance for ease of comparison. The variants are tagged by the number of hidden units and hidden layers, and our choice is $(256, 4)$. Our choice yields the best results among all variants, balancing between quality and computational complexity.



Stage 1: deferred shading vs. forward shading



Stage 2: deferred shading vs. forward shading

Figure 12. The depth/shadow cue visualization and rendering comparison between forward and deferred shading. Forward shading produces better geometry, while deferred shading produces better shadows.

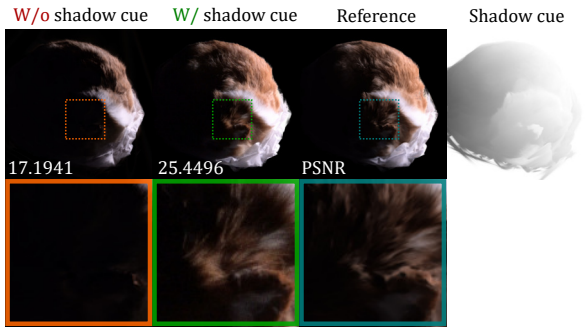


Figure 13. The quality difference at shadow regions with/without the shadow cues. With the shadow cues applied, the network can further improve the quality of dark pixels, as well as provide a clearer boundary for the shadow region.

Table 3. NVS Comparison of our neural appearance model and the vanilla SH-based 3DGS under static environment lighting. We provide (from left to right) PSNR(↑), SSIM (↑) and LPIPS (↓) for comparison, and the prevailing results are marked as **bold**. Our neural radiance representation is more flexible and powerful than the SHs in terms of NVS quality. Note that in this case, the shadow cue and depth refinement MLP are disabled in our method.

Scene	Vanilla 3DGS	Ours (forward shading only)
Armadillo	45.8581 | 0.9959 | 0.0023	**49.2875** | **0.9976** | **0.0009**
CupPlane	43.1947 | 0.9957 | 0.0021	**47.3267** | **0.9974** | **0.0010**
Ficus	36.9734 | 0.9937 | 0.0038	**39.8921** | **0.9964** | **0.0019**
Flowers	36.0157 | 0.9918 | 0.0049	**37.2329** | **0.9941** | **0.0036**
HairBlue	38.5114 | 0.9766 | 0.0197	**39.5742** | **0.9811** | **0.0146**
Hotdog	35.3799 | 0.9941 | 0.0047	**42.6534** | **0.9972** | **0.0015**
Lego	42.0764 | 0.9962 | 0.0021	**44.9111** | **0.9981** | **0.0009**
Average	39.7157 | 0.9920 | 0.0057	**42.9826** | **0.9946** | **0.0035**

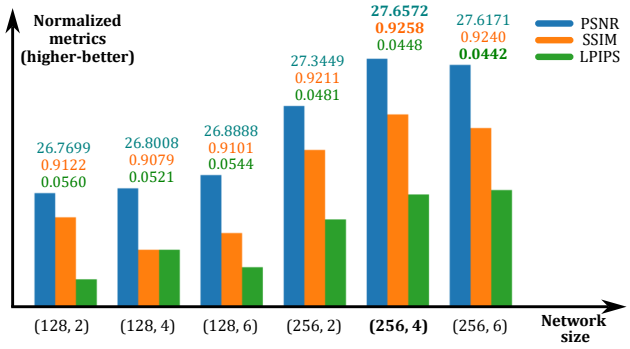


Figure 14. The comparison of variants with different network sizes. We test on FURBALL dataset. All metrics are normalized to be higher-better, and the best values of each metric are marked as **bold**. Our chosen configuration $(256, 4)$ achieves the balance between quality and complexity.

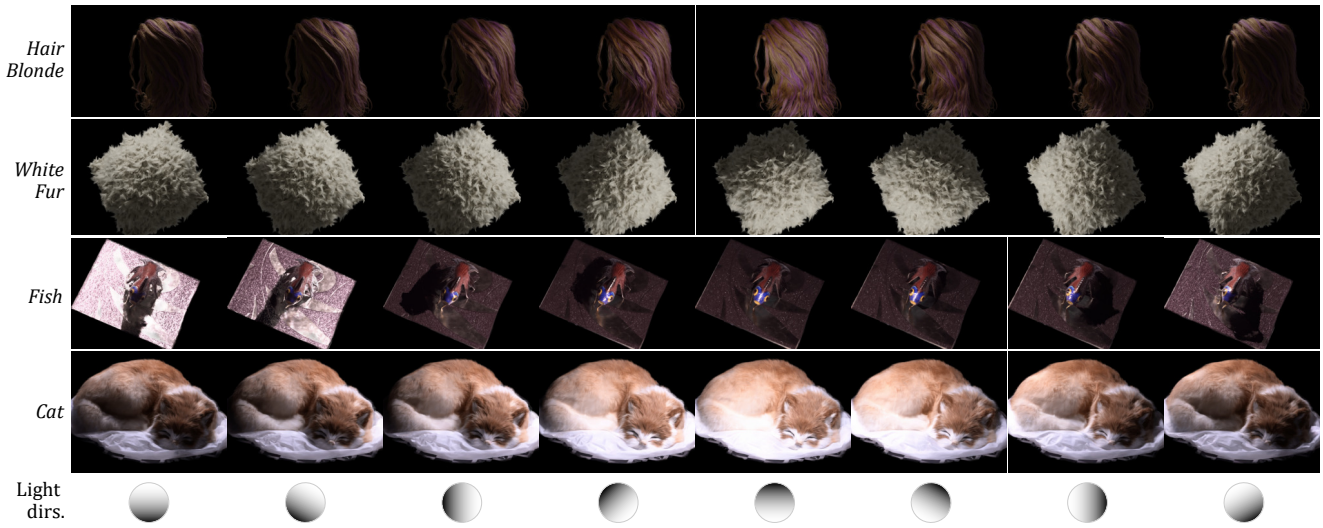


Figure 15. The relighting results of RNG. Each column shows a different point light direction. Our model can render scenes in novel views and lights with realistic appearance with high-quality details, and can properly model the self-shadowing effects.