To appear in the Proceedings of the 24th IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2025)

Real-Time Physically-Based Relighting and Composition of Radiance Fields with Proxy Meshes

Yang Xu Jinyang Bo Junfeng Wang He Tian Yuhe Zhang* Kang Li Guohua Geng

Northwest University, China

94 fps 96 fps

Figure 1: Relighting and composition results of our method in the GARDEN scene represented by 3DGS. Two mesh models, BUNNY and TEAPOT, are inserted into the scene. Two extracted radiance fields, the vase from the GARDEN and the statue from the FAMILY, are also inserted into the scene and relit. In the right image, the entire scene is also relit by an environment map (UFFIZI GALLERY). Our method can simulate a full global illumination solution and capture the inter-object effects such as soft shadows and glossy interreflections at real-time frame rates.

ABSTRACT

Radiance fields, such as neural radiance fields (NeRFs) and 3D Gaussian splatting (3DGS), are the new primitives to represent 3D scenes. Relighting and composition of radiance fields are critical for modeling the complex 3D world in computer graphics. However, it is difficult to relight and composite radiance fields because traditional physically-based rendering techniques, such as path tracing, cannot be directly applied to radiance fields. We propose a physically-based relighting and composition method for radiance fields with proxy meshes. A unified framework is presented to enable us to use radiance fields as the traditional assets in computer graphics. We generate proxy meshes of the radiance fields by reconstructing the geometries of the scenes using Gaussian-based surface reconstruction and the materials using physically-based differentiable rendering. We leverage differential rendering, which is previously used in augmented reality (AR) and mixed reality (MR), to evaluate the radiance change on the proxy meshes introduced by the changing lighting condition, the inserted radiance fields, or the inserted mesh models. Proxy meshes can help us utilize hardware-accelerated ray tracing to perform real-time path tracing. Experimental results show that our method outperforms the baselines in terms of relighting performance and can achieve photorealistic relighting and composition of radiance fields in real-time.

Index Terms: Radiance field, relighting, scene composition, differential rendering, differentiable rendering, path tracing, Gaussian splatting.

*Corresponding author: zhangyuhe0601@nwu.edu.cn

1 INTRODUCTION

In recent years, radiance fields, such as NeRFs [2, 3, 34, 36] and 3DGS [24], have gained much attention in the computer graphics and computer vision communities. As novel view synthesis methods, radiance fields can be utilized to directly represent a scene using the outgoing radiance instead of explicitly modeling the scene properties such as geometry, materials, and light sources, which improves the reconstruction quality, especially for the real world because the scene properties of a real-world scene cannot be accurately estimated.

If we regard radiance fields as a new type of 3D asset in computer graphics, relighting and composition of radiance fields are necessary because one scene may consist of many different models, including radiance fields and even traditional mesh models, and the scene itself can also be represented by mesh models. However, realistic relighting and composition of radiance fields is difficult because traditional photorealistic rendering techniques such as path tracing cannot be directly applied to radiance fields or the efficiency is very low. Many attempts have been made to relight and composite radiance fields. But most of them neglect the inter-object effects, such as soft shadows and interreflections, which leads to a lack of realism [11]. R3DG [13] can produce shadow effects but cannot render interreflections, PRTGS [16] only supports low-frequency indirect illumination, and IRGS [15] cannot work with unbounded scenes. Additionally, most existing relighting and composition methods are specially designed for a specific type of radiance field (typically a variant of 3DGS) and cannot generalize for all types of radiance fields, including new types in the future.

Our goal consists of four aspects: (1) Relight radiance fields themselves with HDR environment maps; (2) composite multiple radiance fields into a scene; (3) insert traditional mesh models into the scene represented by a radiance field; (4) insert radiance fields into a scene represented by traditional polygonal meshes. We propose a unified rendering pipeline to put all these tasks together, which enables us to use radiance fields as the traditional assets

in computer graphics, such as polygonal meshes. We borrow the idea of differential rendering [10, 33] from AR/MR to compute the change in outgoing radiance introduced by the changing lighting condition, the inserted radiance fields, or the inserted mesh models. We generate proxy meshes of radiance fields through GS-based surface reconstruction and physically-based differentiable rendering for material estimation. With proxy meshes, we can leverage the hardware-accelerated ray tracing of modern GPUs to perform path tracing at real-time frame rates. Experiments demonstrate that our method can relight a radiance field with a new environment map, seamlessly integrate radiance fields or mesh models into a scene represented by another radiance field, and insert radiance fields into a scene represented by meshes at real-time frame rates.

The contributions of our work are summarized as follows:

- A unified framework based on differential rendering on proxy meshes for relighting and composition of radiance fields.
- A real-time rendering pipeline based on hardware-accelerated differential path tracing to relight and composite radiance fields.
- A two-stage inverse rendering pipeline based on 3DGS and physically-based differentiable rendering to generate proxy meshes for radiance fields.

Our method can work in the unbounded scenes without providing masks of the interested objects. Our method is independent of the training process and the representation of a radiance field, which means that different types of radiance fields can be directly utilized in our method.

2 Related Works

Radiance Fields NeRF [34] learns a multi-layer perceptron (MLP) representation of the scene through differentiable volumetric rendering for novel view synthesis. Instant-NGP [36] accelerates training and rendering of NeRFs with fully-fused MLPs and multiresolution hash encoding. Mip-NeRF 360 [2] uses a non-linear scene parameterization, online distillation, and a distortion-based regularizer to support unbounded scenes. Zip-NeRF [3] integrates the techniques from Mip-NeRF 360 and Instant-NGP. 3DGS [24] learns a 3D Gaussian representation of the scene through differentiable splatting. HiFi4G [21] introduces a compact 4D Gaussian representation for human performance rendering. Mip-Splatting [59] introduces a 3D smoothing filter and a 2D Mip filter to eliminate the artifacts of 3DGS. Scaffold-GS [29] utilizes anchor points to distribute local 3D Gaussians and dynamically predicts their attributes based on the viewing direction and distance within the frustum. 2DGS [19] collapses 3D Gaussians into a set of 2D oriented planar Gaussian disks to provide view-consistent geometry. 3DGRT [35] builds proxy geometries and a bounding volume hierarchy (BVH) for 3D Gaussians to support hardware-accelerated ray tracing.

Relighting of Radiance Fields NeRV [45] trains MLPs to predict volume density, normal, and material parameters along with visibility and termination depth in any direction at each location to enable relighting. NeRD [6] decomposes shape, SVBRDFs, and illumination represented by spherical Gaussians (SG) using a NeRF-like coordinate-based neural representation to enable relighting. Neural-PIL [7] replaces SG with a pre-integrated lighting network that can convert costly light integration during rendering into a simple network query. Zhang et al. [64] trains MLPs to predict indirect illumination and visibility of direct illumination to recover SVBRDFs and environment light sources. NeRF-OSR [44] learns a neural scene representation that decomposes spatial occupancy, illumination, shadowing, and diffuse albedo from outdoor data captured at different viewpoints and illuminations to enable relighting. SOL-NeRF [47] decomposes outdoor scenes into geometry, reflectance, and lighting. A hybrid lighting representation composed of SG and spherical harmonics (SH) is introduced to estimate shadows of skylights. TensoIR [23] introduces an inverse rendering method based on tensor factorization to enable relighting of TensoRF. Visibility and indirect lighting can be computed online to provide second-bounce shading effects. NRHints [60] models both the local and indirect radiance at each point by a relightable radiance MLP that leverages shadow and highlight hints to model high-frequency light transport effects. However, the NeRF-based methods cannot achieve real-time relighting. GS^3 [4] trains spatial and angular Gaussians to enable relighting, which replaces SH with a Lambertian and a mixture of angular Gaussians for each spatial Gaussian. The self-shadow values are captured by splatting all spatial Gaussians towards the light source, and global illumination is captured by using an MLP to add an RGB tuple for each spatial Gaussian. RNG [12] uses neural Gaussians to achieve faster training and rendering. However, NRHints [60], GS^3 [4], and RNG [12] use point-lit input images, which limits their applications. R3DG [13] learns normals, BRDFs, and incident lights and performs shading for each 3D Gaussian. The shaded colors of all Gaussians are alpha-blended to enable relighting, and the visibility is computed by ray tracing on the Gaussians. PRTGS [16] precomputes radiance transfer for each 3D Gaussian to enable real-time relighting but only supports soft shadows and diffuse interreflections due to the low-frequency nature of the SH basis. 3DGS and 2DGS can be relit by performing deferred shading on the rendered depth and normal maps (G-buffer) [22, 27, 11, 9, 15, 54]. GS-IR [27] models indirect lighting with baked occlusion represented by SH, but fails to model high-frequency specular indirect illumination. GI-GS [9] performs ray marching on the G-buffer to compute indirect illumination. IRGS [15] employs 2D Gaussian ray tracing to directly query the incident indirect radiance and visibility. However, the radiance of each Gaussian becomes invalid as the illumination changes, which requires the split-sum approximation to query indirect radiance when performing relighting. Poirier-Ginter et al. [39] propose a radiance field relighting method based on diffusion models, which fine-tunes a diffusion model conditioned on the dominant lighting direction to augment a single-illumination dataset to a multi-illumination one and then train a multi-illumination 3DGS on it. But the multi-illumination dataset of the same scene is required to fine-tune the model.

Composition of Radiance Fields NeRFs can be composited by learning a compositional NeRF [48, 50, 55] or generating neural depth fields that quickly determine the spatial relationship between objects by allowing direct intersection of rays and implicit surfaces [14], but the illumination change and inter-object effects are neglected or only hard shadows are considered [14]. Multiple 3D Gaussian point clouds can be easily composited by putting all Gaussians together, but the illumination difference and inter-object effects should be further considered to enhance realism. Some radiance field relighting methods can also composite radiance fields [11, 13, 16, 15].

Inserting Mesh Models into Radiance Fields Wang et al. [51] learns a neural intrinsic field of the scene and converts it into an explicit mesh. Primary rays are traced on the neural field, and the secondary rays are traced on the mesh to enable relighting and insertion of mesh models with shadows. Qiao et al. [41] inserts mesh models into a NeRF using a hybrid rendering pipeline combining the light transport equations for both NeRF and meshes. Ye et al. [58] precompute spherical signed distance fields for the inserted mesh models to render shadows cast on a NeRF. However, they suffer from low rendering efficiency or simplification in rendering. 3DGRT [35] and 3DGUT [53] can insert mesh models into a 3D Gaussian radiance field, but only support simple lighting effects such as hard shadows and mirror reflections/refractions because in-

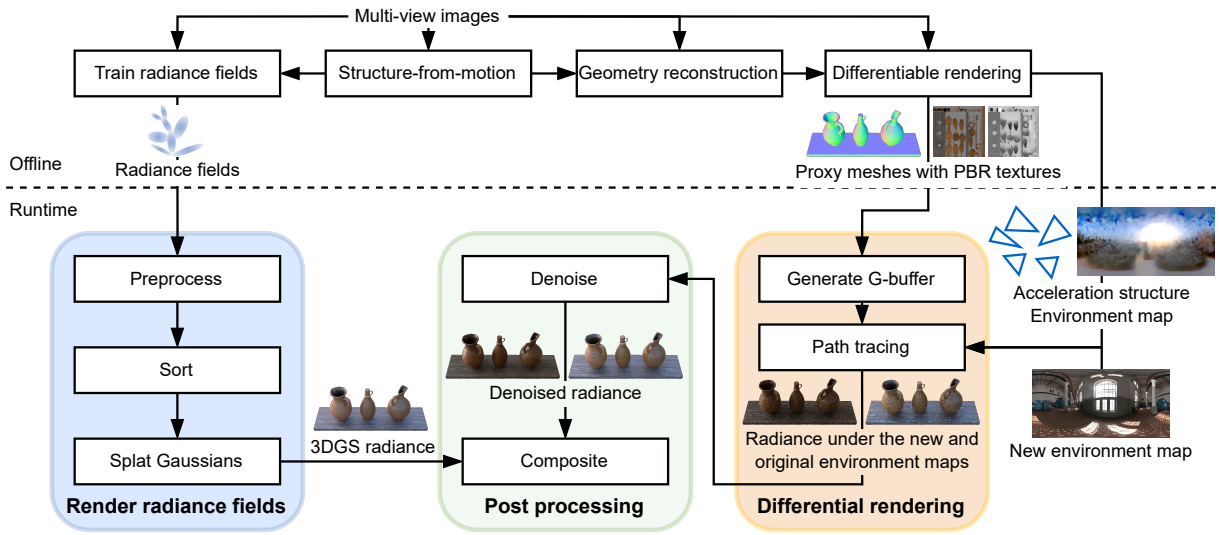


Figure 2: Overview of our method. In the offline part, we generate the proxy meshes with PBR textures, acceleration structures, and environment maps through physically-based differentiable rendering. In the runtime part, we compute the outgoing radiance under the new environment map and the original environment map of the inserted radiance field to relight it.

verse rendering is not performed and the intersection efficiency is not high enough.

Physically-Based Differentiable Rendering Physically-based differentiable rendering on mesh models has been widely studied to achieve inverse rendering. PhySG [62] represents specular BRDFs and environment light sources using mixtures of SGs to efficiently simulate light transport. Zhao et al. [38] reconstructs lighting and material parameters using differentiable rendering with a texture-space sampling scheme. Luan et al. [30] employs physically-based differentiable rendering with several shape and material priors to reconstruct geometry and SVBRDFs from multi-view images. Munkberg et al. [37] employs a differentiable rasterizer to learn geometries, materials, and lighting from images. Hasselgren et al. [17] introduces a differentiable Monte Carlo renderer incorporating multiple importance sampling (MIS) and denoising. NeILF [56] introduces neural incident light fields to handle occlusions and indirect lights and optimizes material and lighting through differentiable rendering. NeILF++ [61] unifies the incident and outgoing light fields through inter-reflections between surfaces to reconstruct scene geometries and materials. Sun et al. [46] combines neural-based object reconstruction with physically-based differentiable rendering to reconstruct object shape, material, and illumination. In this work, we use physically-based differentiable path tracing to optimize the SVBRDFs and the environment map of a proxy mesh.

Differential Rendering Differential rendering proposed by Debevec [10] is a rendering technique for object insertion in AR/MR. Kán et al. [26] introduces differential irradiance caching, which uses one-pass differential rendering that evaluates mixed and real irradiances together in a single pass by using multiple ray types. Mehta et al. [33] introduces a two-mode path tracing that performs mesh-based ray tracing for virtual objects and screen-space ray tracing for real objects. Rhee et al. [42] employs differential rendering to seamlessly composite virtual objects into a 360° video. Rohmer et al. [43] introduces several differential rendering techniques based on ray tracing, including environment map sampling, distance impostor tracing, and voxel cone tracing. Ma et al. [31] introduces a neural compositing method that leverages convolutional neural networks to composite rendering layers of virtual objects with real images. In this work, we use differential path tracing to compute the radiance change caused by the changing lighting or the inserted radiance fields/mesh models.

3 Overview

In this work, we generate proxy meshes along with the SVBRDFs and HDR environment maps for the radiance fields to support traditional physically-based rendering. We use triangular meshes to present the proxy meshes, which benefit from the hardware-accelerated ray tracing of modern GPUs. Based on the proxy meshes with SVBRDFs and environment maps, we use differential path tracing to compute the radiance change introduced by the new lighting condition or the inserted radiance fields to relight the radiance fields and simulate the inter-object effects when compositing multiple radiance fields.

As illustrated in Fig. 2, our method is composed of offline and runtime parts. In the offline part, we generate the proxy meshes for radiance fields along with the SVBRDFs and HDR environment maps using physically-based differentiable rendering. In the runtime part, we perform physically-based relighting and composition of radiance fields at real-time frame rates.

In the runtime part, our real-time rendering pipeline includes three stages. In the first stage, we render the radiance fields. In the second stage, we perform physically-based differential path tracing in a unified framework. And post processing is conducted in the last stage to denoise and blur the rendering results.

4 Proxy Mesh Generation

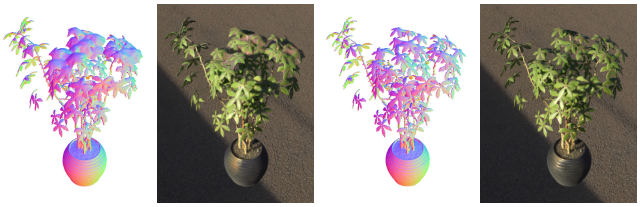
We use a two-stage inverse rendering pipeline to generate the proxy mesh for a radiance field.

4.1 Geometry Reconstruction

Multi-view images of the scene are used to reconstruct the triangular mesh via planar-based Gaussian splatting reconstruction representation (PGSR) [8]. To handle strong specular reflections, we leverage the monocular normal prior from StableNormal [57] to supervise the rendered normal in PGSR using the following loss function:

$$\mathcal{L}_{\text{normal}} = \sum (1 - \mathbf{n}_{\text{render}} \cdot \mathbf{n}_{\text{mono}}), \tag{1}$$

where $\mathbf{n}_{\text{mono}}$ is the estimated normal prior from StableNormal, and $\mathbf{n}_{\text{render}}$ denotes the rendered normal via alpha blending.



w/o $\mathcal{L}_{\mathrm{O}}$ w/ $\mathcal{L}_{\mathrm{O}}$

Figure 3: Comparison of the reconstructed geometries and the final relighting results without and with the binary cross-entropy loss.

For the scenes with masks, we follow R3DG [13] and IRGS [15] to constrain the geometry using a binary cross-entropy loss

$$\mathcal{L}_{\mathrm{O}} = -M \log O - (1-M)\log(1-O), \tag{2}$$

where M is the object mask and O is the accumulated opacity. The reconstructed geometries and the final relighting results w/o and w/ $\mathcal{L}_{\mathrm{O}}$ are shown in Fig. 3.

Since the triangular meshes are used for ray tracing in the rendering pipeline, we decimate the mesh to reduce the triangle count. A texture atlas is generated for the mesh to enable texture mapping in the following differentiable rendering stage to reconstruct the SVBRDFs.

4.2 Environment Map and SVBRDF Estimation

Since differential rendering is used in our rendering pipeline, the light source and material of a scene should be acquired along with the geometry. We model the light source of a scene as an HDR environment map, and the materials of a scene are modeled using SVBRDFs represented by the Disney principled BRDF [18]. Once the geometry of a scene is obtained, we estimate the light sources and the material of the scene using physically-based differentiable rendering. We optimize the HDR environment map and the spatially-varying base color and roughness simultaneously. We employ a loss function as follows to optimize these parameters

$$\mathcal{L} = \mathcal{L}_{\mathrm{render}} + \lambda_{\mathrm{basecolor}}\mathcal{L}_{\mathrm{basecolor}} + \lambda_{\mathrm{rough}}\mathcal{L}_{\mathrm{rough}} + \lambda_{\mathrm{light}}\mathcal{L}_{\mathrm{light}}, \tag{3}$$

where $\mathcal{L}_{\mathrm{render}}$ is the rendering loss given by the L1 losses between the target image I_j and the rendered image $\hat{I}_j$ using the estimated parameters

$$\mathcal{L}_{\mathrm{render}} = \sum_{j=1}^{N} \left\| I_j - \hat{I}_j \right\|_1, \tag{4}$$

$\mathcal{L}_{\mathrm{basecolor}}$, $\mathcal{L}_{\mathrm{rough}}$, and $\mathcal{L}_{\mathrm{light}}$ are the regularization losses of base color, roughness, and environment light source. These three regularization losses are explained in detail as follows:

Regularization Spatially-varying base color and roughness are difficult to optimize even with the known geometry because ambiguity exists between the base color and roughness. Additionally, the physicially-based differentiable renderer we used can produce Monte Carlo noise, which results in noisy base color and roughness maps, and the optimized environment map can also be noisy due to Monte Carlo sampling. To obtain smooth base color, roughness, and environment maps, we regularize the base color, roughness, and environment maps using total variation losses. Taking the roughness as an example, we define the total variation loss of the roughness map $r(x,y)$ as

$$\mathcal{L}_{\mathrm{rough}} = \sum_{i,j} [|r(i+1,j) - r(i,j)| + |r(i,j+1) - r(i,j)|], \tag{5}$$

where $r(i,j)$ denotes the value of the (i,j)-th texel of the roughness map.

5 Relighting and Composition of Radiance Fields

5.1 Differential Rendering

Our method is inspired by differential rendering, which is introduced by Debevec [10] and previously applied in AR and MR to seamlessly integrate virtual objects into the images of the real scenes. The geometry, materials, and light sources of the real scene are modeled in advance. At runtime, we render the modeled real scene to obtain a rendering result L_{real}. Then we insert the virtual objects into the modeled real scene and render the mixed scene to acquire another rendering result L_{mixed}. The composition result can be obtained by superposing the difference between the two rendering results $\Delta L = L_{\mathrm{mixed}} - L_{\mathrm{real}}$, which represents the change in the outgoing radiance introduced by the inserted virtual objects onto the image of the real scene. Finally, the outgoing radiance of the final composition image L_{final} can be expressed as

$$L_{\mathrm{final}} = M \odot L_{\mathrm{mixed}} + (1-M) \odot (L_{\mathrm{cam}} + L_{\mathrm{mixed}} - L_{\mathrm{real}}), \tag{6}$$

where L_{cam} is the radiance from the camera image of the real scene, M is the mask of the inserted virtual objects, and $\odot$ denotes the element-wise product.

Differential rendering can reduce the impact of the material estimation error of the real scene because the estimated materials only affect the change in illumination introduced by the virtual objects. Since the radiance change is lower than the outgoing radiance itself, the error can be suppressed.

5.2 Relighting of Radiance Fields

To relight an object, we can obtain the geometry and materials of the object and render it under another lighting condition. However, it is difficult to represent the materials of an object using a radiance field, and physically-based rendering of the radiance field under another lighting condition in real-time is also hard to achieve.

Fortunately, differential rendering can provide us with an approach to relight an object. We can compute the change in illumination between different lighting conditions and superpose the change on the original radiance field to relight a radiance field. Given a radiance field $\mathcal{R}$ and its proxy mesh $\mathcal{P}$, along with the new lighting condition $\mathcal{E}'$ and the estimated original light condition of the radiance field $\hat{\mathcal{E}}$, the relit radiance of $\mathcal{R}$ under $\mathcal{E}'$ can be approximated by

$$\hat{L}^{\mathcal{E}'}_{\mathcal{R}} = L^{\mathcal{E}}_{\mathcal{R}} + L^{\mathcal{E}'}_{\mathcal{P}} - L^{\hat{\mathcal{E}}}_{\mathcal{P}}, \tag{7}$$

where $L^{\mathcal{E}}_{\mathcal{R}}$ is the outgoing radiance of $\mathcal{R}$ under the original lighting condition $\mathcal{E}$, $L^{\mathcal{E}'}_{\mathcal{P}}$ is the rendering result of $\mathcal{P}$ under $\mathcal{E}'$, and $L^{\hat{\mathcal{E}}}_{\mathcal{P}}$ is the rendering result of $\mathcal{P}$ under $\hat{\mathcal{E}}$. From Eq. (7) we can see that $\mathcal{E}$, $L^{\mathcal{E}'}_{\mathcal{P}}$ and $L^{\hat{\mathcal{E}}}_{\mathcal{P}}$ should be rendered to compute the difference, and the proxy mesh along with the SVBRDFs and the original lighting condition of the radiance field should be modeled and estimated to relight the radiance field under a new lighting condition.

5.3 Composition of Radiance Fields

As for the composition of radiance fields, we consider three different scenarios as follows:

Inserting Mesh Models into a Radiance Field Scene Inserting mesh models into a scene represented by a radiance field can be seen as an AR scenario that inserts mesh models into an image. The outgoing radiance from the radiance field needs to be adjusted according to the inserted models, which can be achieved by differential rendering. Given N mesh models $\{\mathcal{M}_i\}(i=1,2,\ldots,N)$ to be inserted into the radiance field $\mathcal{R}$, the composition result $\hat{L}^{\mathcal{E}}_{\mathcal{R}+\sum\mathcal{M}_i}$ can be expressed as

$$\hat{L}^{\mathcal{E}}_{\mathcal{R}+\sum\mathcal{M}_i} = M \odot L^{\hat{\mathcal{E}}}_{\mathcal{P}+\sum\mathcal{M}_i} + (1-M) \odot (L^{\mathcal{E}}_{\mathcal{R}} + L^{\hat{\mathcal{E}}}_{\mathcal{P}+\sum\mathcal{M}_i} - L^{\hat{\mathcal{E}}}_{\mathcal{P}}), \tag{8}$$

where M is the mask of $\{\mathcal{M}_i\}$, $L^{\hat{\mathcal{E}}}_{\mathcal{P}+\sum\mathcal{M}_i}$ is the rendering result of the proxy mesh $\mathcal{P}$ of $\mathcal{R}$ along with $\{\mathcal{M}_i\}$ under the estimated lighting condition $\hat{\mathcal{E}}$, and $L^{\hat{\mathcal{E}}}_{\mathcal{P}}$ is the rendering result of $\mathcal{P}$ under $\hat{\mathcal{E}}$. From Eq. (8) we can see that $L^{\hat{\mathcal{E}}}_{\mathcal{P}+\sum\mathcal{M}_i}$ and $L^{\hat{\mathcal{E}}}_{\mathcal{P}}$ should be computed to adjust the outgoing radiance of $\mathcal{R}$ to seamlessly integrate $\{\mathcal{M}_i\}$ into $\mathcal{R}$.

Inserting Radiance Fields into a Radiance Field Scene If we insert radiance fields into a scene represented by a radiance field or mesh models, the inserted radiance fields need to be relighted, and the outgoing radiance of the scene needs to be adjusted as well. Given N radiance fields $\{\mathcal{R}_i\}(i=1,2,\ldots,N)$ to be inserted into a radiance field $\mathcal{R}$, the composition result $\hat{L}^{\mathcal{E}}_{\mathcal{R}+\sum\mathcal{R}_i}$ can be computed by

$$\hat{L}^{\mathcal{E}}_{\mathcal{R}+\sum\mathcal{R}_i} = \sum M_i \odot \hat{L}^{\mathcal{E}}_{\mathcal{R}_i} + (1-M)\odot(L^{\mathcal{E}}_{\mathcal{R}} + L^{\hat{\mathcal{E}}}_{\mathcal{P}+\sum\mathcal{P}_i} - L^{\hat{\mathcal{E}}}_{\mathcal{P}}), \quad (9)$$

where M_i is the mask of $\mathcal{R}_i$, M is the mask of $\{\mathcal{R}_i\}$, $L^{\hat{\mathcal{E}}}_{\mathcal{P}+\sum\mathcal{P}_i}$ is the rendering result of the proxy mesh $\mathcal{P}$ of $\mathcal{R}$ along with the proxy meshes $\{\mathcal{P}_i\}$ of $\{\mathcal{R}_i\}$ under the estimated lighting condition $\hat{\mathcal{E}}$, $L^{\hat{\mathcal{E}}}_{\mathcal{P}}$ is the rendering result of $\mathcal{P}$ under $\hat{\mathcal{E}}$, and $\hat{L}^{\mathcal{E}}_{\mathcal{R}_i}$ is the relit result of $\mathcal{R}_i$ under $\mathcal{E}$ that can be computed by

$$\hat{L}^{\mathcal{E}}_{\mathcal{R}_i} = L^{\mathcal{E}_i}_{\mathcal{R}_i} + L^{\hat{\mathcal{E}}}_{\mathcal{P}+\sum\mathcal{P}_i} - L^{\hat{\mathcal{E}}_i}_{\mathcal{P}_i}, \quad (10)$$

where $L^{\mathcal{E}_i}_{\mathcal{R}_i}$ is the outgoing radiance of $\mathcal{R}_i$ and $L^{\hat{\mathcal{E}}_i}_{\mathcal{P}_i}$ is the rendering result of $\mathcal{P}_i$ under the estimated lighting condition $\hat{\mathcal{E}}_i$ of $\mathcal{R}_i$.

Inserting Radiance Fields into a Mesh Scene If we insert radiance fields $\{\mathcal{R}_i\}$ into a scene represented by mesh models $\mathcal{M}$, the composition result $\hat{L}^{\mathcal{E}}_{\mathcal{M}+\sum\mathcal{R}_i}$ is

$$\hat{L}^{\mathcal{E}}_{\mathcal{M}+\sum\mathcal{R}_i} = \sum M_i \odot \hat{L}^{\mathcal{E}}_{\mathcal{R}_i} + (1-M)\odot L^{\hat{\mathcal{E}}}_{\mathcal{M}+\sum\mathcal{P}_i}, \quad (11)$$

where $L^{\hat{\mathcal{E}}}_{\mathcal{M}+\sum\mathcal{P}_i}$ is the rendering result of $\mathcal{M}$ along with the proxy meshes $\{\mathcal{P}_i\}$ of $\{\mathcal{R}_i\}$ under $\hat{\mathcal{E}}$, and $\hat{L}^{\mathcal{E}}_{\mathcal{R}_i}$ is the relit result of $\mathcal{R}_i$ under $\mathcal{E}$ that can be computed by

$$\hat{L}^{\mathcal{E}}_{\mathcal{R}_i} = L^{\mathcal{E}_i}_{\mathcal{R}_i} + L^{\hat{\mathcal{E}}}_{\mathcal{M}+\sum\mathcal{P}_i} - L^{\hat{\mathcal{E}}_i}_{\mathcal{P}_i}. \quad (12)$$

5.4 A Unified Relighting and Composition Equation

We can put Eq. (7) to Eq. (11) into a unified equation to relight and composite radiance fields and mesh models under a new lighting condition $\mathcal{E}'$ as

$$\begin{aligned}\hat{L}^{\mathcal{E}'}_{\mathcal{R}+\sum\mathcal{R}_i+\sum\mathcal{M}_j} = &\sum M_i \odot \hat{L}^{\mathcal{E}'}_{\mathcal{R}_i} + M_{\{j\}} \odot L^{\mathcal{E}'}_{\mathcal{P}+\sum\mathcal{P}_i+\sum\mathcal{M}_j} \\ &+(1-M)\odot(L^{\mathcal{E}}_{\mathcal{R}} + L^{\mathcal{E}'}_{\mathcal{P}+\sum\mathcal{P}_i+\sum\mathcal{M}_j} - L^{\hat{\mathcal{E}}}_{\mathcal{P}}),\end{aligned} \quad (13)$$

where M_i is the mask of $\mathcal{R}_i$, $M_{\{j\}}$ is the mask of $\{\mathcal{M}_i\}$, M is the mask of $\{\mathcal{R}_i\}$ and $\{\mathcal{M}_i\}$, and $\hat{L}^{\mathcal{E}'}_{\mathcal{R}_i}$ is the relit result of $\mathcal{R}_i$ under $\mathcal{E}'$ that can be expressed by

$$\hat{L}^{\mathcal{E}'}_{\mathcal{R}_i} = L^{\mathcal{E}_i}_{\mathcal{R}_i} + L^{\mathcal{E}'}_{\mathcal{P}+\sum\mathcal{P}_i+\sum\mathcal{M}_j} - L^{\hat{\mathcal{E}}_i}_{\mathcal{P}_i}. \quad (14)$$

In our work, we use path tracing, an unbiased rendering technique that can produce photorealistic rendering results, to achieve differential rendering. In differential path tracing, we classify the pixels of the rendered image into three categories: radiance field as a scene, proxy meshes of the inserted radiance fields, and mesh models. Different rendering strategies are applied to different categories, and the implementation details are described in Section 6.2.

However, the textures of a proxy mesh and its corresponding radiance field may be different because the SVBRDFs of the proxy mesh and the radiance field are trained with different methods. If we use Eq. (13) to relight and composite radiance fields, the difference between texture details can overdarken and overbrighten the results in the texture-rich region. In addition, the relighting and composition results are sensitive to the estimation error of SVBRDFs.

Fortunately, Eq. (6) can be rewritten under the assumption that all surfaces are diffuse [33, 52]

$$L_{\text{final}} = M \odot L_{\text{mixed}} + (1-M)\odot(L_{\text{cam}}\frac{E_{\text{mixed}}}{E_{\text{real}}}), \quad (15)$$

where E_{real} and E_{mixed} are the irradiance before and after inserting the virtual objects. We find that Eq. (15) can be also applied to non-diffuse surfaces. Hence, Eq. (6) can be written as

$$L_{\text{final}} = M \odot L_{\text{mixed}} + (1-M)\odot(L_{\text{cam}}\frac{L_{\text{mixed}}}{L_{\text{real}}}), \quad (16)$$

in which we use the radiance ratio instead of the radiance difference to adjust the outgoing radiance of the real scene. With Eq. (16), we can rewrite Eq. (13) as

$$\begin{aligned}\hat{L}^{\mathcal{E}'}_{\mathcal{R}+\sum\mathcal{R}_i+\sum\mathcal{M}_j} = &\sum M_i \odot \hat{L}^{\mathcal{E}'}_{\mathcal{R}_i} + M_{\{j\}} \odot L^{\mathcal{E}'}_{\mathcal{P}+\sum\mathcal{P}_i+\sum\mathcal{M}_j} \\ &+(1-M)\odot(L^{\mathcal{E}}_{\mathcal{R}}\frac{L^{\mathcal{E}'}_{\mathcal{P}+\sum\mathcal{P}_i+\sum\mathcal{M}_j}}{L^{\hat{\mathcal{E}}}_{\mathcal{P}}}),\end{aligned} \quad (17)$$

and Eq. (14) can be rewritten as

$$\hat{L}^{\mathcal{E}'}_{\mathcal{R}_i} = L^{\mathcal{E}_i}_{\mathcal{R}_i}\frac{L^{\mathcal{E}'}_{\mathcal{P}+\sum\mathcal{P}_i+\sum\mathcal{M}_j}}{L^{\hat{\mathcal{E}}_i}_{\mathcal{P}_i}}. \quad (18)$$

Another issue that should be noticed is that since $L^{\hat{\mathcal{E}}}_{\mathcal{P}}$ acts as the denominator in Eq. (17), the result can be invalid when $L^{\hat{\mathcal{E}}}_{\mathcal{P}}$ is close to zero, which can be caused by Monte Carlo noise of path tracing or the near-zero values of one or more channels of the base color. To avoid this problem, we can demodulate the base color and then denoise the numerator and denominator separately before dividing. Demodulating the base color before denoising can also prevent blurring the high-frequency texture details.

5.5 Post Processing

With differential rendering, we can obtain the change in illumination. However, since differential rendering is performed on the proxy meshes, the spatial frequencies of the radiance change are high at the boundaries of the meshes, which leads to aliasing artifacts in the relighting and composition results. To suppress the aliasing, we should reduce the spatial frequency of the radiance change to match the spatial frequency of the radiance fields.

We use a simple approach to reduce the spatial frequency of the radiance change. We just blur the rendering results with a small kernel after denoising and before computing the ratio, most of the artifacts can be suppressed by this blur operation. It should be noted that only the pixels belonging to radiance fields should be blurred, the pixels belonging to the mesh models should be kept unchanged.

6 Real-Time Rendering Pipeline

We implement our real-time rendering pipeline using the Vulkan API with the support of hardware-accelerated ray tracing. As mentioned in Section 3, the real-time rendering pipeline can be divided into three stages: rendering radiance fields, differential rendering, and post processing.

6.1 Radiance Field Rendering

We use 3DGS to represent the radiance field in our implementation because 3DGS can achieve real-time rendering while maintaining high quality novel view synthesis. We do not use PGSR in the proxy mesh generation stage because its novel view synthesis performance is lower than the vanilla 3DGS, which also affects the relight performance in our method. Since our rendering pipeline is built upon a graphics API, we leverage the graphics pipeline instead of the tile-based software rasterizer in the original 3DGS paper [24] to render the 3DGS, which is similar to the 3DGS renderer implemented by Unity [40].

6.2 Differential Rendering

The stage of differential rendering includes two passes: G-buffer generation and differential path tracing. In the G-buffer generation pass, apart from depths, normals, motion vectors, and material parameters, we also store the object indices to identify the category of pixels it belongs to. The G-buffer can also be regarded as the primary hits in the following path tracing pass, and the depth test of mesh models and radiance fields is also accomplished in the G-buffer generation pass using the proxy meshes. In the differential path tracing pass, ray query is performed in the fragment shader to implement differential path tracing. Next event estimation (NEE) is employed to directly sample the environment map, and MIS with the balance heuristic [49] is applied to combine BRDF sampling and environment map sampling, and an alias table is built for the environment map to importance sample it. An individual environment map is associated with a radiance field inserted into the scene, and we use the object indices in the G-buffer to determine which environment map should be used to represent the original lighting condition and determine whether the fragment belongs to a radiance field as the scene, an inserted radiance field, or a mesh model. For the pixels belonging to the mesh models, we only need to perform path tracing once. For the pixels belonging to radiance fields, two path tracings should be conducted to compute the radiance change, and we store the random seeds used by the first path tracing at primary hits and reuse them in the second path tracing, so the variance in direct illumination under the same environment map can be fully eliminated. For the radiance field as a scene, the proxy mesh for G-buffer generation and path tracing is the same. Otherwise, for the radiance field as an object to be inserted into the scene, the proxy mesh for G-buffer generation only contains the object in the scene we want to insert, but the proxy mesh for path tracing (acceleration structure) must be the whole scene because the surrounding geometry may block or bounce light from the environment map. Finally, we demodulate the base color for the following denoising. The pseudocode of our differential path tracing is given in Algorithm 1, where NEE and MIS are omitted for clarity.

Algorithm 1: Differential path tracing for relighting and composition of radiance fields

Input: proxy mesh of the radiance field as a scene $\mathcal{P}$, proxy meshes of the inserted radiance fields $\{\mathcal{P}_i\}$, proxy meshes of the radiance fields where $\{\mathcal{P}_i\}$ are extracted from $\{\mathcal{P}_i^0\}$, inserted mesh models $\{\mathcal{M}_j\}$, new environment map $\mathcal{E}'$, original environment map of the radiance field $\hat{\mathcal{E}}$, original environment maps of the inserted radiance fields $\{\hat{\mathcal{E}}_i\}$, surface point $\mathbf{x}$

Output: outgoing radiance under $\mathcal{E}'$, $\hat{\mathcal{E}}$, and $\{\hat{\mathcal{E}}_i\}$

```
// Render under the new environment map
for k ← 1 to maxBounces do
    sample the BRDF in direction ω_i
    trace ray (x_k, ω_i) against P + ΣP_i + ΣM_j
    if (x_k, ω_i) does not hit the scene then
        accumulate E' in ω_i
        break
    end
end
// Render under the original environment maps
if x does not belongs to {M_j} then
    i ← index of {P_i} that x belongs to
    for k ← 1 to maxBounces do
        sample the BRDF in direction ω_i
        if path vertex x_k belongs to P then
            trace ray (x_k, ω_i) against P
        else
            trace ray (x_k, ω_i) against P_i^0
        end
        if (x_k, ω_i) does not hit the scene then
            if path vertex x_k belongs to P then
                accumulate Ê in ω_i
            else
                accumulate Ê_i in ω_i
            end
            break
        end
    end
end
```

6.3 Post Processing

The post processing stage consists of a denoising pass and a composition pass. In the denoising pass, a cross-bilateral filter with the weight functions considering depth and normal is applied to denoise the two demodulated radiance maps computed by differential path tracing. We remodulate the base color after denoising for the pixels belonging to meshes. In the composition pass, we first apply the spatial Gaussian filter with a small kernel to the two radiance maps and then compute the radiance ratio to modulate the outgoing radiance of radiance fields.

7 Experiments

7.1 Training Details

In the first stage of inverse rendering, the weight for the normal prior loss $\mathcal{L}_{\text{normal}}$ is set to 0.15 and the weight for the binary cross-entropy loss $\mathcal{L}_{\text{O}}$ is set to 0.05 in PGSR. After acquiring the mesh models via PGSR, we decimate the mesh models to 500 k triangles using quadric error metrics, and then we use Blender [5] to generate the texture atlas of the mesh model. Then we use Mitsuba 3 [20], a physically-based differentiable renderer, to optimize the HDR environment map and the SVBRDFs of the scene. In Mitsuba 3, we use the path replay backpropagation integrator with 32 samples per pixel (spp), and the longest path depth is set to infinity. The resolutions of the base color map, roughness map, and environment map are 4090×4096, 2048×2048, and 256×128, respectively. The initial values of the environment map and the base color map are set to 0.5, and the initial roughness is set to 0.8. We randomly pick an image from the training set of images for each iteration, and 5000 iterations are performed in total. We use the Adam optimizer with an initial learning rate of 0.005 for the base color, roughness, and environment map. We set $\lambda_{\text{basecolor}}$ to 0.02 and λ_{rough} to 0.01 in Eq. (3). And λ_{light} in Eq. (3) to 0.001. We use a computer equipped with an Intel Core i7-13700K CPU with 32 GB RAM and an NVIDIA GeForce RTX 4080 GPU with 16 GB VRAM in our experiments to generate proxy meshes and perform real-time rendering.

7.2 Relighting Performance

We use TensoIR [23] and Synthetic4Relight [64] datasets to evaluate the relighting performance of our method. In differential path tracing, 128 spp are used, 64 for BRDF sampling and

Table 1: Quantitative comparison of relighting results on TENSOIR dataset. Our method outperforms all the baselines.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	Time ↓
NeRFactor [63]	23.38	0.908	0.131	>100 h
InvRender [64]	23.97	0.901	0.101	15 h
TensoIR [23]	28.58	0.944	0.081	5 h
GS-IR [27]	24.37	0.885	0.096	<1 h
GI-GS [9]	24.70	0.886	0.106	<1 h
IRGS [15]	30.64	0.935	0.076	<1 h
Ours	32.40	0.958	0.049	<1 h

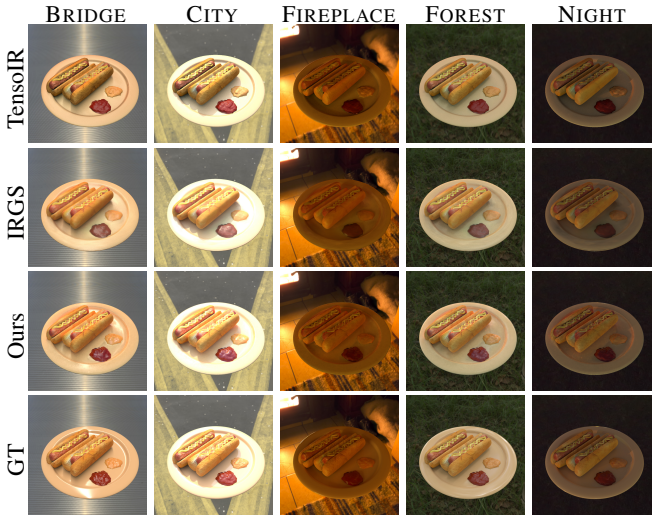


Figure 4: Qualitative comparison of relighting results under different environment maps on TENSOIR dataset.

Table 2: Quantitative comparison of relighting results on SYNTHETIC4RELIGHT dataset. Our method outperforms all the baselines.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	Time ↓
NeRFactor [63]	21.54	0.875	0.171	>48 h
InvRender [64]	28.67	0.950	0.091	14 h
TensoIR [23]	29.69	0.951	0.079	3 h
GS-IR [27]	25.40	0.924	0.083	<1 h
R3DG [13]	32.82	0.967	0.052	<1 h
GI-GS [9]	27.36	0.945	0.070	<1 h
IRGS [15]	34.80	0.963	0.056	<1 h
Ours	35.63	0.976	0.040	<1 h

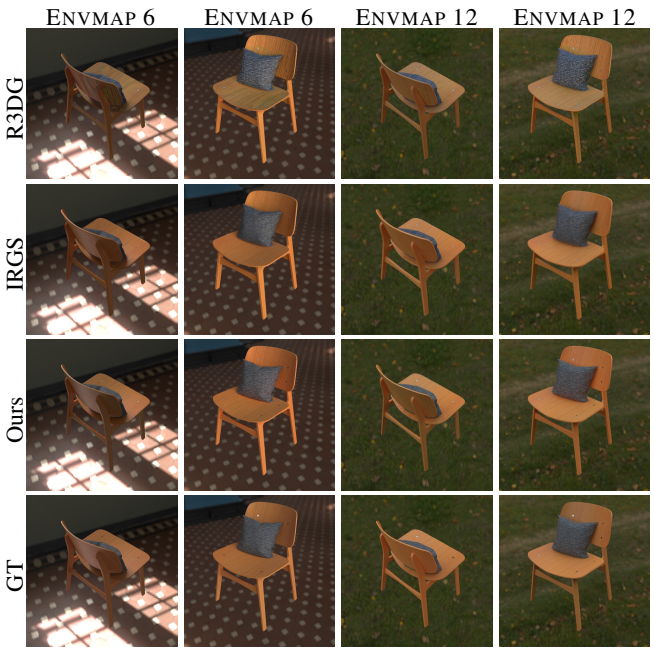


Figure 5: Qualitative comparison of relighting results under different environment maps on SYNTHETIC4RELIGHT dataset.

the other 64 for environment map sampling (NEE). The maximum number of bounces is unlimited, and Russian roulette is applied after 5 bounces to terminate a path. Edge-aware spatial denoising is applied to the rendering results to reduce the noise. The window size of the spatial denoiser is set to 9. To mitigate the inherent ambiguity between the estimated albedo and lighting, we follow the previous works [27, 9, 15] to scale each RGB channel of the base color map by a global scalar according to the ground truth.

The training process takes ∼40 minutes for geometry reconstruction and ∼10 minutes for SVBRDF estimation. We report PSNR, SSIM, and LPIPS metrics and compare them to the inverse rendering baselines based on radiance fields. It should be noted that we use the official implementations of R3DG [13] and IRGS [15] to generate the results, which slightly differ from the reported results in the original papers. The quantitative comparisons in Tab. 1 and Tab. 2 demonstrate that our method has the best relighting performance in terms of all metrics while the training time is acceptable. We also provide the qualitative comparisons on the two datasets in Fig. 4 and Fig. 5. We can see that our method can produce realistic relighting results. Compared to the baselines, the specular reflections of our method are more accurate.

7.3 Real-Time Relighting and Composition

To achieve real-time performance, we use 2-spp path tracing, and temporal denoising with a history length of 20 is applied to reduce the noise. The resolution of all rendered images is 1920×1080. We show the relighting and composition results in the GARDEN, KITCHEN, and ROOM from the MIP-NERF 360 dataset [2], and the TRUCK and FAMILY from the TANKS AND TEMPLES dataset [25].

The training process takes ∼2 hours for geometry reconstruction and 0.5∼2 hours for material estimation, depending on the complexity of a scene and the resolution of images. We extract the vase from the GARDEN and the statue from the FAMILY as the radiance fields to be inserted into scenes represented by radiance fields or mesh models. We also insert two mesh models, BUNNY (144 k triangles) and TEAPOT (15.7 k triangles) [32], into these scenes. Additionally, we use AMAZON LUMBERYARD BISTRO (4 M triangles) [1] as a scene represented by meshes to insert these mesh models and radiance fields into.

Fig. 6 gives the rendering results of the naïve composition and our method in the scenes represented by radiance fields (GARDEN, KITCHEN, ROOM, TRUCK, and FAMILY) and the scene represented by meshes (AMAZON LUMBERYARD BISTRO). Since the naïve composition does not consider the radiance change of inserted radiance fields and the inter-object effects such as the shadows cast on the scenes and the glossy interreflections, all inserted objects seem to float in the air, and the appearances of the inserted radiance fields do not match the scenes. Our method makes the radiance fields relightable and captures the inter-object effects including glossy interreflections.

The timing breakdown of our real-time rendering method in these scenes is provided in Tab. 3. All measured timings are av-

Table 3: Timing breakdown in ms of the real-time rendering pipeline

Scene	Render radiance fields			Differential rendering		Post processing		Total
	Preprocess	Sort	Splat Gaussians	G-buffer	Path tracing	Denoise	Composite	
GARDEN	0.59	0.98	2.59	0.35	3.75	0.72	0.16	9.14
KITCHEN	0.24	0.40	3.29	0.40	3.93	0.75	0.16	9.17
ROOM	0.22	0.23	2.97	0.35	8.23	0.76	0.15	12.91
TRUCK	0.31	0.34	3.12	0.31	3.69	0.57	0.18	8.52
FAMILY	0.26	0.29	2.60	0.26	2.53	0.51	0.17	6.62
BISTRO	0.04	0.15	0.36	0.83	8.55	0.78	0.15	10.86

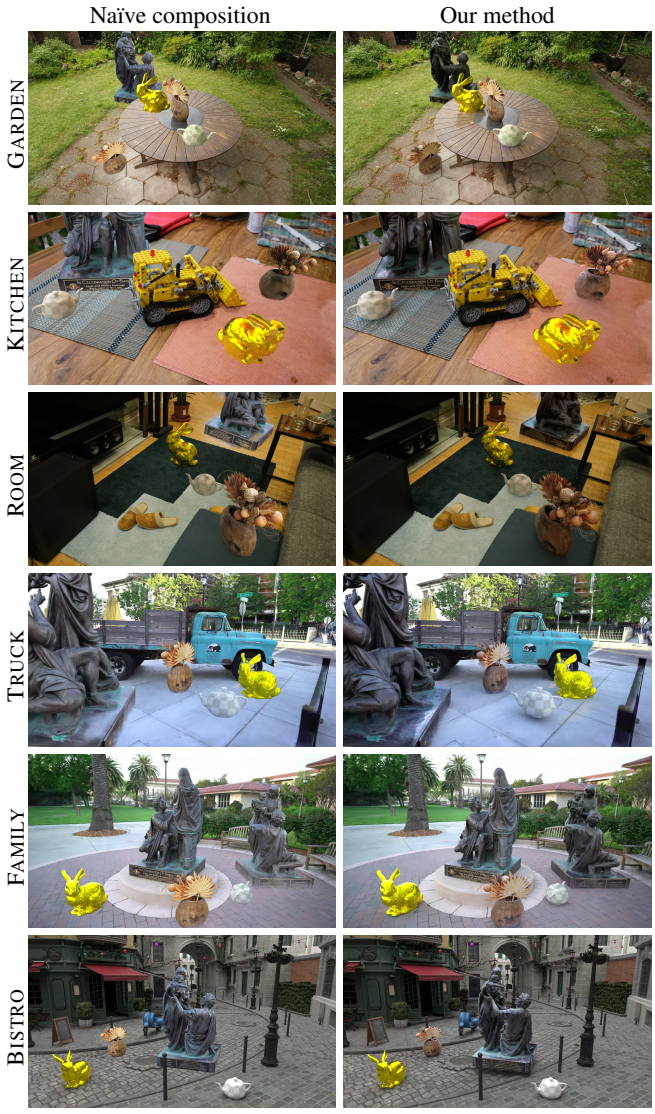


Figure 6: Rendering results in the scenes represented by radiance fields (GARDEN, KITCHEN, ROOM, TRUCK, and FAMILY) and the scene represented by meshes (AMAZON LUMBERYARD BISTRO). Left: Naïve radiance field composition method without considering the inter-object effects and relighting of radiance fields themselves lacks realism. Right: our method, which simulates a full global illumination solution through path tracing, can relight the inserted radiance fields, capture glossy interreflections, and achieve photorealistic rendering.

Table 4: Ablation studies on different components of our method on TENSOIR and SYNTHETIC4RELIGHT datasets.

	TENSOIR [23]			SYNTHETIC4RELIGHT [64]		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Mesh only	31.49	0.937	0.074	34.63	0.955	0.060
Difference	31.78	0.947	0.060	34.66	0.966	0.050
w/o blur	32.19	0.954	0.053	35.42	0.975	0.041
Full	**32.40**	**0.958**	**0.049**	**35.63**	**0.976**	**0.040**

Figure 7: Ablation studies on different components of our method.

eraged over 100 frames. From the timings, we can see that the most time-consuming steps are the splatting of the Gaussians and path tracing. The time to splat Gaussians depends on the number of visible Gaussians in the viewport. Therefore, the BISTRO scene takes less time than other scenes to splat Gaussians because only the vase and the statue are rendered. The ROOM scene takes more time in path tracing than other scenes represented by radiance fields because the environment lighting is partially blocked by the ceiling, which increases the path length in path tracing. The BISTRO scene takes the longest time in G-buffer generation and path tracing because the triangle count is higher than that of any other scene.

7.4 Ablation Study

We conduct ablation studies on different components of our method to evaluate their influences on the relighting performance. The metrics on TENSOIR and SYNTHETIC4RELIGHT datasets are provided in Tab. 4 and the visual comparisons are given in Fig. 7. From Tab. 4 and Fig. 7 we can see that differential rendering can improve the relighting performance compared to only using meshes ("mesh only"). From the comparison of the rendering results using Eq. (13) ("difference") and Eq. (17) ("full"), we can see the overdarkening and overbrightening effects in "difference," and this overdarkening and overbrightening are eliminated in "full", and the quantitative metrics of "full" are also better than those of "difference" in Tab. 4. From the comparison of unblurred ("w/o blur") and blurred ("full") results, we can see that the high-frequency aliasing eliminates after blurring, and the scores also improve in Tab. 4.

Triangle Count We analyze the impact of the triangle count of the proxy mesh on the relighting performance. We decimate the proxy meshes into 4 k, 20 k, 100 k, 500 k, and 2.5 M, respectively. The PSNR, SSIM, LPIPS, and average rendering times are reported

Table 5: Ablation on the triangle count of the proxy mesh on TensoIR and Synthetic4Relight datasets.

Tris.	TensoIR [23]			Synthetic4Relight [64]			Rendering
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	time (ms) ↓
4 k	31.388	0.9493	0.0574	34.775	0.9744	0.0422	15.99
20 k	32.097	0.9540	0.0528	35.509	0.9760	0.0403	18.91
100 k	32.171	0.9563	0.0504	35.637	0.9763	0.0398	21.82
500 k	32.395	0.9576	0.0490	35.630	0.9762	0.0398	24.14
2.5 M	32.388	0.9580	0.0486	35.637	0.9761	0.0398	29.14

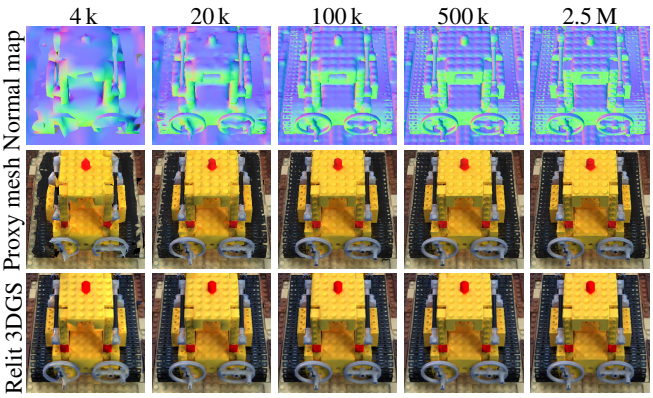


Figure 8: Qualitative comparison of different triangle counts of the proxy meshes.

in Tab. 5, and we also show the proxy meshes visualized by the normal maps, the relighted proxy meshes, and the relighting results of our method in Fig. 8. From Tab. 5 and Fig. 8 we can observe that an extremely low triangle count leads to poor relighting results, but the improvement is marginal when the triangle count is high enough. Furthermore, more triangles consume more time to render.

Sample Count We also investigate the influence of the sample count on the relighting performance. We provide the relighting results of 16, 32, 64, 128, and 256 spp, respectively. The error metrics and the average rendering times are given in Tab. 6 and the visual comparisons are shown in Fig. 9. From Tab. 6 and Fig. 9 we can see that increasing spp can improve the rendering quality, but the rendering time also increases. Since the improvement from 128 spp to 256 spp is marginal, we use 128 spp in our previous experiments.

8 Limitations and Future Work

Complex Materials Due to the use of the isotropic SVBRDF model based on GGX distribution in inverse rendering, we only support opaque objects and cannot handle anisotropic BRDFs, subsurface scattering, or translucency. Due to the use of a hybrid rendering pipeline based on G-buffer, transparent or translucent objects cannot be directly inserted, but we can render transparent objects in a separate pass or use a full ray tracing pipeline to render the opaque and transparent objects simultaneously.

More Accurate Inverse Rendering Since we use proxy meshes with the estimated SVBRDFs and environment maps to compute the radiance change in differential rendering, the relighting and composition results are affected by the performance of inverse rendering. Although we eliminate the error in direct illumination introduced by base color by Eq. (16) and Eq. (18), the estimation error of base color also affects indirect illumination. Now the quality of reconstructed geometries is not very high, and ambiguities still exist in the estimated SVBRDFs and environment maps. In the future, more advanced geometry and material estimation methods may increase the resolution and accuracy of inverse rendering and improve the quality of our method.

Table 6: Ablation on the sample count when relighting on TensoIR and Synthetic4Relight datasets.

spp	TensoIR [23]			Synthetic4Relight [64]			Rendering
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	time (ms) ↓
16	32.234	0.9547	0.0547	35.433	0.9737	0.0460	4.88
32	32.323	0.9562	0.0519	35.543	0.9751	0.0428	7.80
64	32.371	0.9571	0.0501	35.602	0.9758	0.0408	13.49
128	32.395	0.9576	0.0490	35.630	0.9762	0.0398	24.14
256	32.407	0.9578	0.0484	35.643	0.9764	0.0393	45.96

Figure 9: Qualitative comparison of the relighting results using different sample counts.

Non-Distant Environment Light Sources We assume distant environment lighting, so we use the environment map as the light source of the scene. Although environment maps are very efficient for rendering, this assumption is not accurate in most real-world scenes, which may introduce estimation error in SVBRDFs or rendering error in relighting and composition. Several recent works use NeILF [56], NeRF [28, 65, 58] or both of them [61] to represent the non-distant environment light sources, but the inference of neural networks is time-consuming. Modeling the incident radiance of the non-distant environment light source in a more accurate way while keeping fast rendering should be investigated in the future.

Inserting Animated Radiance Fields Now we use the original 3DGS model in our experiments, so we only support static radiance fields. Animated radiance field models such as HiFi4G [21] are impressive to represent humans or animals, which can be integrated into our method to support animated radiance fields.

9 Conclusion

In this work, we propose a physically-based method to relight and composite radiance fields in real-time. We first generate proxy meshes with SVBRDFs and environment maps for the scenes using differentiable rendering. Then a unified differential rendering framework using hardware-accelerated ray tracing is performed on the proxy meshes to compute the radiance change introduced by the new lighting condition or the inserted radiance fields. Furthermore, our unified framework also supports compositing mesh models with radiance fields. Results demonstrate that our method can achieve photorealistic relighting and composition at real-time frame rates.

It should be noted that our method is not limited to relighting and composition of 3DGS. Although we use 3DGS as the radiance field in our implementation, the main components of our method are not directly related to 3DGS. We use 3DGS to represent the radiance field in this work mainly because the reconstruction quality and rendering efficiency of 3DGS are both very high. Additionally, since our method can achieve realistic scene synthesis, our method may potentially be misused in the context of manipulated media.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. This work was supported by the National Natural Science Foundation of China (62302393), the National Key R&D Program of China (2024YFF0907604), the Xi'an Science and Technology Plan Project (24SFSF0002), the Technology Innovation Leading Project of Shaanxi (2024QY-SZX-11), and the Open Project Program of the State Key Laboratory of CAD&CG (A2420), Zhejiang University.

REFERENCES

[1] Amazon Lumberyard. Amazon Lumberyard Bistro, open research content archive (ORCA), July 2017. https://developer.nvidia.com/orca/amazon-lumberyard-bistro. 7

[2] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *Proc. CVPR*, pp. 5460–5469. IEEE Computer Society, Los Alamitos, CA, USA, 2022. doi: 10.1109/CVPR52688.2022.00539 1, 2, 7

[3] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. Zip-NeRF: Anti-aliased grid-based neural radiance fields. In *Proc. ICCV*, pp. 19640–19648. IEEE Computer Society, Los Alamitos, CA, USA, 2023. doi: 10.1109/ICCV51070.2023.01804 1, 2

[4] Z. Bi, Y. Zeng, C. Zeng, F. Pei, X. Feng, K. Zhou, and H. Wu. GS^3: Efficient relighting with triple Gaussian splatting. In *Proc. SIGGRAPH Asia*, p. 12. Association for Computing Machinery, New York, NY, USA, 2024. doi: 10.1145/3680528.3687576 2

[5] Blender Foundation. Blender 4.2.3 LTS, Oct. 2024. https://www.blender.org. 6

[6] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. P. Lensch. NeRD: Neural reflectance decomposition from image collections. In *Proc. ICCV*, pp. 12664–12674. IEEE Computer Society, Los Alamitos, CA, USA, 2021. doi: 10.1109/ICCV48922.2021.01245 2

[7] M. Boss, V. Jampani, R. Braun, C. Liu, J. Barron, and H. P. Lensch. Neural-PIL: Neural pre-integrated lighting for reflectance decomposition. In *Proc. NeurIPS*, vol. 34, pp. 10691–10704. Curran Associates, Inc., Red Hook, NY, USA, 2021. 2

[8] D. Chen, H. Li, W. Ye, Y. Wang, W. Xie, S. Zhai, N. Wang, H. Liu, H. Bao, and G. Zhang. PGSR: Planar-based Gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 31(9):6100–6111, Sept. 2025. doi: 10.1109/TVCG.2024.3494046 3

[9] H. Chen, Z. Lin, and J. Zhang. GI-GS: Global illumination decomposition on Gaussian splatting for inverse rendering. In *Proc. ICLR*, 2025. To appear. 2, 7

[10] P. Debevec. Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proc. SIGGRAPH*, p. 189–198. Association for Computing Machinery, New York, NY, USA, 1998. doi: 10.1145/280814.280864 2, 3, 4

[11] K. Du, Z. Liang, and Z. Wang. GS-ID: Illumination decomposition on Gaussian splatting via diffusion prior and parametric light source optimization. In *Proc. ICCV*, 2025. To appear. 1, 2

[12] J. Fan, F. Luan, J. Yang, M. Hasan, and B. Wang. RNG: Relightable neural Gaussians. In *Proc. CVPR*, 2025. To appear. 2

[13] J. Gao, C. Gu, Y. Lin, Z. Li, H. Zhu, X. Cao, L. Zhang, and Y. Yao. Relightable 3D Gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing. In A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, and G. Varol, eds., *Proc. ECCV*, pp. 73–89. Springer Nature Switzerland, Cham, Switzerland, 2024. doi: 10.1007/978-3-031-72995-9_5 1, 2, 4, 7

[14] X. Gao, Z. Yang, Y. Zhao, Y. Sun, X. Jin, and C. Zou. A general implicit framework for fast NeRF composition and rendering. In *Proc. AAAI*, pp. 1833–1841. AAAI Press, Washington, DC, USA, 2024. doi: 10.1609/aaai.v38i3.27952 2

[15] C. Gu, X. Wei, Z. Zeng, Y. Yao, and L. Zhang. IRGS: Inter-reflective Gaussian splatting with 2D Gaussian ray tracing. In *Proc. CVPR*, 2025. To appear. 1, 2, 4, 7

[16] Y. Guo, Y. Bai, L. Hu, Z. Guo, M. Liu, Y. Cai, T. Huang, and L. Ma. PRTGS: Precomputed radiance transfer of Gaussian splats for real-time high-quality relighting. In *Proc. MM*, pp. 5112–5120. Association for Computing Machinery, New York, NY, USA, 2024. doi: 10.1145/3664647.3680893 1, 2

[17] J. Hasselgren, N. Hofmann, and J. Munkberg. Shape, light, and material decomposition from images using Monte Carlo rendering and denoising. In *Proc. NeurIPS*, vol. 35, pp. 22856–22869. Curran Associates, Inc., Red Hook, NY, USA, 2022. 3

[18] S. Hill, S. McAuley, B. Burley, D. Chan, L. Fascione, M. Iwanicki, N. Hoffman, W. Jakob, D. Neubelt, A. Pesce, and M. Pettineo. Physically based shading in theory and practice. In *SIGGRAPH Courses*, p. 22. Association for Computing Machinery, New York, NY, USA, 2015. doi: 10.1145/2776880.2787670 4

[19] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao. 2D Gaussian splatting for geometrically accurate radiance fields. In *Proc. SIGGRAPH*, p. 32. Association for Computing Machinery, New York, NY, USA, 2024. doi: 10.1145/3641519.3657428 2

[20] W. Jakob, S. Speierer, N. Roussel, M. Nimier-David, D. Vicini, T. Zeltner, B. Nicolet, M. Crespo, V. Leroy, and Z. Zhang. Mitsuba 3 renderer, 2022. https://mitsuba-renderer.org. 6

[21] Y. Jiang, Z. Shen, P. Wang, Z. Su, Y. Hong, Y. Zhang, J. Yu, and L. Xu. HiFi4G: High-fidelity human performance rendering via compact Gaussian splatting. In *Proc. CVPR*, pp. 19734–19745. IEEE Computer Society, Los Alamitos, CA, USA, 2024. doi: 10.1109/CVPR52733.2024.01866 2, 9

[22] Y. Jiang, J. Tu, Y. Liu, X. Gao, X. Long, W. Wang, and Y. Ma. GaussianShader: 3D Gaussian splatting with shading functions for reflective surfaces. In *Proc. CVPR*, pp. 5322–5332. IEEE Computer Society, Los Alamitos, CA, USA, 2024. doi: 10.1109/CVPR52733.2024.00509 2

[23] H. Jin, I. Liu, P. Xu, X. Zhang, S. Han, S. Bi, X. Zhou, Z. Xu, and H. Su. TensoIR: Tensorial inverse rendering. In *Proc. CVPR*, pp. 165–174. IEEE Computer Society, Los Alamitos, CA, USA, 2023. doi: 10.1109/CVPR52729.2023.00024 2, 6, 7, 8, 9

[24] B. Kerbl, G. Kopanas, T. Leimkuehler, and G. Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139, July 2023. doi: 10.1145/3592433 1, 2, 6

[25] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun. Tanks and temples: benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4):78, July 2017. doi: 10.1145/3072959.3073599 7

[26] P. Kán and H. Kaufmann. Differential irradiance caching for fast high-quality light transport between virtual and real worlds. In *Proc. ISMAR*, pp. 133–141. IEEE Computer Society, Los Alamitos, CA, USA, 2013. doi: 10.1109/ISMAR.2013.6671773 3

[27] Z. Liang, Q. Zhang, Y. Feng, Y. Shan, and K. Jia. GS-IR: 3D Gaussian splatting for inverse rendering. In *Proc. CVPR*, pp. 21644–21653. IEEE Computer Society, Los Alamitos, CA, USA, 2024. doi: 10.1109/CVPR52733.2024.02045 2, 7

[28] J. Ling, R. Yu, F. Xu, C. Du, and S. Zhao. NeRF as a non-distant environment emitter in physics-based inverse rendering. In *Proc. SIGGRAPH*, p. 39. Association for Computing Machinery, New York, NY, USA, 2024. doi: 10.1145/3641519.3657404 9

[29] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai. Scaffold-GS: Structured 3D Gaussians for view-adaptive rendering. In *Proc. CVPR*, pp. 20654–20664. IEEE Computer Society, Los Alamitos, CA, USA, 2024. doi: 10.1109/CVPR52733.2024.01952 2

[30] F. Luan, S. Zhao, K. Bala, and Z. Dong. Unified shape and SVBRDF recovery using differentiable Monte Carlo rendering. *Computer Graphics Forum*, 40(4):101–113, July 2021. doi: 10.1111/cgf.14344 3

[31] S. Ma, Q. Shen, Q. Hou, Z. Ren, and K. Zhou. Neural compositing for real-time augmented reality rendering in low-frequency lighting environments. *Science China Information Sciences*, 64(2), Feb. 2021. doi: 10.1007/s11432-020-3024-5 3

[32] M. McGuire. Computer graphics archive, July 2017. https://casual-effects.com/data. 7

[33] S. U. Mehta, K. Kim, D. Pajak, K. Pulli, J. Kautz, and R. Ramamoorthi. Filtering environment illumination for interactive physically-based rendering in mixed reality. In *Proc. EGSR*, pp. 107–118. The Eurographics Association, Geneva, Switzerland, 2015. doi: 10.2312/sre.20151172 2, 3, 5

[34] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proc. ECCV*, pp. 405–421. Springer Nature Switzerland, Cham, Switzerland, 2020. doi: 10.1007/978-3-030-58452-8_24 1, 2

[35] N. Moenne-Loccoz, A. Mirzaei, O. Perel, R. de Lutio, J. Martinez Esturo, G. State, S. Fidler, N. Sharp, and Z. Gojcic. 3D Gaussian ray tracing: Fast tracing of particle scenes. *ACM Transactions on Graphics*, 43(6):232, Nov. 2024. doi: 10.1145/3687934 2

[36] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics

primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4):102, July 2022. doi: 10.1145/3528223.3530127 1, 2

[37] J. Munkberg, W. Chen, J. Hasselgren, A. Evans, T. Shen, T. Müller, J. Gao, and S. Fidler. Extracting triangular 3D models, materials, and lighting from images. In *Proc. CVPR*, pp. 8270–8280. IEEE Computer Society, Los Alamitos, CA, USA, 2022. doi: 10.1109/CVPR52688.2022.00810 3

[38] M. Nimier-David, Z. Dong, W. Jakob, and A. Kaplanyan. Material and lighting reconstruction for complex indoor scenes with texture-space differentiable rendering. In *Proc. EGSR*, pp. 73–84. The Eurographics Association, Geneva, Switzerland, 2021. doi: 10.2312/sr.20211292 3

[39] Y. Poirier-Ginter, A. Gauthier, J. Phillip, J.-F. Lalonde, and G. Drettakis. A diffusion approach to radiance field relighting using multi-illumination synthesis. *Computer Graphics Forum*, 43(4):e15147, July 2024. doi: 10.1111/cgf.15147 2

[40] A. Pranckevičius. Gaussian splatting playground in Unity, 2023. https://github.com/aras-p/UnityGaussianSplatting. 6

[41] Y.-L. Qiao, A. Gao, Y. Xu, Y. Feng, J.-B. Huang, and M. C. Lin. Dynamic mesh-aware radiance fields. In *Proc. ICCV*, pp. 385–396. IEEE Computer Society, Los Alamitos, CA, USA, 2023. doi: 10.1109/ICCV51070.2023.00042 2

[42] T. Rhee, L. Petikam, B. Allen, and A. Chalmers. MR360: Mixed reality rendering for 360° panoramic videos. *IEEE Transactions on Visualization and Computer Graphics*, 23(4):1379–1388, Apr. 2017. doi: 10.1109/TVCG.2017.2657178 3

[43] K. Rohmer, J. Jendersie, and T. Grosch. Natural environment illumination: Coherent interactive augmented reality for mobile and non-mobile devices. *IEEE Transactions on Visualization and Computer Graphics*, 23(11):2474–2484, Nov. 2017. doi: 10.1109/TVCG.2017.2734426 3

[44] V. Rudnev, M. Elgharib, W. Smith, L. Liu, V. Golyanik, and C. Theobalt. NeRF for outdoor scene relighting. In *Proc. ECCV*, pp. 615–631. Springer Nature Switzerland, Cham, Switzerland, 2022. doi: 10.1007/978-3-031-19787-1_35 2

[45] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron. NeRV: Neural reflectance and visibility fields for relighting and view synthesis. In *Proc. CVPR*, pp. 7491–7500. IEEE Computer Society, Los Alamitos, CA, USA, 2021. doi: 10.1109/CVPR46437.2021.00741 2

[46] C. Sun, G. Cai, Z. Li, K. Yan, C. Zhang, C. Marshall, J.-B. Huang, S. Zhao, and Z. Dong. Neural-PBIR reconstruction of shape, material, and illumination. In *Proc. ICCV*, pp. 18000–18010. IEEE Computer Society, Los Alamitos, CA, USA, 2023. doi: 10.1109/ICCV51070.2023.01654 3

[47] J.-M. Sun, T. Wu, Y.-L. Yang, Y.-K. Lai, and L. Gao. SOL-NeRF: Sunlight modeling for outdoor scene decomposition and relighting. In *Proc. SIGGRAPH Asia*, p. 31. Association for Computing Machinery, New York, NY, USA, 2023. doi: 10.1145/3610548.3618143 2

[48] J. Tang, X. Chen, J. Wang, and G. Zeng. Compressible-composable NeRF via rank-residual decomposition. In *Proc. NeurIPS*, vol. 35, pp. 14798–14809. Curran Associates, Inc., Red Hook, NY, USA, 2022. 2

[49] E. Veach and L. J. Guibas. Optimally combining sampling techniques for Monte Carlo rendering. In *Proc. SIGGRAPH*, p. 419–428. Association for Computing Machinery, New York, NY, USA, 1995. doi: 10.1145/218380.218498 6

[50] Y. Wang, W. Wu, and D. Xu. Learning unified decompositional and compositional NeRF for editable novel view synthesis. In *Proc. ICCV*, pp. 18201–18210. IEEE Computer Society, Los Alamitos, CA, USA, 2023. doi: 10.1109/ICCV51070.2023.01673 2

[51] Z. Wang, T. Shen, J. Gao, S. Huang, J. Munkberg, J. Hasselgren, Z. Gojcic, W. Chen, and S. Fidler. Neural fields meet explicit geometric representations for inverse rendering of urban scenes. In *Proc. CVPR*, pp. 8370–8380. IEEE Computer Society, Los Alamitos, CA, USA, 2023. doi: 10.1109/CVPR52729.2023.00809 2

[52] J. Wu and L. Wang. Panoramic ray tracing for interactive mixed reality rendering based on 360° RGBD video. *IEEE Computer Graphics and Applications*, 44(1):62–75, Jan.–Feb. 2024. doi: 10.1109/MCG.2023.3327383 5

[53] Q. Wu, J. M. Esturo, A. Mirzaei, N. Moënne-Loccoz, and Z. Gojcic. 3DGUT: Enabling distorted cameras and secondary rays in Gaussian splatting. In *Proc. CVPR*, 2025. To appear. 2

[54] T. Wu, J.-M. Sun, Y.-K. Lai, Y. Ma, L. Kobbelt, and L. Gao. DeferredGS: Decoupled and relightable Gaussian splatting with deferred shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(8):6307–6319, Aug. 2025. doi: 10.1109/TPAMI.2025.3560933 2

[55] B. Yang, Y. Zhang, Y. Xu, Y. Li, H. Zhou, H. Bao, G. Zhang, and Z. Cui. Learning object-compositional neural radiance field for editable scene rendering. In *Proc. ICCV*, pp. 13759–13768. IEEE Computer Society, Los Alamitos, CA, USA, 2021. doi: 10.1109/ICCV48922.2021.01352 2

[56] Y. Yao, J. Zhang, J. Liu, Y. Qu, T. Fang, D. McKinnon, Y. Tsin, and L. Quan. NeILF: Neural incident light field for physically-based material estimation. In *Proc. ECCV*, pp. 700–716. Springer Nature Switzerland, Cham, Switzerland, 2022. doi: 10.1007/978-3-031-19821-2_40 3, 9

[57] C. Ye, L. Qiu, X. Gu, Q. Zuo, Y. Wu, Z. Dong, L. Bo, Y. Xiu, and X. Han. StableNormal: Reducing diffusion variance for stable and sharp normal. *ACM Transactions on Graphics*, 43(6):250, Nov. 2024. doi: 10.1145/3687971 3

[58] K. Ye, H. Wu, X. Tong, and K. Zhou. A real-time method for inserting virtual objects into neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 31(9):4896–4907, Sept. 2025. doi: 10.1109/TVCG.2024.3422814 2, 9

[59] Z. Yu, A. Chen, B. Huang, T. Sattler, and A. Geiger. Mip-splatting: Alias-free 3D Gaussian splatting. In *Proc. CVPR*, pp. 19447–19456. IEEE Computer Society, Los Alamitos, CA, USA, 2024. doi: 10.1109/CVPR52733.2024.01839 2

[60] C. Zeng, G. Chen, Y. Dong, P. Peers, H. Wu, and X. Tong. Relighting neural radiance fields with shadow and highlight hints. In *Proc. SIGGRAPH*, p. 73. Association for Computing Machinery, New York, NY, USA, 2023. doi: 10.1145/3588432.3591482 2

[61] J. Zhang, Y. Yao, S. Li, J. Liu, T. Fang, D. McKinnon, Y. Tsin, and L. Quan. NeILF++: Inter-reflectable light fields for geometry and material estimation. In *Proc. ICCV*, pp. 3578–3587. IEEE Computer Society, Los Alamitos, CA, USA, 2023. doi: 10.1109/ICCV51070.2023.00333 3, 9

[62] K. Zhang, F. Luan, Q. Wang, K. Bala, and N. Snavely. PhySG: Inverse rendering with spherical Gaussians for physics-based material editing and relighting. In *Proc. CVPR*, pp. 5449–5458. IEEE Computer Society, Los Alamitos, CA, USA, 2021. doi: 10.1109/CVPR46437.2021.00541 3

[63] X. Zhang, P. P. Srinivasan, B. Deng, P. Debevec, W. T. Freeman, and J. T. Barron. NeRFactor: neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics*, 40(6):237, Dec. 2021. doi: 10.1145/3478513.3480496 7

[64] Y. Zhang, J. Sun, X. He, H. Fu, R. Jia, and X. Zhou. Modeling indirect illumination for inverse rendering. In *Proc. CVPR*, pp. 18622–18631. IEEE Computer Society, Los Alamitos, CA, USA, 2022. doi: 10.1109/CVPR52688.2022.01809 2, 6, 7, 8, 9

[65] Y. Zhuang, Q. Zhang, X. Wang, H. Zhu, Y. Feng, X. Li, Y. Shan, and X. Cao. A pre-convolved representation for plug-and-play neural illumination fields. In *Proc. AAAI*, pp. 7828–7836. AAAI Press, Washington, DC, USA, 2024. doi: 10.1609/aaai.v38i7.28618 9