IBL-NeRF: Image-Based Lighting Formulation of Neural Radiance Fields

Changwoon Choi*
Seoul National University
changwoon.choi@gmail.com

Juhyeon Kim*
Seoul National University
cjdeka3123@snu.ac.kr

Young Min Kim
Seoul National University
youngmin.kim@snu.ac.kr

Abstract

We propose IBL-NeRF, which decomposes the neural radiance fields (NeRF) of large-scale indoor scenes into intrinsic components. Previous approaches for the inverse rendering of NeRF transform the implicit volume to fit the rendering pipeline of explicit geometry, and approximate the views of segmented, isolated objects with environment lighting. In contrast, our inverse rendering extends the original NeRF formulation to capture the spatial variation of lighting within the scene volume, in addition to surface properties. Specifically, the scenes of diverse materials are decomposed into intrinsic components for image-based rendering, namely, albedo, roughness, surface normal, irradiance, and prefiltered radiance. All of the components are inferred as neural images from MLP, which can model large-scale general scenes. By adopting the image-based formulation of NeRF, our approach inherits superior visual quality and multi-view consistency for synthesized images. We demonstrate the performance on scenes with complex object layouts and light configurations, which could not be processed in any of the previous works.

1. Introduction

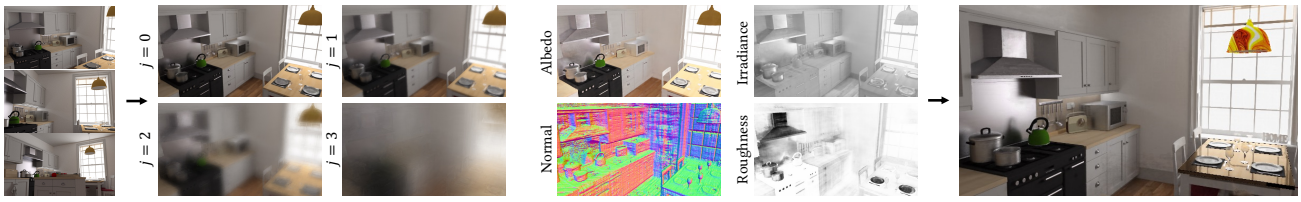
Neural radiance fields (NeRF) [20] prospers for their superior quality in novel-view synthesis with a simple formulation. A neural network is trained to overfit a colored density volume to directly match multiple posed input images. The formulation is ignorant of any intermediate representations of traditional rendering pipelines, namely surface geometry, light transport, or BRDF. The trained volumetric representation does not trace iterative inter-reflections of rays, or model complex occlusion of the surface geometries. Nonetheless, NeRF can produce detailed subtleties of global illumination and parallax effects.

While NeRF can capture complex effects in general scenes, the implicit formulation limits further analysis or edits of the scenes. Inverse rendering is an attractive choice as it decomposes the captured scene into intrinsic components that can be further manipulated to edit the scene. However, intrinsic decomposition is inherently an ill-posed problem and requires enforcing additional priors or constraints. Prior works often extract an isolated object, selected with exhaustive segmentation masks, for inverse rendering with NeRF. They assume low-dimensional environment lighting and incorporate additional knowledge for reflectance properties, such as priors on BRDFs or images captured under different illuminations. Under the constrained set-up, they partially transform the segmented objects into forward rendering with Monte-Carlo integration which can be computationally expensive. Furthermore, such approximation with environment light prohibits viewpoints inside the scene, or a local variation of lights caused from common light fixtures or windows. By relinquishing the flexibility of the original NeRF, existing inverse rendering with NeRF approaches cannot represent everyday environments composed of diverse unsegmented objects.

Instead of extensively simulating multiple bounces of rays with approximated explicit representation, we propose incorporating constraints from the image spaces, extending the NeRF formulation. Specifically, we train a decomposed neural volume, coined IBL-NeRF, to optimize for the implicit light distribution of neural images. This neural representation captures detailed spatial variations of lighting, in contrast to low-dimensional environment mapping. Then we can substitute the illumination integration process into a simple network query for the irradiance. The specular reflection of different surface roughness values is fetched from prefiltered radiance fields of appropriate prefilter levels, similar to texture mipmap. We additionally enforce priors on the intrinsic components for input images, acquired from existing methods for decomposing individual images. By incorporating image-based lighting with implicit intrinsic components, we can efficiently render general scenes without sacrificing the rendering quality of the original NeRF as shown in Fig. 1. We can further edit scenes by changing materials or adding objects, including highly reflective or transparent objects.

In summary, our approach fully leverages the high-

*Authors contributed equally to this work.



(a) Input Images (b) Prefiltered Radiance (c) Decomposed Intrinsic Components (d) Scene Editing

Figure 1. We propose IBL-NeRF, a neural volume representation with prefiltered radiance field inspired by image-based lighting formulation. (a) Given multi-view images, we optimize the (b) prefiltered radiance field and estimate (c) reflectance properties of the material (albedo, roughness), lighting information (irradiance, prefiltered radiance) and the geometry (normal). (d) One can manipulate the neural scene easily by modifying the decomposed components.

quality novel view images of the original NeRF formulation, and yet enables efficient re-generation with approximations inspired from image-based rendering. Our contributions can be listed as following:

- We propose IBL-NeRF, which handles global illuminations with spatially varying lighting and diverse materials given a set of unsegmented images.
- We model the prefiltered radiance of the scene with a neural network of NeRF, and efficiently approximate rendering equations with image-based lighting.
- Our neural representation extracts physically interpretable components of the complex indoor scenes which can be altered to render images with different attributes.

The results are presented with large-scale scenes containing multiple objects, which can not be modeled with previous works employing a single environment lighting or Monte-Carlo integration.

2. Related Works

While NeRF [20] can synthesize photo-realistic novel-view images, one of its limitations is that the radiance information is baked within the implicit neural representation. Several subsequent works propose to distill intrinsic components, such as illumination and reflectance property, and try to achieve inverse rendering with implicit representation, in contrast to reconstructing explicit mesh geometry with multi-view stereo [25, 8]. They optimize components to match the input images by adopting Monte Carlo (MC) integration, which requires heavy computation. Neural Reflectance Fields [2] and NeRV [28] adapt ray-marching to account for reflectance, and model the illumination with a single point light and environment light, respectively. Both approaches require multiple images with known lighting configurations as input. NeRFactor [33], NeRD [4], and PhySG [32], on the other hand, factorize radiance fields from unknown light. They concurrently optimize for low-dimensional environment light in a coarse resolution (NeRFactor) or spherical Gaussian (NeRD, PhySG).

In contrast, IBL-NeRF proposes to efficiently synthesize images without explicit Monte-Carlo integration, and utilizes prefiltered radiance which can be evaluated with a single ray sample. Several concurrent works [29, 5] also adapt integrated illumination for efficient rendering. They are either implicitly conditioned on the surface reflectance property, or propose components without physical interpretation. However, all of the previous works employ environment lighting and therefore are limited to modeling an isolated object.

Inverse rendering for general scenes requires modeling spatially-varying lighting. With increased degrees of freedom for the already under-constrained problem, scene decomposition requires strong assumptions. Commonly used priors include piece-wise constant albedos [6, 16, 17, 18], or sparsity of extracted albedo values [19, 9]. A few works exploit data-driven priors instead of hand-crafted priors [1, 34, 27, 15, 22], which can be subject to domain discrepancy. IBL-NeRF takes inspiration from the aforementioned prior works using single images, and adds constraints in the image space. Because the neural volume of NeRF is trained with images, the formulation can readily be applied to handle challenging indoor scenes without simplifying the illumination model. Furthermore, IBL-NeRF can naturally find multi-view consistent components, which is not possible with single-image decomposition.

3. Method

3.1. IBL-NeRF Formulation

3.1.1 Preliminaries

Ray-tracing engines approximate the light transport with samples of rays, which is computationally expensive. The original rendering equation [11] formulates the outgoing radiance at surface $\mathbf{x}_{\text{surf}}$ as a combination of reflected rays of

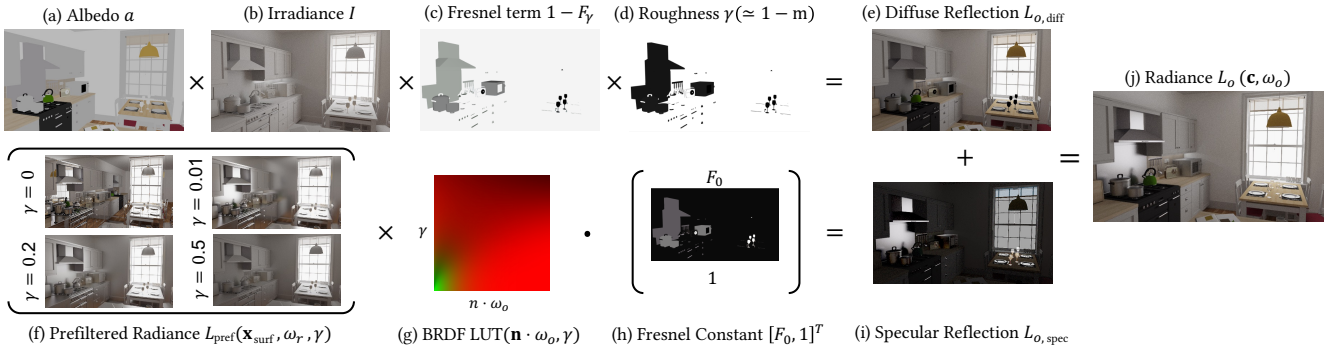


Figure 2. Overview of the radiance approximation used in IBL-NeRF (Eq. 2). With the combination of inferred (a) albedo, (b) Irradiance, (c) Fresnel term, and (d) roughness, one can obtain diffuse reflection. Also, with (f) multi-level prefiltered radiance, (g) fetched value from LUT and (h) Fresnel constant, we can calculate (i) specular reflection. (j) Final approximated radiance is achieved by the sum of diffuse and specular reflection.

incoming radiance L_i

$$L_o(\mathbf{x}_{\mathrm{surf}}, \omega_o) = \int_\Omega f_r(\mathbf{x}_{\mathrm{surf}}, \omega_i, \omega_o) L_i(\mathbf{x}_{\mathrm{surf}}, \omega_i)(\mathbf{n} \cdot \omega_i) d\omega_i, \tag{1}$$

where $\mathbf{n}$ and f_r are the surface normal and BRDF at surface $\mathbf{x}_{\mathrm{surf}}$, and ω_i and ω_o are incoming and outgoing direction. Given the scene properties ($\mathbf{n}$ and f_r), the rendered output relies on the diverse distribution of light transport, L_i and L_o, which are 5D functions.

The approximation within game engines [12] replaces the recursive calls of radiances $L_i \rightarrow L_o$ into a single sample of integrated light. L_o is approximated as the sum of two components, namely the diffuse term and the specular term:

$$\begin{aligned} L_o(\mathbf{x}_{\mathrm{surf}}, \omega_o) = & \underbrace{\gamma \times (1 - F_\gamma(\omega_o, \mathbf{n}, \gamma)) \times a \times I}_{L_{o,\mathrm{diff}}} \\ & + \underbrace{L_{\mathrm{pref}}(\mathbf{x}_{\mathrm{surf}}, \omega_r, \gamma) \times [F_0, 1]^T \, \mathrm{LUT}(\omega_o \cdot \mathbf{n}, \gamma)}_{L_{o,\mathrm{spec}}}. \end{aligned} \tag{2}$$

The diffuse term depends on irradiance I which integrates all the incoming radiance. Additionally, it is proportional to the surface albedo a, roughness γ [1] and approximated Fresnel term F_γ. Calculating the specular term $L_{o,\mathrm{spec}}$ involves directional components of rays. The split-sum approximation simplifies the specular term into the product of two terms: The first component L_{pref} is the *prefiltered environment map* which summarizes the effects of reflected lights to efficiently mimic specular highlights. It is filtered according to the surface roughness level γ and fetched at the reflected direction. The second component is also precalculated as a 2D lookup texture (LUT). IBL-NeRF allows decomposition of NeRF by utilizing neural network

[1] In [12], (1-metallic) is used. We approximate it to roughness.

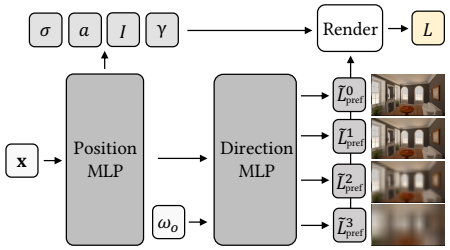


Figure 3. Architecture of IBL-NeRF. The scene properties dependent on position (volume density σ, albedo a, irradiance I and roughness γ) are extracted from position MLP. Those dependent on the viewing direction (each level of prefiltered radiance $\tilde{L}^j_{\mathrm{pref}}$) are obtained from direction MLP.

to represent the pre-computed volumetric light distribution. Detailed descriptions of the approximation are available in supplementary material.

3.1.2 Rendering Pipeline with IBL-NeRF

NeRF synthesizes a photo-realistic image applying a volume rendering on a neural volume

$$L_o(\mathbf{c}, \omega_o) = \int_0^\infty V(\mathbf{x}(t), \mathbf{c}) \sigma(\mathbf{x}(t)) L_e(\mathbf{x}(t), \omega_o) dt, \tag{3}$$

where $\mathbf{x}(t) = \mathbf{c} - t\omega_o$ represents points on a ray initiated from the camera position $\mathbf{c}$, and $V(\mathbf{x}(t), \mathbf{c}) = \exp\left(-\int_0^t \sigma(\mathbf{x}(s)) ds\right)$ is the visibility. Given a position $\mathbf{x}$ and an outgoing direction ω_o, the neural volume of NeRF is trained to regress for density σ from the positional MLP and the emitted radiance L_e from the directional MLP. The training objective is to match the results of volume rendering with the pixels in the input images, which enables creating images of the scene only from a set of multiple-view

images.

To decompose the radiance of NeRF into physically interpretable components of the scene, we can adapt components ignorant of light transport as presented in Sec. 3.1.1. For each ray, we evaluate albedo a, irradiance I, and roughness γ with the volume density σ. We accumulate the values along the ray using volume rendering following the NeRF formulation in Eq. 3. Also, at the estimated surface point, the network evaluates the prefiltered radiance field L_{pref} of the reflected direction. Due to the computational complexity, the reflected rays are evaluated only at the surface hit position of the ray, which is estimated as $\mathbf{x}_{\text{surf}} = \mathbf{c} - d\omega_o$ [33, 28]. The termination depth $d(\mathbf{c}, -\omega_o)$ of the ray defines the surface point $\mathbf{x}_{\text{surf}}$ and can be obtained with density $\int_0^\infty \exp\left(-\int_0^\infty \sigma(\mathbf{c} - s\omega_o)ds\right) t\sigma(\mathbf{c} - t\omega_o)dt$. We obtain surface normal from the numerical gradient of the termination depth d, which is further discussed in the supplementary material. ($\mathbf{n}(\mathbf{x}_{\text{surf}}) = \nabla_{\mathbf{x}} d(\mathbf{x}, \omega) / \|\nabla_{\mathbf{x}} d(\mathbf{x}, \omega)\|$.) All the values are combined using Eq. 2 to find the output radiance corresponding to the pixel, which is also visualized in Fig. 2.

Fig. 3 shows the modified neural network architecture. The positional MLP infers the components that do not have view dependency, namely, albedo a, irradiance I, and roughness γ, in addition to the volume density σ in the vanilla NeRF. The directional component is encoded as prefiltered radiance field L_{pref}, and is the output of the subsequent directional MLP. It is modulated by roughness γ and combined to generate the final image. The following subsection further explains the formulation and approximation used for the prefiltered radiance fields.

3.2. Prefiltered Radiance Fields

The prefiltered environment map $L_{\text{pref}}(\mathbf{x}_{\text{surf}}, \omega_r, \gamma)$ in Eq. 2 accounts for the specular reflection with directional components that reside in a high-dimensional space as a single sample. Let us denote the camera observation direction as $-\omega_o$ and its mirror reflection with respect to the surface normal $\mathbf{n}$ at the surface point as ω_r. Unless the surface is a perfect mirror (roughness 0), the reflected rays are evaluated within angular distribution near the reflection direction. As the surface roughness increases γ, prefiltered radiance should be filtered with a wider range kernel. Fig. 4 illustrates the procedure, where the pre-filtered radiance at ω_r is depicted with cones with yellow shade, whose angle indicates the size of convolution kernel for the roughness value.

While there exist several works that approximate specular illumination from a hit point, IBL-NeRF alleviates the need for Monte-Carlo integration and greatly reduces the computational burden. Table 1 summarizes the comparison of IBL-NeRF against NeRFactor [33], which is a representative formulation with environment light [33, 28, 2].

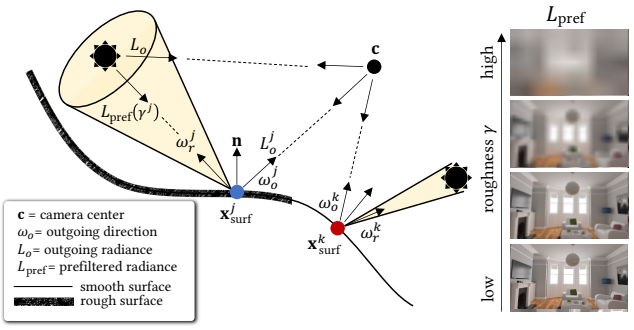


Figure 4. Specular reflection of IBL-NeRF. The prefiltered radiance field is fetched only from the estimated surface point $\mathbf{x}_{\text{surf}}$ with a single reflected ray toward the direction of mirror reflection ω_r. The point on rough surface (intersection of j^{th} ray) fetches the prefiltered radiance convoluted with a wide kernel. In contrast, the point on smooth surface (intersection of k^{th} ray) reads prefiltered radiance field filtered with a narrow kernel.

	NeRF	NeRFactor	IBL-NeRF (Ours)
Rendering	Volume	Surface	Surface
L_o	Baked	Monte Carlo Integration	Approx w. Eq. 2
L_i	-	Env light w. Visibility Infer	MLP Inference
Time Complexity	$\mathcal{O}(N_s)$	$\mathcal{O}(N_s + N_d N_r)$	$\mathcal{O}(N_s + N_r)$

Table 1. We compare IBL-NeRF with NeRF and recent method decomposing NeRF's radiance. The time complexity is measured for the entire training phase. N_s and N_r are the numbers of samples along a camera ray and a reflected ray, and N_d is the number of directional samples over a hemisphere.

Specifically, the Monte-Carlo integration aggregates N_d directional samples of reflected rays from the surface points as shown in shaded cones in Fig. 4. In addition to the N_s samples along the camera ray for the volume rendering of NeRF, each reflected ray is evaluated with N_r samples of towards the surrounding environment lighting. The variants using Monte-Carlo integration therefore require evaluating $\mathcal{O}(N_s + N_d N_r)$ samples. On the other hand, IBL-NeRF proposes fetching a single ray of the prefiltered radiance field $L_{\text{pref}}(\mathbf{x}_{\text{surf}}, \omega_r, \gamma)$ in the place of the environment map, leading to evaluating $\mathcal{O}(N_s + N_r)$ samples.

Additionally, IBL-NeRF can process general scenes with diverse lighting or viewpoints as long as the original NeRF converges. The prefiltered radiance fields is defined for the entire scene volume for any position $\mathbf{x}_{\text{surf}}$ and or direction ω_r. This is in contrast to the approaches relying on environment light, as they assume an isolated object distant from other scene properties, especially lighting. Therefore it cannot render from viewpoints within the volume, diverse

objects spread throughout the scene, or indoor scenes with interior lighting.

Our specular reflection is evaluated as a single ray for the given roughness value within the scene volume since the prefiltered radiance $L_{\text{pref}}(\mathbf{x}_{\text{surf}}, \omega_r, \gamma)$ already aggregates the the directional rays. Specifically, IBL-NeRF outputs prefiltered radiance fields L^j_{pref} with different convolution level j. The prefiltered radiance of the desired roughness γ at a certain point $\mathbf{x}$ with direction ω uses trilinear interpolation as

$$L_{\text{pref}}(\mathbf{x}, \omega, \gamma) = \sum_j w^j(\gamma) L^j_{\text{pref}}(\mathbf{x}, \omega), \tag{4}$$

where $w^j(\gamma)$ is the weight of jth mipmap that depends on the roughness γ as described in Fig. 4. Therefore, we evaluate the prefiltered radiance by fetching a sample of a single ray, similar to texture mipmap.

The prefiltered radiance $\tilde{L}^j_{\text{pref}}$ is inferred from the directional MLPs using the similar volume rendering equation

$$L^j_{\text{pref}}(\mathbf{c}, -\omega_o) = \int_0^\infty V(\mathbf{x}(t), \mathbf{c}) \sigma(\mathbf{x}(t)) \tilde{L}^j_{\text{pref}}(\mathbf{x}(t), -\omega_o) dt. \tag{5}$$

For training, we use a set of images blurred with a discrete set of Gaussian filters from the camera position $-\omega_o$. During the inference of the image, the values of $\tilde{L}^j_{\text{pref}}$ are fetched to render the surface point $\mathbf{x}_{\text{surf}}$ as explained in Sec. 3.1.2 and Eq. 4. Note that the training target is the blurred images observed from the camera $(\mathbf{c}, -\omega_o)$, whereas the inference is evaluated from the reflected direction $(\mathbf{x}_{\text{surf}}, \omega_r)$. The formulation relies on the assumption that training images contain observations of the reflected rays.

3.2.1 Image-Space Approximation

The prefiltered radiance L^j_{pref} of IBL-NeRF incorporates the image-based rendering within the implicit volume of NeRF and achieves computational efficiency. We further analyze the practical considerations with the image-space approximation of Gaussian filters to emulate the specular reflection blobs of different surface roughness. The jth prefiltered radiance L^j_{pref} is approximated for the roughness value γ_j as

$$L^j_{\text{pref}} = \int_\Omega L_i(\mathbf{x}, \omega_i) p(\omega_i | \mathbf{x}, \omega, \gamma_j) d\omega_i \tag{6}$$

$$= \int_S L_i(s_i) p_S(s_i | \mathbf{x}, \omega, \gamma_j) ds. \tag{7}$$

Previous approaches approximate the sampling distribution by inferring radiance multiple times in hemispherical domain Ω (Eq. 6) which is computationally heavy [33, 28]. Our method converts the domain into the image space S of the current view as Eq. 7, where s_i is the screen space coordinate that corresponds to direction ω_i. When rendering for a viewpoint, the viewing direction ω_o can be assumed to be constant, and we can use a globally consistent kernel $L^j_{\text{pref}}(\mathbf{x}, \omega) = K^j(L(s))$, where $K^j(s_i) \propto p_S(s_i | \mathbf{x}, \omega, \gamma_j)$. In supplementary material, we include the full derivation of our approximation and plots of $K^j(s_i)$. The overall shape of $K^j(s_i)$ is similar to that of the Gaussian function, which is used to approximate $L^j_{\text{pref}}(\mathbf{x}, \omega)$ in our implementation. Supplementary materials contain additional discussion on our approximation.

3.3. Training IBL-NeRF

IBL-NeRF imposes the constraints on the rendered images to train the neural volume, similar to vanilla NeRF. The objective function is composed of four terms:

$$\mathcal{L} = \mathcal{L}_{\text{render}} + \mathcal{L}_{\text{pref}} + \mathcal{L}_{\text{prior}} + \lambda_{I,\text{reg}} \mathcal{L}_{I,\text{reg}}. \tag{8}$$

The first two components are rendering losses to match the rendered images with the input images. For each pixel of the camera ray $r = (\mathbf{c}, -\omega_o)$, the rendering loss $\mathcal{L}_{\text{render}}$ of approximated radiance is defined as

$$\mathcal{L}_{\text{render}} = \| L_o(r) - \hat{L}_o(r) \|_2^2, \tag{9}$$

where $\hat{L}_o$ is ground truth radiance and L_o is our approximated radiance calculated with Eq. 2. $\mathcal{L}_{\text{pref}}$ is the rendering loss of prefiltered radiance defined as

$$\mathcal{L}_{\text{pref}} = \sum_j \| L^j_{\text{pref}}(r) - L^j_{\text{G}}(r) \|_2^2. \tag{10}$$

L^j_{pref} is inferred prefiltered radiance of j^{th} level and L^j_{G} is the radiance convolved with j^{th} level Gaussian convolution, where $L^0_{\text{G}} = L$.

Inverse rendering is under-constrained in nature, and the remaining two losses incorporate additional prior knowledge to estimate intrinsic components. We obtain the pseudo albedo $\hat{a}$ and irradiance $\hat{I}$ for our input images by applying intrinsic decomposition for single images [1], and use them as data-driven prior. The prior loss $\mathcal{L}_{\text{prior}}$ encourages our inferred albedo a to match the pseudo albedo

$$\mathcal{L}_{\text{prior}} = \| a(r) - \hat{a}(r) \|_2^2. \tag{11}$$

In addition, $\mathcal{L}_{I,\text{reg}}$ is the irradiance regularization loss

$$\mathcal{L}_{I,\text{reg}} = \| I(r) - \mathbb{E}[\hat{I}] \|_2^2, \tag{12}$$

where $\mathbb{E}[\hat{I}]$ is the mean of irradiance (shading) values in training set images. Although the results from single-image decomposition are inconsistent for different viewpoints, our neural volume learns multi-view consistent and smooth results. We provide more detailed comparison between IBL-NeRF and results from single-image decomposition methods in Sec. 4.1.

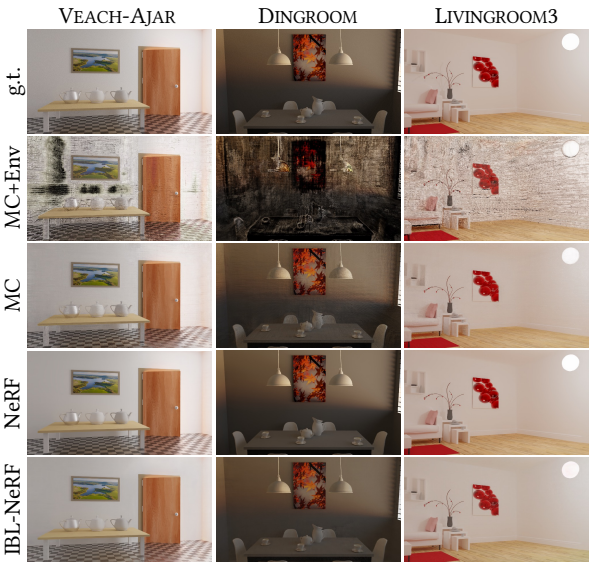


Figure 5. Qualitative results of novel-view image synthesis.

Method	MSE ↓	PSNR ↑	SSIM ↑	Time per step (s) Train	Infer
MC + Env [33]	0.0369	16.107	0.2763	0.4686	0.1062
MC	0.0016	**30.052**	0.8348	0.4941	0.1084
NeRF	**0.0008**	**34.707**	**0.9253**	**0.0984**	**0.0055**
IBL-NeRF	**0.0014**	29.962	**0.9009**	**0.1559**	**0.0211**

Table 2. Quantitative results of view synthesis.

4. Experiments

Dataset & Implementation Details First, we test IBL-NeRF in 12 realistic synthetic indoor scenes [3], which are capable of obtaining ground-truth intrinsic components. We render 100 multi-view images for both training and test set with the OptiX [23] based path tracer [13]. All of the scenes in our dataset exhibit complex lighting with windows or interior lighting and contain multiple objects with challenging material, which cannot be modeled with an environment light. This is in contrast to previous works for decomposing NeRF, which present results with isolated objects [33, 28]. Furthermore, we test IBL-NeRF in real-world scenes from ScanNet dataset [7] and our own captured scene. For ScanNet scenes, we use train/test split from [31]. The camera poses are estimated with COLMAP [26] for real scenes.

The neural network architecture is illustrated in Fig. 3. IBL-NeRF is trained for 120k steps with 512 ray samples, and follows the training schedule below to stabilize the process. For the first 10k steps, we only optimize L^j_{pref} and σ with $\mathcal{L}_{\text{pref}}$. Once we obtain stable geometry and prefiltered radiance fields, we additionally optimize for $\mathcal{L}_{\text{pref}} + \mathcal{L}_{\text{render}}$ without prior. Then we freeze roughness and apply priors $\mathcal{L}_{\text{prior}}, \mathcal{L}_{I,\text{reg}}$ for last 20k steps. We use $\lambda_{I,\text{reg}} = 0.1$ and assume monochromatic irradiance for simplicity. We describe full implementation details in supplementary material.

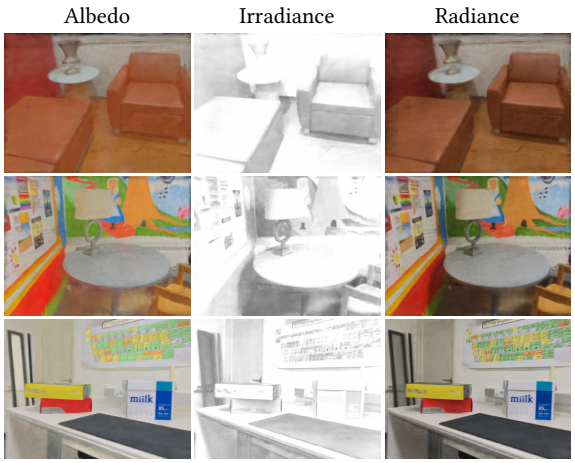


Figure 6. Qualitative results of intrinsic decomposition and view synthesis on real-world datasets.

4.1. View Synthesis & Intrinsic Decomposition

We compare IB-NeRF with two baselines with Monte Carlo (MC) sampling over a hemisphere of environment light. The first baseline (MC) is a variant of IBL-NeRF, which estimates prefiltered radiance L_{pref} and calculates integration with MC sampling. The second baseline (MC + Env) estimates environment light as L_i and employs MC integration. MC + Env emulates the approach from NeRFactor [33] which is the state-of-the-art work in the decomposition of NeRF. The results for MC + Env do not incorporate the albedo prior, as it achieves better performance. We report the results for MC + Env with $\mathcal{L}_{\text{prior}}$ and implementation details for baselines in our supplementary material.

We report the quantitative results of the novel-view synthesis in Table 2 and intrinsic decomposition in Table 3 in terms of MSE, PSNR, and SSIM. IBL-NeRF models outgoing radiance as the combination of various intrinsic components and concurrently generates images whose quality is comparable to vanilla NeRF. Notably, our approach outperforms the method from NeRFactor (MC + Env) in both intrinsic decomposition and image synthesis results for all error metrics, which supports our claim that using environment lighting with MC sampling is inadequate to express complex indoor scenes. The reconstruction quality is much better by alleviating the environment light and instead adapting our formulation in Eq. 2. Theoretically, the MC baseline should have better results in the expense of computation time, which is almost 3 times slower in training phase and 5 times slower in inference phase than IBL-NeRF. However, since there exists a number of invalid samples in the incident radiance that are invisible from train-

	Albedo			Irradiance			Roughness			View Synthesis		
	MSE ↓	PSNR ↑	SSIM ↑	MSE ↓	PSNR ↑	SSIM ↑	MSE ↓	PSNR ↑	SSIM ↑	MSE ↓	PSNR ↑	SSIM ↑
w/ GT $\mathbf{n}$	**0.0551**	**14.134**	**0.7465**	0.0376	15.986	0.7717	**0.0623**	14.216	0.8220	0.0015	29.774	**0.9033**
w/o L_{prior}	0.0664	13.423	0.7107	0.0403	15.609	0.7553	0.0717	15.413	0.8613	0.0011	32.023	0.8978
w/o $L_{I,\text{reg}}$	**0.0551**	14.077	0.7362	**0.0337**	16.215	0.7586	0.0710	14.316	0.8588	0.0012	31.104	0.8960
w/o all priors	0.0775	11.601	0.6911	0.0674	12.147	0.7015	0.0709	15.527	0.8637	**0.0010**	**32.672**	0.9013
IBL-NeRF	0.0553	**14.114**	**0.7455**	0.0351	**16.435**	**0.7778**	**0.0707**	**15.545**	**0.8653**	0.0014	29.962	0.9009

Table 3. Quantitative results of intrinsic decomposition.

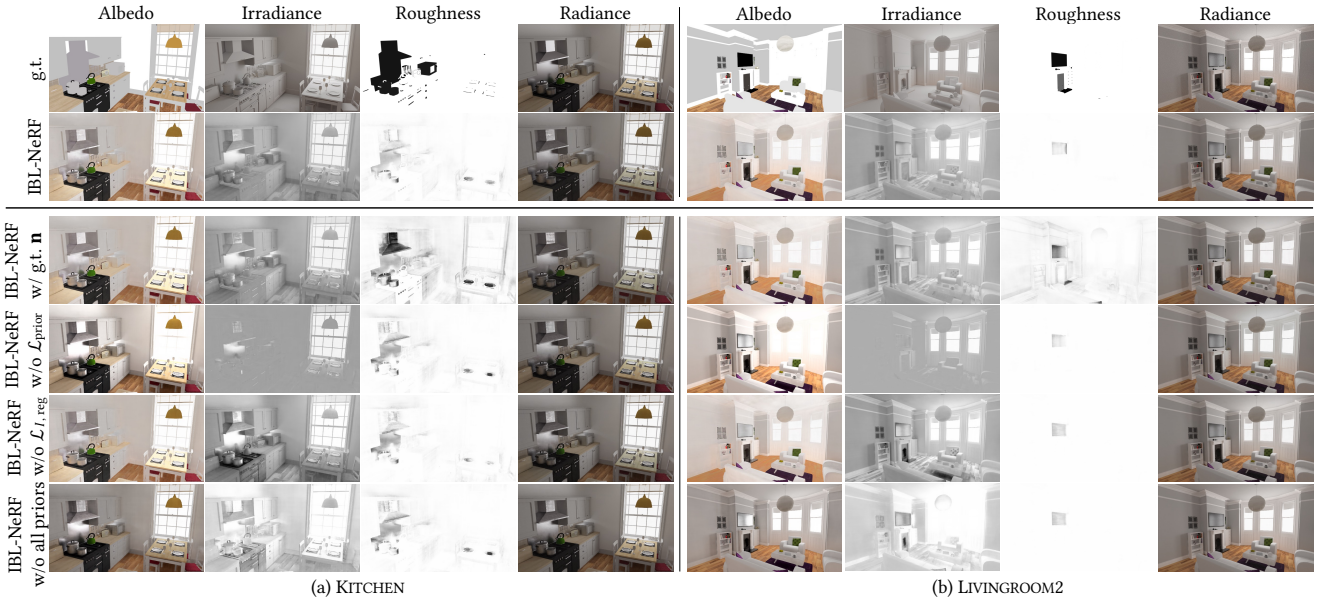


Figure 7. Qualitative results of intrinsic decomposition and view synthesis on our synthetic datasets.

ing viewpoints, the decomposition of MC is comparable to ours.

We demonstrate the qualitative results of novel-view synthesis and intrinsic decomposition in synthetic scenes in Fig. 5 and 7, real scenes in Fig. 6. Our approach and MC approach with prefiltered radiance field reconstruct high-quality images in novel viewpoints, which are comparable to vanilla NeRF. On the other hand, objects in large-scale indoor scenes are often occluded by other structures within the scene, and therefore cannot be illuminated appropriately with environment light (MC + Env). The quality images are significantly worse as it suffers from notable dark and noisy artifacts created from missing viewpoints or ambiguous regions. Fig. 6 and Fig. 7 show that IBL-NeRF successfully decomposes the scene attributes in both synthetic and real-world scenes. IBL-NeRF estimates low roughness at metallic surfaces, for example, the ventilator, metallic wall, buttons in oven, and pots in KITCHEN, TV in LIVINGROOM2 in Fig. 7. However, our method fails to discover metallic surface that does not have specular variation with respect to viewing direction in training set. (For example, the fireplace in LIVINGROOM2 has consistent color in the training images.) Additional results are available in the supplementary material. Furthermore, IBL-NeRF can easily achieve the inherent multi-view consistency and smoothness of our optimizing process as shown in Fig. 8. While the intrinsic decomposition algorithms for single-view images [1, 34] fail to maintain consistent results, it provides a useful guidance for the intrinsic decomposition.

Ablation Studies Fig. 7 and Table 3 also contain results for ablated versions of IBL-NeRF to analyze the important components of the proposed method. The qualitative results with ground-truth normal $\mathbf{n}$ shows cleaner roughness than our original model. The effect of roughness is tightly coupled with the direction of mirror reflection, which is obtained from the surface normal. Recent methods [21, 30] propose to reconstruct high-quality geometry with NeRF formulations, from which IBL-NeRF can learn better decomposition.

Since the inverse rendering is an under-constrained problem, prior knowledge on intrinsic components plays a cru-

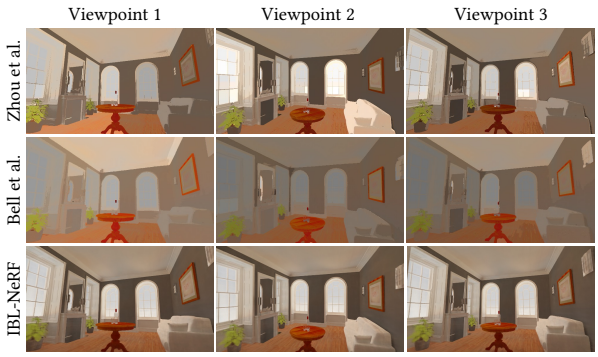


Figure 8. Visual comparison of albedo estimation between IBL-NeRF and single-image based methods.

cial role to disambiguate each components. When we remove $\mathcal{L}_{\text{prior}}$ the albedo contains illumination information which should belong irradiance, and the irradiance is clipped to the mean value by $\mathcal{L}_{I,\text{reg}}$. Also, without $\mathcal{L}_{I,\text{reg}}$, one cannot estimate correct irradiance especially on the surface with dark albedo. (e.g., Oven in Fig. 7 should have irradiance similar to nearby furniture, but the dark pixels encourage estimating lower irradiance without $\mathcal{L}_{I,\text{reg}}$.) Removing both priors shows the worst results. Without prior loss the accuracy of the view synthesis increases since the neural network can focus on minimizing the rendering loss with more flexibility.

4.2. Scene Editing

After IBL-NeRF decomposes intrinsic components, one can render realistic novel-view images of altered scenes by modifying the value of each component. For example, we replace roughness of the dining table and albedo of the lamp to edit KITCHEN scene in Fig. 1(d). We demonstrate more results in Fig. 9. In the first row of Fig. 9, we replace albedo of two kettles in VEACH-AJAR to green and red respectively while preserving illumination information. In the second row of Fig. 9, we reduce the roughness of the picture in frame, drawer, and closet door, which results in mirror-like material in BEDROOM scene. We also change the albedo of the middle door of the closet to white and the conference logo is marked on the left door by modifying roughness. In the third row of Fig. 9, we modify the albedo and roughness of the desk pad in our real-world scene to express the marble-like material. Also, one can insert 3D objects inside our trained neural volume with prefiltered radiance field. In Fig. 19, we add 3 objects with different roughness and transparency inside the KITCHEN. The red blobby object is highly reflective and the surrounding scene is clearly reflected on its surface. The green dragon also has a low roughness value but has translucency so the shape of the green kettle behind the object is visible. Finally, the blue teapot has high roughness value and moderate translucency.

(a) Original Scene (b) Edited Scene

Figure 9. Example of changing intrinsic components of the scene.

Figure 10. Example of adding new objects to the scene.

The blurry reflection on the teapot accounts for its a high roughness value. Additional samples of edited scenes with videos can be found in our supplementary material. Note that scene editing could be achieved similarly using Monte Carlo method with our prefiltered radiance, but IBL-NeRF outperforms them in terms of speed (Table 2, Infer time).

5. Conclusion

We propose IBL-NeRF, a neural volume representation with prefiltered radiance field. Our approach successfully decomposes the intrinsic components in a large-scale scene with an efficient approximation and prefiltered radiance field, which could not be processed in prior works with Monte Carlo integration of environment light. Furthermore, one can easily edit the scene by modifying each decomposed component or inserting 3D models in our neural volume. Although IBL-NeRF can handle both Lambertian reflection and specular reflection, IBL-NeRF has a limitation in expressing transparent objects or perfect-mirror reflection. One can resolve the ambiguity in a mirror with user interaction as [10].

References

[1] Sean Bell, Kavita Bala, and Noah Snavely. Intrinsic images in the wild. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014.

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

[3] Benedikt Bitterli. Rendering resources, 2016. https://benedikt-bitterli.me/resources/.

[4] 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.

[5] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, and Hendrik P.A. Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[6] Lechao Cheng, Chengyi Zhang, and Zicheng Liao. Intrinsic image transformation via scale space decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 656–665, 2018.

[7] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[8] Sylvain Duchêne, Clement Riant, Gaurav Chaurasia, Jorge Lopez-Moreno, Pierre-Yves Laffont, Stefan Popov, Adrien Bousseau, and George Drettakis. Multi-view intrinsic images of outdoors scenes with an application to relighting. *ACM Transactions on Graphics*, page 16, 2015.

[9] Elena Garces, Adolfo Munoz, Jorge Lopez-Moreno, and Diego Gutierrez. Intrinsic images by clustering. In *Computer graphics forum*, volume 31, pages 1415–1424. Wiley Online Library, 2012.

[10] Yuan-Chen Guo, Di Kang, Linchao Bao, Yu He, and Song-Hai Zhang. Nerfren: Neural radiance fields with reflections. *arXiv preprint arXiv:2111.15234*, 2021.

[11] James T Kajiya. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150, 1986.

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

[13] Juhyeon Kim and Young Min Kim. Fast and Lightweight Path Guiding Algorithm on GPU. In Sung-Hee Lee, Stefanie Zollmann, Makoto Okabe, and Burkhard Wünsche, editors, *Pacific Graphics Short Papers, Posters, and Work-in-Progress Papers*. The Eurographics Association, 2021.

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

[15] Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2475–2484, 2020.

[16] Zhengqi Li and Noah Snavely. Cgintrinsics: Better intrinsic image decomposition through physically-based rendering. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–387, 2018.

[17] Zhengqi Li and Noah Snavely. Learning intrinsic image decomposition from watching the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9039–9048, 2018.

[18] Wei-Chiu Ma, Hang Chu, Bolei Zhou, Raquel Urtasun, and Antonio Torralba. Single image intrinsic decomposition without a single intrinsic image. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 201–217, 2018.

[19] Abhimitra Meka, Mohammad Shafiei, Michael Zollhöfer, Christian Richardt, and Christian Theobalt. Real-time global illumination decomposition of videos. *ACM Transactions on Graphics (TOG)*, 40(3):1–16, 2021.

[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. In *European conference on computer vision*, pages 405–421. Springer, 2020.

[21] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5589–5599, October 2021.

[22] Rohit Pandey, Sergio Orts Escolano, Chloe Legendre, Christian Häne, Sofien Bouaziz, Christoph Rhemann, Paul Debevec, and Sean Fanello. Total relighting: Learning to relight portraits for background replacement. *ACM Trans. Graph.*, 40(4), jul 2021.

[23] Steven G Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, et al. Optix: a general purpose ray tracing engine. *Acm transactions on graphics (tog)*, 29(4):1–13, 2010.

[24] 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*, 32:8026–8037, 2019.

[25] Julien Philip, Sébastien Morgenthaler, Michaël Gharbi, and George Drettakis. Free-viewpoint indoor neural relighting from multi-view stereo. *ACM Trans. Graph.*, 40(5), sep 2021.

[26] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[27] Soumyadip Sengupta, Jinwei Gu, Kihwan Kim, Guilin Liu, David W Jacobs, and Jan Kautz. Neural inverse rendering

of an indoor scene from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8598–8607, 2019.

[28] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7495–7504, 2021.

[29] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022.

[30] 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. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 27171–27183. Curran Associates, Inc., 2021.

[31] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5610–5619, October 2021.

[32] 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.

[33] 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. *arXiv:2106.01970*, 2021.

[34] Tinghui Zhou, Philipp Krahenbuhl, and Alexei A Efros. Learning data-driven reflectance priors for intrinsic image decomposition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3469–3477, 2015.

A. BRDF Model

IBL-NeRF adapts the microfacet BRDF model of Unreal Engine [12] and approximates the surface reflectance property with a set of decomposed intrinsic terms. The BRDF (bidirectional reflectance distribution function) $f_r(\mathbf{x}, \omega_i, \omega_o)$ is the ratio between the incoming and outgoing radiance and it is a function of the surface location $\mathbf{x}$, incoming direction ω_i, and the outgoing direction ω_o. Then the outgoing radiance is evaluated by attenuating the incoming radiance L_i by the BRDF f_r, cosine term $\mathbf{n} \cdot \omega_i$ and integrating over all of the incoming directions [11]

$$L_o(\mathbf{x}, \omega_o) = \int_\Omega f_r(\mathbf{x}, \omega_i, \omega_o) L_i(\mathbf{x}, \omega_i)(\mathbf{n} \cdot \omega_i) d\omega_i, \quad (13)$$

where $\mathbf{n}$ is the surface normal at $\mathbf{x}$. The radiance is composed of two terms, namely the diffuse term $L_{o,\text{diff}}$ and the specular term $L_{o,\text{spec}}$. The distinction between the two terms stems from the conventional parameterization of BRDF into the two respective terms $f_r = f_{\text{diff}} + f_{\text{spec}}$.

A.1. Diffuse Reflection

We use a simple Lambertian model for the diffuse BRDF

$$f_{\text{diff}} = (1 - m)(1 - F(\omega_o, h))\frac{a}{\pi}, \quad (14)$$

where a is the surface albedo and m parameterizes how metallic it is[2]. While a and m are constants over incident angles, the diffuse term also varies according to viewing directions due to the Fresnel effect F. The Fresnel effect models how much light is reflected and refracted. The Unreal Engine employs the Fresnel-Schlick approximation [**?**]

$$F(\omega_o, \mathbf{h}) = F_0 + (1 - F_0)(1 - (\omega_o \cdot \mathbf{h}))^5, \quad (15)$$

where $\mathbf{h} = \frac{\omega_o + \omega_i}{\|\omega_o + \omega_i\|}$ is the halfway vector and $F_0 = \text{lerp}(0.04, a, m)$ is a constant term defined as the linear interpolation between 0.04 and a.

Now we can find the diffuse component of the outgoing radiance by combining Eq. (13) with (14):

$$\begin{aligned} &L_{o,\text{diff}}(\mathbf{x}, \omega_o) \\ &= (1 - m)\frac{a}{\pi} \int_\Omega (1 - F(\omega_o, \mathbf{h})) L_i(\mathbf{x}, \omega_i)(\mathbf{n} \cdot \omega_i) d\omega_i, \end{aligned} \quad (16)$$

since a and m are constants over ω_i. We can further simplify the integrand by assuming a constant Fresnel term. One naïve way is to replace the dependency of $\mathbf{h}$ with $\mathbf{n}$, and substitute $F(\omega_o, \mathbf{n})$. But such formulation largely deviates from the diffuse property and incurs excessive reflection near the edge. To alleviate the phenomenon, we inject roughness γ to the Fresnel term as [**?**]

$$F_\gamma(\omega_o, \mathbf{n}, \gamma) = F_0 + (\max(1 - \gamma, F_0) - F_0)(1 - (\mathbf{n} \cdot \omega_o))^5. \quad (17)$$

After factoring out the approximated Fresnel term F_γ, our diffuse component is formulated as following:

$$(1-m)(1-F_\gamma(\omega_o, \mathbf{n}, \gamma))a \underbrace{\left\{ \frac{1}{\pi} \int_\Omega L_i(\mathbf{x}, \omega_i)(\mathbf{n} \cdot \omega_i) d\omega_i \right\}}_{\text{Irradiance } I}. \quad (18)$$

The remaining integrand is the sum of total incident radiance received at point $\mathbf{x}$, which is also known as the *irradiance* I. In real-time image-based rendering, irradiance is fetched at the normal direction $\mathbf{n}$ from an additional environment map, which is precalculated and stored. However, IBL-NeRF implicitly represents irradiance at each position using MLP instead of an environment map.

A.2. Specular Reflection

The specular BRDF follows the Cook-Torrance model [**?**]

$$f_{\text{spec}} = \frac{D(\mathbf{h}, \mathbf{n}, \gamma) \cdot F(\omega_o, \mathbf{h}) \cdot G(\omega_i, \omega_o, \mathbf{n}, \gamma)}{4(\mathbf{n} \cdot \omega_o)(\mathbf{n} \cdot \omega_i)}. \quad (19)$$

Note that the roughness term γ, introduced in the approximate Fresnel term in Eq. (17), plays a crucial role in the specular reflection. In addition to the Fresnel equation defined in Eq. (15), the specular reflection is also dependent on the normal distribution function D and the geometry function G. The normal distribution is adopted from Trowbridge-Reitz GGX [**?**]

$$D(\mathbf{h}, \mathbf{n}, \gamma) = \frac{\alpha^2}{\pi((\mathbf{n} \cdot \mathbf{h})^2(\alpha^2 - 1) + 1)^2}, \quad (20)$$

where $\alpha = \gamma^2$. The geometry function G describes self-shadowing according to Smith's Schlick-GGX [**?**]

$$G(\omega_i, \omega_o, \mathbf{n}, \gamma) = G_{\text{Schlick}}(\omega_i, \mathbf{n}, \gamma) G_{\text{Schlick}}(\omega_o, \mathbf{n}, \gamma), \quad (21)$$

where $G_{\text{Schlick}}(\omega, \mathbf{n}, \gamma) = \frac{(\mathbf{n} \cdot \omega)}{(\mathbf{n} \cdot \omega)(1-k)+k}$ and $k = \frac{\alpha^2}{2}$. Basically, roughness γ affects the sharpness of angular distribution of reflected radiance.

The outgoing radiance of specular component involves a highly complex distribution compared to the diffuse component. The specular term can be computed using importance sampling as follows:

$$L_{o,\text{spec}}(\mathbf{x}, \omega_o) = \frac{1}{N} \sum_{k=1}^{N} \frac{f_{\text{spec}}(\mathbf{x}, \omega_i^k, \omega_o) L_i(\mathbf{x}, \omega_i)(\mathbf{n} \cdot \omega_i^k)}{p(\omega_i^k | \mathbf{x}, \mathbf{n}, \omega_o, \gamma)}.$$

The sampling PDF $p(\omega_i | \mathbf{x}, \mathbf{n}, \omega_o, \gamma)$ can be found from the normal distribution function D. [**?**]. We split the above sum

[2] Metal does not refract at all, so f_{diff} is attenuated by $(1 - m)$.

into two components, adapting the *split-sum approximation* [12]

$$\underbrace{\left\{\frac{1}{N}\sum_{k=1}^{N} L_i(\mathbf{x},\omega_i^k)\right\}}_{\simeq L_{\text{pref}}(\mathbf{x},\omega_r,\gamma)}\underbrace{\left\{\frac{1}{N}\sum_{k=1}^{N}\frac{f_{\text{spec}}(\mathbf{x},\omega_i^k,\omega_o)(\mathbf{n}\cdot\omega_i^k)}{p(\omega_i^k|\mathbf{x},\mathbf{n},\omega_o,\gamma)}\right\}}_{\simeq[F_0,1]^T\text{LUT}(\omega_o,\cdot\mathbf{n},\gamma)}. \tag{22}$$

The approximation separates the effect of lighting (first term) from BRDF (second term), and allows accelerated computation of radiance with precalculated maps. We further discuss the approximation of the two terms in the following subsections.

A.2.1 Prefiltered Radiance Fields

Considering the sampling PDF, the first term is dependent on three terms $[\omega_o, (\omega_o \cdot \mathbf{n}), \gamma]$. Such multiple dependencies greatly increase the amount of computation and memory required to store precalculated results. Thus, it is often assumed that $\omega_o = \mathbf{n} = \omega_r$, where $\omega_r = \text{mirror}(\omega_o, \mathbf{n})$ is the direction of mirror reflection of ω_o with respect to the surface normal. Note that ω_r is chosen since $p(\omega_i|\mathbf{x},\mathbf{n},\omega_o,\gamma) = \delta(\omega_r)$ if $\gamma = 0$. With this isotropic approximation, now the first term only depends on ω_r and γ. We precalculate the first term for each direction ω_r with different roughness values and store it as an environment map with several mipmap levels, which is known as *prefiltered environment map* L_{pref} [12] [3]. Similar to texture mipmap, we can fetch the prefiltered radiance of the desired roughness in the mirror-reflected direction ω_r using trilinear interpolation as

$$L^j_{\text{pref}}(\mathbf{x},\omega_r,\gamma) = \sum_j w^j(\gamma) L^j_{\text{pref}}(\mathbf{x},\omega_r), \tag{23}$$

where L^j_{pref} is jth mipmap and $w^j(\gamma)$ is the weight of γ for jth mipmap. In IBL-NeRF, the prefiltered environment map is stored implicitly in MLP, as irradiance in the previous section. Therefore we will rather refer it as *prefiltered radiance fields*.

A.2.2 Precomputing BRDF Integration

By substituting the Fresnel term in Eq. (15), the second term in Eq. (22) can be formulated as an affine function of F_0:

$$\begin{aligned}
&\int_\Omega f_{\text{spec}}(\mathbf{x},\omega_i,\omega_o)(\mathbf{n}\cdot\omega_i)d\omega_i \\
&\quad= F_0\int_\Omega \frac{f_{\text{spec}}(\mathbf{x},\omega_i,\omega_o)}{F(\omega_o,\mathbf{h})}(1-(1-(\omega_o\cdot\mathbf{h}))^5)(\mathbf{n}\cdot\omega_i)d\omega_i \\
&\qquad+\int_\Omega \frac{f_{\text{spec}}(\mathbf{x},\omega_i,\omega_o)}{F(\omega_o,\mathbf{h})}(1-(\omega_o\cdot\mathbf{h}))^5(\mathbf{n}\cdot\omega_i)d\omega_i \\
&\quad= [F_0,1]^T\text{LUT}(\omega_o\cdot\mathbf{n},\gamma).
\end{aligned}$$

[3] We will consider L_{pref} from incident direction similar to L_i

After integration over Ω, we can remove the dependency on ω_i, and the scale and bias term depend only on γ and $(\mathbf{n}\cdot\omega_o)$ [12]. The integral can be precalculated and stored as a 2D lookup texture (LUT), as illustrated in Fig.2 (g) in main manuscript.

A.3. Final Radiance Approximation

Combining previous sections, the outgoing radiance $L_o(x,\omega_o)$ is approximated as following without Monte Carlo integration,

$$\begin{aligned}
L_o(\mathbf{x},\omega_o) &= (1-m)\times(1-F_\gamma(\omega_o,\mathbf{n},\gamma))\times a\times I \\
&\quad+ L_{\text{pref}}(\mathbf{x},\omega_r,\gamma)\times[F_0,1]^T\text{LUT}(\omega_o\cdot\mathbf{n},\gamma),
\end{aligned} \tag{24}$$

where the first term is the diffuse component and the second term is the specular component. The formulation in Eq. (24) is also visualized in Fig.2 of main manuscript. Given the precomputed maps, we only need to estimate the surface normal, albedo, irradiance, and roughness of the scene to find the diffuse and specular components.

B. Implementation Details

IBL-NeRF is implemented with PyTorch [24]. We will release the code and data publicily upon publication.

B.1. Network Architecture

Following the vanilla NeRF [20], we encode the input position ($\mathbf{x}$) and direction (ω) with 10 and 4 levels of periodic functions, respectively, before feeding them into our network. Also, we use 8 fully-connected ReLU layers, each with 256 channels for the position MLP with a skip connection in the fifth layer's activation. Given the output feature of the positional encoder, additional layers are trained to output the volume density (σ), albedo (a), irradiance (I), and roughness (γ). The encoded direction vector is concatenated to the feature vector from the position MLP, and it is sent as the input to the direction MLP. The direction MLP is a single fully-connected layer with 128 channels. The final parallel 4 layers emit the prefiltered radiance field (L^j_{pref}) for each roughness level $0 \le j \le 3$.

B.2. Hyperparameter Setup & Training Details

In this section, we report the hyperparameter setup used in experiments for IBL-NeRF. We use the Adam optimizer [14] with a learning rate of 5×10^{-4} following the vanilla NeRF, and use the default values for other hyperparameters of Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-7}$). We sample 512 ray batches per training step. We adop importance sampling strategy from NeRF, which samples $N_c = 64$ coordinates along the ray for the coarse volume and $N_f = 128$ samples in the fine volume. To sum up, we

(a) KITCHEN (b) BATHROOM2

(c) LIVINGROOM2 (d) DININGROOM

Figure 11. Samples from our synthetic dataset

sample $N_s = 192$ coordinate samples per ray batch. However, we use the 64 uniform samples for the reflected ray (i.e. $N_r = 64$). For stable training, we train IBL-NeRF to optimize the following objective function for each pixel of the camera ray $r = (\mathbf{c}, -\omega_o)$:

$$\mathcal{L} = \mathcal{L}_{\text{pref}} \tag{25}$$

for the first 10k steps,

$$\mathcal{L} = \mathcal{L}_{\text{pref}} + \mathcal{L}_{\text{render}} \tag{26}$$

before 100k steps, and the full loss function

$$\mathcal{L} = \mathcal{L}_{\text{pref}} + \mathcal{L}_{\text{render}} + \mathcal{L}_{\text{prior}} + \lambda_{I,\text{reg}}\mathcal{L}_{I,\text{reg}} \tag{27}$$

after 100k steps. We set $\lambda_{I,\text{reg}} = 0.1$. Furthermore, we freeze the roughness MLP after 100k steps, which gives better results.

Baselines For all the baselines with Monte Carlo approaches, we use 32×16 resolution environment light following NeRFactor [33]. Also, we use equal-area stratified sampling over hemisphere with 64 samples.

B.3. Data

Synthetic Data We use the 12 realistic synthetic scenes [BATHROOM, BATHROOM2, BEDROOM, CLASSROOM, DININGROOM, KITCHEN, LIVINGROOM, LIVINGROOM2, LIVINGROOM3, STAIRCASE, VEACH-AJAR, VEACH-DOORSIMPLE] released by [3] in our experiments. We collect images from each scene by randomly rotating (at most 30 degree deviation) and translating (at most 10 percent compared to total scene size) camera poses from the original pose. We remove invalid cases where the camera is trapped within an object in the scene and cannot observe anything. For each scene, we render 100 multi-view images for train set and test set with the OptiX [23] based path tracing renderer [13] with 1024 samples per pixel. Furthermore, we obtain ground-truth intrinsic images including albedo, roughness, irradiance in order to measure the quantitative results, which are not available for training. A few sample images from our dataset are presented in Fig. 11.

Real-world Data We use 8 real-world scenes from ScanNet [7] dataset following NerfingMVS [31]. All images have 484×648 resolution. We use same train/test split for all of ScanNet scenes with NerfingMVS. 32 images are used to train IBL-NeRF, 8 images are used for evaluation. Furthermore, we collect our own real-world scene with a smartphone camera. All of the capturing setups including white balance and exposure are set to auto while capturing. We use 80 images for training, 4 images for evaluation.

C. Geometry learned from IBL-NeRF

To evaluate the radiance at $\mathbf{x}_{\text{surf}}$, we need to evaluate normal at the estimated terminal depth $d(\mathbf{c}, -\omega_o)$ for each evaluated ray. The normal can be found from the gradient of surface geometry, which can be estimated from the volume density σ of NeRF. The estimated geometry exhibits unique noise characteristics derived from NeRF training, and we observed that the quality of normal varies depending on how we impose the gradient, as presented in Fig. 12.

Previous works [28, **?**] calculate normal from the direct gradient of the volume density σ with respect to the position x

$$\mathbf{n}(\mathbf{x}) = -\frac{\nabla_{\mathbf{x}}\sigma(\mathbf{x})}{\|\nabla_{\mathbf{x}}\sigma(\mathbf{x})\|}. \tag{28}$$

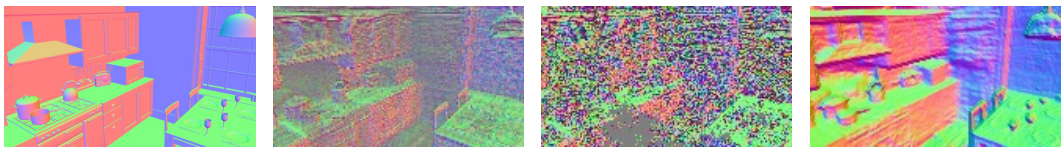
We can substitute Eq.(28) and apply the volume rendering equation to estimate normal along the ray direction (Fig. 12-(b)), or we can calculate the normal only at the terminal point $\mathbf{x}_{\text{surf}}$ (Fig. 12-(c)). However, both methods produce noisy results.

Interestingly, we found that we can obtain better normal by differentiating the depth map on the rendered image plane (Fig. 12-(d)). After deriving the depth map from the terminal depth value at every pixel of the image, we can numerically differentiate the depth values in the image domain by calculating the depth differences between adjacent pixels and estimate normals. Inspired by the successful result from the depth map, we tried to directly differentiate depth in the NeRF pipeline. Normal at $\mathbf{x}_{\text{surf}}$ could be calculated from gradient of depth $d(\mathbf{x}, \omega)$ with respect to either $\mathbf{x}$ or ω:

$$\mathbf{n}(\mathbf{x}_{\text{surf}}) = \frac{\nabla_{\mathbf{x}} d(\mathbf{x}, \omega)}{\|\nabla_{\mathbf{x}} d(\mathbf{x}, \omega)\|} \tag{29}$$

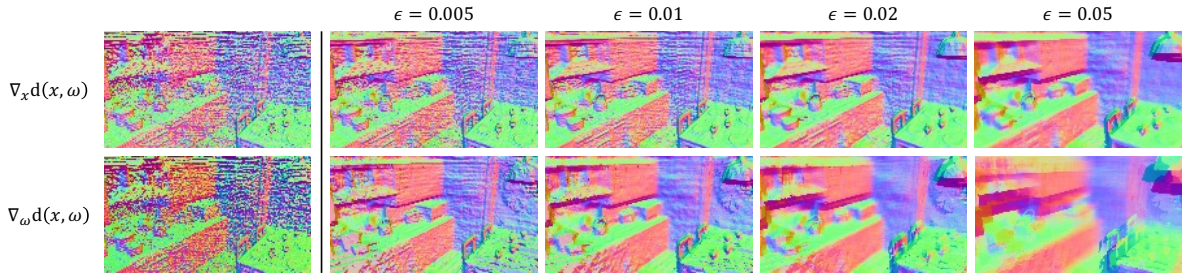
and

$$\mathbf{n}(\mathbf{x}_{\text{surf}}) = \frac{(\nabla_{\omega} d(\mathbf{x}, \omega)) - \omega}{\|(\nabla_{\omega} d(\mathbf{x}, \omega)) - \omega\|}. \tag{30}$$



(a) Ground truth (b) $\nabla_x \sigma(x)$ at volume (c) $\nabla_x \sigma(x)$ at surface (d) From inferred depth map

Figure 12. Comparison between several methods to estimate ($n(x_{\text{surf}})$) and ground-truth normal.



(a) Auto Differentiation (b) Numerical Differentiation

Figure 13. Comparison between normals with different ϵ values

Note that we should subtract ω from the gradient with respect to the angle ω. Both methods give better results than directly differentiating σ (Fig. 13-(a)).

When we compute the gradient of the depth in the actual rendering pipeline, we apply numerical differentiation instead of auto differentiation. The auto differentiation back-propagates the gradients through the neural network and involves more computation than numerical differentiation. The numerical differentiation, on the other hand, simply evaluates neighboring values within ϵ and uses the differences. Not only it is faster, but also it is more robust to noise as shown in Fig. 13. We, therefore, adopt the numerical differentiation with a small value ϵ.There exist a trade-off in choosing appropriate ϵ; large ϵ can produce smoother normal but loses detail. We found that $\epsilon = 0.01$ gives the best result in practice (Fig. 13-(b)).

D. Prefiltered Radiance

D.1. Prefiltered Radiance in Image Space

In this section, we discuss the choice of Gaussian convolution kernels, which approximates the radiance fields with different roughness γ. The prefiltered radiance L_{pref} in our approximation is calculated in the image space S of the current view. Recall that, for specular reflection, the radiance field is fetched from the set of the radiance values with different sharpness, according to the roughness value of the surface point. The radiance value for $\mathbf{x}$ along the direction ω is

$$L^j_{\text{pref}}(\mathbf{x}, \omega) = \mathbb{E}_{\omega_i \sim p(\omega_i|\mathbf{x},\omega,\gamma_j)}[L_i(x, \omega_i)] \tag{31}$$

$$= \int_\Omega L_i(\mathbf{x}, \omega_i) p(\omega_i|\mathbf{x}, \omega, \gamma_j) d\omega_i \tag{32}$$

$$= \int_S L_i(s_i) p_S(s_i|\mathbf{x}, \omega, \gamma_j) ds \tag{33}$$

where s_i is the screen space coordinate that corresponds to direction ω_i. The jth radiance field contains the approximated values for the roughness value of γ_j. The sampling probability on the screen space p_S could be calculated as following

$$p_S(s_i|\mathbf{x}, \omega, \gamma_j) = p(h|\mathbf{x}, \omega, \gamma_j) \left\| \frac{\partial h}{\partial \omega_i} \right\| \left\| \frac{\partial \omega_i}{\partial s_i} \right\| \tag{34}$$

$$= D(\mathbf{h}, \omega, \gamma_j)(\mathbf{h} \cdot \omega) \left(\frac{1}{4(\omega_i \cdot \mathbf{h})} \right) \left(\frac{(\omega_i \cdot v)}{f^2 + s_i^2} \right), \tag{35}$$

where $\mathbf{h}$ is the halfway vector between ω and ω_i, D is the normal distribution function, v is the viewing direction of the camera, and f is focal length of the camera. Now we assume $\omega = v$ in order to use a convolution kernel that is globally consistent. Then the convolution kernel for roughness γ_j in the image space can be designed as

$$K^j(s_i) \propto p_S(s_i|\mathbf{x}, \omega, \gamma_j) \tag{36}$$

for each pixel s_i. Thus $L^j_{\text{pref}}(\mathbf{x}, \omega) = K^j(L(s))$, where s is the screen space coordinate that corresponds to ω. The examples of K^j are plotted in Fig. 14 for different roughness values. The overall shape is similar to that of Gaussian

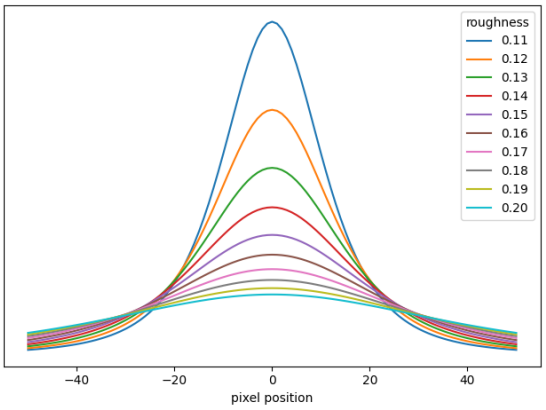


Figure 14. Kernel shapes in screen space for different roughness values.

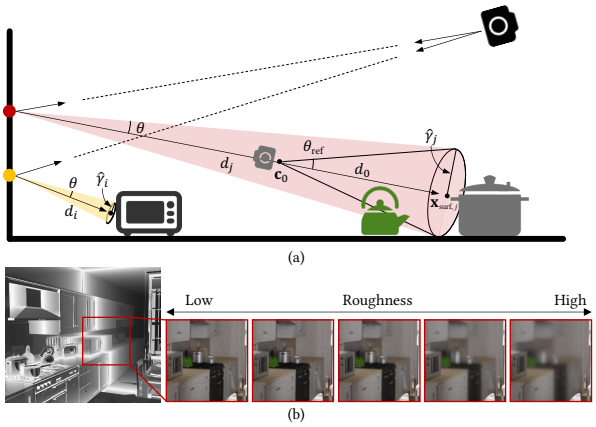


Figure 15. (a) The kernel sizes of image-space filtering depends on the observation angle ($\tan\theta$) while the effective kernel in the 3D spatial domain ($\hat{\gamma}$) depends on the distance to the object. (b) We illustrate the specular reflection on the surface of the back wall in KITCHEN scene. The left image shows the distance to the hit point reflected from the back wall, indicated with a red box. The right image shows the reflected radiance when the wall is assigned with different roughness values. Even though the roughness of the wall is constant, the reflection is blurred with different roughness γ^{ref} that accounts for distance variations; the reflection of closer object (microwave) is sharper, whereas the farther objects (kettle and pot) are blurrier.

function, which is used to approximate $L^j_{\text{pref}}(\mathbf{x}, \omega)$ in our implementation. We did not rigorously calibrate the parameters of Gaussian functions to K^j, which could be further studied in the future work.

D.2. Convolution Level Adaptation

While image-space filtering is an efficient means to aggregate neighboring rays, the approximate filter size to be applied on the image should depend on both the surface roughness and the distance that the reflected ray travels, denoted d. The size of the image filter is determined by the tangent of the observation angle, $\tan\theta$ which is assumed to be proportional to the convolution level of prefiltered radiance γ. Fig. 15(a) shows the effective kernel size for image-space filtering and the corresponding roughness values. The reflection ray at the red point hits farther objects than the yellow point and needs to be blurred with a larger kernel even though observing the left wall having a constant roughness. When the filter is projected near the surface point $\mathbf{x}_{\text{surf}}$, the same observation angle θ corresponds to different sizes of effective range, denoted $\hat{\gamma} = d\tan\theta$, where d is the distance to $\mathbf{x}_{\text{surf}}$.

However, it is infeasible to consider different d values for every surface point shown on the radiance field, and moreover, the distances are unknown before training the neural network. Therefore we simply train L_{pref} assuming a constant distance d_0 for all the pixels, and we set d_0 to be the mean value of the near and far plane. It is illustrated in Fig. 15(a) where the small virtual camera observes the hit point of reflected radiance from d_0, and therefore assuming the same $\hat{\gamma}$ with given level γ. At the inference phase, we can calculate the reflected distance d and compensate for the discrepancy compared to the trained depth d_0. Specifically, we scale the inferred roughness γ with the depth ratio and and use $\gamma^{\text{ref}} = (d/d_0)\cdot\gamma$ to find the appropriate level of prefilter. Fig. 15(b) shows the effect of depth on the normalized filter size. When the objects are farther from the hit point (kettle and pot), the reflected radiance is blurrier than the closer ones (microwave).

D.3. Prefiltered Radiance at Reflected Direction

Prefiltered radiance L_{pref} at $\mathbf{x}_{\text{surf}}$ toward ω_r trained with screen-space prefiltered radiance approximation is visualized in Fig. 16. Given the predefined set of roughness values γ_j, L^j_{pref} represents the jth level prefiltered radiance. First four rows of Fig. 16 show L^j_{pref} with different levels. Since the higher level of j performs convolution using a Gaussian kernel with a wider range, L^j_{pref} with higher j looks more blurry. Fifth and sixth row show roughness γ and normalized roughness ($\gamma^{\text{ref}} = (d/d_0)\cdot\gamma$) which are used for prefiltered radiance fetching index. d is distance from $\mathbf{x}_{\text{surf}}$ to the next hit point along ω_r. Last two rows show L_{pref}s, which is the result of trilinear interpolation of L^j_{pref}'s whose weights are deduced from γ^{ref} and γ respectively. Note that without using normalization, the convolution occurs with a constant kernel regardless of d, which is not physically correct (orange box).

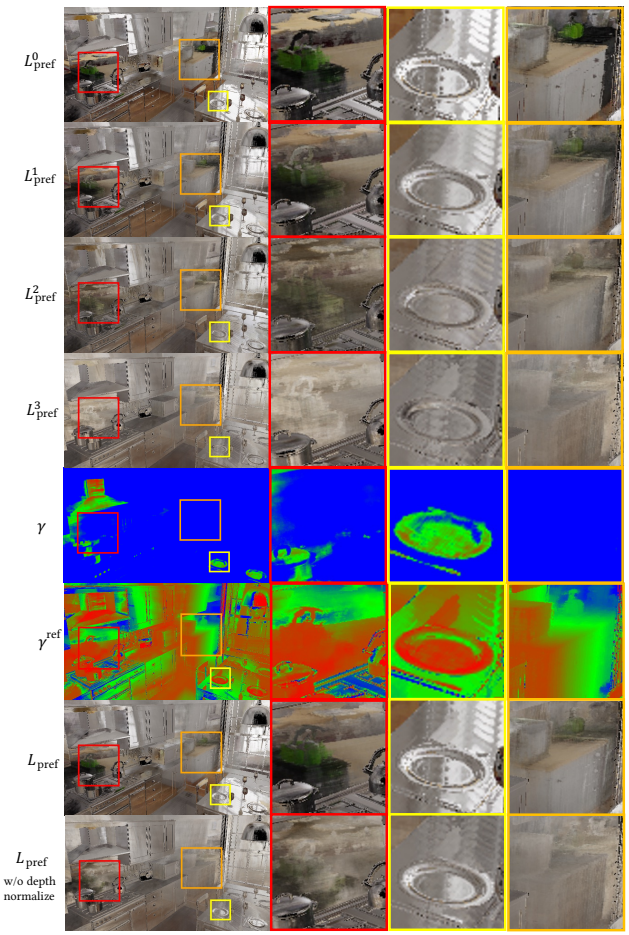


Figure 16. Prefiltered radiance with different levels (L^j_{pref}). The 5, 6th row shows the roughness value γ and normalized roughness value γ^{ref} respectively, where red indicates low and blue indicates high. The last two rows show the combined results after trilinear interpolation (L_{pref}) using γ^{ref} and γ respectively.

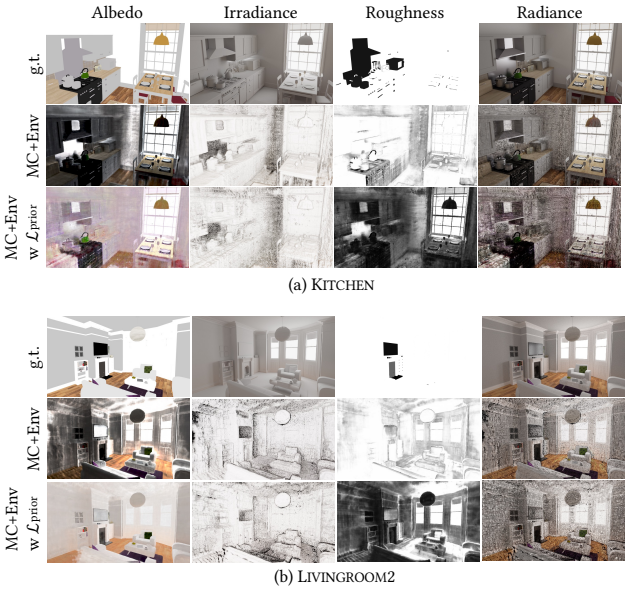


Figure 17. Qualitative results of MC + Env baseline with applying $\mathcal{L}_{\text{prior}}$.

(a) Original Scene (b) Edited Scene

Figure 18. Failure cases of our scene editing task.

E. Additional Results

E.1. Scene Editing

We display additional examples of inserting 3D object into our learned neural volume in Fig. 19. We vary roughness, reflectivity, and translucency with different levels. Thanks to our prefiltered radiance, one can render high-quality image with different roughness and transparency.

Also, we report a failure case of our scene editing task in Fig. 18. We lower the roughness of the picture in the frame in the VEACH-AJAR. The viewpoints in the training image set of the VEACH-AJAR scene are highly restricted. Most of the viewpoints are facing the front of the desk where the kettles are placed. Since there is no visual information in the backward region, the prefiltered radiance is poorly optimized in the unseen area. Therefore, failure in editing scenes with restricted viewpoint is a natural result.

E.2. View Synthesis & Intrinsic Decomposition

First, we report the second baseline method (MC + Env) with $\mathcal{L}_{\text{prior}}$ incorporated in Fig. 17. We observe that Monte Carlo approach with environment light shows inferior intrinsic decomposition performance with our albedo prior loss. Especially, MC + Env totally fails to estimate valid roughness with $\mathcal{L}_{\text{prior}}$.

In addition to Fig. 7 in the main manuscript, we display more quantitative results of novel view synthesis and intrinsic decomposition in Fig. 20, 21, 22 and 23.

Figure 19. Additional samples of inserting new objects inside the scene.

References

[1] Sean Bell, Kavita Bala, and Noah Snavely. Intrinsic images in the wild. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014.

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

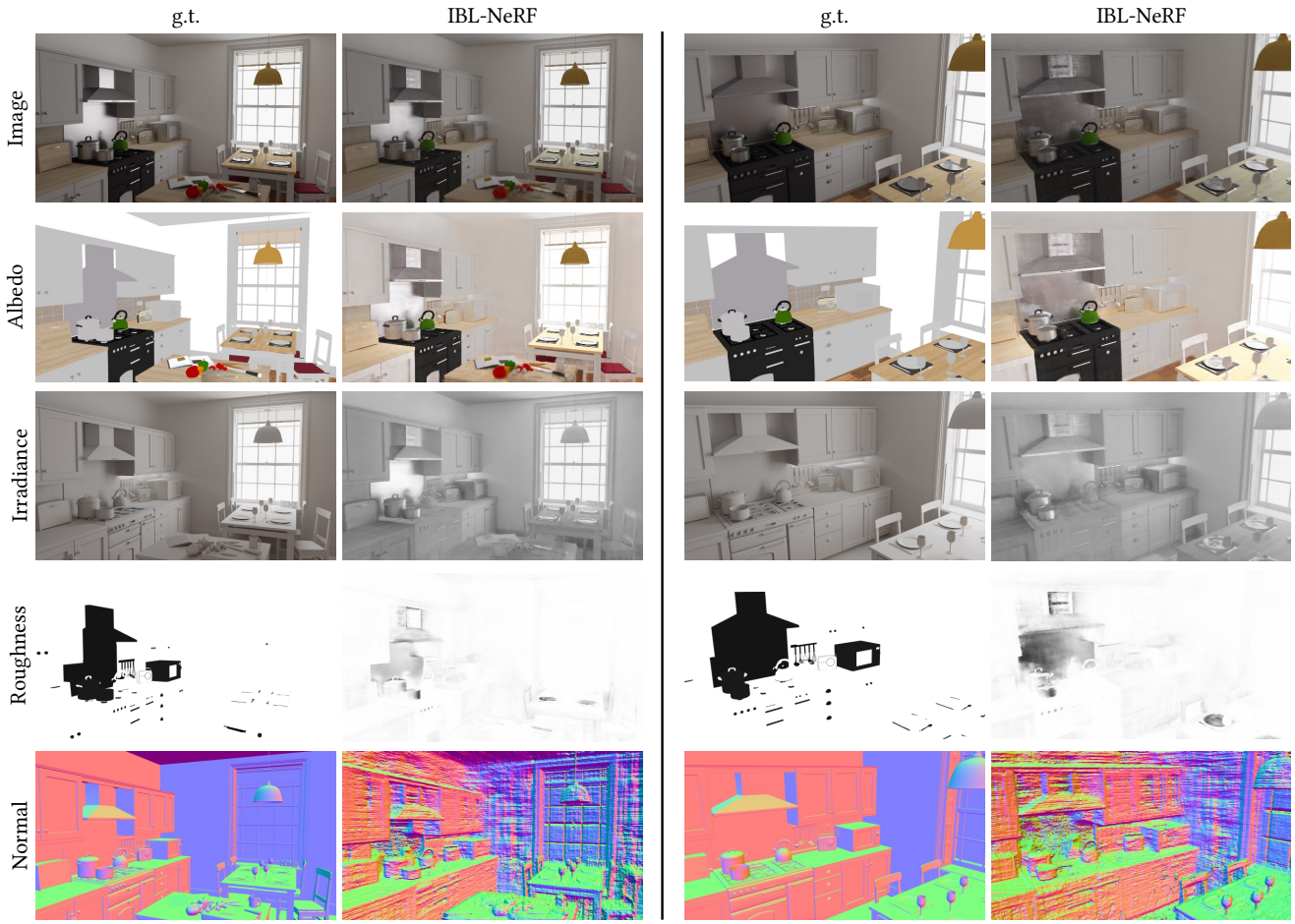


Figure 20. Additional qualitative results of novel view image synthesis and intrinsic decomposition in KITCHEN.

[3] Benedikt Bitterli. Rendering resources, 2016. https://benedikt-bitterli.me/resources/.

[4] 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.

[5] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, and Hendrik P.A. Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[6] Lechao Cheng, Chengyi Zhang, and Zicheng Liao. Intrinsic image transformation via scale space decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 656–665, 2018.

[7] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[8] Sylvain Duchêne, Clement Riant, Gaurav Chaurasia, Jorge Lopez-Moreno, Pierre-Yves Laffont, Stefan Popov, Adrien Bousseau, and George Drettakis. Multi-view intrinsic images of outdoors scenes with an application to relighting. *ACM Transactions on Graphics*, page 16, 2015.

[9] Elena Garces, Adolfo Munoz, Jorge Lopez-Moreno, and Diego Gutierrez. Intrinsic images by clustering. In *Computer graphics forum*, volume 31, pages 1415–1424. Wiley Online Library, 2012.

[10] Yuan-Chen Guo, Di Kang, Linchao Bao, Yu He, and Song-Hai Zhang. Nerfren: Neural radiance fields with reflections. *arXiv preprint arXiv:2111.15234*, 2021.

[11] James T Kajiya. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150, 1986.

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

[13] Juhyeon Kim and Young Min Kim. Fast and Lightweight Path Guiding Algorithm on GPU. In Sung-Hee Lee, Stefanie Zollmann, Makoto Okabe, and Burkhard Wünsche, editors, *Pacific Graphics Short Papers, Posters, and Work-in-Progress Papers*. The Eurographics Association, 2021.

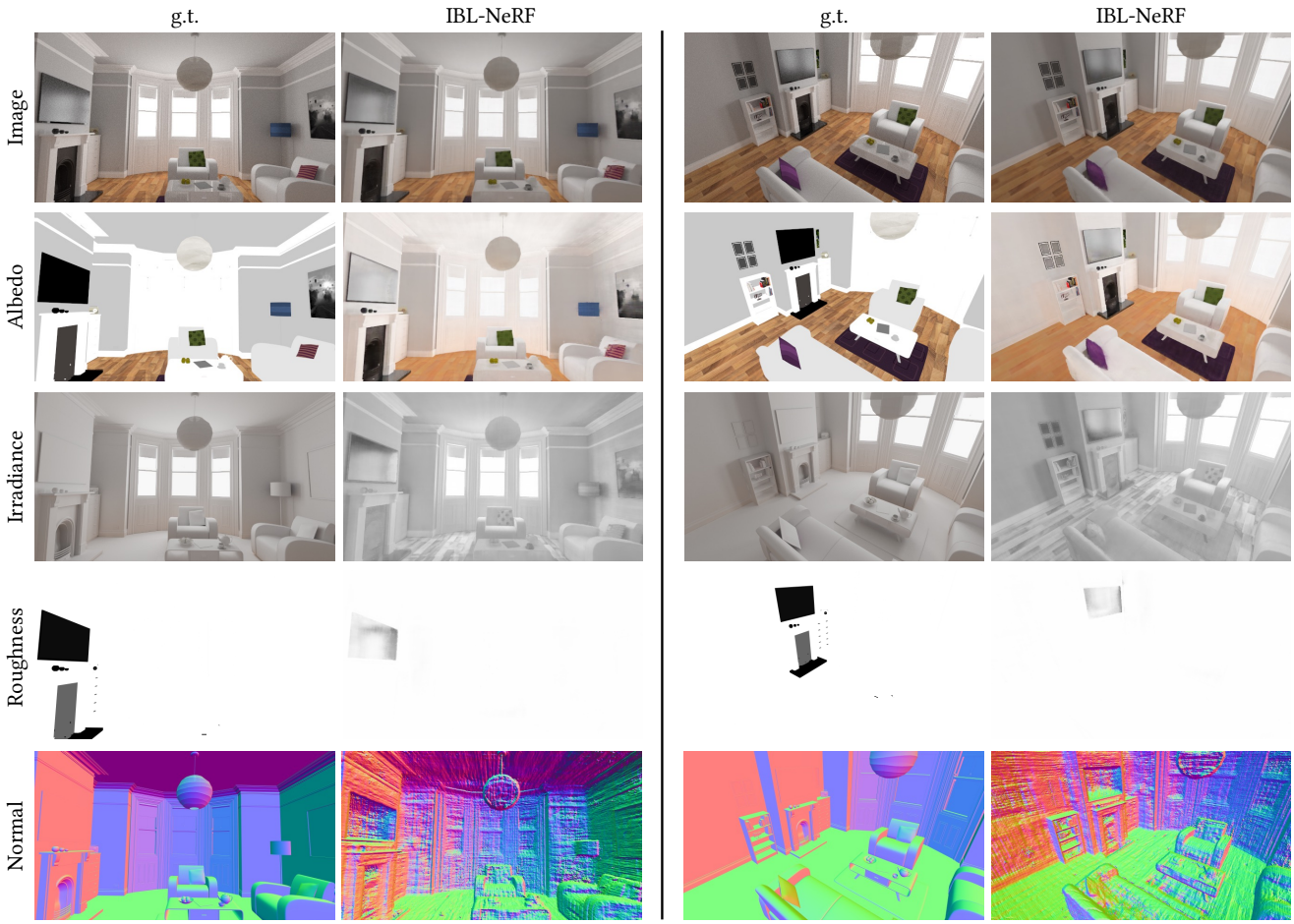


Figure 21. Additional qualitative results of novel view image synthesis and intrinsic decomposition in LIVINGROOM2.

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

[15] Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2475–2484, 2020.

[16] Zhengqi Li and Noah Snavely. Cgintrinsics: Better intrinsic image decomposition through physically-based rendering. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–387, 2018.

[17] Zhengqi Li and Noah Snavely. Learning intrinsic image decomposition from watching the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9039–9048, 2018.

[18] Wei-Chiu Ma, Hang Chu, Bolei Zhou, Raquel Urtasun, and Antonio Torralba. Single image intrinsic decomposition without a single intrinsic image. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 201–217, 2018.

[19] Abhimitra Meka, Mohammad Shafiei, Michael Zollhöfer, Christian Richardt, and Christian Theobalt. Real-time global illumination decomposition of videos. *ACM Transactions on Graphics (TOG)*, 40(3):1–16, 2021.

[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. In *European conference on computer vision*, pages 405–421. Springer, 2020.

[21] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5589–5599, October 2021.

[22] Rohit Pandey, Sergio Orts Escolano, Chloe Legendre, Christian Häne, Sofien Bouaziz, Christoph Rhemann, Paul Debevec, and Sean Fanello. Total relighting: Learning to relight portraits for background replacement. *ACM Trans. Graph.*, 40(4), jul 2021.

[23] Steven G Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, et al.

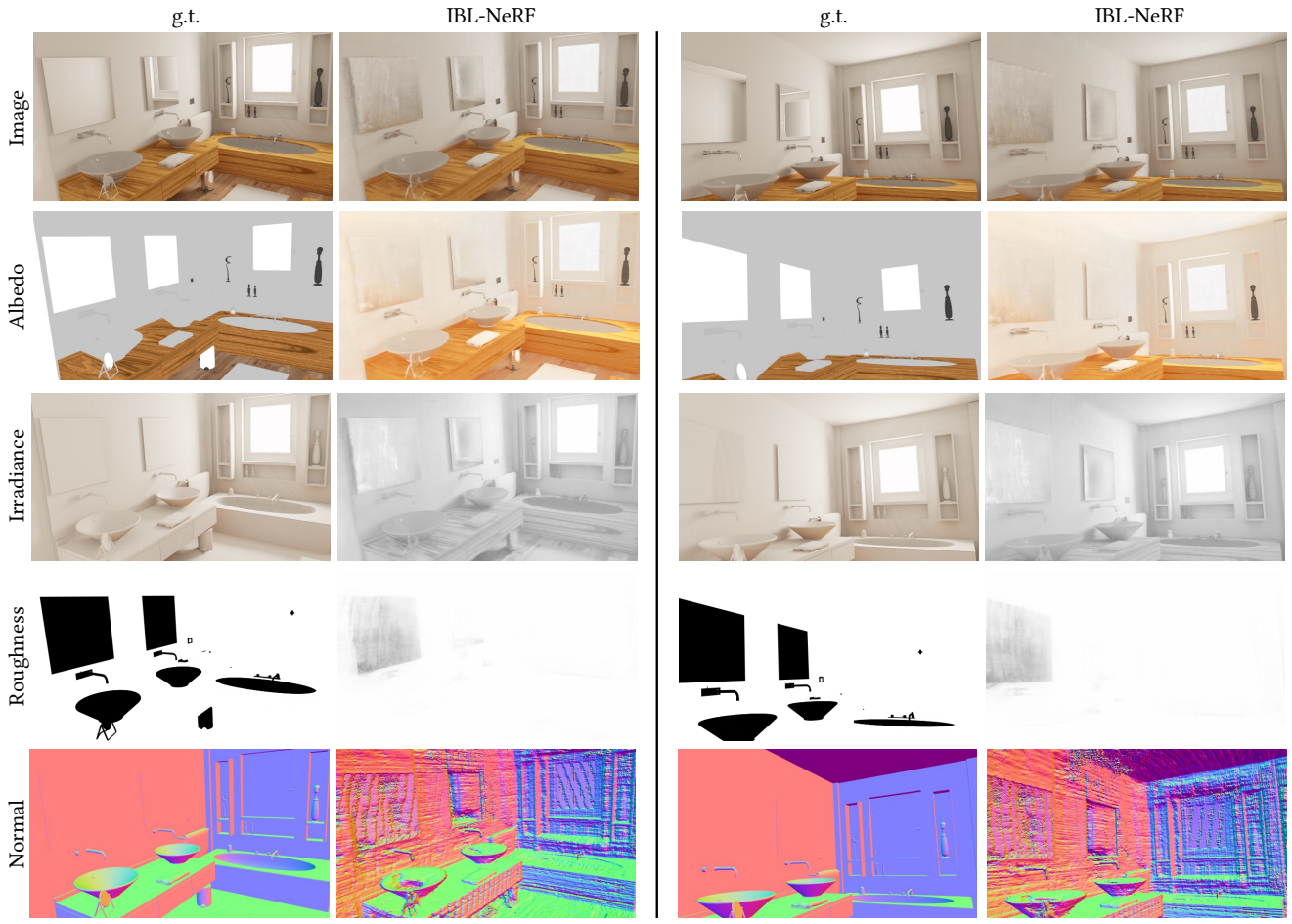


Figure 22. Additional qualitative results of novel view image synthesis and intrinsic decomposition in BATHROOM2.

Optix: a general purpose ray tracing engine. *Acm transactions on graphics (tog)*, 29(4):1–13, 2010.

[24] 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*, 32:8026–8037, 2019.

[25] Julien Philip, Sébastien Morgenthaler, Michaël Gharbi, and George Drettakis. Free-viewpoint indoor neural relighting from multi-view stereo. *ACM Trans. Graph.*, 40(5), sep 2021.

[26] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[27] Soumyadip Sengupta, Jinwei Gu, Kihwan Kim, Guilin Liu, David W Jacobs, and Jan Kautz. Neural inverse rendering of an indoor scene from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8598–8607, 2019.

[28] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7495–7504, 2021.

[29] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022.

[30] 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. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 27171–27183. Curran Associates, Inc., 2021.

[31] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5610–5619, October 2021.

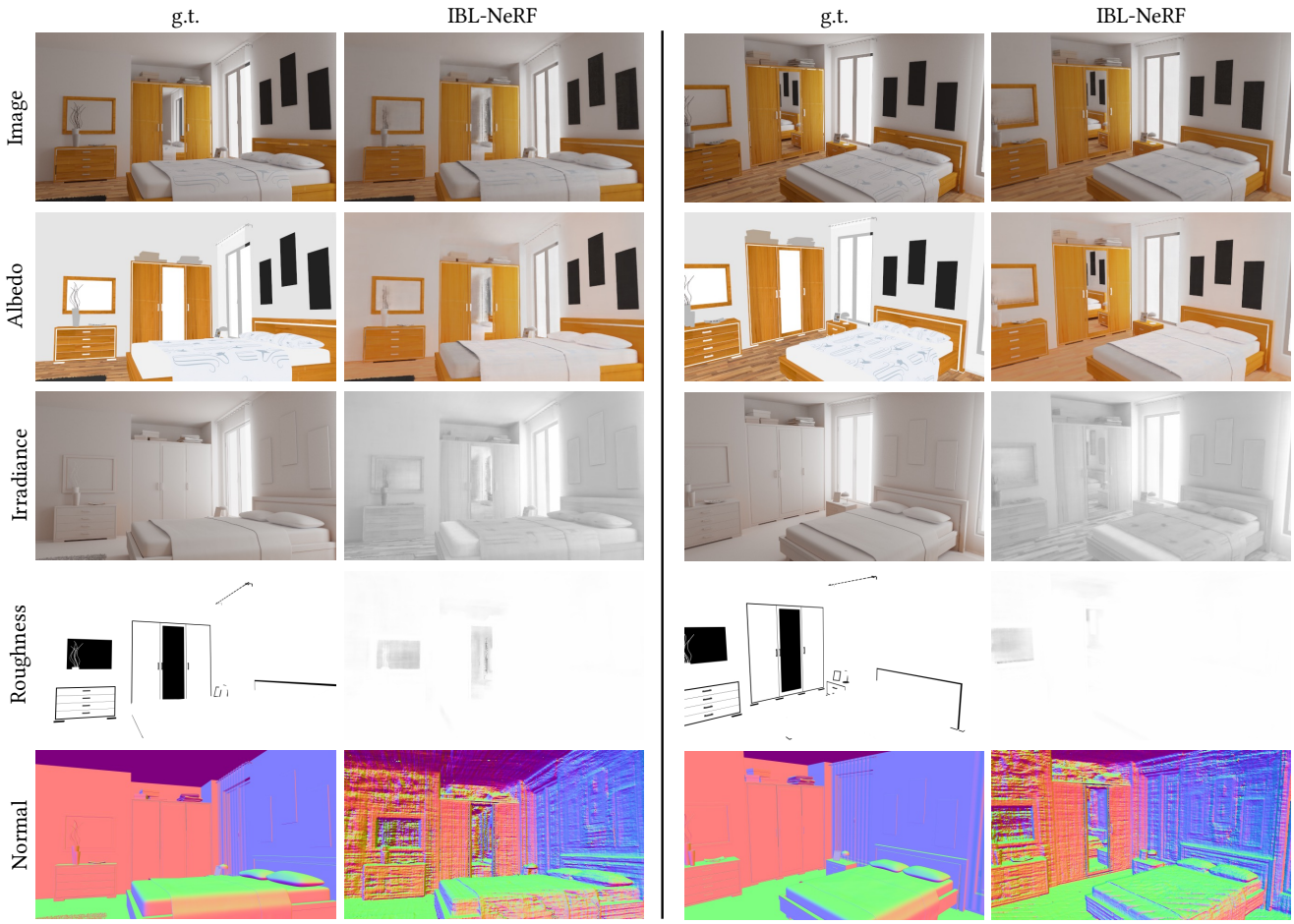


Figure 23. Additional qualitative results of novel view image synthesis and intrinsic decomposition in BEDROOM.

[32] 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.

[33] 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. *arXiv:2106.01970*, 2021.

[34] Tinghui Zhou, Philipp Krahenbuhl, and Alexei A Efros. Learning data-driven reflectance priors for intrinsic image decomposition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3469–3477, 2015.