Neural Global Illumination: Interactive Indirect Illumination Prediction under Dynamic Area Lights

Duan Gao, Haoyuan Mu, Kun Xu

Abstract—We propose neural global illumination, a novel method for fast rendering full global illumination in static scenes with dynamic viewpoint and area lighting. The key idea of our method is to utilize a deep rendering network to model the complex mapping from each shading point to global illumination. To efficiently learn the mapping, we propose a neural-network-friendly input representation including attributes of each shading point, viewpoint information, and a combinational lighting representation that enables high-quality fitting with a compact neural network. To synthesize high-frequency global illumination effects, we transform the low-dimension input to higher-dimension space by positional encoding and model the rendering network as a deep fully-connected network. Besides, we feed a screen-space neural buffer to our rendering network to share global information between objects in the screen-space to each shading point. We have demonstrated our neural global illumination method in rendering a wide variety of scenes exhibiting complex and all-frequency global illumination effects such as multiple-bounce glossy interreflection, color bleeding, and caustics.

Index Terms—Global illumination, Deep Learning

1 Introduction

Fast global illumination (GI) is an important but challenging research problem. In photorealistic rendering, global illumination provides many realistic visual effects such as color bleeding, glossy interreflection, and caustics. Existing global illumination rendering methods can be classified into offline rendering methods and interactive rendering methods. Classic offline rendering methods such as path tracing [26, 33, 63] and photon mapping [25, 19, 18] can achieve photorealistic rendering quality. However, these methods are time-consuming and take minutes or even hours to render a single noise-free frame which makes them not suitable for interactive applications.

Existing interactive global illumination approaches could be classified into three categories. The first category includes approximated methods such as screen-space rendering methods [49, 48, 40, 68] or volume-based methods [28, 29, 8]. Such methods are usually limited to low-frequency global illumination effects. The second category exploits filtering [51] or denoising [5] to reconstruct high-quality images from the noisy path-traced images generated at low sample rates. However, such methods cannot achieve real-time performance for high-quality global illumination due to the high cost of ray tracing. The third category is precomputation-based methods including lightmaps [23] and precomputed radiance transfer [54]. However, existing precomputation-based methods are usually restricted to certain conditions, i.e., static lighting [23, 58] or point light sources [47].

To address these limitations, we propose neural global illumination for generating rich global illumination effects in static scenes under dynamic area light sources. The motivation of our method is to reduce the gap between offline and interactive methods by applying deep learning to the global illumination problem. The key idea of our method is to utilize a deep neural network to represent the complex mapping from input scene information (including attributes of each shading point, the viewpoint, and the incident lighting information) to global illumination. Our method can be regarded as an advancement on classic precomputation-based methods (e.g. PRT, lightmaps, and light probes) and supports full global illumination for glossy materials and dynamic area lighting. The training data generation and network training serve as sampling and fitting of the scene's radiance fields. The trained network is a compact learning-based scene representation that can generate high-frequency global illumination during runtime rendering. Our method can be applied to interactive applications that rely on precomputed global illumination techniques (similar configuration as PRT, i.e. static scenes with dynamic lighting).

We use a deep fully-connected neural network (also known as multilayer perceptron or MLP) to model the global illumination. Global illumination is a high-dimensional, highly non-linear function, thus learning high-frequency global illumination effects efficiently with a compact neural network is non-trivial. We introduce three strategies to address the issue. The first strategy is to map the low-dimension input vector into higher dimensional space by positional encoding technique [62, 59]. Positional encoding enables the MLP to learn high-frequency global illumination successfully. The second strategy is to represent the incident lighting by a combinational lighting representation that takes advantage of different lighting cues which is crucial for generalizing our method in complex scenes with dynamic area light sources. The last strategy is to share global information between shading points in screen space by a

- *Duan Gao, Haoyuan Mu, Kun Xu are with BNRist, Department of Computer Science and Technology, Tsinghua University, Beijing, China.*
- *Kun Xu is the corresponding author, email: xukun@tsinghua.edu.cn.*

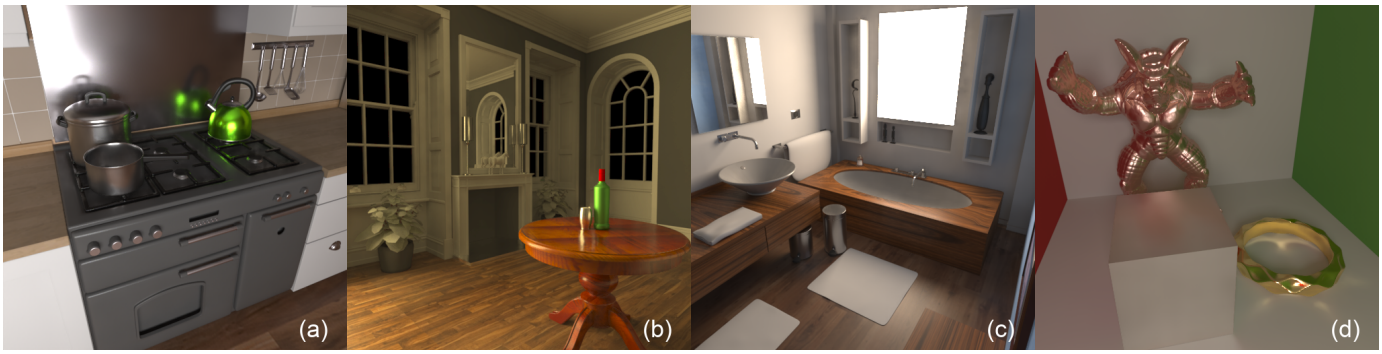


Fig. 1. Realistic rendering results with neural global illumination. (a) *Kitchen* with multiple glossy interreflection, (b) *Living room* with mirror reflection and glossy reflection, (c) *Bathroom-2* with strong indirect illumination and mirror reflection and (d) *Cornell box* with color-bleeding, glossy interreflection and caustics.

convolutional neural network (CNN) which enlarges the receptive fields and makes the training process easier and more robust.

Our method can generate all-frequency global illumination in static scenes with dynamic area lighting at 22 FPS. Our method supports many realistic global illumination effects such as glossy interreflection (e.g. Figure 1 (a, b)), caustics (e.g. Figure 1 (d)), mirror reflection (e.g. Figure 1 (b, c)) and color bleeding (e.g. Figure 1 (d)). Besides, our screen-space CNN naturally supports high resolution due to the fully-convolutional design and our MLP rendering network can generate global illumination independently for each shading point. Therefore, our method can scale up to higher resolution image generation without retraining. Thanks to the proposed efficient learning-based representation, the storage cost of our method is only 55.9 MB for each scene which is compact compared to existing precomputed-based methods.

In summary, our main contributions are:

- a carefully designed framework for interactively rendering global illumination with dynamic area light sources;
- an end-to-end method that generates realistic global illumination only from direct lighting and other screen space buffers which can be rendered by any existing renderers easily;
- a combinational incident lighting representation suitable for dynamic area light sources; and
- a screen space neural buffer that informs the rendering network on how to compute the global illumination efficiently.

2 Related Work

In this section, we will review recent work on interactive global illumination and neural rendering.

2.1 Precomputed Global Illumination

Precompute radiance transfer (PRT)

In PRT [54, 55, 41, 46], the light transport for each shading point is precomputed and stored as coefficients of basis functions. The run-time direct/global illumination is then reduced to a dot product. Early methods represent lighting by spherical harmonics function and only support low-frequency global illumination effects. Later methods [65, 16] extend to use wavelets or spherical Gaussian as the lighting basis and achieve all frequency global illumination. However, all these methods assume distance lighting. Wang and Ramamoorthi [64] presents a novel analytic derivation for spherical harmonic coefficients from uniform polygon area lights. Recently, Wu et al. [66] proposes an analytic formula for the spatial gradients of the spherical harmonic coefficients which enables PRT to support many polygon area lights in real-time. However, existing PRT-based methods [32, 64, 66] that support local light sources cannot handle high-frequency global illumination well.

Lightmaps and light probes

Lightmap stores the precomputed diffuse global illumination and is interpolated at run time to generate global illumination. The lightmap can be generated by offline rendering algorithms such as radiosity [6] or path tracing [26]. For non-diffuse objects, light probes are often used in production to generate real-time global illumination. The light probe was first introduced by Greger et al. [17] that supports global illumination for dynamic diffuse objects. Recently, McGuire et al. [36] proposed light fields probe that stores the full light field and visibility in static scenes and supports real-time global illumination. Rodriguez et al. [50] presented glossy light probes that store glossy light paths. The glossy lighting is reprojected from these precomputed glossy light probes and is added to diffuse lighting to generate the final global illumination. However, all these methods are limited to static lighting. In contrast, our method can generate full global illumination with dynamic area lighting.

Regression-based global illumination

Ren et al. [47] proposed a radiance regression function to model the radiance fields of each shading point in a static scene. They consider point lighting only while our method can generate full global illumination for scenes with dynamic area lighting. To support dynamic area lighting, we propose a combinational lighting representation that represents incident lighting more efficiently. Besides, unlike Ren et al. [47], we do not rely on complex spatial data structure at run time and utilize a single network to fit radiance fields of the whole scene.

2.2 Screen Space Global Illumination

Classic screen space approach

Classic screen space global illumination is a separate post-processing pass to generate approximate global illumination effects from screen-space buffers. Dachsbacher and Stamminger [9] proposed a reflective shadow map technique that extends the classic shadow map to handle single bounce indirect lighting for diffuse objects. Ritschel et al. [48] extended screen-space ambient occlusion to handle more indirect illumination effects such as directional shadow and diffuse color bleeding. However, these methods are limited to producing an approximate result for local indirect illumination. Robison and Shirley [49] proposed to blur a perfectly specular reflection buffer at run time to generate blurry reflections and soft shadows. However, there are still artifacts for glossy interreflection.

Learning-based screen space approach

Learning-based screen space global illumination approaches take several screen buffers as input and predict global illumination directly from the input. Nalbach et al. [40] proposed a Deep Shading framework that can predict indirect illumination from screen space buffer. Recently, Xin et al. [68] proposed a lightweight neural network that generates single-bounce indirect illumination for diffuse scenes. However, as with all the screen-space global illumination approaches, synthesizing global illumination from only screen-space information is a highly under-constrained and ambiguous problem. The mapping from screen-space buffer to global illumination is a one-to-many mapping that cannot be learned by neural network efficiently. Besides, these methods fail to produce full global illumination effects such as color bleeding within multiple diffuse objects or glossy interreflection. Our method can generate high-quality full global illumination in static scenes with dynamic area lighting.

2.3 Real time ray tracing

The latest GPU embedded with ray tracing cores supports much faster path tracing [67] compared with CPU-based path tracing. Real-time ray tracing (RTRT) utilizes neural network-based denoiser [5, 2, 69] to denoise the noisy image generated by GPU path tracing with extremely low samples per pixel (typically only one sample per pixel). However, classic RTRT fails to generate high-frequency global illumination such as multiple bounce glossy reflection because a single sample per pixel is not sufficient for such complex light transport and the latest denoisers are not suitable in these cases either.

Recently, Bitterli et al. [4] proposed a novel algorithm (ReSTIR) for real-time ray tracing by utilizing nearby samples in both spatial and temporal domains. ReSTIR supports high-performance direct illumination rendering for scenes with many lights. However, ResTIR is limited in direct illumination and cannot be easily extended to global illumination. Concurrently, Müller et al. [39] proposed neural radiance caching (NRC) method for fast rendering of global illumination in dynamic scenes. The main idea is online training the radiance cache while path tracing rendering. However, NRC fails to capture high-frequency global illumination effects which are unrelated to the local surface features such as shadows and caustics. To achieve real-time performance, NRC relies on a deep-learning based real-time denoiser [20] to produce noise-free images. Latest denoising algorithms fail to reconstruct high-frequency indirect lighting such as glossy reflection and caustics; thus, NRC will generate blurry results for high-frequency global illumination effects after denoising. In contrast, our method supports high-frequency global illumination such as caustics (as shown in Figure 1 (d)) and can generate noise-free results on the fly without applying any denoisers. Besides, NRC requires a lot of complex engineering to achieve high performance while our method can be integrated into existing rendering engines and deep learning frameworks smoothly without specific engineering optimization.

2.4 Neural rendering

Combining deep learning with computer graphics is rapidly evolving. Neural rendering have successfully been utilized in many traditional computer graphics research problems ranges from appearance modeling [35, 10, 13], image-based rendering [37, 14, 61], scene representation [12, 15], and light transport [38, 71]. We will focus this review of related work on light representation in neural rendering. For more details, we refer to the overview of neural rendering [60].

Efficient lighting representation is crucial for photorealistic neural rendering with dynamic lighting. The goal of lighting representation is to properly provide information about incident lighting to the neural network. There are a variety of lighting representations proposed for different lighting setups. Ren et al. [47] proposed to encode incident point lighting by the coordinate of the point light source. The position of the light source provides global lighting information to each shading point and is sufficient for the point light source. However, the area light source cannot be determined only by the position. Sun et al. [57] relit human face under natural lighting and model the lighting by a low-resolution environment map. Granskog et al. [15] proposed a neural scene representation that disentangles geometry, material, and lighting. Lighting was represented by a compact neural vector. However, both the environment map and disentangled neural vector are not efficient for our setup: a static scene with dynamic area light sources. Recently, Gao et al. [14] proposed radiance cues as the lighting representation for image-based relighting. Radiance cues are an image-space incident lighting representation and are suitable for different types of lighting such as local point lighting and distant environment lighting. The limitation of such screen-space representation is the lack of global lighting description which is important for local lighting. We focus on the incident lighting representation of area light sources and an efficient lighting representation enables accurate global illumination with dynamic incident lighting. We propose a combinational lighting representation that combines both global lighting information and screen-space representation.

In a concurrent work, Diolatzis et al. [11] proposed a Markov-Chain-Monte-Carlo-based approach for efficient training data generation. The neural renderer combined with an explicit parameterization scene representation can render global illumination for dynamic scenes. Rainer et al. [45] proposed neural PRT for global illumination renderings of static scenes under dynamic environment lighting.

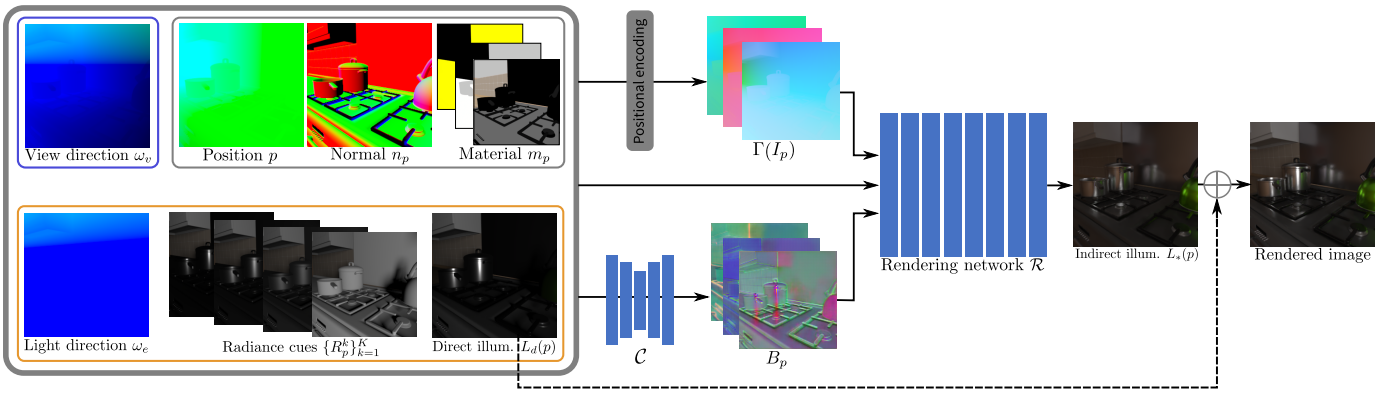


Fig. 2. Overview of Neural Global Illumination. The input I_p includes: attributes of each shading point $g(p) = \{p, n_p, m_p\}$, view information ω_v and area lighting information $L_e(p) = \{L_d(p), \{R_p^k\}_{k=1}^K, \omega_e\}$. First, we transform the input I_p to higher dimension space by positional encoding to get encoded feature $\Gamma(I_p)$. Next, a screen-space neural buffer B_p is generated by a convolutional neural network $\mathcal{C}$: $B_p = \mathcal{C}(I_p)$. Finally, the original input I_p, positional encoded feature $\Gamma(I_p)$ and screen-space neural buffer B_p are concatenated and fed into rendering network $\mathcal{R}$ to predict the indirect illumination $L_*(p)$. Adding direct illumination $L_d(p)$ with the predicted indirect illumination $L_*(p)$ produces the final rendering image.

3 METHOD

3.1 Overview

By assuming there are no participating media in the scene, the exitant radiance could be formalized by the rendering equation [26]:

$$L(p, \omega_v) = L_e(p, \omega_v) + \int_{S^2} f_p(\omega_v, \omega_i) L_i(p, \omega_i) |n_p \cdot \omega_i| \, \mathrm{d}\omega_i, \tag{1}$$

where p, n_p, f_p denote the position, normal and BSDF of the shading point, respectively, ω_v is the view direction, $L_e(p, \omega_v)$ is the emitted radiance, and $L_i(p, \omega_i)$ is the incident radiance from incident direction ω_i.

Based on where the incident radiance comes from, i.e., light sources or other non-emissive objects, the global illumination defined in Equation 1 can be divided into direct illumination $L_d(p, \omega_v)$ and indirect illumination $L_*(p, \omega_v)$. The direct illumination from area light sources can be computed efficiently by Linearly Transformed Cosines (LTCs) [21, 22]. Rendering indirect illumination from area light sources is a much more challenging problem and takes a rather long time to compute in general.

Our key observation is that indirect illumination $L_*(p, \omega_v)$ of static scenes is determined by the shading point, view direction, and incident area lighting since the indirect illumination is synthesized from only these inputs in offline rendering algorithms. Based on this observation, we can rewrite the indirect illumination of static scene as:

$$L_*(p, \omega_v) = \mathcal{F}(g(p), \omega_v, L_e(p)), \tag{2}$$

where $g(p)$ are the attributes of the shading point including: position p, normal n_p and material parameters m_p, $L_e(p)$ is the representation of area light sources and $\mathcal{F}$ represents the complex, highly non-linear mapping from these inputs to indirect illumination.

Our main idea is to utilize a deep neural network to represent the complex mapping $\mathcal{F}$. The pipeline of our method includes preprocess stage and rendering stage. During preprocessing, we will construct the training dataset by rendering images with full global illumination using offline rendering algorithms and then train a deep neural network on the training dataset in an end-to-end manner. The trained deep neural network is a compact representation for global illumination of the scene. This neural representation can predict high-frequency global illumination effects with dynamic area lighting and viewpoints and does not need any complex data structure or expensive storage cost. In the rendering stage, we can render the direct illumination and other buffers through a real-time rendering pipeline easily and feed these inputs into our trained network to generate the indirect illumination. The overview of our method is summarized in Figure 2.

Although deep neural network is well-suited to fit high dimensional function, there remain challenges to represent the high-frequency global illumination from a low dimensional input with a compact network. In the rest of this section, we first introduce how to encode the input information in a neural-network-friendly way in Section 3.2. Then in Section 3.3 we propose a screen-space neural buffer that shares the input information between nearby shading points and informs long-distance global information to each shading point. After that, we will describe the MLP-based neural rendering network that predicts indirect illumination for each shading point in Section 3.4. We will describe the training process in Section 3.5 and how to use our method in rendering in Section 3.6. The implementation details are listed in Section 3.7.

3.2 Neural-network-friendly inputs

As we mentioned earlier, the indirect illumination of the shading point is determined by the shading point, viewpoint, and incident area lighting. The view information can be represented efficiently by the view direction ω_v of each shading point.

In theory, position p is sufficient to represent a shading point in static scenes. However, given the 3D position only for each shading point, the neural network needs to predict global illumination together with other auxiliary attributes such as spatially-varying normal vector and other material parameters implicitly. Combining position with other auxiliary attributes as the representation for each shading point

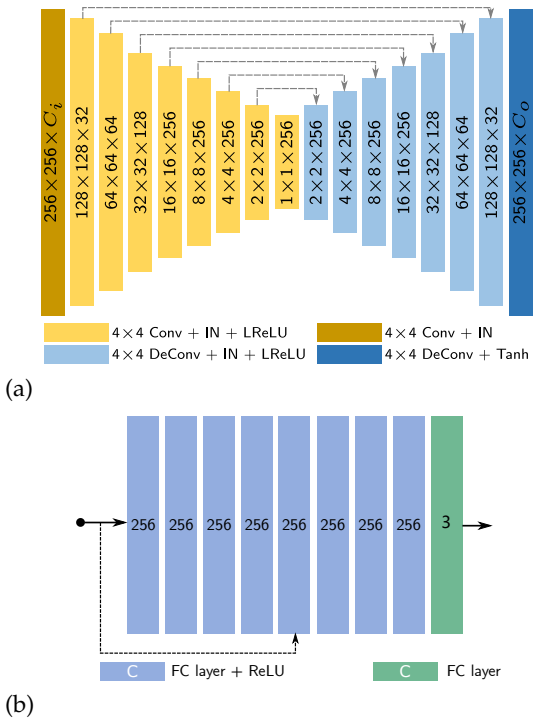


Fig. 3. The network structure of CNN screen space neural buffer extracting network $\mathcal{C}$ (a) and MLP rendering network $\mathcal{R}$ (b). Note that there is no activation applied after the last layer of $\mathcal{R}$.

allows a more efficient approximation with a compact deep neural network which is important to achieve interactive performance. In our implementation, we concatenate the position p to normal vector n_p and material parameters m_p (including diffuse albedo, specular albedo and roughness) to get the input vector $g(p)$ for each shading point. Note that for the shading point with perfectly specular material, we cast a specular reflection ray to get the intersection and assign the input vector of the intersection to the original shading point.

The representation of incident area lighting $L_e(p)$ is more complicated. Although the vertex positions of area light source are sufficient to describe incident lighting in theory, deep neural network cannot learn the complex mapping from such naive lighting representation efficiently. We propose to encode the incident lighting in a neural-network-friendly manner. More specifically, the incident area lighting is encoded by a combinational lighting representation that includes both screen-space lighting cues and global lighting cues. The redundancy in lighting representation allows efficient approximation with a compact deep neural network since the network can take advantage of different input cues in different cases. For example, the color bleeding effects between nearby diffuse objects can be synthesized more efficiently from screen-space lighting cues while the glossy reflection from off-screen objects will benefit from global lighting cues.

3.2.1 Combinational lighting representation

Our combinational lighting representation includes screen-space lighting cues (direct illumination and radiance cues) and a global lighting direction map.

Direct illumination. Direct illumination L_d is an integral of the product of BRDF and incident radiance emitted directly from light sources. Direct illumination is a physically-based representation considering both material and incident lighting; thus, direct illumination provides strong cues about incident lighting of each shading point to the deep neural network. For the fast rendering of direct lighting under area lighting, we first use Linearly Transformed Cosines [21] to approximate the BRDF; then the direct illumination integral of BRDF and polygon area lighting can be analytically computed in real-time. For soft shadows of direct illumination, we use the ratio estimator proposed by Heitz et al. [22] to combine stochastic shadows with analytic direct illumination.

Radiance cues. For a highly glossy or perfectly specular reflection surface, direct illumination is always black since the BRDF is a delta distribution whose value is zero almost everywhere. In such cases, direct illumination provides almost no information about the incident lighting which makes the neural network has no knowledge about varying lighting. We take inspiration from Deferred Neural Lighting [14] that uses radiance cues to represent incident lighting for scenes with unknown geometry and materials. We adapt radiance cues as an additional incident lighting representation mainly for highly glossy surfaces which provide cues to the neural network on the impact of varying incident lighting. Radiance cues $R_p^k = L_d(p, \omega_v | b^k)$ is a set of direct illumination images synthesized with K predefined basis materials $\{b^k\}_{k=1}^K$. Unlike Gao et al. [14], we do not use homogeneous basis material for every object, but instead replace only the non-diffuse material with homogeneous basis material because the direct illumination is sufficient to represent the incident lighting for diffuse surfaces. Besides, we consider direct illumination only in the rendering of radiance cues since global illumination is expensive to compute.

In our implementation, we use a set of $K = 4$ basis materials that cover different frequencies. More specifically, we use a pure Lambertian BRDF and three Cook-Torrance BRDF [7] with roughness parameters 0.05, 0.13, 0.34 respectively. We can render radiance cues together with direct illumination in a deferred rendering pipeline efficiently by sharing common steps. For example, the G-buffer except materials can be shared in rendering the unshadowed shadings and the ray-traced visibility can be shared in rendering the soft shadow by ratio estimator.

Global lighting direction map. Alongside the screen-space lighting cues, we propose to encode global lighting information by the lighting direction ω_e of each shading point. For area light source, we use the center position of area light source to approximate the light position. This global lighting direction map enables each shading point to be aware of the lighting position in the scene and allows the neural network to learn long-distance global illumination effects. Global lighting direction map also helps to overcome the ambiguity issues of pure screen-space representation: in some cases, two different incident lighting will have similar shading effects which confuse the neural network.

In summary, our combinational lighting representation can be represented by: $L_e(p) = \{L_d(p), \{R_p^k\}_{k=1}^K, \omega_e\}$.

3.2.2 Positional encoding

If we directly use the attributes of shading point $g(p)$, view direction ω_v and incident lighting $L_e(p)$ as the inputs

for the rendering neural network, the neural network will generate blurry results and is not able to capture the high-frequency global illumination details. This is consistent with the observation proposed by earlier works [44, 37, 59].

We use the positional encoding technique similar to Mildenhall et al. [37] to transform the inputs to higher dimensional space by a predefined Fourier feature mapping:

$$\begin{aligned} &\gamma(x) = \{\gamma_0(x), \ldots, \gamma_L(x)\}, \\ &\text{where } \gamma_l(x) = \{\sin(2^{l-1}\pi x), \cos(2^{l-1}\pi x)\}, l \in [0, L] \end{aligned} \tag{3}$$

The mapping $\gamma(\cdot)$ is applied to each component of original inputs $I_p = \{g(p), \omega_v, L_e(p)\}$ independently. Practically, we use $L = 9$ for all inputs. Note that unlike in Mildenhall et al. [37], we apply positional encoding to not only position and view direction but also the attributes of each shading point and incident lighting. Since both the movement of viewpoint and lighting will lead to high-frequency appearance changes. And the mapping from the attributes of each shading point to global illumination effects is also high-frequency.

Applying positional encoding to all inputs enables the rendering network to capture high-frequency global illumination effects. We will compare the positional encoding with other alternative techniques in subsection 4.3. The encoded high-dimensional features $\Gamma(I_p)$ are concatenated with the original inputs I_p to feed into our rendering network.

3.3 Screen-space neural buffer

The neural-network-friendly input noted in Section 3.2 contains only the information of a single shading point without considering information of nearby shading points which is quite useful in some cases. For example, glossy reflection between two nearby objects and the color bleeding effects that occurred between two nearby diffuse planes strongly rely on the nearby information. Predicting these effects from each shading point independently without considering nearby information is inefficient.

We propose to predict a screen-space neural buffer B from I_p by a convolutional neural network $\mathcal{C}$ which helps to exploit context information, i.e., the relationship between different objects within screen space. In our implementation, we use the fully-convolutional U-Net [24] architecture. The encoder transforms the per-pixel inputs to a compact latent space vector while the decoder propagates the latent space vector back to each pixel. The output neural buffer has the same spatial resolution as the input and stores a high-dimension neural feature vector B_p in each pixel that contains both global information in screen space and local features corresponding to the pixel.

The advantages of the proposed CNN module are:

- the screen-space neural buffer extends the receptive fields of the MLP rendering network and informs global information in screen-space to each shading point.
- the fully-convolutional structure supports different resolutions naturally without any retraining.

Summary

We summarize the inputs $I_p^+ = \{I_p, \Gamma(I_p), B_p\}$ before introducing the details about rendering network as following:

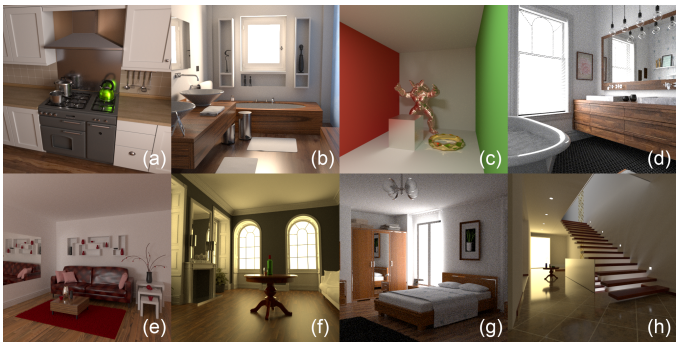


Fig. 4. Our eight test scenes. (a) *Kitchen*, (b) *Bathroom-2*, (c) *Cornell box*, (d) *Bathroom*, (e) *Living room-3*, (f) *Living room*, (g) *Bedroom* and (h) *Staircase-2*.

1) neural-network-friendly inputs I_p contains attributes of each shading point, view direction, and incident lighting;
2) high-dimensional feature $\Gamma(I_p)$ that is transformed from I_p by a positional encoding; and
3) a learned neural feature vector B_p that contains global information within screen-space.

3.4 Rendering network

The rendering network $\mathcal{R}$ takes I_p^+ as input and generates indirect illumination $L_*(p, \omega_v)$ of the corresponding shading point. We follow the network architecture of Park et al. [42]. As shown in Figure 3 (b), we use an 8 layer MLP network which gets a good balance between running performance and network capacity. The MLP network is trained together with the CNN network (Figure 3 (a)) that extracts screen-space neural buffer. The screen-space neural buffer informs the MLP rendering network on how to compute the global illumination efficiently and the rendering network guides the CNN network on how to extract useful features in screen space.

The reasons that we use an MLP neural network instead of classic CNN to synthesize the indirect illumination are: MLP learns the mapping from each shading point independently while CNN takes nearby pixels into account; thus MLP is more suitable for learning a disentangled mapping by utilizing the consistency of multiple batch data of each shading point efficiently. For example, given two batch data of a shading point, while the input data are the same except for incident lighting, the shared attributes (e.g. viewpoint, position, normal) will be ignored by an MLP to synthesize the appearance differences caused by different incident lighting. However, for a CNN network, the shared attributes will be distorted with nearby pixels in screen space during convolution which makes the network more difficult to disentangle input attributes. In a typical rendering dataset, there are always multiple samples for each attribute that help the MLP network to learn a disentangled mapping from the input attribute to the output.

3.5 Neural network training

As noted before, the MLP rendering network R and the CNN-based screen-space neural buffer extracting network C

are trained together for each scene:

$$\mathcal{R}^*, \mathcal{C}^* = \underset{\mathcal{R},\mathcal{C}}{\operatorname{argmax}} \sum_{i}^{N} \mathcal{L}(I_p^+, L_p|\mathcal{R}, \mathcal{C}), \tag{4}$$

where $\mathcal{L}$ is the training loss function, L_p is the global illumination of shading point p, N is the number of training data.

The training loss function $\mathcal{L}$ is defined as the sum of a pixel loss term and a perceptual loss term:

$$\mathcal{L}(a, b) = \mathcal{L}_{pixel}(a, b) + \lambda \mathcal{L}_{perceptual}(a, b), \tag{5}$$

where λ is a weight to balance the two terms and we set λ to 1.0 in our implementation, the pixel loss $\mathcal{L}_{pixel}$ is defined as per-pixel L_1 distance on log-encoded pixel values:

$$\mathcal{L}_{pixel}(a, b) = \| \log(a + \epsilon) - \log(b + \epsilon) \|_1, \tag{6}$$

where $\epsilon = 1.0/\mathrm{e}$. The perceptual loss $L_{perceptual}$ is defined following Zhang et al. [70] which measures the similarity of two images in a way that accounts for human perception. The perceptual similarity is measured by the distance between deep features extracted by pre-trained neural network. In our implementation, we use 5 Conv layers from the VGG network [52] in computing the perceptual loss as suggested by Zhang et al. [70]. We will discuss in Section 4.3 that combining both losses yields plausible and visually accurate results.

3.6 Rendering

After training the neural network, the global illumination under dynamic viewpoint and area lighting can be synthesized by our method. First, we render G-Buffer (including position, view direction, lighting direction, normal vector, and material maps), direct illumination, and radiance cues by real-time rendering pipeline. Then we get the high-dimensional encoded inputs by positional encoding and CNN-based global feature extracting. After that, we concatenate all these inputs and feed them into our MLP rendering network to generate indirect illumination. Global illumination is the sum of rendered direct illumination and predicted indirect illumination.

It is straightforward to support multiple area light sources thanks to the linearity of light transport. During rendering, the rendered images lit by each area light source can be synthesized by our pipeline and the sum of all these rendered images is the final rendered image lit by multiple area light sources. The computational complexity is proportional to the number of area light sources. In application, we can parallelize the rendering of each area lighting since the computation is independent. As shown in Figure 12 (q, r), our method is able to generate plausible rendering results for multiple area light sources.

3.7 Implementation details

3.7.1 Training

We implemented our method in TensorFlow [1] and use Adam optimizer [30] to train our framework using a learning rate of 10^{-4} and set $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We train our framework for $1{,}000k$ iterations using a batch size of 1. Training takes 22 hours for each scene on an NVIDIA RTX 2080Ti GPU.

TABLE 1
Quantitative results of test data for each scene. The errors are computed over 100 test images with novel lighting and viewpoint for each scene.

	#Input	#Vert.	MAE	MSE	SSIM	PSNR	$\text{LPIPS}_{\text{Alex}}$
Kitchen	4,900	$443k$	0.0064	0.00034	0.984	37.82	0.0270
Bathroom-2	4,915	$696k$	0.0074	0.00020	0.983	39.96	0.0206
Cornell box	5,250	$1{,}127k$	0.0053	0.00017	0.992	40.99	0.0133
Bathroom	5,300	$258k$	0.0062	0.00041	0.985	36.85	0.0249
Living room-3	5,217	$3{,}596k$	0.0088	0.00028	0.988	36.74	0.0146
Living room	5,786	$117k$	0.0018	0.00004	0.996	47.10	0.0105
Bedroom	6,250	$1{,}052k$	0.0083	0.00050	0.985	36.67	0.0233
Staircase-2	6,000	$101k$	0.0054	0.00011	0.991	42.78	0.0169

TABLE 2
Quantitative comparison to selected prior work on *Kitchen* scene. The errors are computed over 100 test images with novel lighting and viewpoint. Ours* is a variant of our method that does not use perceptual loss during training. The best results are marked in bold and the second best results are underlined.

	MAE	MSE	SSIM	PSNR	$\text{LPIPS}_{\text{Alex}}$
RRF [47]	0.0226	0.00199	0.899	29.64	0.1111
BCN [68]	0.0190	0.00231	0.917	30.41	0.1579
CNNR [15]	0.0215	0.00606	0.949	28.33	0.0925
RTRT	**0.0051**	**0.00021**	**0.988**	<u>39.56</u>	0.0403
Ours*	**0.0051**	<u>0.00025</u>	<u>0.987</u>	**39.71**	<u>0.0334</u>
Ours	0.0064	0.00034	0.984	37.82	**0.0270**

TABLE 3
Temporal stability comparison to selected prior work on *Kitchen* scene. The errors are computed over continuous video sequences with dynamic view or dynamic lighting. The best results are marked in bold and the second best results are underlined.

	MABD_{view}	MABD_{light}
RRF [47]	0.000588	0.000124
BCN [68]	<u>0.000502</u>	0.000122
CNNR [15]	0.000550	**0.000072**
RTRT	0.000839	0.000140
Ours	**0.000437**	<u>0.000097</u>

3.7.2 Training data generation

For each scene, we render around 5000 images as training data (see the first column of Table 1 for a summary of the number of rendered images per scene). For viewpoint generation, we randomly sample the camera position and look-at target position from a given bounding box. For area light source generation, we randomly sample the size from a given range and sample the center position from a given bounding box. We use the GPU-based path tracer in PBRT-v4 [43] to render our training dataset at 256×256 resolution with 1024 samples per pixel (spp). To capture full global illumination, we set the max bounce of path tracing to 16. Note that the rendered images are denoised further by OptiX Denoiser [5]. The rendering of training data takes 20-26 hours for each scene with two RTX 2080 GPUs.

4 Results

4.1 Rendering results

To demonstrate the effectiveness of our method for a variety of global illumination effects, we validate with 8 complex scenes [3, 50] exhibiting complex global illumination effects as shown in Figures 1 and 4. We show the scene complexity by the number of vertices of each scene in the second column of Table 1. The scenes we used in our paper are: *Kitchen* (with glossy interreflection of multiple objects), *Bathroom-2*

Fig. 5. Qualitative comparison of results from a novel viewpoint and novel lighting that are rendered by ours and reference. Left: reference results. Middle: our rendering results. Right: differences ($\times 5$) between our results and reference results.

(with strong multi-bounce indirect illumination and mirror reflection), *Cornell box* (with strong color bleeding, glossy reflection and caustics), *Bathroom* (with difficult specular light transport due to a large mirror and relatively area light sources), *Living room-3* (with rich texture details, color bleeding, and mirror reflection), *Living room* (with glossy interreflection and mirror reflection), *Bedroom* (with strong multi-bounce indirect illumination and mirror reflection) and *Staircase-2* (with strong glossy interreflection). Please refer to the supplementary material for the nearest neighbors from the training set to better understand the distribution of training and test samples.

Figures 5 and 12 show the rendering results generated by our method in multiple challenging cases and demonstrate that our method is able to produce visually plausible results. We also show the scenes rendered with direct illumination and the indirect illumination rendered by our method in Figure 15. Our method is robust to different lighting conditions such as multiple area light sources (see Figure 12 (q, r)) and different size of area lighting (see Figure 10). Please refer to the supplemental material for more visualizations of each scene.

The quantitative evaluation results are summarized in Table 1. In all our quantitative results, we use five error metrics (MAE, MSE, SSIM, PSNR, and $\text{LPIPS}_{\text{Alex}}$) to measure the distance between the predicted results and the reference results. Note that we use $\text{LPIPS}_{\text{VGG}}$ as the perceptual loss function in training while the backbone network is different from $\text{LPIPS}_{\text{Alex}}$.

The storage size of our trained neural network is 55.9 MB for each scene. Our method is able to achieve 22 FPS at run time with 256×256 resolution. More specifically, we averaged the running time over all our eight test scenes with an NVIDIA RTX 2080Ti GPU. At run time, there are three main steps: G-Buffer generation, direct illumination rendering (includes direct lighting and radiance cues), and network inference:

- G-Buffer generation takes 0.7 ms on average;
- Direct illumination rendering includes both direct lighting images and four radiance cues images. For a single pass of analytic direct lighting without shadows, the average rendering time is 1.4 ms. The total time of analytic direct lighting rendering is 6 ms. Besides, rendering stochastic shadow takes 10 ms in total (we set the spp equals to 4 as Heitz et al. [22] suggested);
- Network inference is the most time-consuming step and the inference time is independent of the complexity of scenes since our network takes only screen-space buffers as inputs. The network inference takes 28.3 ms on average.

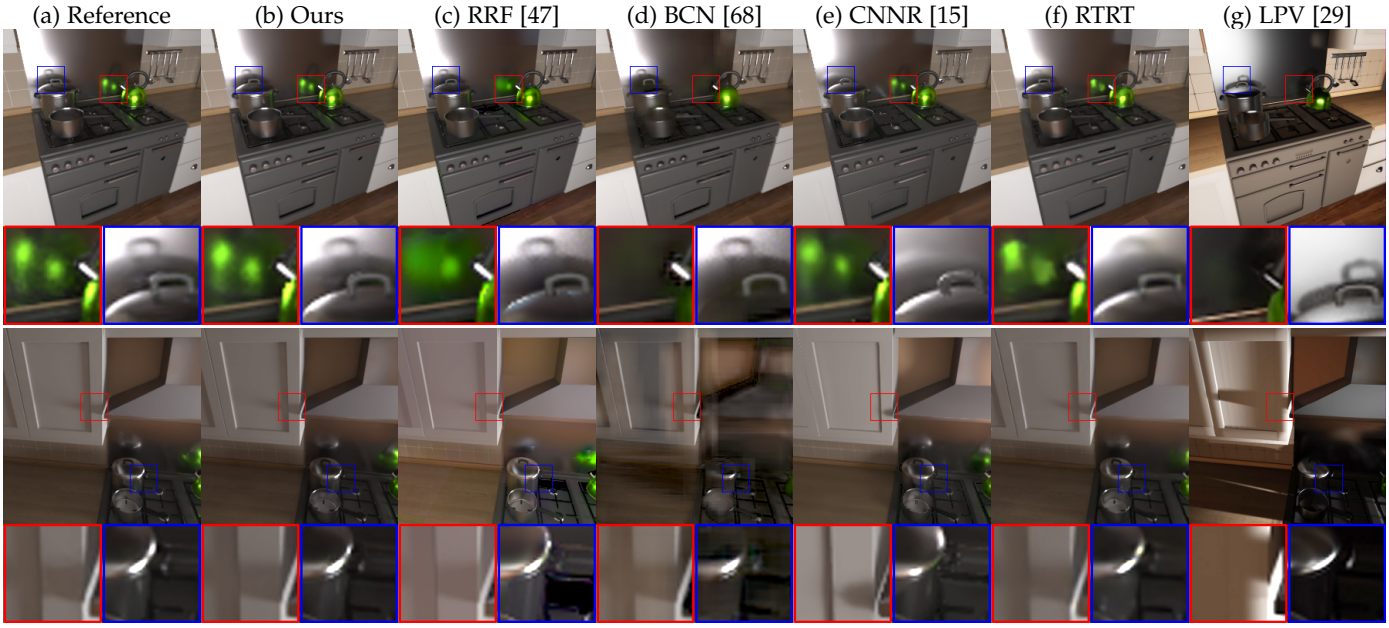


Fig. 6. Qualitative comparison to selected prior works on the *Kitchen* scene. Our method (b) is able to generate plausible glossy interreflection with dynamic viewpoint and lighting compared with the reference (a). *RRF* (c) fails to fully capture glossy interreflection. *BCN* (d) cannot produce plausible results for such a challenging scene with multiple glossy objects. *CNNR* (e) produces blurry results (e.g. the wood table) and the indirect illumination is not as accurate as ours even taking three additional images with full global illumination as inputs. *RTRT* (f) generates overblurred results due to insufficient samples and denoising. *LPV* (g) fails to generate plausible global illumination for glossy materials such as glossy interreflections.

4.2 Comparisons

To our knowledge, there is no existing prior work that renders full global illumination in static scenes under dynamic viewpoint and area lighting. Thus we try our best to select prior works that solve similar problems. We compare the *Kitchen* scene across all methods. The visual comparison is shown in Figure 6.

We select the following five prior works to compare with: *RRF* [47] predicts global illumination in static scenes with dynamic lighting. However, they assume point light sources and use the position of the point light source to represent the incident lighting. We use the center of the area light source to approximate the position of the light. *RRF* proposes a partition-based strategy to handle complex scenes. In our implementation, we first decompose the whole scene into roughly $1,000$ partitions using Kd-Tree and then train a small RRF network in each partition. The storage size of each small RRF network is 52 KB (3 hidden layers, the width of each layer: 128, 64, 32) and thus the final RRF network has a similar capacity as ours. As shown in Figure 6 (c), *RRF* can generate plausible results but fails to fully capture glossy interreflection. The main differences compared with the reference (Figure 6 (a)) are the multiple bounce glossy reflection which demonstrates light position only is not sufficient to represent complex light sources like area light sources. *BCN* [68] is a state-of-the-art screen-space global illumination method and generates single bounce indirect illumination for diffuse objects from screen-space buffers. Although their approach supports dynamic scenes without any retraining, we train their model on our static *Kitchen* scene for a fair comparison. As shown in Figure 6 (d), *BCN* produces blurry results and cannot generate glossy indirect illumination well. This indicates that the screen-space approach is not well-suited to such challenging scenes with multiple bounce glossy interreflection. *CNNR* [15] proposed a neural scene representation that disentangles geometry, material, and lighting and can be used to predict indirect illumination. We simplify their method to only consider dynamic lighting and keep geometry and material static. Note that for the dynamic lighting scenario, we should first render three additional images with full global illumination under the same lighting and different viewpoints and then feed these three images and G-Buffers together to their pipeline to generate the final result. And this is not a practical solution for dynamic lighting because the overhead of preparing three images is more expensive than rendering the output directly. As shown in Figure 6 (e), the global illumination results predicted by *CNNR* are not as accurate as ours (e.g. the glossy reflection of the steel pot in the first example and the soft shadow in the second example). Besides, *CNNR* fails to capture all specular highlights and texture details (e.g. wood texture of both table and floor) faithfully. *RTRT* renders noisy images with low samples per pixel and denoises the rendered images by a post-processing pass. In our implementation, we use a fast path tracer that is based on the Falcor rendering framework [27] to render images. We set the spp equal to 16 for the equal-time comparison. We utilize the Optix Denoiser [5] to denoise the rendered images due to its high performance and easy implementation. Ideally, we should have trained a scene-specific denoiser for a completely fair comparison, however, we did not do it since it is non-trivial for retraining the Optix Denoiser. As shown in Figure 6 (f), *RTRT* can capture the overall global illumination effects but fails to generate high-frequency, artifact-free results due to insufficient samples. Besides, the errors lead to temporal flickering, especially for dynamic viewpoints. *LPV* uses multiple virtual point lighting (VPL) to approximate indirect illumination and store

Fig. 7. Ablation study of neural global illumination. Our method (b) is able to produce plausible global illumination compared to the reference (a). Using the convolutional neural network of [72] (c) fails to generate sharp glossy reflection and introduces artifacts in other cases. Decreasing the network size (d, e) produces less accurate results. We also observe that the quality loss of decreasing the network width (d) is larger than decreasing the depth (e) and this indicates the network width is more important in generating high-quality global illumination. Screen space neural buffer plays an important role in sharing screen-space object-level information. The results without screen-space neural buffer (f) produce an incorrect intensity of global illumination and generate artifacts in dark areas (e.g. shadow areas) or bright areas (e.g. glossy reflection of a light source). Positional encoding is important to generate high-frequency details as demonstrated in (g-i). Although our method without perceptual loss had lower numeric errors, the visual quality of our method without perceptual loss (j) is worse than ours, especially on indirect illumination. This indicates that perceptual loss enables the training process to focus on minimizing the errors of indirect illumination areas. Decreasing the number of input images (k, l) fails to capture indirect illumination accurately and indicates that the number of input images has a significant impact on the visual quality.

the indirect illumination into a 3D volume texture. Light propagation volume (LPV) [28, 29] is widely used in real-time applications and is able to generate plausible global illumination for diffuse objects in real-time. We adopt the publicly available implementation in Unreal Engine and import the *Kitchen* scene manually. Note that the material models and lighting models are slightly different in Unreal Engine and other methods resulting in some appearance differences. As shown in Figure 6 (g), *LPV* fails to generate plausible global illumination for glossy materials. Note that we retrained the neural network of *RRF*, *BCN*, and *CNNR* for each scene in the same condition as ours for a fair comparison.

Figure 11 shows additional comparisons of *RRF* [47], *BCN* [68], and *RTRT* in the *Cornell box* scene. *RRF* and *BCN* can produce plausible diffuse indirect lighting but fail to predict caustics and high-frequency glossy interreflections. *RTRT* can capture the overall appearance but fails to generate high-quality glossy reflection and caustics due to insufficient samples and denoising.

To further validate the effectiveness of our method, the quantitative evaluation is shown in Table 2. The errors are computed over 100 rendered images with novel views and lighting. Our method has a consistently lower error in all error metrics compared with RRF, BCN, and CNNR. Compared with RTRT, the errors of our method are slightly larger in MAE, MSE, SSIM, and PSNR. However, our method is able to capture sharper and more accurate indirect lighting effects compared with RTRT (as shown in Figures 6 and 11) and the perceptual errors agree with the visual comparison. The MAE/MSE/SSIM/PSNR errors of our method without perceptual loss are comparable to RTRT while the perceptual error is lower.

Furthermore, RTRT fails to generate temporal-consistent results as shown in the supplemental video while our method is able to generate high-quality videos without temporal flickering. We use the temporal consistency error metric (MABD) proposed by Lai et al. [34] to measure the video stability. The optical flows used in MABD are predicted by PWC-Net [56]. The temporal quantitative comparison results are shown in Table 3. We can see that: 1. the temporal instabilities when changing the viewpoints are more obvious than changing the lighting. 2. the temporal stability of RTRT is the worst among all methods which are consistent with the visual results. 3. Our method is able to produce high-quality and temporal consistent results.

4.3 Ablation Study

Impact of the network architecture of rendering network

We use a deep fully-connected network to represent the rendering neural network. As noted in Section 3.4, MLP network architecture is more suitable than classic CNN

TABLE 4
Quantitative evaluation for the ablation study. The errors are computed over 100 test images with novel lighting/viewpoint not part of the training set. The best results are marked in bold and the second best results are underlined.

Ablation Variant	MAE	MSE	SSIM	PSNR	$LPIPS_{Alex}$
[72]-net	0.0078	0.00050	0.977	36.15	0.0399
Width 128	0.0082	0.00055	0.976	35.86	0.0403
Depth 4	0.0077	0.00048	0.977	36.52	0.0370
w/o CNN.	0.0099	0.00054	0.968	35.70	0.0412
w/o CNN-D12	0.0101	0.00055	0.970	35.85	0.0391
w/o CNN-W512	0.0093	0.00044	0.972	36.37	0.0354
w/o CNN-W1024/D16	0.0084	0.00037	0.976	37.30	0.0320
w/o Percept.	**0.0051**	**0.00025**	**0.987**	**39.71**	0.0334
w/o Pos.	0.0071	0.00042	0.981	37.15	0.0335
SIREN	0.0112	0.00108	0.963	33.28	0.0647
SIREN*	0.0106	0.00099	0.967	33.60	0.0570
1,000 images	0.0137	0.00130	0.951	32.34	0.0734
2,500 images	0.0108	0.00092	0.964	33.76	0.0567
Light Pos.	0.0207	0.00152	0.926	31.73	0.0604
Light Pos. ext.	0.0109	0.00157	0.969	32.60	0.0466
Light Pos. + Dir.	0.0079	0.00040	0.981	37.49	0.0279
Light Pos. + Dir. + 1 Rad.	0.0068	0.00039	0.982	37.67	0.0277
Light Pos. + Dir. + 2 Rad.	0.0067	0.00038	0.983	37.75	0.0275
Ours	0.0064	0.00034	0.984	37.82	**0.0270**

networks when used for predicting complex mapping from low-dimension input. Our MLP rendering network (Figure 7 (b)) is able to capture global illumination effects more accurate compared with a classic CNN network architecture that follows the generator design of [72] (Figure 7 (c)). The quantitative errors listed in Table 4 agree with the qualitative comparisons.

Impact of the size of rendering network

We use a fully connected network with 8 layers in depth and 256 channels in width to represent the complex mapping from input to global illumination. In Figure 7 (d, e), we explore the impact of the size of our rendering network. We can see that decreasing the size of the rendering network (either the width or depth) results in significant quality reduction. At the same time, the size of the rendering network affects inference performance while a larger network takes longer times to evaluate. Therefore, we opt for using the current network size for a balance between rendering quality and inference performance.

Impact of screen-space neural buffer

The global illumination can be determined completely by the shading point, view, and incident lighting. However, the relationship among objects is still useful to inform the rendering neural network to compute the global illumination efficiently since the global illumination is generated by multiple bounce scattering between different objects. We use the screen-space neural buffer to propagate global information in screen space to each shading point so that the complex mapping can be learned by a compact MLP network.

To better validate the effectiveness of the screen-space neural buffer, we first compared the results of our method with and without the screen-space neural buffer. Figure 7 (f) shows the result of this ablation experiment. Screen-space neural buffers help to produce more accurate global illumination and reduce visual artifacts (global intensity mismatch). The numeric errors in Table 4 also show that the screen-space neural buffer improves the results significantly.

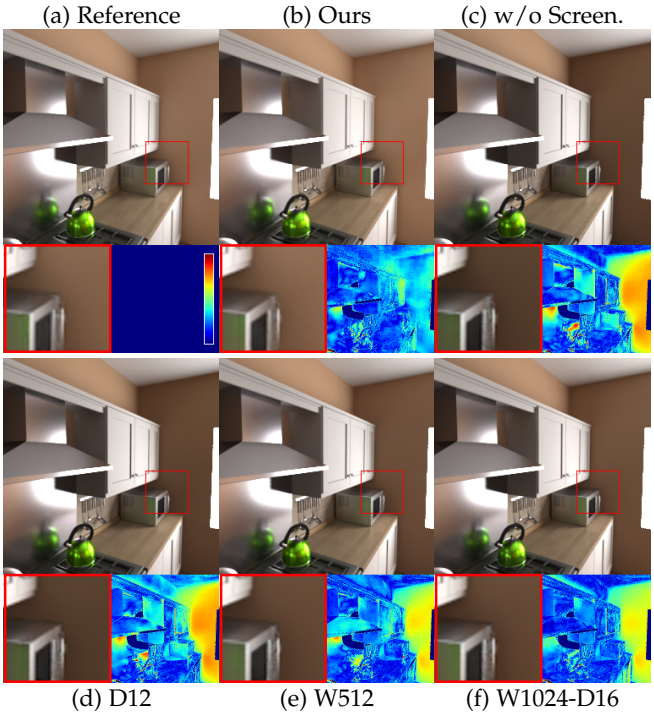


Fig. 8. Additional experiments about screen-space neural buffer. The rendering results of different variants are shown in the first row. The close-up views of certain areas and error maps are visualized in the second row. Our method (b) is able to produce high-quality rendering results compared to the reference (a). Other variants (MLP only) fail to capture the full dynamic range and cannot generate plausible results in dark areas (e.g. the wall behind the area light).

The previous experiment demonstrated that CNN plays an important role in our method. Since the global illumination of a shading point can be determined completely by the shading point itself, a natural question arises on whether a large MLP performs similarly to our method. We compare our solution to several alternative MLP-only solutions (as shown in Figure 8):

1) w/o Screen.: Removing the CNN from our method.
2) D12: Increase the depth of MLP (depth = 12).
3) W512: Increase the width of MLP (width = 512).
4) W1024-D16: Increase both the depth and the width of MLP (depth = 16, width = 1024).

We can see that increasing the capacity can improve the rendering results (Figure 8 (d-f) compared with (c)) but MLP-only solutions fail to generate accurate global illumination in dark areas. The quantitative results (Table 4) further demonstrate that our method can produce more accurate results than MLP-only variants. Besides, increasing the size of MLP will affect the run-time performance significantly. More specifically, the network inference of our method takes 28.3 ms and the other three variants (D12, W512, W1024-D16) take 30.5, 53.2, and 129.1 ms respectively. Therefore, the screen-space neural buffer is crucial to improve the rendering quality and enables the compact MLP to learn complex mapping efficiently.

Impact of positional encoding

Positional encoding enables MLP to learn high frequency function. There are lots of high-frequency effects in global

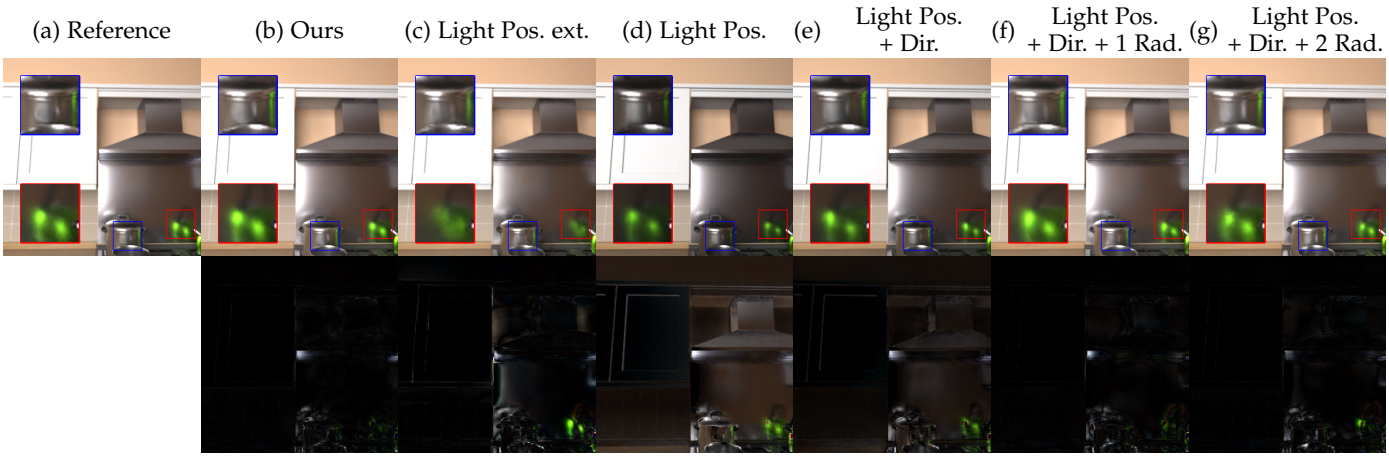


Fig. 9. Comparison of the rendering quality with different lighting representations. The errors ($\times 2$) are visualized in the second row. We can see that our full lighting representation (Light position, direct lighting, and radiance cues) faithfully produces global illumination effects. Representing lighting by position only (d) produces a less accurate result (see the error map). The naive extension (c) fails to generate plausible glossy interreflection. This indicates that screen-space lighting representation is more efficient. Combining light position with direct illumination (e) can generate plausible results but still have artifacts, especially in glossy interreflection areas. Adding one (f) or two radiance cues (g) can improve the glossy interreflections further. The results (b, e-g) show that radiance cues play an important role in providing incident lighting information for glossy surfaces.

illumination such as glossy interreflection, mirror reflection, and caustics. Figure 7 (b, g) shows the comparison with or without applying positional encoding. For a fair comparison, we keep the number of input channels constant by replacing the positional encoding with naive repeating. We can see that our method can generate overall plausible results even without positional encoding. However, transforming low-dimensional input to higher dimensional space by positional encoding can improve our results further. Furthermore, we also compare positional encoding with the latest SIREN [53] (SInusoidal REpresentation Networks) that shows great success in novel view synthesis. Figure 7 (h) shows the results of SIREN and Figure 7 (i) shows the results of SIREN* (combining naive repeating with SIREN to keep the number of input channels constant). The results of two SIREN variants are even worse than the results with naive repeating in our task which indicates the effectiveness of ReLU activation function and positional encoding.

Impact of training loss function

We use a per-pixel L_1 distance and a perceptual loss to train our pipeline. The perceptual loss enables the training process to focus on visually important areas instead of noises or over-saturated areas. We argue that although adding perceptual loss will increase the numeric errors (MAE, MSE, SSIM, PSNR in Table 4), the overall visual quality and perceptual error will be improved as shown in Figure 7 (j) and Table 4 $\text{LPIPS}_{\text{Alex}}$. The role of perceptual loss is not to decrease the L1/L2 distance between the rendered images and the reference images but to change the distribution of errors. In other words, the perceptual loss would encourage the neural network to place more errors on less important areas and focus on improving the quality in important areas. The perceptual loss may be useful for other rendering tasks to improve the visual quality.

Impact of the number of input images

To generate plausible rendering results for dynamic viewpoint and lighting conditions, good coverage of viewpoint/lighting combination is important. In Figure 7 (k, l), we show the results trained with $1{,}000$ and $2{,}500$ images sampled uniformly from our $5{,}000$ input images. We can see that the network trained with $1{,}000$ and $2{,}500$ images generates blurry results and cannot generate indirect illumination faithfully. This indicates that dense samples of viewpoint and lighting space play an important role in generating visually plausible results. Besides, these artifacts lead to temporal artifacts in continuous sequences. We use around $5{,}000$ - $6{,}000$ input images for all our scenes that provide a good balance between accuracy and rendering performance since rendering high-quality reference images is time-consuming.

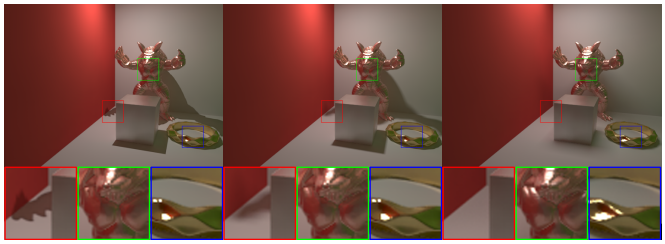


Fig. 10. Results of the impact of different area lighting sizes demonstrated on the *Cornell box* scene. Our method can produce plausible results with hard shadow boundaries for very small area light sources or even point light sources (a). For a larger area light source (b), the soft shadow can be generated by our method. For a very large area light source (c), the shadows almost disappeared as expected.

Impact of lighting representation

We represent incident area lighting by both global lighting position and screen-space lighting cues (direct illumination and radiance cues). Lighting position only is not sufficient to describe the area light source since both the size and orientation are ignored. Figure 9 (d) and Table 4 demonstrate that screen-space lighting representation is important to produce plausible global illumination results. Since the area light source can be described by its vertices, a natural question arises on whether this naive extension of light position is sufficient. In this naive extension, we represent

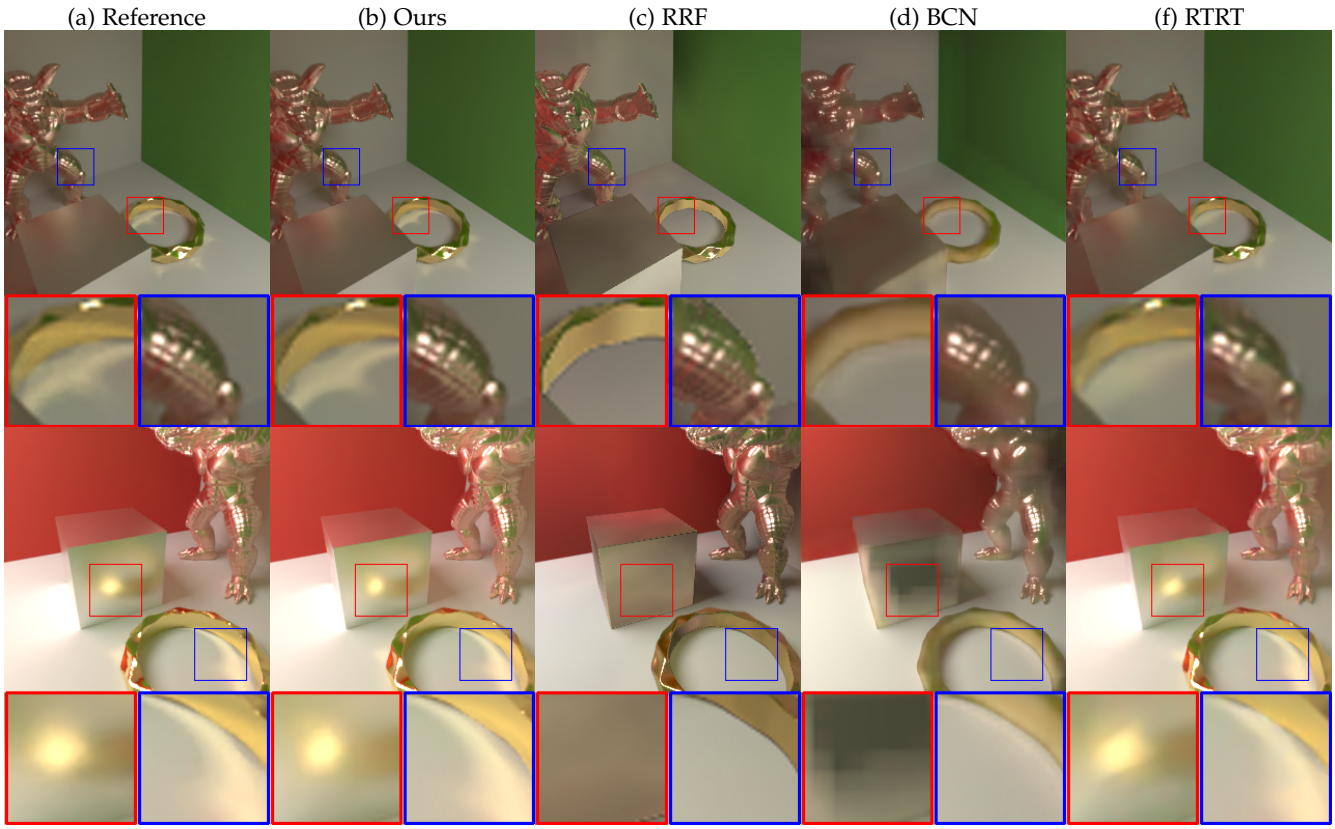


Fig. 11. Additional qualitative comparison to prior work: *RRF*[47], *BCN*[68], and *RTRT* on *Cornell box* scene. Our method (b) is able to produce high-quality caustics and glossy reflections with dynamic viewpoint and lighting compared with the reference (a). *RRF* (c) and *BCN* (d) can produce plausible diffuse global illumination but fail to predict high-frequency reflections and caustics. *RTRT* generates overblurred glossy reflections and can not reproduce high-quality caustics.

the area light source by its center position, positions of each vertex, and the normal orientation and keep other parts (the number of input channels, positional encoding, etc.) fixed. Figure 9 (c) and Table 4 show that such a naive extension fails to produce high quality global illumination effects under dynamic area lighting. For the screen-space lighting representation, we demonstrate that direct lighting is an efficient lighting representation (Figure 9 (e)) but combining radiance cues with direct lighting generates the most accurate result (Figure 9 (b)). In Figure 9 (f, g) we further explore the impact of the number of radiance cues while keeping the total number of channels constant by repeating. The results of a single radiance cue (diffuse only, Figure 9(f)) and two radiance cues (diffuse + 1 specular basis, Figure 9(g)) are slightly better compared with the results of no radiance cues. Our method (with 4 radiance cues) can improve the results further. The quantitative results as shown in Table 4 confirm our observation. We should note that multiple radiance cues can be rendered in the same pass in the real-time rendering pipeline with negligible additional costs.

5 Discussion

5.1 Scenarios of our method

We will re-clarify our scenarios and potential applications in this section. The scenarios and potential applications of our method are quite similar to classic precomputed-based methods (both assume static scenes with dynamic lighting) which can be used in many applications. Our method can be regarded as an advancement on classic precomputed-based methods (e.g. PRT, light map). The training data generation and neural network training serve as the pre-computation stage. The parameters of the trained network are a compact representation of the given scene. The rendering of neural network inputs and network inference serves as the run-time rendering stage. Compared with classic precomputed-based methods, our method supports local lighting and high-frequency global illumination without any complex data structures.

5.2 Relationship with screen-space learning-based approach

Although the inputs of our method are the attributes of each shading point and screen-space neural buffer, there are fundamental differences between our method with existing screen-space approaches. Screen-space learning-based approach [40, 68] takes screen-space buffers as input and predicts global illumination by a deep neural network for dynamic scenes. The role of their neural network is to predict global illumination from the screen-space buffer of an arbitrary scene and there is no scene-specific information stored in the neural network. By contrast, our method assumes static scenes and synthesizes global illumination of the specific scene with dynamic viewpoint and lighting. The training dataset generation and the network training can be regarded as the sampling and fitting process of the radiance field of the specific scene respectively. Our rendering network

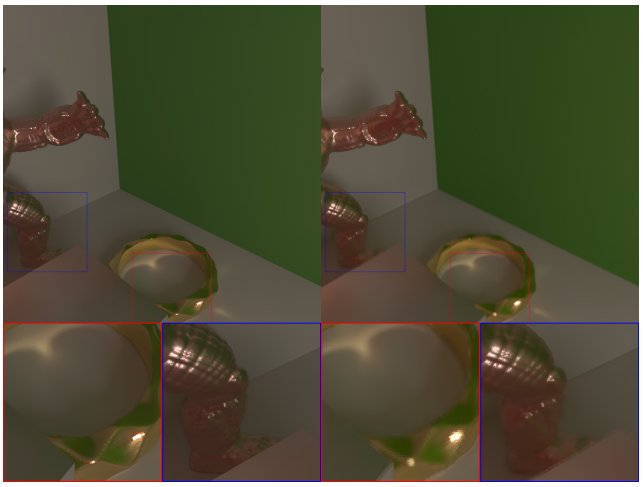
Fig. 12. Rendering results of our method from novel viewpoint/lighting combinations. These scenes exhibit rich global illumination effects: glossy interreflection (a, b, d, e, f, i, j, l, o), mirror reflection (c, d, g, h, i, k, m, n), and color bleeding (e, f, i, j).

is a compact representation of the radiance field of the scene. At run time, given input of a certain shading point as a query, the global illumination with dynamic viewpoint and lighting is interpolated from our compact neural scene representation.

In theory, adding explicit global representation such as point clouds or voxel-based volumes will enhance the knowledge about the static scene. However, to keep our method lightweight without complex data structure or pre-sampling process at run time, we use screen-space input that can be generated easily in the real-time rendering pipeline as the only input information. To make the mapping from input to global illumination both easy to learn and smooth, we propose to utilize carefully designed neural-network-friendly inputs that include the information of shading point, viewpoint, and lighting.

5.3 Joint bilateral upsampling based high-resolution pipeline

As mentioned before, our method is able to extend to higher resolution (e.g. 512×512) without any retraining. In practice, to achieve interactive run-time performance with high resolution, we utilize joint bilateral upsampling guided by high-resolution G-buffer (e.g. normal and position) to scale up the low-resolution indirect illumination generated by our network [31, 68]. The final render image is the sum of upsampled indirect illumination and high-resolution direct illumination which get a good balance between run-time performance and visual quality as shown in Figure 13.



High res. pipeline Bilateral pipeline

Fig. 13. The comparison of the rendering quality with two high-resolution pipelines. Our method supports high-resolution input naturally and is able to generate sharp details in 1024×1024 resolution (left). However, feeding high-resolution input directly into our network takes more time and cannot achieve interactive run-time performance. Instead, we can feed low resolution (256×256) input into our network and scale up to high-resolution indirect illumination result by bilateral upsampling. This bilateral pipeline (right) generates plausible high-resolution results and keep the run-time performance almost the same as low-resolution result.

5.4 Limitations

Our method is able to produce plausible global illumination with dynamic viewpoint and area lighting on a wide variety of scenes; however, it still has some limitations.

First, our method supports arbitrary viewpoints and area lighting. However, our rendering network can only produce plausible results in the view/lighting space covered by the training dataset. As noted in Section 3.5, we randomly sample both viewpoint and area light sources to get good coverage of view and lighting, but there are still some missing combinations of view and lighting and our method would fail in these cases (Figure 14 (b)). Sampling the viewpoint and

(a) Reference (b) Ours (c) Reference (d) Ours

Fig. 14. Failure cases of our method. Our method fails to generate plausible results for some view/light combinations (b) that are not covered by the training dataset. Besides, our method cannot fully reproduce refraction effects (d).

area light source adaptively (sample denser viewpoint and lighting in the region with larger fitting error) is an interesting future direction. The active exploration strategy proposed by Diolatzis et al. [11] is another interesting direction to sample the training data more efficiently.

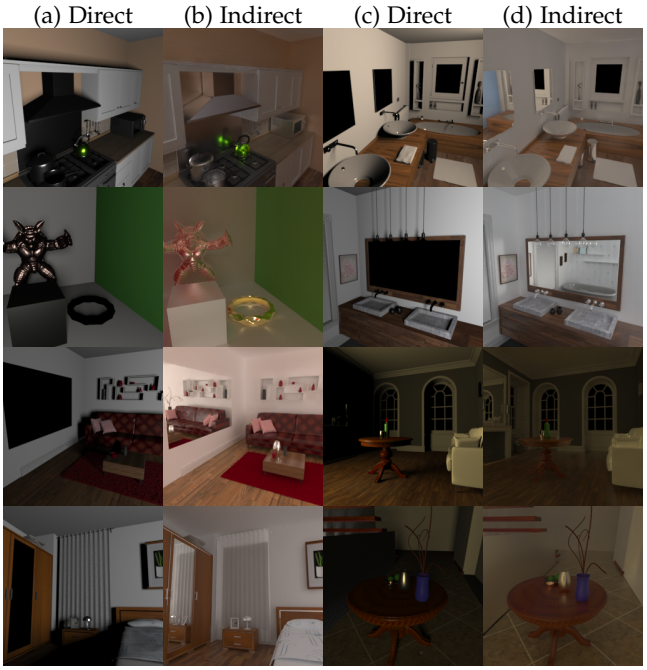
Second, our method supports different materials varying from diffuse to glossy or even mirror. As shown in the *Bathroom* scene (Figure 12 (g, h)), our method can generate plausible results for glass bulbs. However, current material representation (based on the material model of opaque objects) and lighting representation (the radiance cues include reflection only) are not efficient for translucent objects in general cases. As shown in Figure 14 (d), for a variant *Cornell box* scene (replacing the glossy cube with a glass sphere), our method fails to produce all refraction effects. We are interested in exploring other strategies to handle translucent materials.

Third, our method is able to achieve interactive run-time performance at 256 $\times$ 256 resolution. We plan to improve the run-time performance to real-time in future work. One possible solution is that we can split shading points into groups based on the material properties and then use smaller MLP per group (e.g. split shading points into diffuse/specular). Replace a single large MLP with multiple small MLPs can reduce the total number of trainable parameters which means smaller storage cost and faster inference. For high-resolution rendering, we rely on a bilateral upsampling-based pipeline to achieve interactive performance. This bilateral upsampling-based pipeline may miss material-related high-frequency global illumination details (e.g. texture details that are reflected by highly-glossy objects). We are interested in exploring efficient strategies that can preserve such high-frequency details without affecting performance.

Finally, our method is able to generate high-quality rendering results for indoor scenes exhibiting a wide range of global illumination effects (e.g. glossy interreflection, caustics, color bleeding), material properties (e.g. diffuse, glossy, and mirror) and geometrical complexity (e.g. grass-like carpet in *Living room-3* and *Bedroom*, plants in *Living room*, *Staircase-2*, and *Living room-3*). However, our method is limited to moderate-size indoor scenes and cannot scale with open-world large scenes.

6 Conclusions and Future Work

In this paper, we presented a novel neural global illumination approach for generating global illumination in static scenes with dynamic viewpoint and area lighting. The complex



(a) Direct (b) Indirect (c) Direct (d) Indirect

Fig. 15. Visualization of scenes rendered with direct illumination (a, c) and the indirect illumination rendered with our method (b, d). As shown here, these scenes exhibit strong multiple bounce indirect illumination.

mapping from the input of each shading point to global illumination is modeled by a deep fully-connected network that is well-suited to approximate such complex mapping. The neural-network-friendly input representation plays a crucial role in reducing the requirement of network size without affecting fitting quality. The positional encoding technique and screen-space neural buffer enable the network to learn high-frequency mapping efficiently. Our method takes direct lighting and screen-space buffers as inputs and does not need any complex spatial data structure or precomputation at run time. The trained model of our method is more compact (only 55.9 MB for each scene) compared with other existing precomputed-based GI methods.

For future work, we would like to explore strategies for efficient training data generation. We also intend to improve the run-time performance of our method further and extend our method to handle fully-dynamic scenes.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Project Number: 61932003).

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

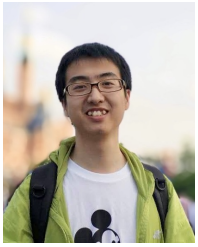
[2] S. Bako, T. Vogels, B. McWilliams, M. Meyer, J. Novák, A. Harvill, P. Sen, T. DeRose, and F. Rousselle, "Kernel-predicting convolutional networks for denoising monte carlo renderings," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2017)*, vol. 36, no. 4, pp. 97:1–97:14, 2017.

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

[4] B. Bitterli, C. Wyman, M. Pharr, P. Shirley, A. Lefohn, and W. Jarosz, "Spatiotemporal reservoir resampling for real-time ray tracing with

dynamic direct lighting," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 148–1, 2020.
[5] C. R. A. Chaitanya, A. S. Kaplanyan, C. Schied, M. Salvi, A. Lefohn, D. Nowrouzezahrai, and T. Aila, "Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–12, 2017.
[6] M. F. Cohen, J. R. Wallace, and P. Hanrahan, *Radiosity and realistic image synthesis*. Morgan Kaufmann, 1993.
[7] R. L. Cook and K. E. Torrance, "A reflectance model for computer graphics," *ACM Transactions on Graphics (ToG)*, vol. 1, no. 1, pp. 7–24, 1982.
[8] C. Crassin, F. Neyret, M. Sainz, S. Green, and E. Eisemann, "Interactive indirect illumination using voxel cone tracing," *Computer Graphics Forum*, vol. 30, no. 7, pp. 1921–1930, 2011.
[9] C. Dachsbacher and M. Stamminger, "Reflective shadow maps," in *Proceedings of the 2005 symposium on Interactive 3D graphics and games*, 2005, pp. 203–231.
[10] V. Deschaintre, M. Aittala, F. Durand, G. Drettakis, and A. Bousseau, "Single-image svbrdf capture with a rendering-aware deep network," *ACM Transactions on Graphics (ToG)*, vol. 37, no. 4, pp. 1–15, 2018.
[11] S. Diolatzis, J. Philip, and G. Drettakis, "Active exploration for neural global illumination of variable scenes," *ACM Transactions on Graphics (TOG)*, 2022.
[12] S. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor *et al.*, "Neural scene representation and rendering," *Science*, vol. 360, no. 6394, pp. 1204–1210, 2018.
[13] D. Gao, X. Li, Y. Dong, P. Peers, K. Xu, and X. Tong, "Deep inverse rendering for high-resolution svbrdf estimation from an arbitrary number of images," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–15, 2019.
[14] D. Gao, G. Chen, Y. Dong, P. Peers, K. Xu, and X. Tong, "Deferred neural lighting: free-viewpoint relighting from unstructured photographs," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–15, 2020.
[15] J. Granskog, F. Rousselle, M. Papas, and J. Novák, "Compositional neural scene representations for shading inference," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 135–1, 2020.
[16] P. Green, J. Kautz, W. Matusik, and F. Durand, "View-dependent precomputed light transport using nonlinear gaussian function approximations," in *Proceedings of the 2006 symposium on Interactive 3D graphics and games*, 2006, pp. 7–14.
[17] G. Greger, P. Shirley, P. M. Hubbard, and D. P. Greenberg, "The irradiance volume," *IEEE Computer Graphics and Applications*, vol. 18, no. 2, pp. 32–43, 1998.
[18] T. Hachisuka and H. W. Jensen, "Stochastic progressive photon mapping," in *ACM SIGGRAPH Asia 2009 papers*, 2009, pp. 1–8.
[19] T. Hachisuka, S. Ogaki, and H. W. Jensen, "Progressive photon mapping," in *ACM SIGGRAPH Asia 2008 papers*, 2008, pp. 1–8.
[20] J. Hasselgren, J. Munkberg, M. Salvi, A. Patney, and A. Lefohn, "Neural temporal adaptive sampling and denoising," in *Computer Graphics Forum*, vol. 39, no. 2. Wiley Online Library, 2020, pp. 147–155.
[21] E. Heitz, J. Dupuy, S. Hill, and D. Neubelt, "Real-time polygonal-light shading with linearly transformed cosines," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–8, 2016.
[22] E. Heitz, S. Hill, and M. McGuire, "Combining analytic direct illumination and stochastic shadows," in *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 2018, pp. 1–11.
[23] id Software, "Quake," https://github.com/id-Software/Quake-III-Arena, 1999.
[24] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.
[25] H. W. Jensen, "Global illumination using photon maps," in *Eurographics workshop on Rendering techniques*. Springer, 1996, pp. 21–30.
[26] J. T. Kajiya, "The rendering equation," in *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, 1986, pp. 143–150.
[27] S. Kallweit, P. Clarberg, C. Kolb, T. Davidovič, K.-H. Yao, T. Foley, Y. He, L. Wu, L. Chen, T. Akenine-Möller, C. Wyman, C. Crassin, and N. Benty, "The Falcor rendering framework," 3 2022, https://github.com/NVIDIAGameWorks/Falcor. [Online]. Available: https://github.com/NVIDIAGameWorks/Falcor
[28] A. Kaplanyan, "Light propagation volumes in cryengine 3," *ACM SIGGRAPH Courses*, vol. 7, p. 2, 2009.
[29] A. Kaplanyan and C. Dachsbacher, "Cascaded light propagation volumes for real-time indirect illumination," in *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, 2010, pp. 99–107.
[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[31] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele, "Joint bilateral upsampling," *ACM Transactions on Graphics (ToG)*, vol. 26, no. 3, pp. 96–es, 2007.
[32] A. W. Kristensen, T. Akenine-Möller, and H. W. Jensen, "Precomputed local radiance transfer for real-time lighting design," in *ACM SIGGRAPH 2005 Papers*, 2005, pp. 1208–1215.
[33] E. P. Lafortune and Y. D. Willems, "Rendering participating media with bidirectional path tracing," in *Eurographics Workshop on Rendering Techniques*. Springer, 1996, pp. 91–100.
[34] W.-S. Lai, J.-B. Huang, O. Wang, E. Shechtman, E. Yumer, and M.-H. Yang, "Learning blind video temporal consistency," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 170–185.
[35] X. Li, Y. Dong, P. Peers, and X. Tong, "Modeling surface appearance from a single photograph using self-augmented convolutional neural networks," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, pp. 1–11, 2017.
[36] M. McGuire, M. Mara, D. Nowrouzezahrai, and D. Luebke, "Real-time global illumination using precomputed light field probes," in *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 2017, pp. 1–11.
[37] 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 *European Conference on Computer Vision*. Springer, 2020, pp. 405–421.
[38] T. Müller, B. McWilliams, F. Rousselle, M. Gross, and J. Novák, "Neural importance sampling," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 5, pp. 1–19, 2019.
[39] T. Müller, F. Rousselle, J. Novák, and A. Keller, "Real-time neural radiance caching for path tracing," *arXiv preprint arXiv:2106.12372*, 2021.
[40] O. Nalbach, E. Arabadzhiyska, D. Mehta, H.-P. Seidel, and T. Ritschel, "Deep shading: convolutional neural networks for screen space shading," in *Computer graphics forum*, vol. 36, no. 4. Wiley Online Library, 2017, pp. 65–78.
[41] R. Ng, R. Ramamoorthi, and P. Hanrahan, "Triple product wavelet integrals for all-frequency relighting," in *ACM SIGGRAPH 2004 Papers*, 2004, pp. 477–487.
[42] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 165–174.
[43] M. Pharr, W. Jakob, and G. Humphreys, *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.
[44] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. Hamprecht, Y. Bengio, and A. Courville, "On the spectral bias of neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 5301–5310.
[45] G. Rainer, A. Bousseau, T. Ritschel, and G. Drettakis, "Neural precomputed radiance transfer," in *Computer Graphics Forum*, vol. 41, no. 2. Wiley Online Library, 2022, pp. 365–378.
[46] R. Ramamoorthi, *Precomputation-based rendering*. NOW Publishers Inc, 2009.
[47] P. Ren, J. Wang, M. Gong, S. Lin, X. Tong, and B. Guo, "Global illumination with radiance regression functions," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–12, 2013.
[48] T. Ritschel, T. Grosch, and H.-P. Seidel, "Approximating dynamic global illumination in image space," in *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, 2009, pp. 75–82.
[49] A. Robison and P. Shirley, "Image space gathering," in *Proceedings of the Conference on High Performance Graphics 2009*, 2009, pp. 91–98.
[50] S. Rodriguez, T. Leimkühler, S. Prakash, C. Wyman, P. Shirley, and G. Drettakis, "Glossy probe reprojection for interactive global illumination," *ACM Transactions on Graphics (SIGGRAPH Asia Conference Proceedings)*, vol. 39, no. 6, December 2020. [Online]. Available: http://www-sop.inria.fr/reves/Basilic/2020/RLPWSD20
[51] C. Schied, A. Kaplanyan, C. Wyman, A. Patney, C. R. A. Chaitanya,

J. Burgess, S. Liu, C. Dachsbacher, A. Lefohn, and M. Salvi, "Spatiotemporal variance-guided filtering: Real-time reconstruction for path-traced global illumination," in *Proceedings of High Performance Graphics*, 2017, pp. 2:1–2:12.
[52] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[53] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," *Advances in Neural Information Processing Systems*, vol. 33, 2020.
[54] P.-P. Sloan, J. Kautz, and J. Snyder, "Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments," in *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002, pp. 527–536.
[55] P.-P. Sloan, J. Hall, J. Hart, and J. Snyder, "Clustered principal components for precomputed radiance transfer," *ACM Transactions on Graphics (TOG)*, vol. 22, no. 3, pp. 382–391, 2003.
[56] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," 2018.
[57] T. Sun, J. T. Barron, Y.-T. Tsai, Z. Xu, X. Yu, G. Fyffe, C. Rhemann, J. Busch, P. E. Debevec, and R. Ramamoorthi, "Single image portrait relighting." *ACM Trans. Graph.*, vol. 38, no. 4, pp. 79–1, 2019.
[58] X. Sun, K. Zhou, Y. Chen, S. Lin, J. Shi, and B. Guo, "Interactive relighting with dynamic brdfs," *ACM Transactions on Graphics*, vol. 26, no. 3, p. 27, 2007.
[59] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," *arXiv preprint arXiv:2006.10739*, 2020.
[60] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. Nießner *et al.*, "State of the art on neural rendering," in *Computer Graphics Forum*, vol. 39, no. 2. Wiley Online Library, 2020, pp. 701–727.
[61] J. Thies, M. Zollhöfer, and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–12, 2019.
[62] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017.
[63] E. Veach and L. J. Guibas, "Optimally combining sampling techniques for monte carlo rendering," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 1995, pp. 419–428.
[64] J. Wang and R. Ramamoorthi, "Analytic spherical harmonic coefficients for polygonal area lights," *ACM Trans. Graph.*, vol. 37, no. 4, jul 2018. [Online]. Available: https://doi.org/10.1145/3197517.3201291
[65] R. Wang, J. Tran, and D. Luebke, "All-frequency relighting of glossy objects," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 2, pp. 293–318, 2006.
[66] L. Wu, G. Cai, S. Zhao, and R. Ramamoorthi, "Analytic spherical harmonic gradients for real-time rendering with many polygonal area lights," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 134–1, 2020.
[67] C. Wyman and A. Marrs, "Introduction to directx raytracing," in *Ray Tracing Gems*. Springer, 2019, pp. 21–47.
[68] H. Xin, S. Zheng, K. Xu, and L.-Q. Yan, "Lightweight bilateral convolutional neural networks for interactive single-bounce diffuse indirect illumination," *IEEE Transactions on Visualization & Computer Graphics*, no. 01, pp. 1–1, 2020.
[69] B. Xu, J. Zhang, R. Wang, K. Xu, Y.-L. Yang, C. Li, and R. Tang, "Adversarial monte carlo denoising with conditioned auxiliary feature," *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2019)*, vol. 38, no. 6, pp. 224:1–224:12, 2019.
[70] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018.
[71] Q. Zheng and M. Zwicker, "Learning to importance sample in primary sample space," in *Computer Graphics Forum*, vol. 38, no. 2. Wiley Online Library, 2019, pp. 169–179.
[72] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.



Duan Gao is a Ph.D. student in the Department of Computer Science and Technology, Tsinghua University. He received his bachelor degree from Department of Computer Science and Technology, Nanjing University in 2017. His research interests include physically-based rendering and neural rendering.

Haoyuan Mu is a senior undergraduate student in the Department of Computer Science and Technology, Tsinghua University. His research interests include computer vision and real-time rendering.

Kun Xu is an associate professor in the Department of Computer Science and Technology, Tsinghua University. Before that, he received his bachelor and doctor degrees from the same university in 2005 and 2009, respectively. His research interests include realistic rendering and image/video editing.