# LightFormer: Light-Oriented Global Neural Rendering in Dynamic Scene

HAOCHENG REN and YUCHI HUO[*], State Key Lab of CAD&CG, Zhejiang University, China
YIFAN PENG, University of Hong Kong, China
HONGTAO SHENG, WEIDONG XUE, HONGXIANG HUANG, JINGZHEN LAN, RUI WANG, and HUJUN BAO[*], State Key Lab of CAD&CG, Zhejiang University, China

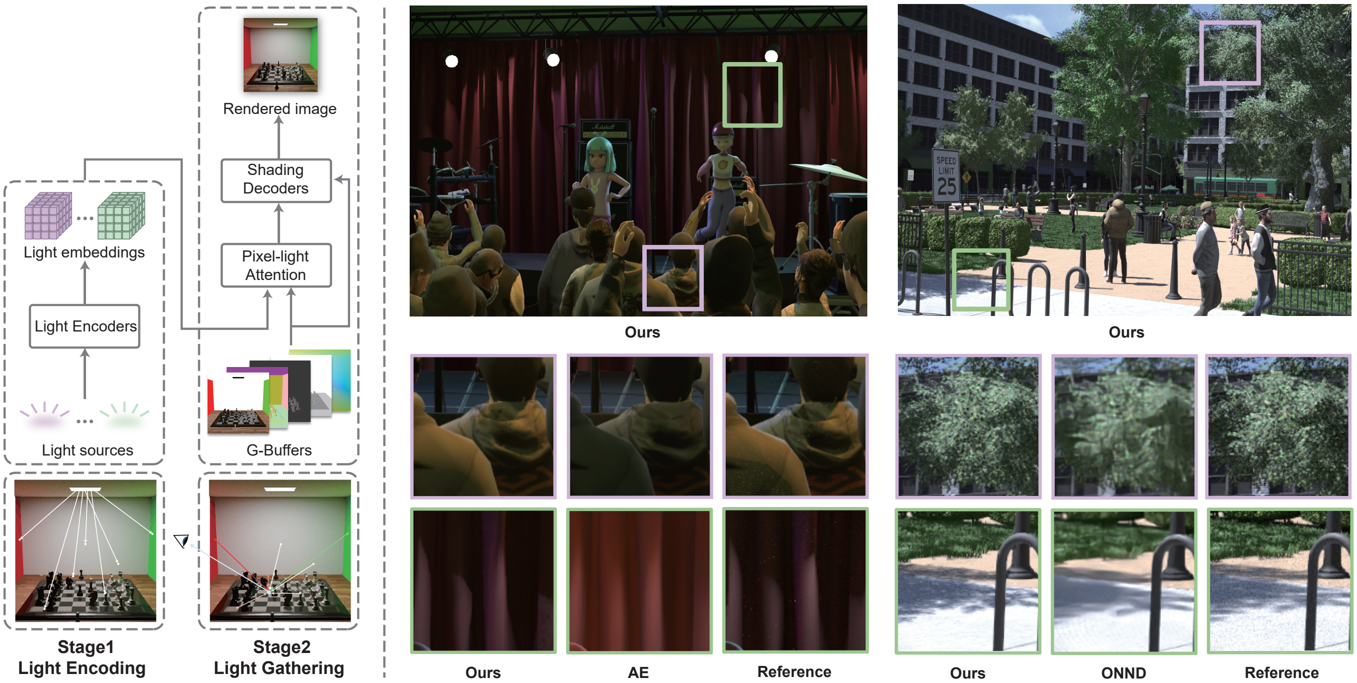


Fig. 1. Leveraging the well-established many-light rendering framework, we propose *LightFormer*, which consists of two stages — Light Encoding and Light Gathering, as shown on the left. By focusing on the neural representation from light sources, our framework can generate the realistic global illumination of fully dynamic and large-scale scenes in real time, which is challenging for state-of-the-art neural rendering techniques, as well as real-time path tracing and denoising methods, as presented on the right.

The generation of global illumination in real time has been a long-standing challenge in the graphics community, particularly in dynamic scenes with complex illumination. Recent neural rendering techniques have shown great promise by utilizing neural networks to represent the illumination of scenes and then decoding the final radiance. However, incorporating object parameters into the representation may limit their effectiveness in handling fully dynamic scenes. This work presents a neural rendering approach, dubbed *LightFormer*, that can generate realistic global illumination for fully dynamic scenes, including dynamic lighting, materials, cameras, and animated objects, in real time. Inspired by classic many-lights methods, the proposed approach focuses on the neural representation of light sources in the scene rather than the entire scene, leading to the overall better generalizability. The neural prediction is achieved by leveraging the virtual point lights and shading clues for each light. Specifically, two stages are explored. In the light encoding stage, each light generates a set of virtual point lights in the scene, which are then encoded into an implicit neural light representation, along with screen-space shading clues like visibility. In the light gathering stage, a pixel-light attention mechanism composites all light representations for each shading point. Given the geometry and material representation, in tandem with the composed light representations of all lights, a lightweight neural network predicts the final radiance. Experimental results demonstrate that the proposed LightFormer can yield reasonable and realistic global illumination in fully dynamic scenes with real-time performance.

[*] Corresponding authors

Authors' addresses: Haocheng Ren, swordigo@zju.edu.cn; Yuchi Huo, State Key Lab of CAD&CG, Zhejiang University, China, huo.yuchi.sc@gmail.com; Yifan Peng, University of Hong Kong, China, evanpeng@hku.hk; Hongtao Sheng, 22251006@zju.edu.cn; Weidong Xue, 22251027@zju.edu.cn; Hongxiang Huang, crisprhhx@zju.edu.cn; Jingzhen Lan, lanjz@zju.edu.cn; Rui Wang, ruiwang@zju.edu.cn; Hujun Bao, State Key Lab of CAD&CG, Zhejiang University, China, bao@cad.zju.edu.cn.

0730-0301/2024/7-ART1 $15.00
https://doi.org/10.1145/3658229

CCS Concepts: • **Computing methodologies → Rendering**; • **Machine learning**;

Additional Key Words and Phrases: Neural rendering, Scene representation, Light transport

**ACM Reference Format:**
Haocheng Ren, Yuchi Huo, Yifan Peng, Hongtao Sheng, Weidong Xue, Hongxiang Huang, Jingzhen Lan, Rui Wang, and Hujun Bao. 2024. LightFormer: Light-Oriented Global Neural Rendering in Dynamic Scene. *ACM Trans. Graph.* 43, 4, Article 1 (July 2024), 14 pages. https://doi.org/10.1145/3658229

## 1 INTRODUCTION

The exploration of neural rendering methods has reached unprecedented heights [Tewari et al. 2022] and significantly empowered various domains, including radiance field representation [Mildenhall et al. 2021; Müller et al. 2022], path tracing and adaptive sampling [Dong et al. 2023; Huo et al. 2020; Müller et al. 2021; Xu et al. 2023], denoising [Balint et al. 2023; Fan et al. 2021; Huo and Yoon 2021; Yu et al. 2021], complex material [Kuznetsov 2021; Vicini et al. 2019] and luminaires [Zhu et al. 2021]. Remarkably, real-time global illumination has been a long-standing challenging task in graphics, yet significant progress has been made recently via the utilization of neural networks. Beyond rendering high-quality global illumination for static scenes [Gao et al. 2022; Guo et al. 2022; Hadadan et al. 2021; Raghavan et al. 2023], several approaches have demonstrated the ability to handle scenes with rigidly transformed objects, variable materials, lights, and cameras [Diolatzis et al. 2022; Granskog et al. 2020, 2021; Zheng et al. 2023]. Nevertheless, manipulating fully dynamic scenes, particularly those with animated objects or on a large scale, remains challenging for existing methods.

State-of-the-art neural light transport methods employ neural networks to represent the complex rendering function. In particular, for representing the scene and lights, screen space methods typically represent the scene using G-Buffers [Nalbach et al. 2017; Xin et al. 2020], with lights represented through direct shading [Xin et al. 2020] and screen-space buffers [Gao et al. 2022]. Such screen-space representation overlooks information beyond the camera's view, potentially leading to significant errors. For instance, altering the color of a wall outside the screen does not impact the results. Diolatzis et al. [2022] address this issue by employing explicit variable parameters alongside G-Buffers to represent the scene and lights. However, representing dynamic scenes with animated objects proves to be impractical when the parameter size exceeds a limited number.

Alternative approaches encode the scene with observations as a supplement to screen-space features [Eslami et al. 2018; Granskog et al. 2020]. However, randomly selecting observation cameras may fail to cover the entire scene adequately, and demanding fully path-traced images as observations is impractical for real-time rendering. Nevertheless, these observations contribute to the generalizability by enabling the capture of scenes and lights, even for previously unobserved scenes. Building upon this insight, NeLT [Zheng et al. 2023] has introduced a comprehensive scene representation by observing the scene from the center of each variable object. However, this approach incurs a linear increase in rendering cost with the number of objects, and it does not support scenes with unseen objects due to the light transport properties of each object being baked into a network.

To tackle existing challenges, we introduce *LightFormer*, a neural rendering framework built upon observations from lights, extending from the many-light rendering framework [Dachsbacher et al. 2014]. In contrast to existing methods fusing the encoding of the scene and lights, the use of light observations provides a physics-based way for light transport. Drawing inspiration from the Transformer model [Vaswani et al. 2017], the proposed architecture consists of two stages — to encode the information from every light present in the scene into a light embedding; and to utilize the G-Buffer to query the corresponding light embedding, representing its incident radiance, for each shading point using an attention mechanism so as to decode them into the final radiance. This encoder-decoder architecture also aligns with the virtual point light (VPL) generation and VPL gathering stages in many-light rendering methods, as depicted in the left part of Figure 1. Built upon light observations, we also generate shading clues for shadow, specularity and indirect lighting respectively to improve rendering quality and generalization.

Our main technical contributions are as follows:

- We introduce a novel neural rendering framework that can handle full dynamic scenes in real-time, while preserving high-frequency shading details such as glossy reflection and shadow.
- We explore a *neural reflective shadow map* which can provide plausible global illumination effects (Sec. 4), in particular, those vital clues like visibility and indirect lighting, for light transport.
- We develop a neural light gathering algorithm tailored with a *pixel-light attention* mechanism (Sec. 5), which can efficiently gather the contribution of dynamic light sources in a many-light manner.

Experiments demonstrate that this framework produces superior results compared to previous neural rendering baselines, achieving better temporal consistency than real-time path tracing and denoising methods. Although we compared it with path tracing methods, we emphasize that our work primarily focuses on exploring the potential of a novel neural rendering framework rather than providing a mature global illumination solution for production rendering.

## 2 RELATED WORK

This work is based upon a wide body of relevant research of neural rendering, implicit scene representation, as well as realistic light transport, that are briefly summarized below.

### 2.1 Precomputed Global Illumination

Precomputed radiance transfer (PRT) [Sloan et al. 2002] has revolutionized real-time rendering of static scenes with dynamic environmental lighting. By storing light transport as coefficients of spherical harmonics (SH) basis functions, efficient integration is achieved. Alternative approaches such as wavelets or spherical Gaussian functions [Ng et al. 2004; Wang et al. 2009] have been explored in subsequent works, to obtain shading results across all frequencies. These basis functions, though useful, exhibit limitations regarding low-frequency representation and lack of rational invariance. Xu et al. [2022] recently presented neural basis functions to address these concerns. While previous works primarily focused on static scenes and distant lighting, there has been a significant effort to extend the PRT system with near-field area lights [Kristensen et al. 2005; Wu et al. 2020], editable materials [Ben-Artzi et al. 2008, 2006; Sun et al. 2007], and relax the restriction of static scene [Sloan et al. 2005; Zhou et al. 2005]. Sloan et al. [2005] employed zonal harmonics to represent the transport function with fast rotations, enabling local deformation in PRT. However, this approach does not capture global effects such as changes in shadowing. Zhou et al. [2005] precomputed shadow fields for each rigid object in the scene, resulting in real-time soft shadows under dynamic lighting.

Despite these advancements, the dynamic geometry in PRT is still limited to rigid object transformations or local surface deformations [Ramamoorthi et al. 2009]. Baking lightmaps and light probes [Greger et al. 1998] are widely used in the industry for real-time global illumination. Recently, the glossy probe projection technique [Rodriguez et al. 2020] has been proposed for the interactive rendering of static complex glossy scenes by handling all types of glossy paths. Guo et al. [2022] introduced a probe-based neural rendering framework, yielding the efficient pre-computing of light probes as well as producing high quality global illumination. We note that all these recent works primarily focus on static scene and lighting condition, and their performance in complex, dynamic scenarios remains elusive.

## 2.2 Learning-driven Light Transfer

In recent years, there has been a significant focus on using deep learning to represent scenes and precompute global illumination [Tewari et al. 2022]. These learning-based methods offer a smaller memory footprint compared to previous techniques, thanks to the compactness of neural networks. Ren et al. [2013] were the pioneers in modeling indirect illumination using neural networks with dynamic *point* lighting and views. Eslami et al. [2018] demonstrated that neural networks can learn the scene representation and render novel views by encoding multiple observation images. To enhance the controlability of neural scene representation, Granskog et al. [2020] disentangled lighting, material, and geometry information from representation, and utilized G-buffers to generate sharp and realistic results. NeRF [Mildenhall et al. 2021] represents the radiance field of a static scene using neural networks defined with a volume. Remarkable reconstruction results in novel views through volume rendering techniques have been realized. Similarly, the Neural Radiosity [Hadadan et al. 2021] represents the radiance filed using a neural network and solves the rendering equation by minimizing the residual norm. We note that the above methods cannot handle dynamic scenes without retraining. Building upon the PRT framework, recent advancements [Raghavan et al. 2023; Rainer et al. 2022] employ neural networks to represent environmental lighting but assume also static scenes and distant lighting.

A recent advancement by Diolatzis et al. [2022] has utilized active learning techniques to sample effective data for reconstructing complex global illumination effects. In contrast to previous works that rely on implicit neural scene representation encoded from observations [Eslami et al. 2018; Granskog et al. 2020], this method uses an explicit parametric scene representation. However, this representation is constrained to specific scene configurations and faces challenges in representing animated objects or textures, limiting its generalizability. Additionally, complex dynamic scenes with numerous variables, such as variable textures, remain challenging due to the size of the scene representation vector. Recently, NeLT [Zheng et al. 2023] has made progress towards a flexible neural scene representation by modeling light transfer related to each geometric object through neural networks. Although it can generate high-quality results for scenes with multiple rigid-transformed geometries, handling complex scenes with numerous dynamic objects remains challenging due to the prohibitively time-consuming composition process, which scales linearly with the number of objects. Notably, our framework's performance is not directly dependent on the number of objects (except for lights), enabling real-time rendering of dynamic, large-scale scenes.

## 2.3 Screen Space Global Illumination

The screen space approach has been extensively utilized in real-time applications, primarily due to its effectiveness and ease of implementation. Screen space ambient occlusion (SSAO) [Shanmugam and Arikan 2007] is a well-established technique for generating plausible ambient occlusion computed from the depth buffer. Building upon SSAO, the screen space directional occlusion (SSDO) [Ritschel et al. 2009] introduces indirect lighting effects such as color bleeding. Subsequent approaches have employed screen space filtering [Robison and Shirley 2009] or screen space ray marching [McGuire and Mara 2014] to obtain glossy reflection effects. However, due to the limitations of screen space information, these methods mostly can only provide approximate results.

In recent years, there has been a surge of screen space global illumination methods combined with neural networks. Deep shading [Nalbach et al. 2017] employs convolutional neural networks to predict various visual effects solely using screen space buffers. While it achieves realistic outcomes, temporal stability is not addressed. Xin et al. [2020] improved temporal consistency by introducing a new temporal loss, but their focus is limited to single bounce diffuse indirect illumination. Datta et al. [2022] applied neural networks to accurately generate soft shadows from screen space shadow buffers, but this approach does not cover indirect illumination under multiple light sources. Recently, Gao et al. [2022] proposed a neural rendering framework for predicting high-quality indirect illumination from screen space buffers and direct illumination, but it is restricted to dynamic area lights in static scenes.

In all, compared to precomputed global illumination, screen space approaches often struggle to produce high-quality results due to the limited information available beyond the screen. Notably, mapping screen space buffers to global illumination is a highly underconstrained and ambiguous problem, posing significant challenges for learning algorithms.

## 2.4 Real-time Ray Tracing

Combined with specialized hardware, namely RTCore, and state-of-the-art denoising algorithms [Balint et al. 2023; Huo and Yoon 2021;

Müller et al. 2021; Nvidia 2023b], real-time ray tracing has demonstrated exceptional physics-based realistic results in recent years. However, maintaining temporal consistency can be challenging due to the nature of its sampling procedure [Balint et al. 2023], especially when working with limited time budgets and hardware capabilities. In often cases, very low samples per pixel are allowed under real-time scenarios [Fan et al. 2021; Nvidia 2023a]. State-of-the-art deep-learning-driven denoisers are not fully capable of addressing this issue. For instance, temporal accumulation methods may further blur the results.

In contrast to real-time ray tracing and denoising methods, our method leverages temporally consistent and clear inputs [Diolatzis et al. 2022; Granskog et al. 2020; Zheng et al. 2023], along with our elaborately designed pipeline, to achieve strong temporal consistency. Despite the path tracing method excels at effects like pure specular reflection and refraction, and it has great generalizability to support complex materials and light transport, our method still outperform it in certain scenarios as demonstrated in Section 6.2. Notably, our method effectively preserves high-frequency shading details like shadow, and provides more temporally stable results even in large scenes, where tracing enough light paths under limited time is challenging.

# 3 OVERVIEW

## 3.1 Problem Statement

Our aim is to explore a neural rendering method for real-time global illumination of fully dynamic scenes, including but not limited to dynamic lights, views, materials, as well as transformable and animated objects. In this context, the rendering process becomes a complex, non-linear mapping given the scene, lights and camera, represented as follows:

$$L(o, \omega) = \mathcal{F}(o, \omega; \mathcal{S}, \mathcal{L}), \tag{1}$$

where $S$ represents the entire scene, $\mathcal{L}$ represent all light sources, $o$ is the origin of the camera, and $\omega$ represents the direction from camera origin to a shading point.

The lack of generalization in prior works hinders their ability to effectively render dynamic scenes. To this end, our focus is on learning the neural network, denoted as $\mathcal{F}$, to function as a neural shader, instead of *baking* the scene $\mathcal{S}$ and light $\mathcal{L}$ directly into the network. To achieve this objective, we aim to extract more comprehensive and useful information from $\mathcal{S}$ and $\mathcal{L}$, which would serve as network-friendly inputs. This process guides the network to concentrate on the task of light transfer, rather than memorizing specific scene configurations.

Our approach to enhancing the rendering of dynamic scenes is based on leveraging observations from light sources. In contrast to random camera observations that combine the encoding of scene and light, light observations offer more valuable information for light transport by separating the encoding of $\mathcal{S}$ and $\mathcal{L}$. First, the use of light observations provides sufficient shading clues, such as classic shadowmap for visibility. Second, akin to the concept of reflective shadowmap [Dachsbacher and Stamminger 2005], the pixels in light observations can be considered as indirect light sources, providing crucial cues for global illumination. Lastly, unlike the per-object representation that scales rendering costs by the number of objects [Zheng et al. 2023], our way of leveraging light observations enables to render large-scale dynamic scenes using neural rendering techniques.

## 3.2 Pipeline Overview

To fully leverage the benefits of light observations for rendering dynamic scenes, we explore a pipeline incorporating features of many-light rendering methods [Dachsbacher et al. 2014] and the Transformer architecture [Vaswani et al. 2017]. Our pipeline consists of two main stages: the Light Encoding Stage and the Light Gathering Stage, corresponding to the light generation-gathering stages of the many-light rendering, and the encoder-decoder stages of the Transformer, respectively.

In the light encoding stage, we first capture the scene from each light using reflective shadow maps [Dachsbacher and Stamminger 2005]. Each texel of the rendered reflective shadow map represents an indirect virtual point light (VPL) emitted from the light source. The depth information in the reflective shadow map is utilized as the traditional shadow map, which is then converted to the screen space to serve as a shading clue. All of these inputs are encoded using a series of encoders to obtain a light embedding for each light source.

In the light gathering stage, our objective is to gather light information for each shading point and generate the final rendered image. For scenes with multiple light sources, we employ a Transformer block to perform attention for all the light embeddings, resulting in an effective composed light embedding for each shading point. This procedure replaces the light culling step in the classic many-light rendering framework, which often requires manually-defined error bounds [Walter et al. 2005] or selecting a small subset of virtual lights [Hašan et al. 2007]. Given the composed light embedding, as well as geometry and material information for each shading point, we utilize shading decoders to generate the final rendered result.

# 4 LIGHT ENCODING

The process of encoding each light in the scene to obtain its light embedding is illustrated in Figure 2. We utilize the VPL to represent lights following the many-light rendering technique [Dachsbacher et al. 2014]. By inheriting excellent traits from the point-based representation, VPL leads to a unified representation for light sources, enabling a unified encoding scheme for different light types. Given the distinct properties of direct and indirect light transfer, we define VPLs into two types accordingly. For direct illumination, only the lighting information from emitted surfaces of lights is necessary, denoted as *direct* VPLs. For indirect illumination, all the illuminated surfaces should be considered as light sources, termed as *indirect* VPLs. Notably, two different encoding networks are applied to obtain their light embedding separately. To aid the framework in producing complex, high-frequency shading results and to enhance the framework's generalization, we explore additional information of $\mathcal{S}$ and $\mathcal{L}$ as network-friendly screen-space buffers, termed *shading clues*. As each shading clue maintains its unique feature, different encoding networks are applied to encode them separately.

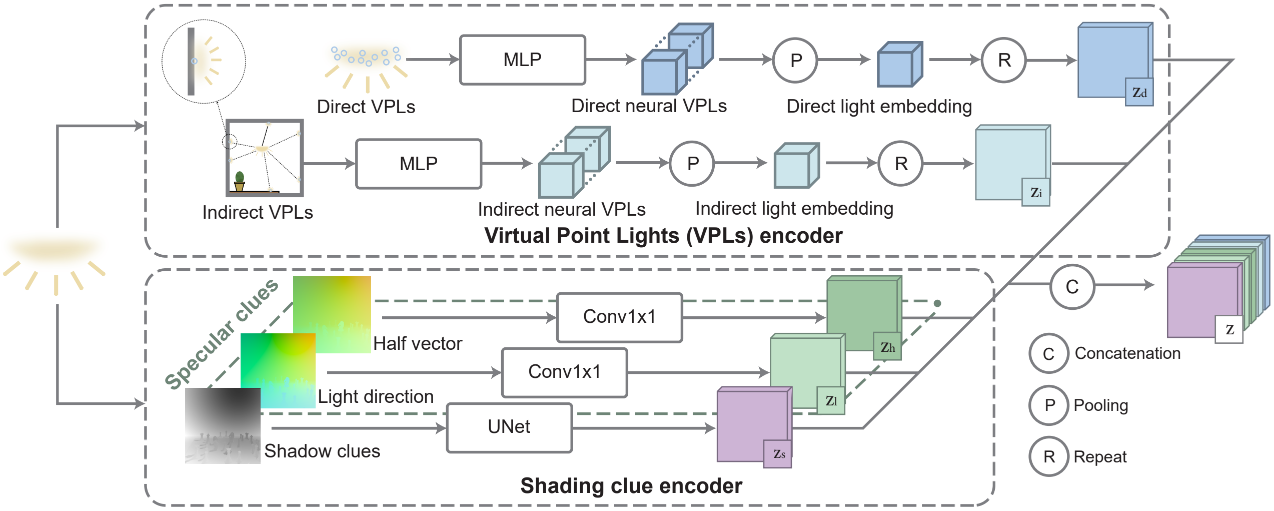


Fig. 2. Light Encoding Stage of LightFormer. For each light in the scene, we apply a series of VPL encoders and shading clue encoders to encode different light properties. These neural features are then concatenated to obtain a unified light embedding for each light.

## 4.1 Direct Light Embedding

For each light present in the scene, we utilize random samples on the emitted surface to generate a series of direct VPLs. Additionally, we apply the importance sampling based on the emitted radiance distribution to obtain representative samples. For each sample, the position, normal, emitted radiance, and sampling probability are recorded. Subsequently, we calculate the expected power per direct VPL and normalize them with the average expected power, following the NeLT work [Zheng et al. 2023]. This operation contributes to the generalization of direct VPLs, especially when dealing with lighting distribution that exhibits a high dynamic range (HDR).

In addition to area light and environment light, our framework supports the Delta light sources, such as point light and directional lights. The invalid fields for some light (e.g., the position of VPLs from environment lighting) are filled with unique constant values to maintain a unified representation. This representation provides ample information, including the shape of arbitrary area lights and their spatially varying emission properties.

We utilize the multilayer perception (MLP) to encode each direct VPL. To achieve a unified neural representation for each light, we employ the average pooling to aggregate them. Details of the network architecture can be found in the supplementary document.

## 4.2 Neural Reflective Shadow Map

The reflective shadow map (RSM) [Dachsbacher and Stamminger 2005] serves as the foundation for our approach to rendering shadow and indirect lighting, which is considered as a *neural reflective shadow map*. Each indirect VPL contains depth, position, normal and reflected flux. In contrast to traditional RSM techniques that employ heuristic rules to select a portion of indirect VPLs, we utilize neural networks to encode and gather information from all indirect VPLs. Similarly, our method also avoids any explicit visibility tests when computing illumination from indirect VPLs for performance consideration.

It is important to note that the original RSMs are designed for delta lights, such as point lights and directional lights. To extend the applicability of RSMs to other types of lights, a point proxy is required. Unlike previous methods that select the center point of the area light [Datta et al. 2022], our method utilizes the mean position of direct VPLs. In such a way, we can obtain a more representative proxy for general lights, since the distribution of direct VPLs is importance sampled considering the emitted radiance distribution.

The RSM is stored in a cubemap for omni-directional light sources like point lights. For directional or spotlights, a single texture is utilized. However, when dealing with area lights, one texture with a perspective view is not sufficient due to the significant distortion. Therefore, we employ a cubemap to store high-quality light observations for area lights, with invisible pixels being discarded to maintain efficiency. Similar to direct VPLs, we utilize the MLP to encode each indirect VPL. In addition, we leverage the depth information in the RSM to generate shading clues of shadows, as we will discuss in the following section.

## 4.3 Screen-space Shading Clues

In the field of neural light transfer, the accurate rendering of shadows and specular effects poses a significant challenge due to their inherent high-frequency properties [Diolatzis et al. 2022; Zheng et al. 2023]. G-buffers, while widely used, often fail to provide enough information about these effects. Although, in theory, direct VPLs can faithfully represent the light source, it is still difficult to learn the complex light transfer function using only direct VPLs as inputs.

To address this challenge, we introduce shading clues for shadows and highlights. By providing a more comprehensive representation of the scene and its lighting, we aim to make it easier for neural networks to learn and improve their generalizability. Our approach

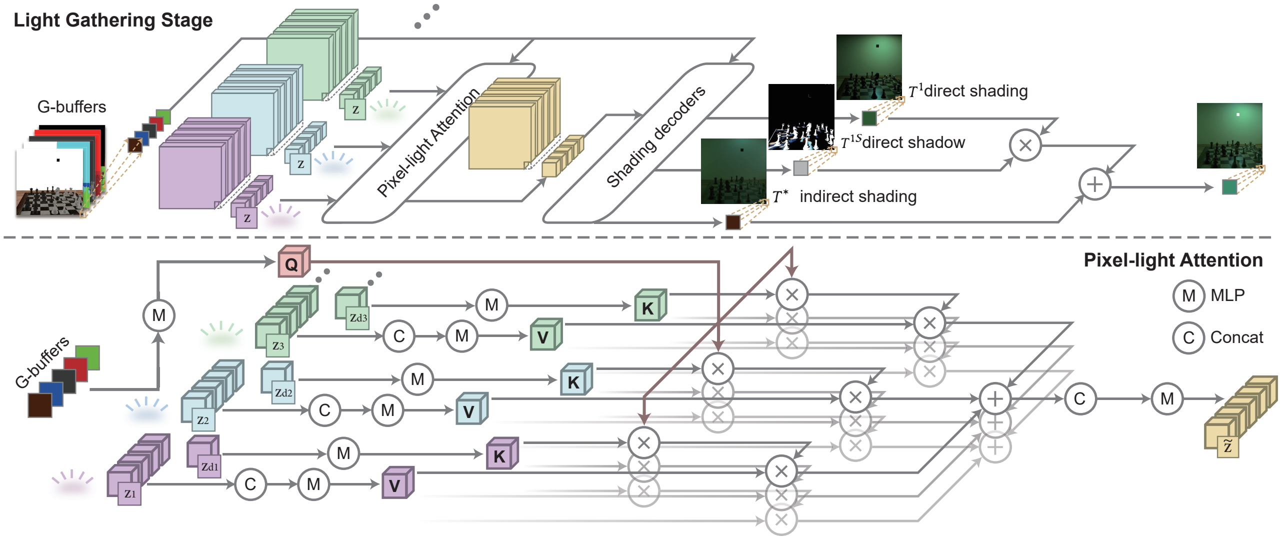


Fig. 3. Light Gathering Stage of LightFormer. Given each light source's screen-space embedding and G-Buffers, we first perform pixel-light attention per pixel. The composed light embedding of each shading point, together with its G-Buffers, are used to predict final radiance by shading decoders. We illustrate our pixel-light attention for one shading point under three light sources.

is motivated by recent advances in relighting techniques like Zeng et al. [2023], which also rely on similar hints to improve rendering accuracy.

*Shadow clues.* Given the generated shadow map, we can resolve the visibility information explicitly. Neural shadow mapping [Datta et al. 2022] employs a compact UNet to generate high-quality soft shadow from a series of screen space shadow buffers. We expand upon this comprehensive study to achieve soft shadows under multiple lights. The input of our shadow clues encoder is a 7-channel buffer:

$$S = \{d - d_f, d/d_f, c_e, c_c, \mathbf{p}\}, \tag{2}$$

where $d$ is the emitter-to-occluder depth, $d_f$ is the pixel-to-emitter distance, $c_e$ and $c_c$ are dot products of normal with light direction and view direction, respectively. $\mathbf{p}$ is the position buffer from G-Buffers. Note that, the $d$ is obtained by projecting the shadow map of RSM to screen space. Similarly, these inputs are encoded by the UNet instead of MLP, considering the importance of spatial information for reconstructing soft shadows. Compared to neural shadow mapping, our method exhibits several key distinctions.

Firstly, our encoder generate an embedding that stores visibility information of each shading point, as opposed to neural shadow mapping which directly predicts the final shadow mask. In scenes with multiple light sources, neural shadow mapping requires predicting individual shadow masks for each light source. In contrast, our method achieves superior performance by fusing shadow embeddings of all lights and decoding them only once. By focusing on optimizing the encoding process, we have managed to reduce the size of the UNet. The detailed architecture can be found in the supplementary document.

Secondly, in addition to the original input buffers [Datta et al. 2022], we incorporate a position buffer $\mathbf{p}$ from the GBuffer as an auxiliary input to enhance shadow quality. The original inputs utilize the distance from occluder to receiver for soft shadow effects, resulting in identical inputs for all shading points not in shadow. This poses a challenge for the convolutional network to differentiate between occluders and receivers, leading to artifacts such as blurred shadows being cast from the receiver onto the occluder's surface. By including the position buffer, we provide clues about the scene's geometry, which helps mitigate this phenomenon.

Note that high-quality shadow clues depend on high-resolution shadow maps. However, the resolution of the reflective shadow map directly impacts the performance of the indirect VPL encoding process. To strike a balance between quality and performance, we render depth buffers at full resolution to ensure accurate visibility, while other buffers have lower resolutions due to the low-frequency nature of indirect illumination. In practice, we employ a resolution of 1024×1024 per cubemap face for depth storage, and a resolution of 64×64 for position, normal, and reflected flux storage.

*Specular clues.* In addition to shadows, predicting the specular component in computer graphics is also challenging due to its high-frequency variation. The appearance of specular shading is determined by factors such as lighting direction, view direction, shading normal and material property. In dynamic scenes, where all these factors are subject to change, rendering specular effects becomes a higher-dimensional mapping compared to static scenes.

Similar to approaches used for shadow clues, we convert light information into screen-space buffers. First, we utilize a light direction buffer which enables each shading point to be aware of the direction of each light source. For area lights, we employ the position of the light's point proxy. Even though the proxy's position is an approximation, we find the network implicitly calibrates the bias and shows no visible impact in most cases. Furthermore, drawing inspiration from the well-established Blinn-Phong shading model [Blinn 1977], we introduce a half-vector buffer as another clue for rendering specular effects. By simply taking the dot product of the normal and half-vector, we can obtain the approximate representation of specular results, thus reducing the complexity of predicting specular effects. We encode the half-vector and light direction separately using MPLs due to their different properties.

For each light source, after encoding aforementioned features separately, we combine them into a unified screen-space light embedding. The embedding of direct and indirect VPLs is represented as a feature vector, while the embeddings for shading clues are in the form of screen-space feature maps. Following the methodology established by Granskog et al. and Diolatzis et al. [2022; 2020], we expand the VPL embedding to match the screen-space dimension and then concatenate all embeddings to obtain the final light embedding.

## 5 LIGHT GATHERING

At this stage, given the light embedding of each light source in the scene, we can illuminate the scene. Akin to the light gathering process in many-light rendering [Dachsbacher et al. 2014], the shading process can be formulated as:

$$L(x, \omega) = \sum_{j} f(x) G(x, y_j) V(x, y_j) L_j, \tag{3}$$

where $x$ is the shading point, $y_j$ represent the $j^{th}$ VPL, $f(x)$ is the BRDF function, $G(x, y_j)$ represents the geometry term and $V(x, y_j)$ represents the visibility between shading point $x$ and VPL $y_j$.

For speed acceleration, we handle all VPLs originating from the *same light source* simultaneously as a light embedding, instead of accumulating contributions from each VPL individually as in classical methods. Specifically, we utilize the neural network to gather all light information from light embeddings implicitly for each shading point. The light gathering stage consists of two steps (Figure 3). First, we compose all light embedding using the pixel-light attention mechanism (Section 5.1). Next, we render results given the composed light embedding of each shading point (Section 5.2). This process is formulated as:

$$\widetilde{\mathbf{z}} = C(g(x); \mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_n}), \tag{4}$$

$$L(x, \omega) = \mathcal{R}(g(x); \widetilde{\mathbf{z}}). \tag{5}$$

Herein, the G-Buffers $g(x)$ contains the position, normal, diffuse color, specular color, and roughness. $\widetilde{\mathbf{z}}$ is the composed light embedding. $\mathbf{z_k}$ represents the light embedding of the $k^{th}$ light. $C$ and $\mathcal{R}$ are the neural networks for composing light embedding (Section 5.1) and rendering (Section 5.2), respectively.

### 5.1 Pixel-light Attention

A straightforward approach to gathering all VPLs is applying the pooling operation like the PointNet [Qi et al. 2017]. However, the importance of different lights varies for each shading point. For example, a light that is far away from the shading point or has a low emitted radiance generally shows a subtle influence on the final radiance. Traditional methods, such as lightcuts [Walter et al. 2005] and MRCS [Hašan et al. 2007], aim to accelerate the gathering stage based on this insight.

In contrast to clustering or culling VPLs explicitly [Prutkin et al. 2012; Walter et al. 2005], our neural light transfer seeks a proper way to compose a compact neural light representation by exploring the adaptive importance of each light. This compact representation aids the decoder to concentrate on the critical aspects of light information, leading to alleviate its workload, thereby enabling the use of a smaller and faster decoder. Build upon the cross attention operation from the Transformer architecture [Vaswani et al. 2017], we explore the *pixel-light attention* to gather all light embeddings effectively. Figure 3 illustrates the details of our pixel-light attention block.

The attention mechanism computes a set of weights given the query $Q$ and the key $K$, the result can be obtained by composing the set of value vectors $V$ by these weights. In our pixel-light attention, the G-Buffers of each shading point are used as $Q$, the light embeddings $\mathbf{z}$ of all lights are regarded as $V$. Unlike using the same embedding as $K$ and $V$ in classic cross-attention, we slice the direct light embeddings $\mathbf{z_d}$ from $\mathbf{z}$ to become $K$. The ablation study shows the computational cost would significantly increase (about 3x compared to ours) using the full light embedding as $K$.

We also utilize the multi-head attention [Vaswani et al. 2017] with 8 heads for better practical performance. After neural light gathering stage, the composed light embedding is fed as input into the decoder networks.

### 5.2 Rendering

With the composed light embedding for each shading point, we then decode the final radiance given geometry and materials information as input, as Eq. 5. Specifically, as the direct and indirect shading rely on different physical factors, we apply the separate decoders to predict them:

$$\begin{aligned} T^1, T^{1S} &= \mathcal{R}_d(g(x); \widetilde{z_d}, \widetilde{z_h}, \widetilde{z_l}, \widetilde{z_s}), \\ T^* &= \mathcal{R}_i(g(x); \widetilde{z_i}), \end{aligned} \tag{6}$$

where $\mathcal{R}_d$ and $\mathcal{R}_i$ represent the direct and indirect decoders, respectively. $\widetilde{z_d}$, $\widetilde{z_h}$, $\widetilde{z_l}$, $\widetilde{z_s}$, and $\widetilde{z_i}$ represent the direct light, half vector, light direction, shadow, and indirect light parts of the composed light embedding $\widetilde{\mathbf{z}}$ (Fig. 2), respectively. $T^1$, $T^{1S}$, and $T^*$ represent different light transport, namely direct shading, direct shadow, and indirect shading. The final radiance is obtained by composing them:

$$L(x, \omega) = T^1 \odot T^{1S} + T^*, \tag{7}$$

where $\odot$ represents the Hadamard product (a.k.a, pixel-wise multiplication). Similar to NeLT [Zheng et al. 2023], we believe adding separate losses on each shading component can improve the performance, and the validation of light transfer decomposition is shown in Sec. 6.3.

For the direct and indirect decoders, we utilize a neural network architecture similar to the prior work by Gao et al. [2022], which is a lightweight variation of the pixel generator [Diolatzis et al. 2022; Granskog et al. 2020; Sitzmann et al. 2019]. Our observations indicate that the implementation of the compact decoder network significantly accelerates the decoding process when compared to the vanilla pixel generator. Simultaneously, it delivers rendering results of satisfactory quality thanks to the powerful light embedding. Note that, the direct decoder generates both direct shading

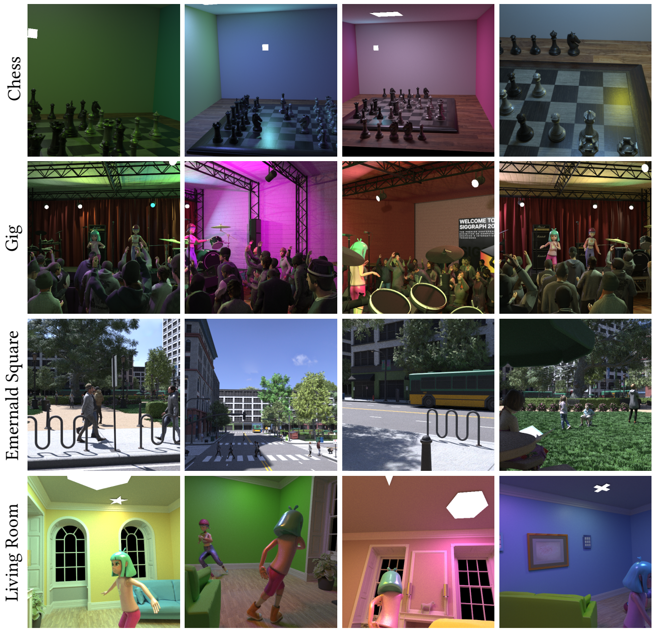


Fig. 4. Visualization of rendered scene samples in our dataset.

and shadow. Specifically, we inject the composed shadow embedding $\widetilde{\mathbf{z}_s}$ in the middle layer of the decoder for predicting shadow. Further details regarding the network architecture are presented in the supplementary document.

## 6 RESULTS

### 6.1 Implementation Details

*Dataset.* To validate the effectiveness of our proposed approach, we have built four highly dynamic scenes, namely *Chess*, *Gig*, *Emernald Square*, and *Living Room*. In all four scenes, we have varied the camera views to ensure the diversity of training data. For each dataset, we render 20,000 random scenes for training and 100 random scenes for testing. In the process of VPL generation, we sample 500 VPLs for each light, except for environmental lighting with 2000 VPLs. Figure 4 showcases selected rendering examples of our datasets.

Specifically, in the *Chess* scene, the chessboard as well as the highly specular chessman are placed within a classic Cornell box. Two area lights are placed on the top and back faces, respectively. Each chessman is animated, and two lights can be transformed. Moreover, the color of two side faces of the Cornell box could be tuned. The *Gig* scene is a modern stage with highly dynamic lighting, comprising six area lights for stage lighting, each with varying intensity and color, and a large emissive textured lighting on the wall. Two stage lights are spinning continuously. The base color of the wall can be randomized to create challenging global illumination scenarios. Additionally, 28 animated characters are present on the stage or on the floor. The *Emernald Square* scene is a large-scale scene with a total of 10.68M triangles, including 40 animated characters and buses. The high-fidelity trees cast realistic yet complex shadows, posing a significant challenge for rendering. The scene is illuminated by environmental lighting, and the environment map can be rotated and changed. The *Living Room* scene depicts a modern indoor scene with two animated characters. The living room is illuminated by three area lights with different shapes. The sofa and lights can involve translation, rotation, and scaling.

We have utilized the Falcor [Kallweit et al. 2022] rendering framework to generate all the training data. The reference images are rendered using path tracing with a resolution of 512×512 and 2,048 samples per pixel. The components of direct shading, shadow, and indirect shading are recorded during path tracing. The size of the shadow map varies across scenes, with most scenes using a resolution of 1024×1024.

To handle the large-scale scene in *Emernald Square*, very high-resolution shadow map is needed to maintain accuracy. The cascaded shadow map (CSM) [Zhang et al. 2006] is a classic approach used to achieve the shadow accuracy within a limited memory footprint. Inspired by the spirit of CSM, we extend our neural reflective shadow map with 4 cascades for high-quality shadow clues in large-scale scenes. Similarly, different resolutions are applied for all RSM buffers in each cascade, with maximal 2,048×2,048 resolution.

*Optimization.* We apply a hybrid loss on the outputs of shading decoders. The loss function can be represented as:

$$\mathcal{L} = \mathcal{L}_1(\hat{T}^1, T^1) + \mathcal{L}_1(\hat{T}^*, T^*) + \mathcal{L}_1(\hat{T}^{1S}, T^{1S}) + \lambda \mathcal{L}_{VGG}(\hat{T}^{1S}, T^{1S}), \tag{8}$$

where the per-pixel $\mathcal{L}_1$ loss is calculated by all shading components.

We observe that applying the structural dissimilarity (DSIIM) loss like previous work [Diolatzis et al. 2022] does not yield any performance gain in our cases. Additionally, we incorporate the VGG loss for better soft shadow, following the approach outlined in the neural shadow mapping [Datta et al. 2022], and empirically set the weight $\lambda$ to 0.1 for approximately equal magnitudes. Akin to the previous methods employed in handling high-dynamic-range images [Xu et al. 2019; Zheng et al. 2023], we also utilize the log transform $log(1 + x)$ prior to calculating the loss function. This step aids to stabilize the training process.

Our model is trained end-to-end by the Adam optimizer, with a learning rate of $10^{-4}$ and a mini-batch size of 4. All the encoders and decoders are trained jointly. The model is optimized for approximately 50 hours on two NVIDIA RTX A6000 graphics cards.

### 6.2 Comparison

To validate the effectiveness of our proposed approach, we compare our results with state-of-the-art neural rendering methods, namely CNSR [Granskog et al. 2020] and AE [Diolatzis et al. 2022]. Furthermore, we employ the latest version of Nvidia OptiX AI-accelerated Denoiser (ONND) [Nvidia 2023b] and Intel Open Image Denoise (OIDN) [Áfra 2024] as strong baselines for real-time path tracing and denoising, following the approach outlined in the recent work by Balint et al. [2023].

To ensure a fair comparison, we train CNSR and AE on the same dataset for each scene. Both models are trained for an equal or longer period to guarantee convergence. For CNSR, we utilize full G-Buffers, as in our proposed methods. Regarding AE, as it is designed for non-generalizable cases, we provide the parameter of the normalized

Table 1. Quantitative results of ours and state-of-the-art neural rendering methods. The timing of CNSR and AE are measured in PyTorch on RTX A6000. The timing of ONND and OIDN are the sum of path tracing time (accelerated by RTCore) in Falcor and the denoising process on an RTX 4090. We present two timings from PyTorch and TensorRT (TRT) version for our method. Methods: CNSR [Granskog et al. 2020], AE [Diolatzis et al. 2022], ONND [Nvidia 2023b], OIDN [Áfra 2024]. Metrics: Relative Square Error (RSE), Learned Perceptual Image Patch Similarity (LPIPS).

| | *Chess* | | | *Gig* | | | *Emernald Square* | | | *Living Room* | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Methods** | **Time (ms) ↓** | **LPIPS ↓** | **RSE ↓** | **Time (ms) ↓** | **LPIPS ↓** | **RSE ↓** | **Time (ms) ↓** | **LPIPS ↓** | **RSE ↓** | **Time (ms) ↓** | **LPIPS ↓** | **RSE ↓** |
| CNSR | 171.37 | 0.0326 | 0.0125 | 171.60 | 0.2798 | 0.1781 | 171.28 | 0.4021 | 0.3589 | 174.19 | 0.1766 | 0.0351 |
| AE | 115.80 | 0.0274 | 0.0040 | 132.83 | 0.3295 | 0.2985 | 118.63 | 0.3010 | 0.1386 | 120.37 | 0.1823 | 0.0189 |
| **Ours** | 98.48 | 0.0222 | 0.0023 | 147.91 | 0.1338 | 0.0125 | 87.89 | 0.1600 | 0.0394 | 113.72 | 0.1820 | 0.0142 |
| ONND | 22.08 (42spp) | 0.0209 | 0.0012 | 46.55 (25spp) | 0.1998 | 2.0660 | 26.64 (13spp) | 0.4652 | 0.0668 | 28.01 (35spp) | 0.1406 | 0.0056 |
| OIDN | 22.29 (32spp) | 0.0181 | 0.0013 | 46.36 (22spp) | 0.1441 | 1.2590 | 25.96 (10spp) | 0.3491 | 0.0557 | 27.85 (28spp) | 0.1412 | 0.0052 |
| **Ours (TRT)** | 22.90 | 0.0222 | 0.0023 | 45.82 | 0.1338 | 0.0125 | 25.26 | 0.1600 | 0.0394 | 27.47 | 0.1820 | 0.0142 |

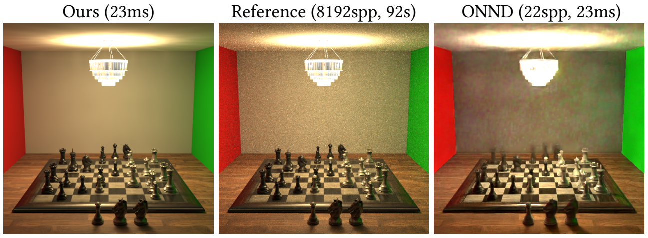

Ours (23ms)
Reference (8192spp, 92s)
ONND (22spp, 23ms)

Fig. 5. Equal-time comparison with ONND under the *Chess* scene and a complex luminary. LPIPS for Ours and ONND are 0.3751 and 0.4843, respectively. Please refer to the supplementary video for dynamic results.

animation time stamp for its best quality. When training AE, we utilize the uniform data sampling strategy to align with our method. We believe that their adaptive data sampling strategy can also benefit our method, and we leave it as future work. For ONND, we employ the latest version 8.0 with temporal [Hasselgren et al. 2020] and kernel-based extensions [Bako et al. 2017], as suggested in the prior work [Balint et al. 2023]. We utilize the latest version 2.2.2 of OIDN, which supports GPU acceleration. Table 1 presents the quantitative comparisons of our results and baselines, with methods grouped into two categories: neural rendering and real-time path tracing. Visual comparison results for all methods are shown in Figure 6.

As shown in the first part of Table 1, our method achieves overall better metrics compared to CNSR and AE in all scenes. Visual comparison in Figure 6 also validate this observation. It is not surprising that CNSR and AE fail in every scene, especially in high-frequency areas like shadows and highlights. Without shading clues, these approaches needs to embed the complex mapping between dynamic scene information and shading results in a compact network, which becomes unsolvable for these highly dynamic scenes. Aside from quality, our method is also faster than these neural rendering methods in most cases as shown in Table 1. Note that our performance scales linearly with the number of lights, which results in inferior performance in the *Gig* scene with 7 lights. We provide further discussion about performance scalability in Section 7.

For comparisons with path tracing and denoising methods, we selected different samples-per-pixel (spp) in each scene to achieve approximately equal-time comparison. As observed in the second part of Table 1, our method can outperform state-of-the-art path tracing and denoising methods in relatively complex and large scenes (i.e., *Gig* and *Emernald Square*), while performing less effectively in others. This is because tracing complex and large scenes is costly, making it difficult to capture sufficient light transport within the tight real-time budget. As clearly depicted in the third and fourth rows of Figure 6, a significant amount of shading details in shadow, texture, and geometry are missing in denoising methods. However, our method successfully preserve those high-frequency shading details. Another common side-effect of path tracing lies in its poor temporal and view consistency, resulting in considerable flicking artifacts, which is clearly demonstrated in the Supplementary Video.

We also replace the area lights in our *Chess* scene with a complex luminaire, as shown in Figure 5. There are 18 light bulbs inside the transparent shell, leading to an extremely challenging light transport [Zhu et al. 2021]. In this case, though the triangle number is not high, path tracing is still struggle to generate enough results in real-time budget. However, our methods, make this challenging scene rendering in real-time, as shown in our supplementary video.

Overall, our method and path tracing methods excel in different types of scenarios. Our method provides temporally consistent rendering results, preserving details even under large or complex scenes. However, as a mature rendering method, path tracing exhibits better generalizability which is advantageous for handling more types of materials, light transport, and completely dynamic scene, as discussed further in Section 7. Nevertheless, thanks to the light-oriented architecture that disentangles scene parameters from the network representations, our framework outperforms previous neural rendering methods, demonstrating accurate prediction of the direct shading, shadow, and indirect shading components, even in highly dynamic scenes. To validate our model with challenging scenes featuring multiple freely movable objects, we constructed a scene named *Interior Design*. This also demonstrates its modeling-while-rendering application. More discussions and results are provided in the supplementary material. Figure 7 showcases our results when manipulating the light position and altering the wall color. It is evident that the transformation of the light source leads to changes in high-frequency specular effects. Similarly, altering the wall color induces variations in the indirect shading. Please refer to the supplementary video for more dynamic results. We also provided a set

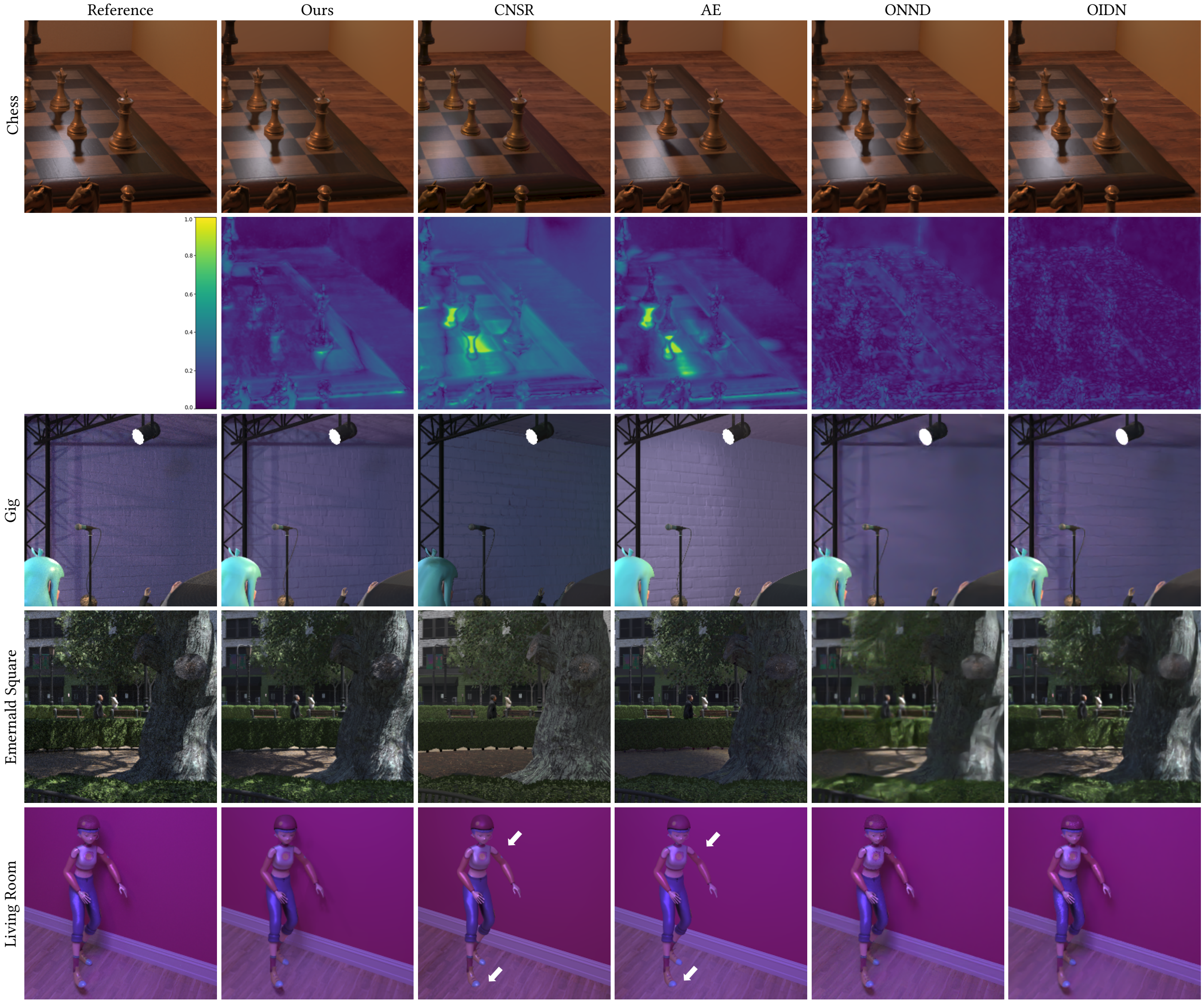


Fig. 6. Comparison results of our method and state-of-the-art neural rendering methods. Note that, white arrows highlight regions showing differences in shadow and shading effects. We show the FLIP [Andersson et al. 2021] error maps in *Chess* scene to illustrate perceptual differences. Please refer to the supplementary material for additional error maps (i.e., RSE, SSIM) in all scenes.

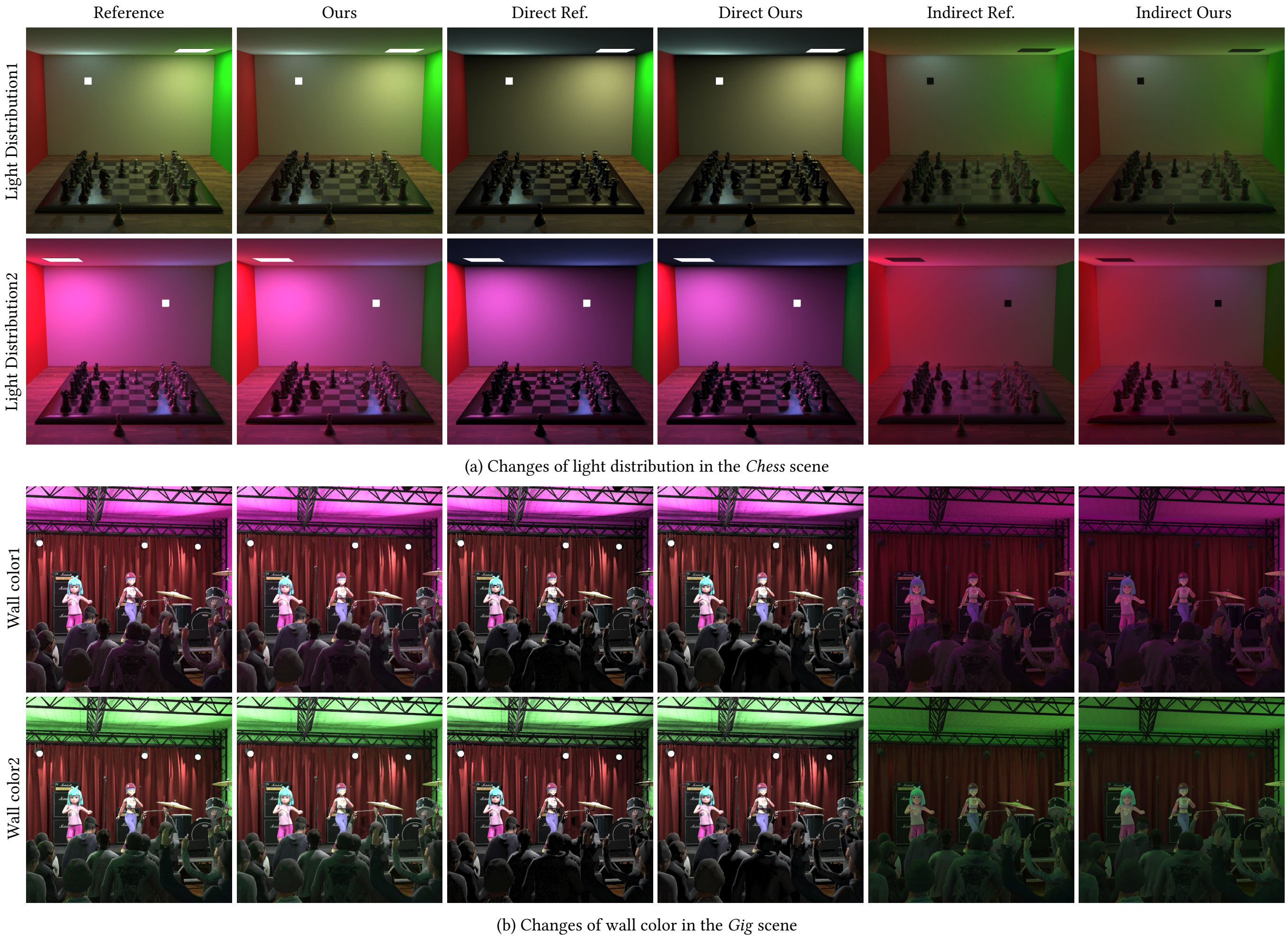
of full-resolution results in a web-based viewer as supplementary material.

### 6.3 Ablation Study

We studies the effectiveness of each component in our designed framework, as depicted in Table 2 and Figure 8.

*Pixel-light attention.* To assess the effectiveness, we compare our pixel-light attention with average pooling. For a fair comparison with the average pooling variant, we add additional layers in the decoder to maintain similar network performance. It is evident that our pixel-light attention produces superior quality compared to average pooling, especially in the cases of multiple light sources.

*Shading clues.* The specular clues in our approach are derived from two screen-space inputs: light direction and half vector. We comprehensively evaluate the impact of each, as presented in Table 2 and Figure 8. We observe that each shading clue shows a noticeable impact on the results. For instance, as shown in the $2^{nd}$ row of Figure 8, lacking half vector results in blur specular highlights, and the self-shadow on the character cannot be faithfully resolved without shadow clues.



(a) Changes of light distribution in the *Chess* scene

(b) Changes of wall color in the *Gig* scene

Fig. 7. Rendering results of editing the lighting (a) and material (b) of dynamic scenes, each is presented with two examples.

Table 2. Quantitative results for ablation studies tested on the *Gig* dataset with a resolution of 512×512.

| Variants | L1 ↓ | RSE ↓ | LPIPS ↓ |
|---|---|---|---|
| Average pooling | 0.0137 | 0.0141 | 0.1442 |
| w/o half vector | 0.0135 | 0.0126 | 0.1473 |
| w/o light direction | 0.0143 | 0.0146 | 0.1417 |
| w/o shadow clues | 0.0151 | 0.0234 | 0.1624 |
| w/o GI decomposition | 0.0135 | 0.0127 | 0.1428 |
| **Ours** | **0.0132** | **0.0125** | **0.1338** |

*GI decomposition.* In addition to predicting the direct and indirect shading, we explore the use of a unified decoder in directly producing the final radiance based on the composed light embedding. For a fair comparison, in experiments we double the size of the unified decoder and train the model with the $\mathcal{L}_1$ loss. The results (Figure 8 and Table 2) reveal that our model outperforms the unified decoder.

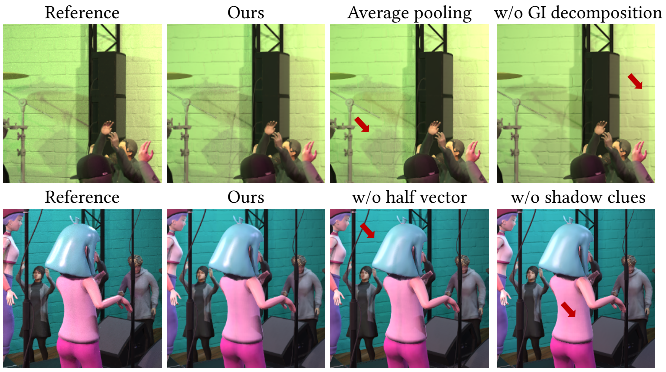


Fig. 8. Visual comparison between different variants. The specular effect of the character's hair varies, self shadow is lost without shadow clues. (Red arrows highlight differences. Please zoom in for details).

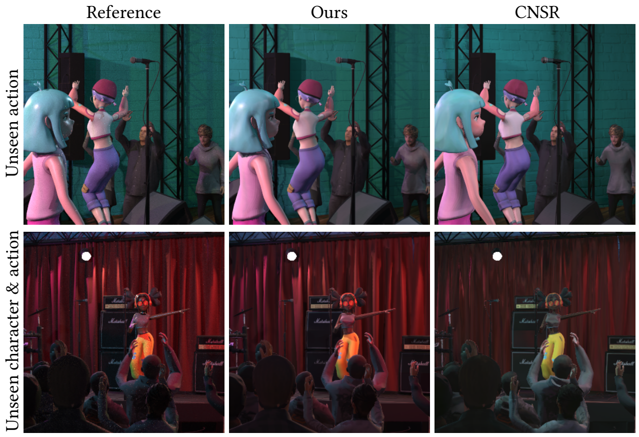


Fig. 9. Generalization study. Our method can generate reasonable and realistic results even for unseen animated actions and unseen objects.

We attribute this improvement to the distinct supervision on each shading component, along with distinct inputs for the decoders. These factors can simplify the learning process and eventually lead to superior results.

### 6.4 Generalization Analysis

Our method exhibits better generalizability than previous methods, enabling the accurate prediction for dynamic scenes. We validate this by testing our model on scenes with novel actions. Specifically, we alter the animated action of the two persons on the stage to novel actions in the *Gig* scene, as shown in the first row of Figure 9. Our method consistently generates reasonable and realistic specular and shadow effects. In contrast, CNSR yield incorrect shading results, particularly in the context of shadows and highlights.

Furthermore, we assess the generalizability of different methods for unseen objects. Introducing an unseen character with a novel action, our method still produces more realistic shading results compared to CNSR, as seen in the second row of Figure 9. We also tested challenging cases involving freely movable objects in the large-scale scene of *Emernald Square*, which were not included in the training set. We conducted tests by freely translating, rotating, and scaling the tree, and moving the bus across the street block. Thanks to its generalizability, our model still produces visually plausible results, as demonstrated in the supplementary video. Notably, our training set consists of one scene only, and we anticipate the performance in generalizing to unseen objects would be significantly improved with a large-scale scene dataset, whose preparation for such experiments can be prohibitively expensive at this moment. However, path tracing methods do not suffer from these problems. Therefore, this remains an important avenue for future research.

## 7 LIMITATIONS AND FUTURE WORK

*High-frequency indirect shading.* While our framework successfully produces reasonable global illumination results, it still faces challenges in handling high-frequency effects in indirect shading. As illustrated in Figure 10, the first row showcases a case where the indirect shadow on the grass in the *Emernald Square* scene exhibits high-frequency patterns similar to the direct shadow, instead of the desired soft indirect shadow on the floor. We attribute this to the complex visibility on the grass, and a potential workaround could involve incorporating shadow clues into the indirect decoder. Alternatively, a more general solution for handling indirect shadow could be offering additional clues about the visibility of indirect VPLs [Ritschel et al. 2008], which we plan to address as future work.

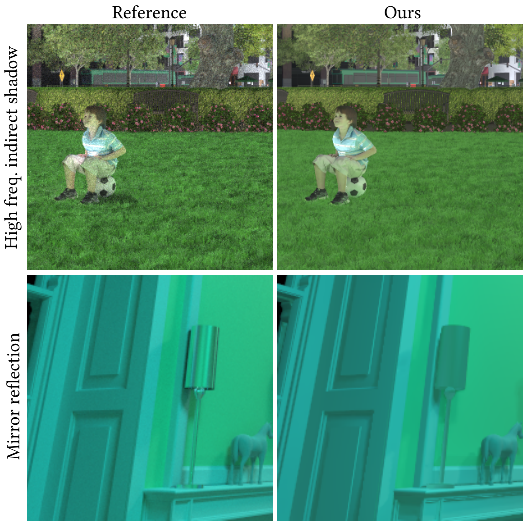


Fig. 10. Illustration of failure cases. Our method struggles to predict accurate indirect shadow and mirror reflection effects. Note that, the $1^{st}$ row shows indirect shading results, whose brightness is amplified 10 times for visualization.

Besides, the second row of Figure 10 reveals the challenge of resolving the highly glossy reflection. One effective workaround is to generate second-bounce G-buffers as shading clues for indirect lighting as mentioned by previous works [Diolatzis et al. 2022; Guo et al. 2022]. Alternative promising avenues could involve applying data importance sampling strategies from auto-encoder (AE) approaches [Diolatzis et al. 2022], and exploring an effective attention scheme for indirect VPLs, potentially helping address the challenges of rendering such high-frequency indirect shading.

*Complex light transport.* Although we provided an example of complex luminaire in Figure 5, many cases of complex light transport remain unexplored, including sub-surface scattering, caustics, and volumetric rendering. Theoretically, our framework has the potential to handling such complex effects, as evidenced by results from previous neural rendering works [Diolatzis et al. 2022; Gao et al. 2022]. However, additional efforts are needed, such as introducing new shading clues to handle complex light transport in highly dynamic scenes, and we consider this as future work.

*Performance Scalability.* LightFormer provides a flexible neural rendering framework supporting rendering highly dynamic large-scale scenes in real-time. However, similar to the object-oriented schema [Zheng et al. 2023] that scales the rendering cost by the number of objects, the computing cost of LightFormer also scales by the number of lights. Specifically, computational cost scales linearly both with the number of pixels and the number of lights in the lighting encoding stage. Unlike decoding one by one and then performing the composition, our framework composes all light information in the latent space, which decreases the cost of multiple decoding process. Moreover, performance of decoding process is only related to the number of pixels except for the number of lights. State-of-the-art techniques, such as light culling [Dachsbacher and Stamminger 2006; Harada et al. 2012] , VPL clustering [Prutkin et al. 2012; Simon et al. 2015] and advanced shadowmap [Olsson et al. 2015], could be applied for handling a large number of lights efficiently. Additionally, we believe combining the concept of light culling with our pixel-light attention to further bootstrap the rendering performance is a valuable direction.

## 8 CONCLUSION

We have proposed a flexible and generalizable neural rendering framework, dubbed *LightFormer*, for high-quality, real-time global illumination of fully dynamic scenes. Specifically, we generate the light observation of each light using the neural reflective shadow map. The light source parameters and important shading clues for specular, shadow, and indirect lighting are encoded into a light embedding for each light. Leveraging the *pixel-light attention*, we compose all light embeddings for each shading point and decode the final radiance. Our implementation results demonstrate superior rendering quality over state-of-the-art neural rendering techniques. Compared with path tracing and denoising methods, our method can provide more temporal stable results, achieving superior quality in large scenes that are costly to trace.

However, *LightFormer* is still not without its challenges, as discussed in Section 7. Although we have improved generalization and support for dynamic scenes, current mature rendering methods based on rasterization or path tracing are definitely more generalizable. Some aspects of complex light transport are not extensively explored yet, making it challenging for production rendering. Nevertheless, further exploration of the potential of neural rendering, particularly within the context of current surge in AI development, presents a challenging yet exciting research direction. We hope that *LightFormer* could play an important role in bridging the worlds of real-time rendering and machine learning, inspiring more exciting research in the future.

## ACKNOWLEDGMENTS

We thank all reviewers for their insightful comments. We also thank Chuankun Zheng, Yuzhi Liang for helpful discussions, and He Zhu for preparing 3D scenes. This work was partially supported by National Key R&D Program of China (No. 2024YDLN0011, No. 2023YFF0905102), NSFC (No. 62441205), Key R&D Program of Zhejiang Province (No. 2023C01039).

## REFERENCES

Attila T. Áfra. 2024. Intel® Open Image Denoise. https://www.openimagedenoise.org.

Pontus Andersson, Jim Nilsson, Peter Shirley, and Tomas Akenine-Möller. 2021. Visualizing errors in rendered high dynamic range images. (2021).

Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony Derose, and Fabrice Rousselle. 2017. Kernel-predicting convolutional networks for denoising Monte Carlo renderings. *ACM Trans. Graph.* 36, 4 (2017), 97–1.

Martin Balint, Krzysztof Wolski, Karol Myszkowski, Hans-Peter Seidel, and Rafał Mantiuk. 2023. Neural Partitioning Pyramids for Denoising Monte Carlo Renderings. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.

Aner Ben-Artzi, Kevin Egan, Frédo Durand, and Ravi Ramamoorthi. 2008. A precomputed polynomial representation for interactive BRDF editing with global illumination. *ACM Transactions on Graphics (TOG)* 27, 2 (2008), 1–13.

Aner Ben-Artzi, Ryan Overbeck, and Ravi Ramamoorthi. 2006. Real-time BRDF editing in complex lighting. *ACM Transactions on Graphics (TOG)* 25, 3 (2006), 945–954.

James F Blinn. 1977. Models of light reflection for computer synthesized pictures. In *Proceedings of the 4th annual conference on Computer graphics and interactive techniques*. 192–198.

Carsten Dachsbacher, Jaroslav Křivánek, Miloš Hašan, Adam Arbree, Bruce Walter, and Jan Novák. 2014. Scalable realistic rendering with many-light methods. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 88–104.

Carsten Dachsbacher and Marc Stamminger. 2005. Reflective shadow maps. In *Proceedings of the 2005 symposium on Interactive 3D graphics and games*. 203–231.

Carsten Dachsbacher and Marc Stamminger. 2006. Splatting indirect illumination. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games*. 93–100.

Sayantan Datta, Derek Nowrouzezahrai, Christoph Schied, and Zhao Dong. 2022. Neural Shadow Mapping. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.

Stavros Diolatzis, Julien Philip, and George Drettakis. 2022. Active exploration for neural global illumination of variable scenes. *ACM Transactions on Graphics (TOG)* 41, 5 (2022), 1–18.

Honghao Dong, Guoping Wang, and Sheng Li. 2023. Neural Parametric Mixtures for Path Guiding. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–10.

SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. 2018. Neural scene representation and rendering. *Science* 360, 6394 (2018), 1204–1210.

Hangming Fan, Rui Wang, Yuchi Huo, and Hujun Bao. 2021. Real-time Monte Carlo Denoising with Weight Sharing Kernel Prediction Network. In *Computer Graphics Forum*, Vol. 40. Wiley Online Library, 15–27.

Duan Gao, Haoyuan Mu, and Kun Xu. 2022. Neural Global Illumination: Interactive Indirect Illumination Prediction under Dynamic Area Lights. *IEEE Transactions on Visualization and Computer Graphics* (2022).

Jonathan Granskog, Fabrice Rousselle, Marios Papas, and Jan Novák. 2020. Compositional neural scene representations for shading inference. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 135–1.

Jonathan Granskog, Till N Schnabel, Fabrice Rousselle, and Jan Novák. 2021. Neural scene graph rendering. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–11.

Gene Greger, Peter Shirley, Philip M Hubbard, and Donald P Greenberg. 1998. The irradiance volume. *IEEE Computer Graphics and Applications* 18, 2 (1998), 32–43.

Jie Guo, Zijing Zong, Yadong Song, Xihao Fu, Chengzhi Tao, Yanwen Guo, and Ling-Qi Yan. 2022. Efficient Light Probes for Real-Time Global Illumination. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–14.

Saeed Hadadan, Shuhong Chen, and Matthias Zwicker. 2021. Neural radiosity. *ACM Transactions on Graphics (TOG)* 40, 6 (2021), 1–11.

Takahiro Harada, Jay McKee, and Jason C Yang. 2012. Forward+: Bringing deferred lighting to the next level.. In *Eurographics (Short Papers)*. 5–8.

Miloš Hašan, Fabio Pellacini, and Kavita Bala. 2007. Matrix row-column sampling for the many-light problem. In *ACM SIGGRAPH 2007 papers*. 26–es.

Jon Hasselgren, Jacob Munkberg, Marco Salvi, Anjul Patney, and Aaron Lefohn. 2020. Neural temporal adaptive sampling and denoising. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 147–155.

Yuchi Huo, Rui Wang, Ruzahng Zheng, Hualin Xu, Hujun Bao, and Sung-Eui Yoon. 2020. Adaptive incident radiance field sampling and reconstruction using deep reinforcement learning. *ACM Transactions on Graphics (TOG)* 39, 1 (2020), 1–17.

Yuchi Huo and Sung-eui Yoon. 2021. A survey on deep learning-based Monte Carlo denoising. *Computational visual media* 7 (2021), 169–185.

Simon Kallweit, Petrik Clarberg, Craig Kolb, Tom'aš Davidovič, Kai-Hwa Yao, Theresa Foley, Yong He, Lifan Wu, Lucy Chen, Tomas Akenine-Möller, Chris Wyman, Cyril Crassin, and Nir Benty. 2022. The Falcor Rendering Framework. https://github.com/NVIDIAGameWorks/Falcor

Anders Wang Kristensen, Tomas Akenine-Möller, and Henrik Wann Jensen. 2005. Precomputed local radiance transfer for real-time lighting design. In *ACM SIGGRAPH 2005 Papers*. 1208–1215.

Alexandr Kuznetsov. 2021. NeuMIP: Multi-resolution neural materials. *ACM Transactions on Graphics (TOG)* 40, 4 (2021).

Morgan McGuire and Michael Mara. 2014. Efficient GPU screen-space ray tracing. *Journal of Computer Graphics Techniques (JCGT)* 3, 4 (2014), 73–85.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)* 41, 4 (2022), 1–15.

Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. 2021. Real-time neural radiance caching for path tracing. *arXiv preprint arXiv:2106.12372* (2021).

Oliver Nalbach, Elena Arabadzhiyska, Dushyant Mehta, H-P Seidel, and Tobias Ritschel. 2017. Deep shading: convolutional neural networks for screen space shading. In *Computer graphics forum*, Vol. 36. Wiley Online Library, 65–78.

Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. 2004. Triple product wavelet integrals for all-frequency relighting. In *ACM SIGGRAPH 2004 Papers*. 477–487.

Nvidia. 2023a. NVIDIA Real-Time Denoisers. https://developer.nvidia.com/rtx/ray-tracing/rt-denoisers

Nvidia. 2023b. OptiX AI-Accelerated Denoiser. https://developer.nvidia.com/optix-denoiser

Ola Olsson, Markus Billeter, Erik Sintorn, Viktor Kämpe, and Ulf Assarsson. 2015. More efficient virtual shadow maps for many lights. *IEEE transactions on visualization and computer graphics* 21, 6 (2015), 701–713.

Roman Prutkin, Anton Kaplanyan, Carsten Dachsbacher, et al. 2012. Reflective Shadow Map Clustering for Real-Time Global Illumination.. In *Eurographics (Short Papers)*. 9–12.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

Nithin Raghavan, Yan Xiao, Kai-En Lin, Tiancheng Sun, Sai Bi, Zexiang Xu, Tzu-Mao Li, and Ravi Ramamoorthi. 2023. Neural Free-Viewpoint Relighting for Glossy Indirect Illumination. In *Computer Graphics Forum*, Vol. 42. Wiley Online Library, e14885.

Gilles Rainer, Adrien Bousseau, Tobias Ritschel, and George Drettakis. 2022. Neural precomputed radiance transfer. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 365–378.

Ravi Ramamoorthi et al. 2009. Precomputation-based rendering. *Foundations and Trends® in Computer Graphics and Vision* 3, 4 (2009), 281–369.

Peiran Ren, Jiaping Wang, Minmin Gong, Stephen Lin, Xin Tong, and Baining Guo. 2013. Global illumination with radiance regression functions. *ACM Trans. Graph.* 32, 4 (2013), 130–1.

Tobias Ritschel, Thorsten Grosch, Min H Kim, H-P Seidel, Carsten Dachsbacher, and Jan Kautz. 2008. Imperfect shadow maps for efficient computation of indirect illumination. *ACM transactions on graphics (tog)* 27, 5 (2008), 1–8.

Tobias Ritschel, Thorsten Grosch, and Hans-Peter Seidel. 2009. Approximating dynamic global illumination in image space. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*. 75–82.

Austin Robison and Peter Shirley. 2009. Image space gathering. In *Proceedings of the Conference on High Performance Graphics 2009*. 91–98.

Simon Rodriguez, Thomas Leimkühler, Siddhant Prakash, Chris Wyman, Peter Shirley, and George Drettakis. 2020. Glossy probe reprojection for interactive global illumination. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–16.

Perumaal Shanmugam and Okan Arikan. 2007. Hardware accelerated ambient occlusion techniques on GPUs. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*. 73–80.

Florian Simon, Johannes Hanika, and Carsten Dachsbacher. 2015. Rich-VPLs for improving the versatility of many-light methods. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 575–584.

Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. 2019. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems* 32 (2019).

Peter-Pike Sloan, Jan Kautz, and John Snyder. 2002. 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*. 527–536.

Peter-Pike Sloan, Ben Luna, and John Snyder. 2005. Local, deformable precomputed radiance transfer. *ACM Transactions on Graphics (TOG)* 24, 3 (2005), 1216–1224.

Xin Sun, Kun Zhou, Yanyun Chen, Stephen Lin, Jiaoying Shi, and Baining Guo. 2007. Interactive relighting with dynamic BRDFs. In *ACM SIGGRAPH 2007 papers*. 27–es.

Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Wang Yifan, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. 2022. Advances in neural rendering. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 703–735.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

Delio Vicini, Vladlen Koltun, and Wenzel Jakob. 2019. A learned shape-adaptive subsurface scattering model. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–15.

Bruce Walter, Sebastian Fernandez, Adam Arbree, Kavita Bala, Michael Donikian, and Donald P Greenberg. 2005. Lightcuts: a scalable approach to illumination. In *ACM SIGGRAPH 2005 Papers*. 1098–1107.

Jiaping Wang, Peiran Ren, Minmin Gong, John Snyder, and Baining Guo. 2009. All-frequency rendering of dynamic, spatially-varying reflectance. In *ACM SIGGRAPH Asia 2009 papers*. 1–10.

Lifan Wu, Guangyan Cai, Shuang Zhao, and Ravi Ramamoorthi. 2020. Analytic spherical harmonic gradients for real-time rendering with many polygonal area lights. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 134–1.

Hanggao Xin, Shaokun Zheng, Kun Xu, and Ling-Qi Yan. 2020. Lightweight bilateral convolutional neural networks for interactive single-bounce diffuse indirect illumination. *IEEE Transactions on Visualization and Computer Graphics* 28, 4 (2020), 1824–1834.

Bing Xu, Liwen Wu, Milos Hasan, Fujun Luan, Iliyan Georgiev, Zexiang Xu, and Ravi Ramamoorthi. 2023. NeuSample: Importance Sampling for Neural Materials. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–10.

Bing Xu, Junfei Zhang, Rui Wang, Kun Xu, Yong-Liang Yang, Chuan Li, and Rui Tang. 2019. Adversarial Monte Carlo denoising with conditioned auxiliary feature modulation. *ACM Trans. Graph.* 38, 6 (2019), 224–1.

Zilin Xu, Zheng Zeng, Lifan Wu, Lu Wang, and Ling-Qi Yan. 2022. Lightweight Neural Basis Functions for All-Frequency Shading. In *SIGGRAPH Asia 2022 Conference Papers*. 1–9.

Jiaqi Yu, Yongwei Nie, Chengjiang Long, Wenjun Xu, Qing Zhang, and Guiqing Li. 2021. Monte Carlo denoising via auxiliary feature guided self-attention. *ACM Trans. Graph.* 40, 6 (2021), 273–1.

Chong Zeng, Guojun Chen, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. 2023. Relighting Neural Radiance Fields with Shadow and Highlight Hints. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.

Fan Zhang, Hanqiu Sun, Leilei Xu, and Lee Kit Lun. 2006. Parallel-split shadow maps for large-scale virtual environments. In *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*. 311–318.

Chuankun Zheng, Yuchi Huo, Shaohua Mo, Zhihua Zhong, Zhizhen Wu, Wei Hua, Rui Wang, and Hujun Bao. 2023. NeLT: Object-oriented Neural Light Transfer. *ACM Transactions on Graphics* (2023).

Kun Zhou, Yaohua Hu, Stephen Lin, Baining Guo, and Heung-Yeung Shum. 2005. Precomputed shadow fields for dynamic scenes. In *ACM SIGGRAPH 2005 Papers*. 1196–1201.

Junqiu Zhu, Yaoyi Bai, Zilin Xu, Steve Bako, Edgar Velázquez-Armendáriz, Lu Wang, Pradeep Sen, Milos Hasan, and Ling-Qi Yan. 2021. Neural complex luminaires: representation and rendering. *ACM Trans. Graph.* 40, 4 (2021), 57–1.