# Efficient Light Probes for Real-time Global Illumination

JIE GUO, ZIJING ZONG, YADONG SONG, XIHAO FU, CHENGZHI TAO, and YANWEN GUO*, State Key Lab for Novel Software Technology, Nanjing University, China
LING-QI YAN, University of California, Santa Barbara, United States of America


Fig. 1. A new rendering pipeline based on precomputed light probes and deep neural networks is proposed that can reconstruct a 1080P (1920×1080) image with complex global illumination in real time (>30 frames per second). The generated images closely match the path traced ground truth and achieve state-of-the-art quality as compared with those generated by previous work, e.g., the glossy probe reprojection (GPR) method [Rodriguez et al. 2020]. Thanks to a neural image reconstruction method, the light probes can be generated with a low resolution and a low sampling rate, saving the precomputation time.

Reproducing physically-based global illumination (GI) effects has been a long-standing demand for many real-time graphical applications. In pursuit of this goal, many recent engines resort to some form of light probes baked in a precomputation stage. Unfortunately, the GI effects stemming from the precomputed probes are rather limited due to the constraints in the probe storage, representation or query. In this paper, we propose a new method for probe-based GI rendering which can generate a wide range of GI effects, including glossy reflection with multiple bounces, in complex scenes. The key contributions behind our work include a gradient-based search algorithm and a neural image reconstruction method. The search algorithm is designed to reproject the probes' contents to any query viewpoint, without introducing parallax errors, and converges fast to the optimal solution. The neural image reconstruction method, based on a dedicated neural network and several G-buffers, tries to recover high-quality images from low-quality inputs due to limited resolution or (potential) low sampling rate of the probes. This neural method makes the generation of light probes efficient. Moreover, a temporal reprojection strategy and a temporal loss are employed to improve temporal stability for animation sequences. The whole pipeline runs in real-time (>30 frames per second) even for high-resolution (1920×1080) outputs, thanks to the fast convergence rate of the gradient-based search algorithm and a light-weight design of the neural network. Extensive experiments on multiple complex scenes have been conducted to show the superiority of our method over the state-of-the-arts.

CCS Concepts: • **Computing methodologies** → **Ray tracing**; *Neural networks.*

Additional Key Words and Phrases: Light probes, Global illumination, Neural networks, Real-time rendering

**ACM Reference Format:**
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 Trans. Graph.* 41, 4, Article 202 (July 2022), 14 pages. https://doi.org/10.1145/3550454.3555452

## 1 INTRODUCTION

Accurately computing physically-based global illumination (GI) is a longstanding problem in computer graphics, and is particularly challenging for real-time scenarios. To accelerate runtime global illumination, many recent real-time rendering engines favor some form of light probes which encode lighting information of a 3D scene into a sparse set of compact caches placed statically in the scene [Hooker 2016; Majercik et al. 2019, 2021; McGuire et al. 2017]. At runtime, the light that hits an object is approximately reconstructed by the precomputed or baked values from a small number of nearest probes to that object. To offer more physical-plausible shading, these light probes can be further coupled with visibility [Majercik et al. 2019; McGuire et al. 2017] or precomputed radiance transfer [Silvennoinen and Lehtinen 2017; Sloan et al. 2002].

*Corresponding author

Authors' addresses: Jie Guo, guojie@nju.edu.cn; Zijing Zong, jimzongzijing@163.com; Yadong Song, MG21330049@smail.nju.edu.cn; Xihao Fu, fuxihao66@gmail.com; Chengzhi Tao, tcz_tao@yeah.net; Yanwen Guo, ywguo@nju.edu.cn, State Key Lab for Novel Software Technology, Nanjing University, China; Ling-Qi Yan, University of California, Santa Barbara, United States of America, lingqi@cs.ucsb.edu.


0730-0301/2022/7-ART202 $15.00
https://doi.org/10.1145/3550454.3555452

Among existing probe-based methods, many rely on simplifying assumptions about path types, such as pure diffuse paths handled by irradiance probes [Majercik et al. 2019] or one-bounce specular paths approximated by environment maps [Blinn and Newell 1976]. Generally, long glossy paths–paths with multiple glossy bounces–are usually excluded from these methods due to the strong correlation between the scattering distribution and the view direction. To efficiently handle all light paths in static scenes, glossy probe reprojection (GPR) [Rodriguez et al. 2020] is recently introduced, enabling interactive rendering of complex glossy scenes. However, this method has a very large computational burden in the precomputation step since high-quality probes with a high sampling rate are required. The long baking time makes the lighting workflow inefficient for artists and difficult for engineers to debug. Moreover, the heuristic gathering and filtering processes in this method may incur artifacts, such as inaccurate glossiness of highlights and specular geometric aliasing along object edges.

With the prevalence of deep learning techniques, neural rendering [Tewari et al. 2020] has become a widespread concern for many graphical applications. In this paper, we investigate the use of neural rendering to handle light probes, as a means to lower the computational cost in the precomputation step, while still maintaining photorealism even for surfaces of varying roughness. Leveraging specialized performance optimization (e.g., TensorRT) for deep learning, the proposed method is expected to achieve real-time performance for GI rendering.

Our method is built upon precomputed light probes and resorts to a neural network for final image reconstruction. The key to our method is the replacement of heuristic and analytical probe interpolation in previous methods by a more robust and flexible learning-based model. In this method, a *gradient-based reflection search* algorithm first reprojects parallax-corrected glossy reflection from multiple probes to the current view. This process ensures that reflection effects on the glossy surfaces are free of parallax errors but may produce noise and incomplete structures due to limited information stored in the probes. To remove the noise and restore the important structures, we design and train a convolutional neural network under the guidance of some G-buffers (including normal, depth and albedo). To better fuse features from different modalities in this network, a *G-buffer Modulation* module is specially designed which improves the quality of reconstructed images.

The proposed learning-based method allows us to precompute light probes (and lightmaps for diffuse reflection) with a very low sampling rate while still reproducing high-quality global illumination with complex light paths (including long glossy paths) from any novel view. This significantly reduces the precomputation time that is required for the generation of light probes and lightmaps and is beneficial for the fast response to scene editing and updating. As another notable advantage, the low-level image statistics priors [Nalbach et al. 2017; Ulyanov et al. 2018] learned by our deep convolutional network, including geometrical structures and texture details, are also beneficial for reducing the specular geometric aliasing along object edges [Rodriguez et al. 2020].

In summary, our paper makes the following contributions:

- We introduce an efficient rendering pipeline for reproducing high-quality global illumination with complex glossy light paths from low-quality light probes.
- We propose a gradient-based reflection search algorithm that explicitly takes the parallax changes between the probe and the novel view into consideration during probe query.
- We design a neural network, with a dedicated G-buffer modulation module, that can better recover high-quality images from low-quality inputs due to imperfect lightmaps and light probes.
- We illustrate the effectiveness and robustness of our method on multiple scenes with complex glossy paths, showing real-time performance when rendering at 1080P resolution on latest GPUs.

## 2 RELATED WORK

In this section, we review existing techniques that are closely related to our work.

### 2.1 Probe-based global illumination

Light probes were first introduced into rendering by Greger et al. [1998] in their seminal irradiance volumes work. Since then, it has become a standard method for approximating global illumination in real time. Probes can be represented in various forms including cube maps [Hooker 2016], octahedron maps [Cigolle et al. 2014], Spherical Gaussians [Currius et al. 2020] or Spherical Harmonics (SH) expansions [Sloan et al. 2002; Tatarchuk 2005]. Considering the tradeoff between accuracy and performance, different information can be stored in a probe. Irradiance probes encode lighting information as SH basis functions to achieve high efficiency and allow dynamic contents update. However, only diffuse reflection can be well captured by irradiance probes. To support the computation of indirect glossy reflection, Křivánek et al. [2005] proposed to cache and interpolate directional incoming radiance instead of irradiance. Such a radiance caching scheme has been successfully applied in both offline [Dubouchet et al. 2017; Jarosz et al. 2008; Jiang and Kainz 2021; Marco et al. 2018; Zhao et al. 2019] and real-time rendering [Müller et al. 2021; Vardis et al. 2014]. Light field probes [McGuire et al. 2017] further extend radiance/irradiance probe structures with additional geometric information to eliminate light and dark leaks. Silvennoinen and Lehtinen [2017] suggested to factorize the transport matrix into global and local transport matrices and sample sparse radiance probes to reconstruct indirect lighting. Majercik et al. proposed Dynamic Diffuse Global Illumination (DDGI) [Majercik et al. 2019] which combines sparse irradiance probes with visibility-aware interpolation to rapidly approximate indirect illumination. Hu et al. [2021] extended DDGI by employing signed distance fields as a representation of the scene, enhancing the details of reconstructed global illumination. By using angularly-filtered radiance, DDGI can also be extended to approximate glossy reflection [Majercik et al. 2021] from a limited set of path types. To handle all types of glossy paths, glossy probe projection [Rodriguez et al. 2020] is proposed at the cost of heavy precomputation and specular geometric aliasing. Our deep learning-based method addresses these problems and allows real-time novel view reconstruction.

## 2.2 Real-time specular/glossy Illumination

Due to the high-frequency range and the view-dependent nature, specular illumination has long been unfavored by precomputation-based techniques [Ramamoorthi 2009], especially for scenes with complex interreflection. In the context of real-time rendering, much effort has been devoted to accurately reproduce specular or highly glossy illumination. Many methods rely heavily on dedicated graphics hardwares [Szirmay-Kalos et al. 2009]. For perfect reflection, environment mapping [Blinn and Newell 1976] has been widely adopted to model reflected distant lighting from an infinite environment map. To recreate parallax effects that are missing in the original environment mapping technique, Szirmay-Kalos et al. [2005] added depth information to the environment map to select the proper pixel inside the environment map. Chen and Arvo [2000a; 2000b] proposed specular path perturbation methods to find reflection points on implicit curved reflectors with a second-order Taylor expansion. Explicit triangle meshes are also used to compute the accurate reflection positions in the scene [Ofek and Rappoport 1998; Roger and Holzschuch 2006]. To simulate glossy reflection, prefiltering strategies can be applied on the environment map [Kautz et al. 2000]. Xu et al. [2014] proposed to analytically compute one-bounce glossy interreflection using Spherical Gaussians, achieving near interactive performance.

The recent advent of hardware-accelerated ray tracing in modern GPUs [Burgess 2020; Harada 2020; Sandy et al. 2018] has spurred the development of real-time ray tracing (RTRT) that naturally supports all types of glossy paths. Currently, many RTRT methods rely on rendering a very noisy image using a very low sampling rate (e.g., 1 sample per pixel) and denoising the results with a highly efficient denoiser [Chaitanya et al. 2017; Fan et al. 2021; Koskela et al. 2019; Schied et al. 2017; Zhuang et al. 2021]. To maximize the quality of path traced images before denoising, Bitterli et al. [2020] proposed ReSTIR which combines importance resampling and classic weighted reservoir sampling to allow the reuse of a large number of samples in constant time. This method focuses on direct illumination from a large number of light sources, but can be further extended to indirect illumination via resampling multi-bounce indirect lighting paths [Ouyang et al. 2021]. Temporal coherence between adjacent frames is quite often considered to further reduce the samples taken at each individual frame [Scherzer et al. 2012; Yang et al. 2020].

## 2.3 Novel view synthesis

Our work is also closely related to novel view synthesis which aims to warp image content from captured views to a new view. Early methods typically employ optimization-based multi-view stereo methods to reconstruct scene geometry (or geometric proxy) and warp observations into the coordinate frame of the novel view [Buehler et al. 2001; Chaurasia et al. 2013; Debevec et al. 1998, 1996; Sinha et al. 2009; Wood et al. 2000]. More recent solutions have trained deep neural networks end-to-end for view synthesis, which has emerged as a new paradigm in this field. Usually, a learned representation of the scene, such as multiplane image (MPI) [Flynn et al. 2019; Mildenhall et al. 2019; Xu et al. 2019; Zhou et al. 2018a], 3D volume [Lombardi et al. 2019], neural light transport [Zhang et al. 2021a] or neural radiance field (NeRF) [Martin-Brualla et al. 2021; Mildenhall et al. 2020; Zhang et al. 2021b], is reconstructed from the observations.

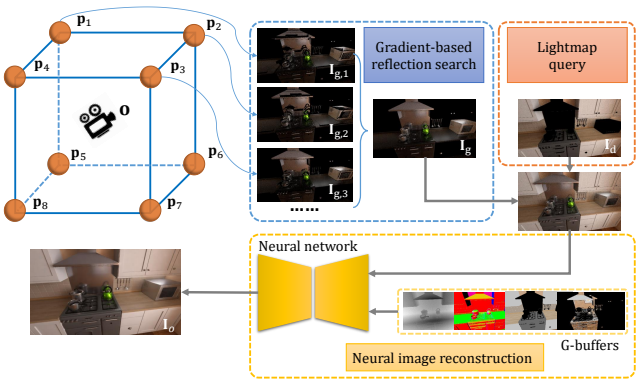



Fig. 2. High-level overview of our pipeline. To generate the diffuse reflection map $\mathbf{I}_d$, we query a lightmap precomputed with a low sampling rate. For glossy reflection, we rely on precomputed light probes and perform gradient-based reflection search to obtain a parallax-corrected glossy reflection image $\mathbf{I}_g$. The final image $\mathbf{I}_o$ is reconstructed by a neural network from $\mathbf{I}_d$ and $\mathbf{I}_g$, with the guidance of several G-buffers.

## 2.4 Neural rendering

Besides novel view synthesis, neural networks have also been successfully applied in other complex and time-consuming aspects of rendering, such as denoising [Bako et al. 2017; Chaitanya et al. 2017; Fan et al. 2021; Vogels et al. 2018], material acquisition [Deschaintre et al. 2018; Dong 2019; Gao et al. 2019; Guo et al. 2021b, 2020], texture synthesis [Zhou et al. 2018b], relighting [Chen et al. 2020; Philip et al. 2019; Xu et al. 2018], spatial/temporal super-resolution [Guo et al. 2021a; Xiao et al. 2020] and deferred rendering [Gao et al. 2020; Thies et al. 2019]. This forms a new and rapidly emerging area named neural rendering which has witnessed a large body of work been published recently in graphics community. Many neural rendering approaches can be broken up into two components: a neural scene representation and a neural (potentially differentiable) renderer. Unlike hand-crafted scene representations used in traditional rendering, neural scene representations [Eslami et al. 2018; Granskog et al. 2020; Kulkarni et al. 2015; Sitzmann et al. 2019] are more flexible and can be tailored to the task at hand. The rendering process can also be replaced by neural networks [Hadadan et al. 2021; Nalbach et al. 2017; Ren et al. 2013], achieving high performance while still preserving physical plausibility. For a complete review of neural rendering, please refer to a recent survey presented by Tewari et al. [2020]. In our work, we design, to our knowledge, the first neural network for final image inference for light probes, showing superiority as compared with traditional image synthesis processes in existing probe-based real-time rendering methods.

# 3 PROBLEM FORMULATION AND OVERVIEW

We aim to solve the rendering equation [Kajiya 1986] in real time, so as to capture every physically-based shading effect for a given scene

with more than 30 frames per second. To achieve this goal, we resort to precomputed lightmaps (for diffuse reflection) and light probes (for glossy reflection). The usage of lightmaps follows the general solution in many previous rendering systems [Rodriguez et al. 2020; Seyb et al. 2020]. Notably, the learning-based image reconstruction method in our pipeline allows us to generate the lightmap with a very low sampling rate (e.g., 256 spp), thus largely reducing the precomputation time.

To reproduce glossy reflection stemming from long glossy paths in real time, we first precompute a sparse set of light probes and organize them on a regular 3D grid in the scene. Similar to the lightmap, these light probes can also be generated with a low sampling rate. For a light probe $\mathbf{M}$ located at a grid position $\mathbf{p}$, we store the accurate radiance from ray queries for all glossy paths originating at $\mathbf{p}$. This information is then reused when shading glossy points at novel view near $\mathbf{p}$. We also store the material ID of the shading point seen directly from the probe's origin and the world position of the reflected geometry seen with once-bounce reflection if every surface is assumed to be mirror-reflected.

At runtime, given a query viewpoint $\mathbf{o}$ we first identify $K$ nearest light probes $\mathbf{M}_k$ $(k = 1, 2, ..., K)$ surrounding $\mathbf{o}$, as shown in Fig. 2. Then, we reproject each light probe to $\mathbf{o}$ via a gradient-based reflection search algorithm (denoted as $\mathcal{P}$):

$$\mathbf{I}_{\mathrm{g},k} = \mathcal{P}\{\mathbf{o}, \mathbf{p}_k, \mathbf{M}_k\} \tag{1}$$

where $\mathbf{p}_k$ is the world position of the $k$-th probe. After that, we obtain $K$ 2D images $\mathbf{I}_{\mathrm{g},k}$ $(k = 1, 2, ..., K)$ storing the glossy reflection seen from the query viewpoint $\mathbf{o}$. The reflection search is performed in a parallax-aware manner, which means parallax changes between the probe and the camera's viewpoint are corrected by $\mathcal{P}$. This plays an important role in handling glossy paths since large parallax errors usually exist, producing unnatural images.

When multiple probes are used ($K > 1$), an interpolation process is typically required to reconstruct the final image from $K$ reprojected images. Existing methods usually rely on analytical blending strategies, which are simple and efficient, but are easily plagued with annoying artifacts such as light/darkness leaks [Hooker 2016], undersampling [Wang et al. 2019], or geometric aliasing [Rodriguez et al. 2020]. In this paper, we propose a learning-based method to recover the final image $\mathbf{I}_o$ at $\mathbf{o}$ with high quality. Mathematically, we formulate this step as

$$\mathbf{I}_o = \Phi_\zeta \left( \sum_{k=1}^{K} W_k \mathbf{I}_{\mathrm{g},k} + \mathbf{I}_\mathrm{d}, \mathbf{D}_o, \mathbf{N}_o, \mathbf{A}_o, \mathbf{B}_o \right) \tag{2}$$

where $\Phi_\zeta$ denotes the network parameterized by $\zeta$. The first item feeding to $\Phi_\zeta$ is the weighted average of $K$ glossy reflected images plus a diffuse reflected image $\mathbf{I}_\mathrm{d}$, while the other items include a depth map $\mathbf{D}_o$, a normal map $\mathbf{N}_o$, an albedo map $\mathbf{A}_o$ and a reflected albedo map $\mathbf{B}_o$. These G-buffers (e.g., $\mathbf{D}_o$, $\mathbf{N}_o$ and $\mathbf{A}_o$) have also been adopted to some other neural rendering methods, such as deferred rendering [Gao et al. 2020; Thies et al. 2019] and Monte Carlo denoising [Bako et al. 2017; Chaitanya et al. 2017; Fan et al. 2021; Vogels et al. 2018].

We train the network $\Phi_\zeta$ in a supervised manner. To this end, we construct a large-scale dataset containing sufficient training examples generated from 3D scenes. Each training example is comprised of $\{\sum_{k=1}^{K} W_k \mathbf{I}_{\mathrm{g},k}, \mathbf{I}_\mathrm{d}, \mathbf{D}_o, \mathbf{N}_o, \mathbf{A}_o, \mathbf{B}_o\}$ and the ground truth $\hat{\mathbf{I}}$. After trained on this dataset with a properly chosen loss function $\mathcal{L}$, the network $\Phi_\zeta$ is expected to output a high-quality image that is close to the ground truth at any novel viewpoint. Since specially-designed neural networks are robust to aliasing and noise, our method is able to eliminate most artifacts mentioned above. The low-level image statistics priors learned by the network even allow us to generate light probes with a low sampling rate, which is beneficial for reducing the precomputation overhead.

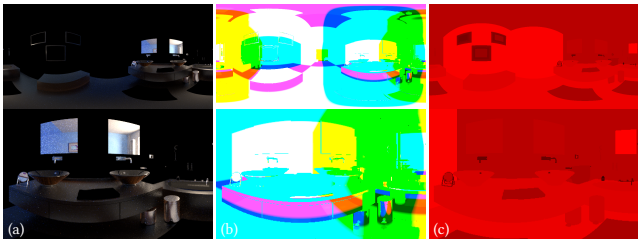



Fig. 3. Data stored at each probe's position: (a) radiance of glossy reflection generated with 128 spp, (b) reflected position and (c) material ID. The first row shows the panoramas and the second row shows the closeups.

## 4 THE PROPOSED RENDERING PIPELINE

In this section, we expose the details of each step in our pipeline.

### 4.1 Lightmap with a low sampling rate

We currently handle diffuse reflection and glossy reflection separately. For diffuse reflection, we adopt the lightmap technique which is the de facto solution in many rendering systems [Seyb et al. 2020]. Generally, the conventional lightmap technique is plagued by a long baking time since a very high sampling rate is required to generate noise-free results. Our learning-based image reconstruction method makes it possible to accept noisy lightmaps synthesized with a low sampling rate. Once trained on a large-scale dataset, our network suppresses noise on the diffuse regions and yields clear diffuse reflection. This significantly reduces the time required by precomputing the lightmap, and makes it convenient for artistic design.

### 4.2 Probe precomputation and storage

To handle glossy reflection, we first precompute a sparse set of light probes in the scene and arrange them on a regular 3D grid. For each probe $\mathbf{M}$ located at a 3D position $\mathbf{p}$, we render a 360° panorama storing surfaces that are visible from the center of the probe. Currently, we cache incident radiance from glossy surface, material ID and reflected position in the panoramas. One case is provided in Fig. 3. The incident radiance is generated by path tracing the scene starting from the center of the probe and only takes glossy reflection into consideration. The reflected position is generated by treating all surfaces as perfect mirrors and storing the world position of geometry seen with one-bounce reflection. Both reflected position and material ID will be used in the following gradient-based reflection search algorithm.

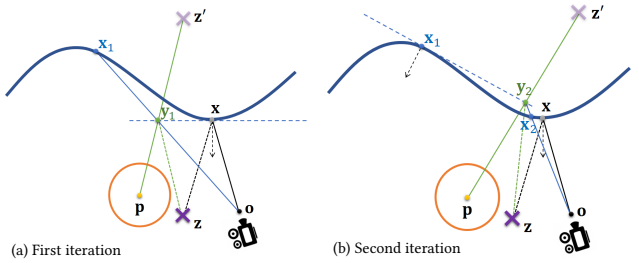



Fig. 4. Illustration of our gradient-based reflection search algorithm. The solid purple cross represents a point **z** seen from the viewpoint **o** with one-bounce mirror reflection, while the transparent purple cross is its virtual image **z′** in the local tangent space. The goal of this algorithm is to search a point on each probe that stores the actual reflection from **z**.

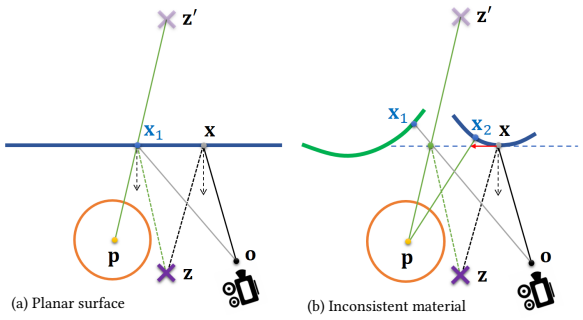



Fig. 5. Two special cases of our gradient-based reflection search algorithm.

Similar to the treatment of lightmaps, our neural method allows the probes to be imperfect, meaning that we can generate noisy probes with a low sampling rate. This differs from some previous methods [McGuire et al. 2017; Rodriguez et al. 2020; Silvennoinen and Lehtinen 2017] which require noise-free probes, and can further reduce the time cost in the precomputation step. Benefiting from the image priors learned by the network and the additional G-buffers (depth, normal and albedo), high-quality images could be recovered from low-quality inputs.

### 4.3 Gradient-based reflection search

One key step in probe-based rendering pipelines is to query probes without producing improper glossy reflection stemming from parallax changes between the probe and the novel view. We address this issue by introducing a new gradient-based reflection search algorithm. This algorithm searches physically-plausible reflection positions in the whole screen space of the novel view and converges fast to the optimal solution with the help of the gradient at each candidate.

At runtime, for each viewpoint **o** we first identify $K = 8$ nearest probes around **o** (see Fig. 2) and then search on each probe the optimal reflection position that matches the glossy shading point at the novel view (see Fig. 4). The optimal reflection position is then

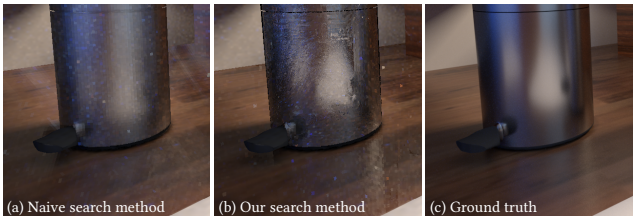



Fig. 6. Illustration of the parallax-corrected glossy reflection generated by our reflection search method (b), as compared with the effect of the naive search method (a) and the ground truth (c).

used to query the probes, yielding 8 images ($\mathbf{I}_{g,1}$, $\mathbf{I}_{g,2}$, ..., $\mathbf{I}_{g,8}$) with parallax-corrected glossy reflection effects.

*The basic algorithm.* The search process runs iteratively for each image's pixels, as visually explained in Fig. 4. Suppose that a primary ray starting from **o** reaches **z** in the scene after being reflected by a glossy shading point **x**. In the first iteration, we obtain the tangent plane at **x** according to the normal of **x**. This tangent plane serves as a mirror reflector that determines the virtual image **z′** of **z** in the local tangent space of **x**. The intersection point of line **pz′** and the tangent plane ($\mathbf{y}_1$ in Fig. 4(a)) yields a candidate $\mathbf{x}_1$ that is used to further validate its correctness. For planar surfaces, it is easy to check that the candidate of the first iteration $\mathbf{x}_1$ happens to be the optimal solution, as explained in Fig. 5(a). For curved surfaces, the iteration continues at $\mathbf{x}_1$ and conducts a further search along its gradient, as shown in Fig. 4(b). Each consequent search is expected to produce candidates closer to the optimal position. The correctness of the $k$-th candidate $\mathbf{x}_k$ is defined as

$$\mathbb{C}(\mathbf{x}_k) = \frac{\mathbf{p}\mathbf{x}_k + \mathbf{z}\mathbf{x}_k}{\|\mathbf{p}\mathbf{x}_k + \mathbf{z}\mathbf{x}_k\|} \cdot \mathbf{N}(\mathbf{x}_k) \tag{3}$$

where $\mathbf{N}(\mathbf{x}_k)$ indicates the normal at $\mathbf{x}_k$. The value of $\mathbb{C}(\mathbf{x}_k)$ reflects the confidence of each candidate $\mathbf{x}_k$. The iteration stops until reaching:

(1) a predefined threshold of the optimal value of $\mathbb{C}(\mathbf{x}_k)$, i.e., $T_{\max}$, or
(2) a predefined maximum search number $N_{\max}$.

After convergence for each probe, a weighted blending operation is performed on the eight images ($\mathbf{I}_{g,1}$, $\mathbf{I}_{g,2}$, ..., $\mathbf{I}_{g,8}$), i.e.,

$$\mathbf{I}_g = \sum_{k=1}^{8} W_k \mathbf{I}_{g,k} \quad \text{with} \quad W_k = \frac{\exp[-\tau \mathbb{C}_k]}{\sum_{k=1}^{8} \exp[-\tau \mathbb{C}_k]} \tag{4}$$

where $\mathbb{C}_k$ is a confidence map for the $k$-th probe, generated according to Eq. (3); $\tau$ is a parameter defining the decay rate of the exponential. As shown in Fig. 6, this method well solves the parallax problem for the glossy reflection, significantly outperforming the naive search method which connects the glossy shading point **x** and the probe's location **p** directly.

*Special cases.* Two special cases should be concerned during the iteration. The first happens when two consecutive candidates (e.g., **x** and $\mathbf{x}_1$ in Fig. 5(b)) lie on different objects. We check this case by comparing the material IDs of **x** and $\mathbf{x}_1$. If their material IDs


Fig. 7. The impact of visibility checking in our search algorithm. Note the ghosting effects highlighted by the red arrows in (a) due to the visibility issue.

are different, our algorithm will still search along the gradient of $\mathbf{x}$ but with a shortened step, as indicated by the red arrow in Fig. 5(b). The step is shortened continuously until the same material ID is achieved. The second special case happens when the projected coordinate of a candidate is out of the viewport. In this case, we perform the same shortened search as in the first case.

*Handling visibility.* Even an optimal position can be found through the above steps, there is still a risk that this optimal position is actually occluded from the queried probe. In this case, this optimal position is invalid and should be dropped. Otherwise, there will be ghosting artifacts in the resulting images, as shown in Fig. 7. To address this issue, we resort to the reflected position map stored in each probe, e.g., Fig. 3(a). We evaluate the distance between $\mathbf{z}$ and the value queried from the reflected position map according to the retrieved optimal position. If the distance is larger than a predefined threshold $D_{\max}$, this optimal value will not be used. This can suppress the ghosting effects due to the improper query.

*Comparison and discussion.* Rodriguez et al. [2020] propose a two-level grid search algorithm to find the optimal reflection position for each pixel. They first use per-pixel curvature to obtain an approximate reflection position $\mathbf{x}'$, and then fine-tune it in its neighborhood by a two-level grid search strategy. A key ingredient necessary for this algorithm is the curvatures to determine the approximation $\mathbf{x}'$. However, per-pixel curvature will be inaccurate for meshes that are tessellated insufficiently or bumpy surfaces with dramatic changes. More importantly, the search is constrained in a fixed local area around $\mathbf{x}'$, which may miss optimal positions that are actually out of this area, leading to obvious parallax errors. In addition, a filter footprint estimation is involved in the filtering step of their search algorithm. This may further introduce bias to the highlights due to the errors from the estimation. To demonstrate this, we show a case from the KITCHEN scene in Fig. 8. As seen, the query results of GPR (either in gathering or in filtering) deviate significantly from the ground truth, resulting in inaccurate glossiness as highlighted in the error maps. In comparison, our results match the ground truth quite well, since we search the optimal solution in the whole viewport. The accuracy of our algorithm is clearly indicated by the error maps and the RMSE values in Fig. 8. As another advantage, our search algorithm runs much faster than the two-level search algorithm in GPR, since for many surfaces our method converges to the optimal solution with only a small number of iterations.

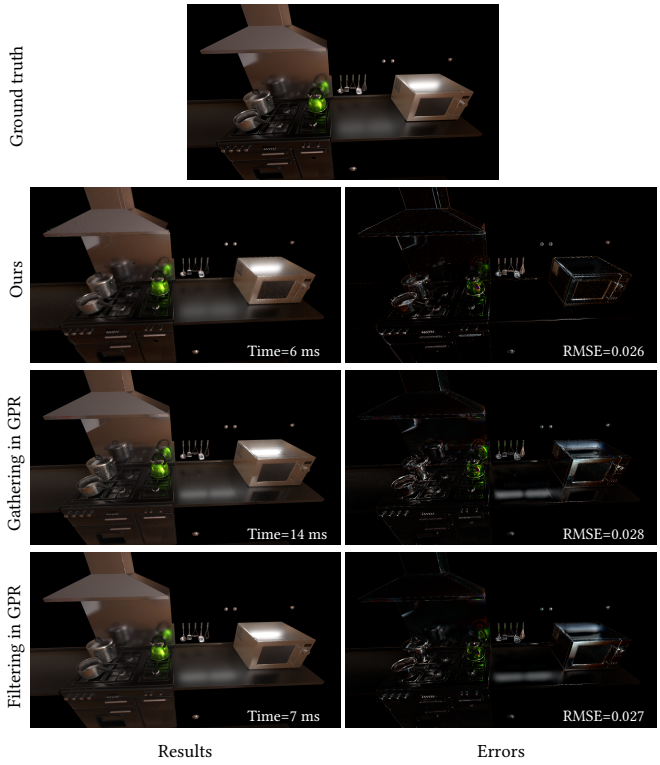



Fig. 8. Visual comparison between our gradient-based reflection search algorithm and the gathering and filtering algorithms in GPR [Rodriguez et al. 2020]. For fair comparisons, we precompute the light probes using the same sampling rate as GPR. Rendering times (tested on an NVIDIA GeForce RTX 3090Ti graphics card) and errors in terms of RMSE are reported for each algorithm.

## 4.4 Neural image reconstruction

Since both lightmaps and light probes in our pipeline are generated with a low sampling rate, there will be noise and aliasing in the images $\mathbf{I}_{\mathrm{d}}$ and $\mathbf{I}_{\mathrm{g}}$. To recover high-quality images from these low-quality inputs, we leverage a specially designed neural network described below.

*Network architecture.* Fig. 9 shows the detailed architecture of the neural network, which is a typical deep encoder-decoder architecture with multiple branches. The encoder consists of two branches. One branch extracts multi-scale feature maps from the input low-quality image $\mathbf{I} = \mathbf{I}_{\mathrm{d}} + \mathbf{I}_{\mathrm{g}}$, while the other branch extracts multi-scale feature maps from several G-buffers, including a depth map $\mathbf{D}$, a normal map $\mathbf{N}$, an albedo map $\mathbf{A}$ and a reflected albedo map $\mathbf{B}$. Note that the albedo map $\mathbf{A}$ is generated for diffuse reflection areas, and the reflected albedo map $\mathbf{B}$ is generated for glossy reflection areas which captures the albedo of reflected geometries after one-bounce reflection. For all convolution operations in both branches, the kernel size is set to $3 \times 3$. Stride-2 convolution is used to downsample the resolution of the feature maps.

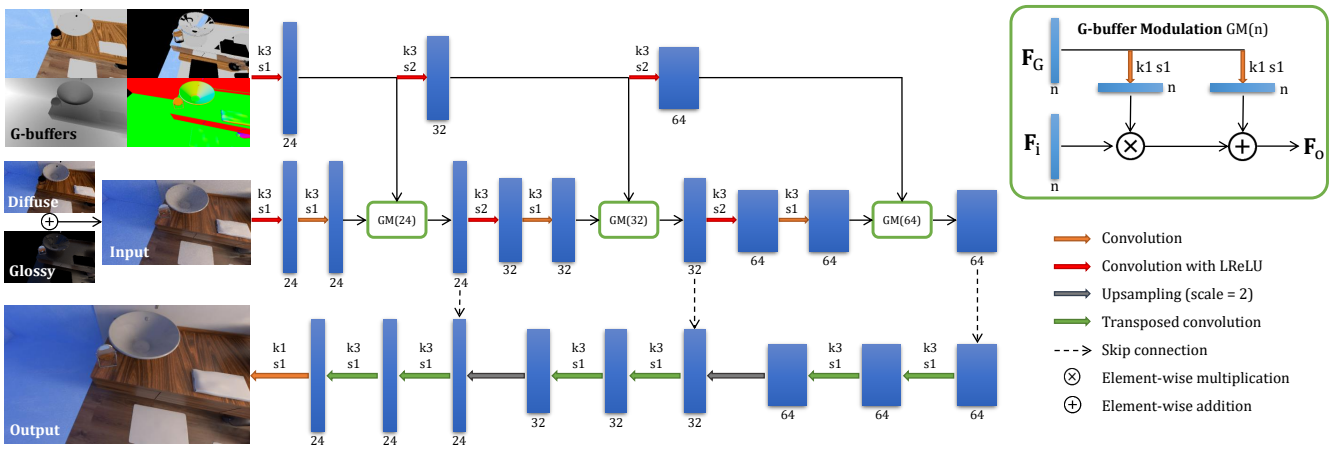



Fig. 9. The architecture of the network used in neural image reconstruction. Each blue rectangle refers to a feature map with the channel number marked below. Each arrow indicates a layer of operation explained at the bottom right. Here, "k$m$ s$n$" means that the convolution operation has a kernel size of $m \times m$ and a stride of $n$.

To fuse the feature maps from different branches in the encoder side, we design and resort to a G-buffer modulation (GM) module shown in the green box of Fig. 9. Unlike simple concatenation, this module, inspired by SPADE in [Park et al. 2019], re-weights the feature maps in the input branch via pixel-wise transformation of the feature maps from the G-buffer branch. We can formulate it as

$$\mathbf{F}_o = \text{Conv1}(\mathbf{F}_G) \otimes \mathbf{F}_i \oplus \text{Conv2}(\mathbf{F}_G) \tag{5}$$

where $\mathbf{F}_i$ and $\mathbf{F}_G$ represent feature maps from the input branch and the G-buffer branch, respectively. $\mathbf{F}_o$ is the output of the GM module. Conv1 and Conv2 are two different convolution operations with a kernel size of $1 \times 1$ and a stride of 1. In this case, the feature maps from the G-buffer branch serve as a condition to help address our image reconstruction problem. This can better exploit features from clean G-buffers while avoiding noise and other potential artifacts from the input image $\mathbf{I}$.

The decoder has a symmetrical structure with the encoder with slight differences. It uses transposed convolution operations to gradually fuse and upsample the features in a sequential fashion from deep feature maps to shallow ones. Leaky ReLU activating function is adopted after every transposed convolution. Moreover, our network also uses skip connections between mirrored layers in the encoder and decoder stacks to preserve fine details. In order to guarantee real-time performance in the inference phase, our network is designed in a light-weight manner with few convolution layers and channels.

*Temporal reprojection.* To improve temporal stability in animation sequences, we leverage a temporal reprojection method, in a similar way as in [Koskela et al. 2019; Meng et al. 2020; Schied et al. 2017]. The key idea behind this method is to accumulate previous frames to the current one by using the rendered motion vectors. Currently, we accumulate two consecutive frames in the input. The decay factors of the two frames are set to 0.3 and 0.1, respectively.

*Loss function.* The loss function we adopt to train our network contains three items. First, we calculate the pixel-wise $L_1$ distance between the ground truth image $\hat{\mathbf{I}}$ and the output image $\mathbf{I}_o$ inferred by the network:

$$\mathcal{L}_{L1} = ||\hat{\mathbf{I}} - \mathbf{I}_o||_1 \tag{6}$$

In addition, to preserve more details in the image, we adopt Learned Perceptual Image Patch Similarity (LPIPS) [Zhang et al. 2018] to calculate the perceptual loss $\mathcal{L}_{\text{LPIPS}}$ between two images. We use pre-trained VGG-16 [Simonyan and Zisserman 2014] as our evaluation network and calculate the layer-wise $L_2$ distance between two images. Specifically, $\mathcal{L}_{\text{LPIPS}}$ is calculated as

$$\mathcal{L}_{\text{LPIPS}} = \sum_l ||\omega_l \odot (\mathbf{F}_l(\hat{\mathbf{I}}) - \mathbf{F}_l(\mathbf{I}_o))||_2^2 \tag{7}$$

where $\mathbf{F}_l$ is the feature map extracted from the $l$-th layer of the VGG-16 network and $\omega_l$ is a vector pre-learned by LPIPS to scale the activations channel-wise. Last, we adopt a temporal loss to suppress the temporal flickering that may be caused by our search algorithm or network inference. Our temporal loss is calculated between two consecutive frames $\mathbf{I}_o$ and $\mathbf{I}'_o$, after $\mathbf{I}'_o$ being warped with motion vector $\mathbf{V}$:

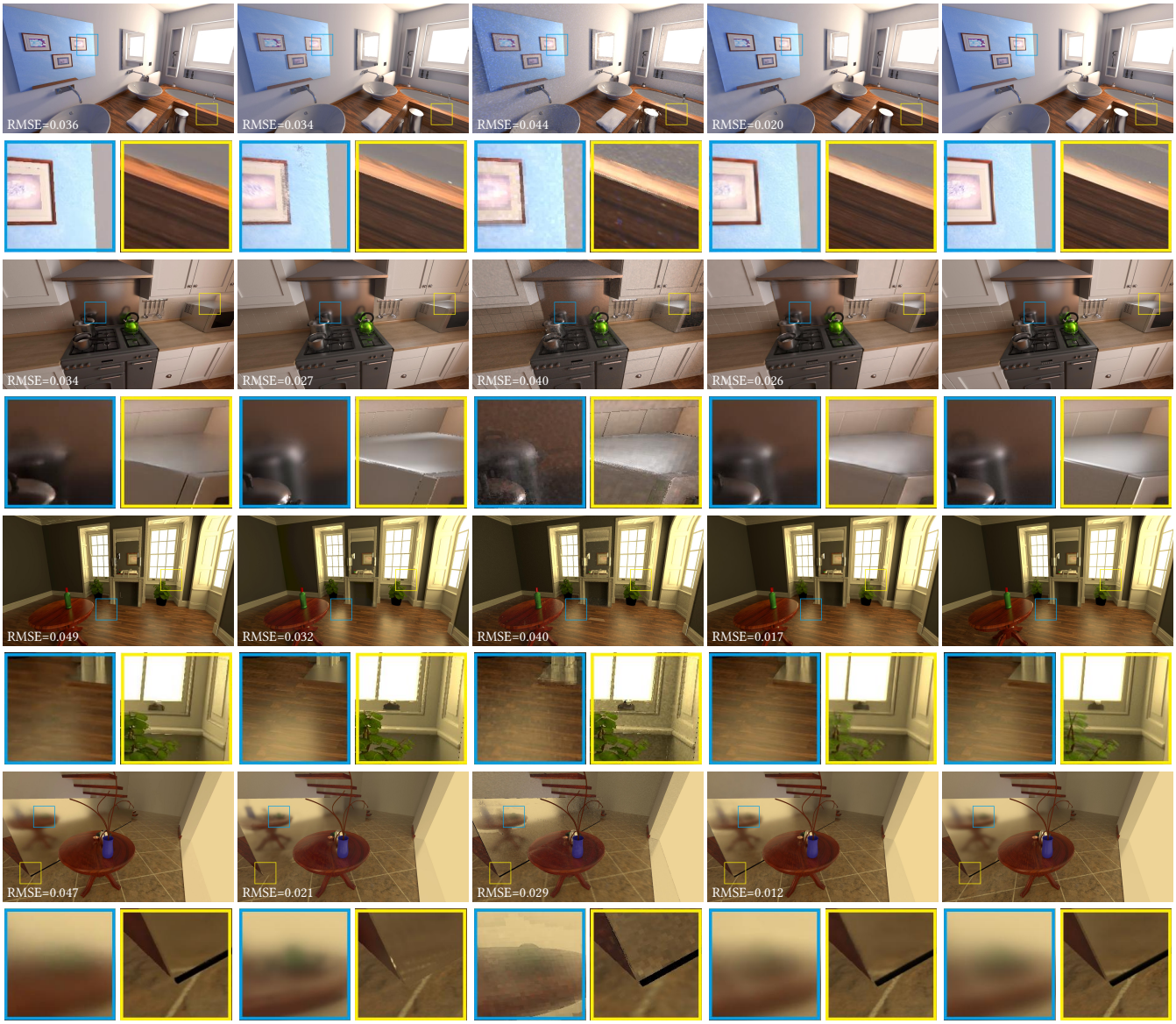
$$\mathcal{L}_T = ||\mathbf{I}_o - \mathcal{W}(\mathbf{I}'_o, \mathbf{V})||_1. \tag{8}$$

The final loss function $\mathcal{L}$ of our network is the summation of the three items above:

$$\mathcal{L} = \lambda_{L1}\mathcal{L}_{L1} + \lambda_{\text{LPIPS}}\mathcal{L}_{\text{LPIPS}} + \lambda_T\mathcal{L}_T \tag{9}$$

where $\lambda_{L1}$, $\lambda_{\text{LPIPS}}$ and $\lambda_T$ are weights to balance the influence of three losses. In our implementation, we set $\lambda_{L1}$ to 0.8, $\lambda_{\text{LPIPS}}$ to 0.03, and $\lambda_T$ to 0.15.

*Training and inference.* To ensure high-quality image reconstruction, we current opt to train the network independently for each scene. The training time and the time used to generate training examples are counted as precomputation time. During this phase, we render 20 ground truth images for a wide range of views covering the whole




(a) RTRT [Benty et al. 2020] (b) GPR [Rodriguez et al. 2020] (c) Ours (Search) (d) Ours (Final) (e) Ground truth

Fig. 10. Visual comparisons against two state-of-the-art methods (RTRT [Benty et al. 2020] and GPR [Rodriguez et al. 2020]) which can also achieve real-time global illumination with complex glossy paths. Errors in terms of RMSE are reported for each method.

scene. Note that these views will no longer be used in the inference phase. The training process is performed using PyTorch library [Paszke et al. 2019], and typically takes around 4 hours (1000 epochs using the Adam optimizer) to converge for each scene. TensorRT is used to accelerate network inference.

At runtime, images from gradient-based reflection search and corresponding G-buffers are first rendered into OpenGL buffers, and then mapped to CUDA tensors for network inference. The mapping process has negligible time cost compared to the consumption of the whole pipeline.

## 5 RESULTS AND DISCUSSION

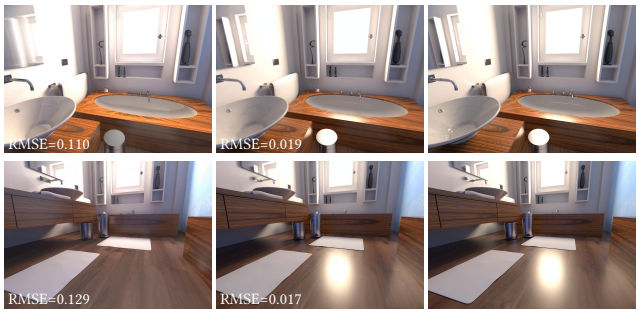
The whole pipeline of our method contains two stages. In the pre-computation stage, we generate the lightmaps and probe data using a modified version of Mitsuba renderer [Jakob 2010]. In our current implementation, each lightmap is rendered at 256 spp and a resolution of $2048 \times 2048$, while each probe is rendered at 128 spp and a resolution of $1024 \times 512$. We typically use 256 probes for each scene. The real-time rendering stage is implemented on top of SIBR [Bonopera et al. 2020] with OpenGL and TensorRT. All

output images are rendered at a resolution of 1920×1080 on a desktop machine with an Intel I7-8700K CPU and an NVIDIA GeForce RTX 3090Ti GPU. The thresholds and parameters used in our reflection search algorithm are set as follows: $T_{\max} = 0.999$, $N_{\max} = 20$, $D_{\max} = 0.1$ and $\tau = 100$. We test four scenes (Bathroom, Kitchen, Livingroom and Staircase) provided by Rodriguez et al. [2020]. These scenes contain many glossy surfaces that are hard to handle in real time. Animation sequences of these four scenes are provided in the supplemental video.

## 5.1 Rendering quality comparisons

We first compare our work with two state-of-the-art methods that can also generate a wide range of GI effects, in particular multi-bounce glossy reflection, with high performance. Pair-wise visual comparisons for the four scenes are provided in Fig. 10. The first method is RTRT implemented atop NVIDIA's Falcor framework [Benty et al. 2020]. To guarantee real-time performance (>30 FPS) and high image quality, we also adopt precomputed lightmaps (generated at 2048 spp) in the RTRT framework to reproduce diffuse reflection. Although RTRT, as a flexible and general solution, can handle long glossy paths in theory, the time budget only allows us to generate interreflection within three bounces. Therefore, secondary glossy reflection effects are absent as illustrated in the first column of Fig. 10. For instance, the Kitchen scene misses highlights in the reflected pot, shown in the closeup. Moreover, RTRT tends to under-estimate the glossiness of the highlights. The second competitor is the GPR method proposed by Rodriguez et al. [2020]. GPR relies on high-quality probes rendered with a high sampling rate (2048 spp) to reproduce glossy reflection. Besides the long baking time required for precomputing the probe data, the probe query algorithm used in this method easily yields inaccurate glossiness of the highlights, especially on large planar surfaces, as we explained previously. Another common artifact that bothers GPR is specular geometric aliasing which can be observed along object edges. It will cause temporal flickering in the animation sequence. Thanks to a more accurate reflection search algorithm and a robust neural image reconstruction model, our method outperforms these competitors, achieving high-quality results that are very close to the path traced ground truth. In Fig. 10, we also provide the images after reflection search from our whole pipeline. Despite the noise and other artifacts stemming from low-quality lightmaps and light probes we adopted in the pipeline, these results (Our (Search)) faithfully capture the basic structures of light paths for each scene and are free from parallax errors.

In Fig. 11, we compare our work with another prevailing probe-based method proposed by McGuire et al. [2017]. This method develops a new probe data structure named Light Field Probes (LFP) to simulate complex light transport in real time. Here, we generate their probes at 2048 spp and 1024×1024 resolution to ensure reasonable image quality. For a fair comparison, the same number of probes are used as ours. As shown in Fig. 11(a), glossy highlights are not correctly reproduced by LFP, although it achieves real-time performance. This is actually a common limitation for most existing probe-based real-time rendering solutions, as the probe structure (either representation or query) is more friendly to diffuse reflection.




(a) LFP (b) Ours (c) Ground truth

Fig. 11. Visual comparisons with the Light Field Probes (LFP) method [McGuire et al. 2017]. Note that specular highlights are absent in the LFP method. Errors in terms of RMSE are reported for each method.

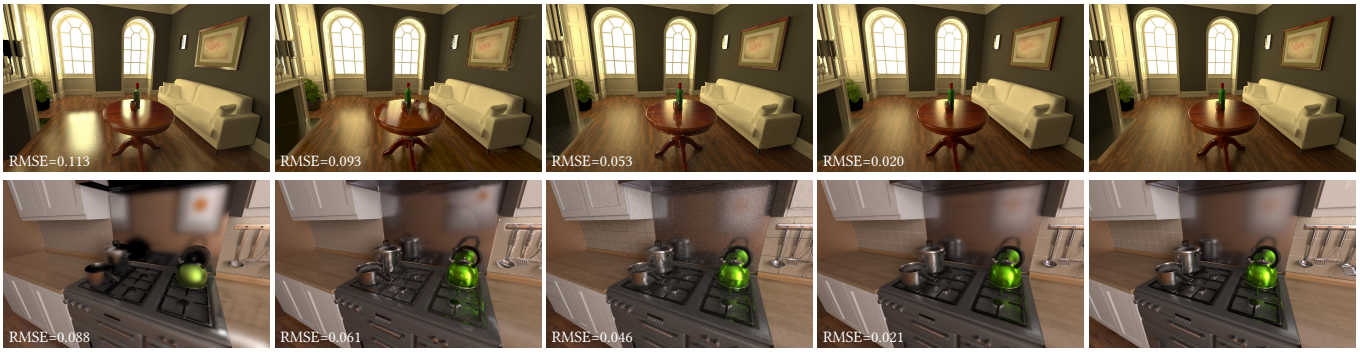
Our learning-based model allows us to generate relatively low-quality images at the reflection search step. Even so, the proposed new search method still outperforms some traditional methods in terms of visual quality. To demonstrate this, we make comparisons against two traditional methods based on Image Space Gathering (ISG) [Robison and Shirley 2009] and Unstructured Lumigraph Rendering (ULR) [Buehler et al. 2001], respectively. ISG approximates glossy reflection by caching specular reflection in a buffer and filtering that buffer during gathering. ULR is a typical image-based rendering method that relies on a large collection of images from different camera positions to achieve novel view image synthesis. Clearly, directly applying these two methods on our probe data (rendered at 2048 spp) will lead to unsatisfactory results that are even worse than those generated by our search algorithm.

## 5.2 Quantitative evaluation

To further validate the image quality of our method, we perform quantitative evaluation using Root Mean Square Error (RMSE), Structural Dissimilarity (DSSIM) and Learned Perceptual Image Patch Similarity (LPIPS) [Zhang et al. 2018]. We make comparisons against ISG, ULR, RTRT and GPR on four scenes. The errors are averaged over 20 frames for each scene, and the results are listed in Table 1. Overall, our work performs consistently better than previous methods in terms of different quantitative metrics. It should be noted that even the images generated by our gradient-based reflection search algorithm with low-quality probes beat those generated by some traditional methods (e.g., ISG and ULR). This indicates the superiority of the proposed search algorithm.

## 5.3 Validation of key design choices

In this section, we validate several key design choices in our pipeline. The first is the number of probes we choose during precomputation. We currently precompute 256 light probes for each scene. This ensures high image quality and reasonable precomputation time and memory cost. Increasing the number of probes will improve the image quality (e.g., preserving more details) as shown in Fig. 13, but at the cost of additional overhead both for time and storage.

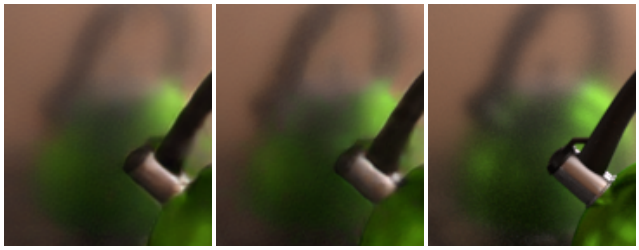



(a) ISG [Robison and Shirley 2009] (b) ULR [Buehler et al. 2001] (c) Ours (Search) (d) Ours (Final) (e) Ground truth

Fig. 12. Visual comparisons against Image Space Gathering (ISG) [Robison and Shirley 2009] and Unstructured Lumigraph Rendering (ULR) [Buehler et al. 2001]. Here, probe data used by ISG and ULR are rendered at 2048 spp, while our method only requires a low sampling rate (128 spp). Errors in terms of RMSE are reported for each method.

Table 1. Quantitative comparisons, in terms of RMSE, DSSIM and LPIPS, against image space gathering (ISG) [Robison and Shirley 2009], unstructured lumigraph (ULR) [Buehler et al. 2001], RTRT implemented in NVIDIA's Falcor framework [Benty et al. 2020] and glossy probe reprojection (GPR) [Rodriguez et al. 2020]. For our method, we list statistics of images generated by our reflection search algorithm (Ours (search)) and the network inference (Ours (Final)) respectively. The best results are highlighted in **bold**.

| Scene | Method | RMSE | DSSIM | LPIPS |
|---|---|---|---|---|
| Bathroom | ISG | 0.087 | 0.131 | 0.093 |
| | ULR | 0.110 | 0.150 | 0.134 |
| | RTRT | 0.057 | 0.068 | 0.055 |
| | GPR | 0.033 | 0.041 | 0.044 |
| | Ours (Search) | 0.043 | 0.093 | 0.139 |
| | Ours (Final) | **0.022** | **0.029** | **0.030** |
| Kitchen | ISG | 0.078 | 0.132 | 0.092 |
| | ULR | 0.060 | 0.117 | 0.103 |
| | RTRT | 0.041 | 0.049 | 0.048 |
| | GPR | 0.028 | 0.035 | 0.049 |
| | Ours (Search) | 0.043 | 0.108 | 0.139 |
| | Ours (Final) | **0.024** | **0.031** | **0.032** |
| Livingroom | ISG | 0.126 | 0.190 | 0.110 |
| | ULR | 0.099 | 0.195 | 0.107 |
| | RTRT | 0.055 | 0.081 | 0.062 |
| | GPR | 0.034 | 0.044 | 0.047 |
| | Ours (Search) | 0.044 | 0.078 | 0.110 |
| | Ours (Final) | **0.022** | **0.022** | **0.020** |
| Staircase | ISG | 0.069 | 0.093 | 0.064 |
| | ULR | 0.054 | 0.083 | 0.061 |
| | RTRT | 0.053 | 0.074 | 0.051 |
| | GPR | 0.020 | 0.021 | 0.030 |
| | Ours (Search) | 0.030 | 0.060 | 0.100 |
| | Ours (Final) | **0.014** | **0.013** | **0.008** |



(a) 256 probes (b) 489 probes (c) Ground truth

Fig. 13. Comparing images generated with different numbers of probes. More details appear when the number of probes increases, at the cost of more time and storage cost.

(a) Concate (b) w/o LPIPS (c) Ours (d) Ground truth

Fig. 14. Validation of our network design. We compare our complete model used in neural image reconstruction with two ablated models: one model fuses feature maps from two different branches using simple concatenation (a) and the other removes the LPIPS loss during training (b).

Next, we show several ablated models for the network used in neural image reconstruction. The proposed network employs a G-buffer modulation module to better exploit features from G-buffers. As compared in Fig. 14(a) and (c), replacing this module with simple concatenation will introduce obvious artifacts due to improper feature fusion. Moreover, an LPIPS loss is added in the loss function


Fig. 15. Impact of the temporal reprojection strategy and the temporal loss. Here, we compare closeups from five consecutive frames generated with and without the temporal components. Note the flickering highlights along the edge of the bin and on the bottle when removing these two components (-Temp).

to make the results perceptually better. Its effectiveness is validated by the pairwise comparison between Fig. 14(b) and (c).

To enforce temporal stability for time-evolving images, we employ a temporal reprojection strategy and a temporal loss in network training. Without these two components, temporal flickering will appear frequently. In Fig. 15, five consecutive frames in an animation sequence are shown. Note the flickering highlights along the edges of the bin and on the bottle, in the case of removing all temporal concerns.

### 5.4 Performance analysis

Time performance for precomputation and real-time rendering is reported in Table 2 and Table 3, respectively. Comparisons are made against GPR to show the superiority of our method.

The precomputation for both our method and GPR runs on a high performance server with four Intel Xeon Gold 5118 CPUs and 128GB RAM. The results are reported in Table 2. GPR requires high-quality lightmaps and probes that are rendered at a very high sampling rate (2048 spp). This results in roughly 20 hours for each scene. In comparison, the low sampling rate adopted in our pipeline reduces this time cost for our method to less than one hour. However, the per-scene training strategy currently incurs 12 hours (including 20 ground-truth images generation and network training) overhead using an NVIDIA RTX 3090Ti GPU. This additional time cost, which should also be counted in the precomputation phase, reduces the performance gain achieved by the efficient generation of low-quality light probes and lightmaps. Ever so, our method still largely reduces the total precomputation time consumption as reported in Table 2.

The runtime performance is reported on a PC with an Intel Core i7-6900K CPU and an NVIDIA RTX 3090Ti GPU. On such a platform, both our method and GPR achieve real-time performance, as shown in Table 3. In particular, the proposed reflection search algorithm requires roughly 6 ms per-frame to achieve the optimal solution. This is much faster than the search steps involved in GPR. Moreover, the light-weight design in our network allows us to infer one 1080P image within 15 ms, guaranteeing real-time performance (>30 FPS) for the whole pipeline.

Table 2. Precomputation time cost breakdown for GPR [Rodriguez et al. 2020] and our method, running on a high performance service with four Intel Xeon Gold 5118 CPUs and 128GB RAM.

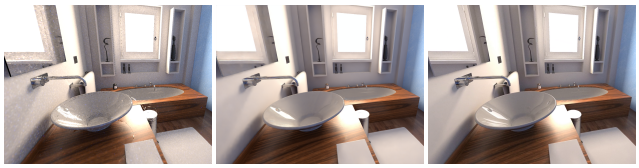
| Scene | Method | Lightmap | Probes | GT | Train | **Total** |
|---|---|---|---|---|---|---|
| Bat.. | GPR | 2.9 h | 18.3 h | - | - | 21.2 h |
| | Ours | 0.08 h | 0.8 h | 8.2 h | 3.9 h | 13.0 h |
| Kit.. | GPR | 1.5 h | 16.5 h | - | - | 18.0 h |
| | Ours | 0.04 h | 0.6 h | 8.1 h | 3.9 h | 12.6 h |
| Liv.. | GPR | 3.1 h | 17.1 h | - | - | 20.2 h |
| | Ours | 0.09 h | 0.9 h | 7.3 h | 3.9 h | 12.2 h |
| Sta.. | GPR | 2.3 h | 19.2 h | - | - | 21.5 h |
| | Ours | 0.06 h | 0.8 h | 6.7 h | 3.9 h | 11.5 h |

Table 3. Rendering time cost breakdown for GPR [Rodriguez et al. 2020] and our method. These statistics are collected from a PC with an Intel i7-6900K CPU and an NVIDIA GeForce RTX 3090Ti GPU. The search step listed here for GPR includes gathering and filtering procedures. Raster. means rasterizing G-buffers.

| Scene | Method | Raster. | Raycast | Search | Infer. | **Total** |
|---|---|---|---|---|---|---|
| Bat.. | GPR | 0.5 ms | 4 ms | 20.5 ms | - | 25.0 ms |
| | Ours | 0.5 ms | 4 ms | 6 ms | 14.6 ms | 25.1 ms |
| Kit.. | GPR | 0.6 ms | 4 ms | 21.6 ms | - | 26.2 ms |
| | Ours | 0.6 ms | 4 ms | 6 ms | 14.6 ms | 25.2 ms |
| Liv.. | GPR | 0.5 ms | 4 ms | 21.6 ms | - | 26.1 ms |
| | Ours | 0.6 ms | 4 ms | 6 ms | 14.6 ms | 25.2 ms |
| Sta.. | GPR | 0.4 ms | 4 ms | 19.5 ms | - | 23.9 ms |
| | Ours | 0.6 ms | 4 ms | 4.4 ms | 14.6 ms | 23.6 ms |

## 6 LIMITATIONS AND FUTURE WORK

*Reduced brightness and inconsistency of highlights.* Due to the logarithm compression for HDR inputs and insufficient G-buffer information, high-frequency highlights may reduce brightness or even disappear after the network inference. Such a failure case is shown in Fig. 16. This is a common limitation for many neural image reconstruction methods, as mentioned in KPCN [Bako et al. 2017] and GradNet [Guo et al. 2019]. It also occasionally causes temporal flickering of small highlights as shown in the supplemental video. Adopting more sophisticated neural network architectures such as GANs [Goodfellow et al. 2014] or Transformers [Ranftl et al. 2021] may alleviate this issue.



(a) Ours (Search) (b) Ours (Final) (c) Ground truth

Fig. 16. Our method misses some highlights in the bathroom bowl after the network inference (Final), even though these highlights are preserved in the reflection search step (Search).

*General model.* We currently train a separate model for each 3D scene in neural image reconstruction. Although this is a widely adopted strategy in many recent neural rendering tasks [Guo et al. 2021a; Mildenhall et al. 2020; Xiao et al. 2020] to ensure high quality for the output frames, it will cause additional precomputation overhead in our pipeline. A general model that is trained on a large number of different scenes and generalizes well to other scenes will be very appealing, since it will dramatically reduce the precomputation time to less than an hour.

*Efficient probe updating.* To control the budget of precomputation time, we currently render the light probes with 128 spp, a fairly low sample rate. Even so, the time needed for precomputing the light probes is still not affordable for dynamic scenes that require efficient probe updating. DDGI [Majercik et al. 2019] supports rapid probe updating, but is limited to diffuse reflection. For glossy reflection, it is still a challenging issue which deserves further research.

*Automatic probe placement.* In our current implementation, probes are simply laid out in a grid-like structure. This facilitates probe interpolation, but may result in inadequate spatial coverage [Cupisz 2012]. Recently, there are some attempts to place light probes adaptively and automatically [Vardis et al. 2021; Wang et al. 2019], according to the input scene's content. Although our deep learning-based solution is more robust to this issue than traditional methods based on analytical interpolation, it is still interesting to see if adaptive probe placement is beneficial for our method. Furthermore, investigating a joint learning scheme of probe placement and novel view reconstruction is also an interesting future work.

## 7 CONCLUSIONS

This paper has presented a learning-based solution for real-time rendering with full global illumination. It builds upon precomputed light probes and employs a data-driven model to faithfully reproduce a wide range of GI effects, including multi-bounce glossy reflection which is forbidden by many previous probe-based solutions. The whole pipeline of the proposed method comprises a gradient-based reflection search algorithm and a dedicated neural network for final image reconstruction. Notably, the new reflection search algorithm only relies on per-pixel gradient and recovers parallax-free reflection at any view point with only several milliseconds. The neural network used in neural image reconstruction succeeds in suppress noise, aliasing and other artifacts that bother other real-time rendering methods, leveraging image priors learned from the labeled training data. To the best of our knowledge, this paper presents the first work dealing with light probes with deep learning techniques. Extensive experiments on multiple scenes validate the effectiveness and efficiency of the proposed method.

## ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable suggestions. This work was supported by the National Natural Science Foundation of China (No. 61972194 and No. 62032011) and the Natural Science Foundation of Jiangsu Province (No. BK20211147).

## REFERENCES

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 (July 2017), 97:1–97:14.

Nir Benty, Kai-Hwa Yao, Petrik Clarberg, Lucy Chen, Simon Kallweit, Tim Foley, Matthew Oakes, Conor Lavelle, and Chris Wyman. 2020. The Falcor Rendering Framework. https://github.com/NVIDIAGameWorks/Falcor https://github.com/NVIDIAGameWorks/Falcor.

Benedikt Bitterli, Chris Wyman, Matt Pharr, Peter Shirley, Aaron Lefohn, and Wojciech Jarosz. 2020. Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 39, 4 (July 2020).

James F. Blinn and Martin E. Newell. 1976. Texture and Reflection in Computer Generated Images. *Commun. ACM* 19, 10 (oct 1976), 542–547.

Sebastien Bonopera, Peter Hedman, Jerome Esnault, Siddhant Prakash, Simon Rodriguez, Theo Thonat, Mehdi Benadel, Gaurav Chaurasia, Julien Philip, and George Drettakis. 2020. sibr: A System for Image Based Rendering. https://gitlab.inria.fr/sibr/sibr_core

Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. 2001. Unstructured Lumigraph Rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. Association for Computing Machinery, New York, NY, USA, 425–432.

John Burgess. 2020. RTX on the NVIDIA Turing GPU. *IEEE Micro* 40, 2 (2020), 36–44.

Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. 2017. Interactive Reconstruction of Monte Carlo Image Sequences Using a Recurrent Denoising Autoencoder. *ACM Trans. Graph.* 36, 4, Article 98 (jul 2017), 12 pages.

Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. 2013. Depth Synthesis and Local Warps for Plausible Image-Based Navigation. *ACM Trans. Graph.* 32, 3, Article 30 (jul 2013), 12 pages.

Min Chen and James Arvo. 2000a. Perturbation methods for interactive specular reflections. *IEEE Transactions on Visualization and Computer Graphics* 6, 3 (2000), 253–264.

Min Chen and James Arvo. 2000b. Theory and Application of Specular Path Perturbation. *ACM Trans. Graph.* 19, 4 (oct 2000), 246–278.

Zhang Chen, Anpei Chen, Guli Zhang, Chengyuan Wang, Yu Ji, Kiriakos N. Kutulakos, and Jingyi Yu. 2020. A Neural Rendering Framework for Free-Viewpoint Relighting. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 5598–5609.

Zina H. Cigolle, Sam Donow, Daniel Evangelakos, Michael Mara, Morgan McGuire, and Quirin Meyer. 2014. A Survey of Efficient Representations for Independent Unit Vectors. *Journal of Computer Graphics Techniques (JCGT)* 3, 2 (17 April 2014), 1–30.

Robert Cupisz. 2012. Light Probe Interpolation using Tetrahedral Tessellations. Game Developers Conference.

R. R. Currius, D. Dolonius, U. Assarsson, and E. Sintorn. 2020. Spherical Gaussian Light-field Textures for Fast Precomputed Global Illumination. *Computer Graphics Forum* 39, 2 (2020), 133–146.

Paul Debevec, Yizhou Yu, and George Borshukov. 1998. Efficient View-Dependent Image-Based Rendering with Projective Texture-Mapping. In *Rendering Techniques '98*. Springer Vienna, Vienna, 105–116.

Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. 1996. Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-Based Approach. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*. Association for Computing Machinery, New York, NY, USA, 11–20.

Valentin Deschaintre, Miika Aittala, Fredo Durand, George Drettakis, and Adrien Bousseau. 2018. Single-Image SVBRDF Capture with a Rendering-Aware Deep Network. *ACM Trans. Graph.* 37, 4, Article 128 (July 2018), 15 pages.

Yue Dong. 2019. Deep appearance modeling: A survey. *Visual Informatics* 3, 2 (2019), 59 – 68.

Renaud Adrien Dubouchet, Laurent Belcour, and Derek Nowrouzezahrai. 2017. Frequency Based Radiance Cache for Rendering Animations. In *Eurographics Symposium on Rendering - Experimental Ideas & Implementations*, Matthias Zwicker and Pedro Sander (Eds.). The Eurographics Association.

S. M. Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S. Morcos, Marta Garnelo, Avraham Ruderman, Andrei A. Rusu, Ivo Danihelka, Karol Gregor, David P. Reichert, Lars Buesing, Theophane Weber, Oriol Vinyals, Dan Rosenbaum, Neil Rabinowitz, Helen King, Chloe Hillier, Matt Botvinick, Daan Wierstra, Koray Kavukcuoglu, and Demis Hassabis. 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. *Computer Graphics Forum* 40, 4 (2021), 15–27.

John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. 2019. DeepView: View Synthesis With Learned Gradient Descent. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2362–2371.

Duan Gao, Guojun Chen, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. 2020. Deferred Neural Lighting: Free-Viewpoint Relighting from Unstructured Photographs. *ACM Trans. Graph.* 39, 6, Article 258 (nov 2020), 15 pages.

Duan Gao, Xiao Li, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. 2019. Deep Inverse Rendering for High-Resolution SVBRDF Estimation from an Arbitrary Number of Images. *ACM Trans. Graph.* 38, 4, Article 134 (July 2019), 15 pages.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2* (Montreal, Canada) *(NIPS'14)*. MIT Press, Cambridge, MA, USA, 2672–2680.

Jonathan Granskog, Fabrice Rousselle, Marios Papas, and Jan Novák. 2020. Compositional Neural Scene Representations for Shading Inference. *ACM Trans. Graph.* 39, 4, Article 135 (July 2020), 13 pages.

Gene Greger, Peter Shirley, Philip M. Hubbard, and Donald P. Greenberg. 1998. The Irradiance Volume. *IEEE Comput. Graph. Appl.* 18, 2 (mar 1998), 32–43.

Jie Guo, Xihao Fu, Liqiang Lin, Hengjun Ma, Yanwen Guo, Shiqiu Liu, and Ling-Qi Yan. 2021a. ExtraNet: Real-Time Extrapolated Rendering for Low-Latency Temporal Supersampling. *ACM Trans. Graph.* 40, 6, Article 278 (dec 2021), 16 pages.

Jie Guo, Shuichang Lai, Chengzhi Tao, Yuelong Cai, Lei Wang, Yanwen Guo, and Ling-Qi Yan. 2021b. Highlight-Aware Two-Stream Network for Single-Image SVBRDF Acquisition. *ACM Trans. Graph.* 40, 4, Article 123 (jul 2021), 14 pages.

Jie Guo, Mengtian Li, Quewei Li, Yuting Qiang, Bingyang Hu, Yanwen Guo, and Ling-Qi Yan. 2019. GradNet: Unsupervised Deep Screened Poisson Reconstruction for Gradient-Domain Rendering. *ACM Trans. Graph.* 38, 6, Article 223 (nov 2019), 13 pages.

Yu Guo, Cameron Smith, Miloš Hašan, Kalyan Sunkavalli, and Shuang Zhao. 2020. MaterialGAN: Reflectance Capture Using a Generative SVBRDF Model. *ACM Trans. Graph.* 39, 6, Article 254 (Nov. 2020), 13 pages.

Saeed Hadadan, Shuhong Chen, and Matthias Zwicker. 2021. Neural Radiosity. *ACM Trans. Graph.* 40, 6, Article 236 (dec 2021), 11 pages.

Takahiro Harada. 2020. Hardware-Accelerated Ray Tracing in AMD Radeon ProRender 2.0. https://gpuopen.com/learn/radeon-prorender-2-0/.

John T. Hooker. 2016. Volumetric Global Illumination at Treyarch. In *Advances in real-time rendering, part I*. ACM SIGGRAPH 2016 Courses.

Jinkai Hu, Milo K. Yip, Guillermo Elias Alonso, Shihao Gu, Xiangjun Tang, and Xiaogang Jin. 2021. Efficient real-time dynamic diffuse global illumination using signed distance fields. *Vis. Comput.* 37, 9 (2021), 2539–2551.

Wenzel Jakob. 2010. Mitsuba renderer. http://www.mitsuba-renderer.org.

Wojciech Jarosz, Craig Donner, Matthias Zwicker, and Henrik Wann Jensen. 2008. Radiance Caching for Participating Media. *ACM Trans. Graph.* 27, 1, Article 7 (mar 2008), 11 pages.

Giulio Jiang and Bernhard Kainz. 2021. Deep radiance caching: Convolutional autoencoders deeper in ray tracing. *Computers & Graphics* 94 (2021), 22–31.

James T. Kajiya. 1986. The Rendering Equation. *SIGGRAPH Comput. Graph.* 20, 4 (aug 1986), 143–150.

Jan Kautz, Pere-Pau Vázquez, Wolfgang Heidrich, and Hans-Peter Seidel. 2000. A Unified Approach to Prefiltered Environment Maps. In *Rendering Techniques 2000*. Springer Vienna, Vienna, 185–196.

Matias Koskela, Kalle Immonen, Markku Mäkitalo, Alessandro Foi, Timo Viitanen, Pekka Jääskeläinen, Heikki Kultala, and Jarmo Takala. 2019. Blockwise Multi-Order Feature Regression for Real-Time Path-Tracing Reconstruction. *ACM Trans. Graph.* 38, 5, Article 138 (June 2019), 14 pages.

Tejas D Kulkarni, William F. Whitney, Pushmeet Kohli, and Josh Tenenbaum. 2015. Deep Convolutional Inverse Graphics Network. In *Advances in Neural Information Processing Systems*, Vol. 28. Curran Associates, Inc.

Jaroslav Křivánek, Pascal Gautron, Sumanta N. Pattanaik, and Kadi Bouatouch. 2005. Radiance caching for efficient global illumination computation. *IEEE Transactions on Visualization and Computer Graphics* 11, 5 (2005), 550–561.

Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. 2019. Neural Volumes: Learning Dynamic Renderable Volumes from Images. *ACM Trans. Graph.* 38, 4, Article 65 (jul 2019), 14 pages.

Zander Majercik, Jean-Philippe Guertin, Derek Nowrouzezahrai, and Morgan McGuire. 2019. Dynamic Diffuse Global Illumination with Ray-Traced Irradiance Fields. *Journal of Computer Graphics Techniques (JCGT)* 8, 2 (5 June 2019), 1–30. http://jcgt.org/published/0008/02/01/

Zander Majercik, Adam Marrs, Josef Spjut, and Morgan McGuire. 2021. Scaling Probe-Based Real-Time Dynamic Global Illumination for Production. *Journal of Computer Graphics Techniques (JCGT)* 10, 2 (3 May 2021), 1–29. http://jcgt.org/published/0010/02/01/

Julio Marco, Adrian Jarabo, Wojciech Jarosz, and Diego Gutierrez. 2018. Second-Order Occlusion-Aware Volumetric Radiance Caching. *ACM Trans. Graph.* 37, 2, Article 20 (jul 2018), 14 pages.

Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. 2021. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *CVPR*.

Morgan McGuire, Mike Mara, Derek Nowrouzezahrai, and David Luebke. 2017. Real-Time Global Illumination Using Precomputed Light Field Probes. In *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (San Francisco, California) *(I3D '17)*. Association for Computing Machinery, New York, NY, USA, Article 2, 11 pages.

Xiaoxu Meng, Quan Zheng, Amitabh Varshney, Gurprit Singh, and Matthias Zwicker. 2020. Real-time Monte Carlo Denoising with the Neural Bilateral Grid. In *Eurographics Symposium on Rendering - DL-only Track*, Carsten Dachsbacher and Matt Pharr (Eds.). The Eurographics Association.

Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. 2019. Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines. *ACM Trans. Graph.* 38, 4, Article 29 (jul 2019), 14 pages.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *Computer Vision – ECCV 2020*. Springer International Publishing, Cham, 405–421.

Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. 2021. Real-Time Neural Radiance Caching for Path Tracing. *ACM Trans. Graph.* 40, 4, Article 36 (jul 2021), 16 pages.

O. Nalbach, E. Arabadzhiyska, D. Mehta, H.-P. Seidel, and T. Ritschel. 2017. Deep Shading: Convolutional Neural Networks for Screen Space Shading. *Computer Graphics Forum* 36, 4 (2017), 65–78.

Eyal Ofek and Ari Rappoport. 1998. Interactive Reflections on Curved Objects. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98)*. Association for Computing Machinery, New York, NY, USA, 333–342.

Yaobin Ouyang, Shiqiu Liu, Markus Kettunen, Matt Pharr, and Jacopo Pantaleoni. 2021. ReSTIR GI: Path Resampling for Real-Time Path Tracing. *Computer Graphics Forum* (2021).

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2337–2346.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc.

Julien Philip, Michaël Gharbi, Tinghui Zhou, Alexei Efros, and George Drettakis. 2019. Multi-view Relighting Using a Geometry-Aware Network. *ACM Transactions on Graphics (SIGGRAPH Conference Proceedings)* 38, 4 (July 2019).

Ravi Ramamoorthi. 2009. Precomputation-Based Rendering. *Found. Trends. Comput. Graph. Vis.* 3, 4 (apr 2009), 281–369.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. 2021. Vision Transformers for Dense Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 12179–12188.

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, Article 130 (jul 2013), 12 pages.

Austin Robison and Peter Shirley. 2009. Image space gathering. In *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 Trans. Graph.* 39, 6, Article 237 (nov 2020), 16 pages.

David Roger and Nicolas Holzschuch. 2006. Accurate Specular Reflections in Real-Time. *Computer Graphics Forum* 25, 3 (Sept. 2006), 293–302. https://hal.inria.fr/inria-

00379310
Matt Sandy, Johan Andersson, and Colin Barré-Brisebois. 2018. DirectX: Evolving Microsoft's Graphics Platform. Game Developers Conference 2018.
Daniel Scherzer, Lei Yang, Oliver Mattausch, Diego Nehab, Pedro V. Sander, Michael Wimmer, and Elmar Eisemann. 2012. Temporal Coherence Methods in Real-Time Rendering. *Comput. Graph. Forum* 31, 8 (dec 2012), 2378–2408.
Christoph Schied, Anton Kaplanyan, Chris Wyman, Anjul Patney, Chakravarty R. Alla Chaitanya, John Burgess, Shiqiu Liu, Carsten Dachsbacher, Aaron Lefohn, and Marco Salvi. 2017. Spatiotemporal Variance-Guided Filtering: Real-Time Reconstruction for Path-Traced Global Illumination. In *Proceedings of High Performance Graphics (HPG '17)*. Association for Computing Machinery, New York, NY, USA, Article 2, 12 pages.
Dario Seyb, Peter-Pike Sloan, Ari Silvennoinen, Michał Iwanicki, and Wojciech Jarosz. 2020. The Design and Evolution of the UberBake Light Baking System. *ACM Trans. Graph.* 39, 4, Article 150 (jul 2020), 13 pages.
Ari Silvennoinen and Jaakko Lehtinen. 2017. Real-Time Global Illumination by Precomputed Local Reconstruction from Sparse Radiance Probes. *ACM Trans. Graph.* 36, 6, Article 230 (nov 2017), 13 pages.
Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).
Sudipta N. Sinha, Drew Steedly, and Richard Szeliski. 2009. Piecewise planar stereo for image-based rendering. In *2009 IEEE 12th International Conference on Computer Vision*. 1881–1888.
Vincent Sitzmann, Michael Zollhoefer, and Gordon Wetzstein. 2019. Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations. In *Advances in Neural Information Processing Systems*, Vol. 32. Curran Associates, Inc.
Peter-Pike Sloan, Jan Kautz, and John Snyder. 2002. Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments. *ACM Trans. Graph.* 21, 3 (jul 2002), 527–536.
László Szirmay-Kalos, Barnabás Aszódi, István Lazányi, and Mátyás Premecz. 2005. Approximate Ray-Tracing on the GPU with Distance Impostors. *Computer Graphics Forum* 24, 3 (2005), 695–704.
László Szirmay-Kalos, Tamás Umenhoffer, Gustavo Patow, László Szécsi, and Mateu Sbert. 2009. Specular Effects on the GPU: State of the Art. *Computer Graphics Forum* 28, 6 (2009), 1586–1617.
Natalya Tatarchuk. 2005. Irradiance Volumes for Games. Game Developers Conference.
A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. Nießner, R. Pandey, S. Fanello, G. Wetzstein, J.-Y. Zhu, C. Theobalt, M. Agrawala, E. Shechtman, D. B Goldman, and M. Zollhöfer. 2020. State of the Art on Neural Rendering. *Computer Graphics Forum* 39, 2 (2020), 701–727.
Justus Thies, Michael Zollhöfer, and Matthias Nießner. 2019. Deferred Neural Rendering: Image Synthesis Using Neural Textures. *ACM Trans. Graph.* 38, 4, Article 66 (jul 2019), 12 pages.
Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. 2018. Deep Image Prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
Kostas Vardis, Georgios Papaioannou, and Anastasios Gkaravelis. 2014. Real-time Radiance Caching using Chrominance Compression. *Journal of Computer Graphics Techniques (JCGT)* 3, 4 (16 December 2014), 111–131.
Konstantinos Vardis, Andreas Alexandros Vasilakis, and Georgios Papaioannou. 2021. Illumination-driven Light Probe Placement. In *Eurographics 2021 Posters*, Jirí Bittner and Manuela Waldner (Eds.). The Eurographics Association.
Thijs Vogels, Fabrice Rousselle, Brian Mcwilliams, Gerhard Röthlin, Alex Harvill, David Adler, Mark Meyer, and Jan Novák. 2018. Denoising with Kernel Prediction and Asymmetric Loss Functions. *ACM Trans. Graph.* 37, 4 (July 2018), 124:1–124:15.
Yue Wang, Soufiane Khiat, Paul G. Kry, and Derek Nowrouzezahrai. 2019. Fast Non-Uniform Radiance Probe Placement and Tracing. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (Montreal, Quebec, Canada) *(I3D '19)*. Association for Computing Machinery, New York, NY, USA, Article 5, 9 pages.
Daniel N. Wood, Daniel I. Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H. Salesin, and Werner Stuetzle. 2000. Surface Light Fields for 3D Photography. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00)*. ACM Press/Addison-Wesley Publishing Co., USA, 287–296.
Lei Xiao, Salah Nouri, Matt Chapman, Alexander Fix, Douglas Lanman, and Anton Kaplanyan. 2020. Neural Supersampling for Real-Time Rendering. *ACM Trans. Graph.* 39, 4, Article 142 (jul 2020), 12 pages.
Kun Xu, Yan-Pei Cao, Li-Qian Ma, Zhao Dong, Rui Wang, and Shi-Min Hu. 2014. A Practical Algorithm for Rendering Interreflections with All-Frequency BRDFs. *ACM Trans. Graph.* 33, 1, Article 10 (feb 2014), 16 pages.
Zexiang Xu, Sai Bi, Kalyan Sunkavalli, Sunil Hadap, Hao Su, and Ravi Ramamoorthi. 2019. Deep View Synthesis from Sparse Photometric Images. *ACM Trans. Graph.* 38, 4, Article 76 (jul 2019), 13 pages.
Zexiang Xu, Kalyan Sunkavalli, Sunil Hadap, and Ravi Ramamoorthi. 2018. Deep image-based relighting from optimal sparse samples. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 126.
Lei Yang, Shiqiu Liu, and Marco Salvi. 2020. A Survey of Temporal Antialiasing Techniques. *Computer Graphics Forum* (2020).
Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 586–595.
Xiuming Zhang, Sean Fanello, Yun-Ta Tsai, Tiancheng Sun, Tianfan Xue, Rohit Pandey, Sergio Orts-Escolano, Philip Davidson, Christoph Rhemann, Paul Debevec, Jonathan T. Barron, Ravi Ramamoorthi, and William T. Freeman. 2021a. Neural Light Transport for Relighting and View Synthesis. *ACM Trans. Graph.* 40, 1, Article 9 (jan 2021), 17 pages.
Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. 2021b. NeRFactor: Neural Factorization of Shape and Reflectance under an Unknown Illumination. *ACM Trans. Graph.* 40, 6, Article 237 (dec 2021), 18 pages.
Yangyang Zhao, Laurent Belcour, and Derek Nowrouzezahrai. 2019. View-Dependent Radiance Caching. In *Proceedings of the 45th Graphics Interface Conference on Proceedings of Graphics Interface 2019* (Kingston, Canada) *(GI'19)*. Canadian Human-Computer Communications Society, Waterloo, CAN, Article 22, 9 pages.
Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. 2018a. Stereo Magnification: Learning View Synthesis Using Multiplane Images. *ACM Trans. Graph.* 37, 4, Article 65 (jul 2018), 12 pages.
Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. 2018b. Non-stationary Texture Synthesis by Adversarial Expansion. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 37, 4 (2018), 49:1–49:13.
Tao Zhuang, Pengfei Shen, Beibei Wang, and Ligang Liu. 2021. Real-time Denoising Using BRDF Pre-integration Factorization. *Computer Graphics Forum* 40, 7 (2021), 173–180.