Photon-Driven Neural Reconstruction for Path Guiding

SHILIN ZHU, University of California San Diego, USA
ZEXIANG XU, Adobe Research, USA
TIANCHENG SUN, University of California San Diego, USA
ALEXANDR KUZNETSOV, University of California San Diego, USA
MARK MEYER, Pixar Animation Studios, USA
HENRIK WANN JENSEN, University of California San Diego and Luxion, USA
HAO SU, University of California San Diego, USA
RAVI RAMAMOORTHI, University of California San Diego, USA

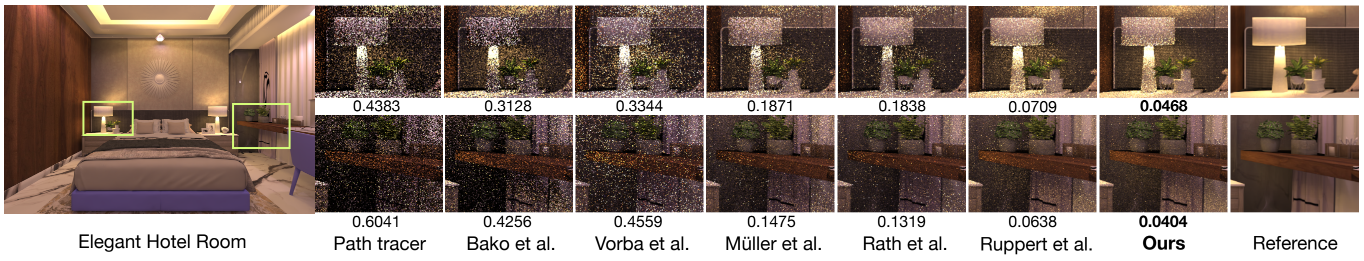

Fig. 1. We present a novel photon-driven neural path guiding approach that can effectively reduce the variance in path tracing. This complex scene is lit by several decorative lights which are very difficult to discover in path tracing. We compare the equal-time (~20 minutes) rendering results with standard path tracing and state-of-the-art path-guiding methods (including Müller et al. [2017], Bako et al. [2019], Rath et al. [2020], and Ruppert et al. [2020]), showing the crops (illuminated mostly by the *lamp lights*) of the rendered results with corresponding relative MSEs (rMSEs). Bako et al. [2019] use an offline trained neural network for path guiding; however, it only supports guiding the first bounce, which is not very effective since this scene is dominated by indirect lighting. While traditional methods allow for multi-bounce path guiding, they are mostly online learning methods and they need longer time to learn the complex sampling functions for this challenging scene. Our method utilizes a trained deep neural network and enables effective path guiding at any path bounces.

Although Monte Carlo path tracing is a simple and effective algorithm to synthesize photo-realistic images, it is often very slow to converge to noise-free results when involving complex global illumination. One of the most successful variance-reduction techniques is path guiding, which can learn better distributions for importance sampling to reduce pixel noise. However, previous methods require a large number of path samples to achieve reliable path guiding. We present a novel neural path guiding approach that can reconstruct high-quality sampling distributions for path guiding from a sparse set of samples, using an offline trained neural network. We leverage photons traced from light sources as the primary input for sampling density reconstruction, which is effective for challenging scenes with strong global illumination. To fully make use of our deep neural network, we partition the scene space into an adaptive hierarchical grid, in which we apply our network to reconstruct high-quality sampling distributions for any local region in the scene. This allows for effective path guiding for arbitrary path bounce at any location in path tracing. We demonstrate that our photon-driven neural path guiding approach can generalize to diverse testing scenes, often achieving better rendering results than previous path guiding approaches and opening up interesting future directions.

CCS Concepts: • **Computing methodologies** → **Ray tracing**.

Additional Key Words and Phrases: Global Illumination, Path Guiding, Ray Tracing, Sampling and Reconstruction, Neural Rendering

ACM Reference Format:
Shilin Zhu, Zexiang Xu, Tiancheng Sun, Alexandr Kuznetsov, Mark Meyer, Henrik Wann Jensen, Hao Su, and Ravi Ramamoorthi. 2021. Photon-Driven Neural Reconstruction for Path Guiding. *ACM Trans. Graph.* 1, 1, Article 1 (January 2021), 15 pages. https://doi.org/10.1145/3476828

Authors' addresses: Shilin Zhu, University of California San Diego, USA, shz338@eng.ucsd.edu; Zexiang Xu, Adobe Research, USA, zexu@adobe.com; Tiancheng Sun, University of California San Diego, USA, tis037@eng.ucsd.edu; Alexandr Kuznetsov, University of California San Diego, USA, a1kuznet@eng.ucsd.edu; Mark Meyer, Pixar Animation Studios, USA, mmeyer@pixar.com; Henrik Wann Jensen, University of California San Diego and Luxion, USA, henrik@cs.ucsd.edu; Hao Su, University of California San Diego, USA, haosu@eng.ucsd.edu; Ravi Ramamoorthi, University of California San Diego, USA, ravir@cs.ucsd.edu.

1 INTRODUCTION

Monte Carlo path tracing has been widely used in photo-realistic image synthesis. However, while simple and flexible, path tracing can take a significant amount of time to generate noise-free images for complex scenes (e.g., Fig. 1). One critical challenge for Monte Carlo based methods is to effectively construct light transport paths connecting the light and the camera.

Many path-guiding methods [Müller et al. 2017; Jensen 1995] have been presented to construct advanced distributions (usually approximating incident light fields or some variants of those) for importance sampling at local shading points, guiding the local path sampling for high-energy path construction. The recent successful methods are based on unidirectional guiding [Müller et al. 2017; Rath et al. 2020; Ruppert et al. 2020]; they rely on early path samples to discover high-energy sampling directions. Although the strategy is generally effective, this unidirectional path discovery process can be still slow for a challenging scene that is dominated by indirect illumination. While using light paths is known to be efficient in exploring the path space, previous photon-driven or bidirectional path-guiding methods [Jensen 1995; Vorba et al. 2014] are not yet efficient, requiring sampling a large number of light paths.

We present a novel path guiding approach that can achieve effective path sampling using *a sparse set* of light paths as input, thus successfully advancing the overall rendering speed. Inspired by the original path guiding work [Jensen 1995], we leverage photons to compute local sampling distributions for importance sampling in path tracing, where a sampling distribution at any 3D local region can be obtained by binning local photons according to their directions (i.e., a 2D histogram map). However, such distributions are only reliable with locally dense photons, and are usually low-quality and appear noisy with sparse photons (Figs. 2 and 3).

We propose to use a compact neural network to reconstruct high-quality sampling distributions for path guiding from low-quality noisy histograms that are acquired by binning sparse photons. In essence, we break down the complex path guiding problem, mainly focusing on reconstructing local sampling distributions represented as 2D maps (i.e., images), and thus pose this problem as one of the image-to-image translations that can now be addressed by deep learning techniques. Our neural reconstruction network is effectively trained offline in a scene-independent way, and can recover the shapes of complex sampling distributions on new scenes, enabling guided path tracing with complex global illumination effects.

Our framework is designed to reconstruct high-quality sampling maps at local spatial regions. To make these sampling maps well distributed and locally representative in the scene space, we adaptively partition the entire scene space into a hierarchical grid, according to the complexity of local incident light variations. The neurally reconstructed sampling maps are cached in leaf voxels of the grid, enabling path guiding at different locations in a scene. Therefore, we can support guiding path tracing at multiple bounces. Although our approach also has specific limitations (e.g., reconstructing only low-resolution sampling maps because of memory limit, experiencing uneven photon visibility), we demonstrate that our novel learning-based path guiding often achieves better rendering quality on various challenging scenes than previous state-of-the-art path-guiding methods when GPU resources are available (Figs. 1, 6, and 7). The proposed reconstruction framework serves as a starting point for many extension possibilities. In summary, our main contributions are:

- We present a learning-based approach that leverages photons to reconstruct high-quality sampling distributions locally;
- By combining with an adaptive spatial caching structure, we support building and reusing better sampling distributions at arbitrary bounces to reduce variance of path tracing results.

2 RELATED WORK

Monte Carlo rendering. One central problem of computer graphics is to efficiently evaluate the rendering equation [Kajiya 1986], which describes how light transports globally inside a scene. Monte Carlo methods are among the most effective methods to compute the light transport, which require sampling high-energy paths that connect the camera and light for efficient rendering. Since Monte Carlo path tracing was introduced in the seminal work by Kajiya [1986], numerous work have developed more efficient methods to explore path space, including bidirectional path tracing [Lafortune and Willems 1993; Veach and Guibas 1995a] and metropolis light transport [Veach and Guibas 1997; Pauly et al. 2000]. These methods typically leverage importance sampling to sample sub-path directions at any bounces for each traced path traversing the scene. Since the incident illumination is unknown, the importance sampling usually only considers the reflectance term (with a cosine term) in the rendering equation (more details in Sec. 3); this however is inefficient for challenging scenes with complex indirect lighting. Previous work [Jiang and Kainz 2021] have explored directly reconstructing an approximation of the radiance field. Path guiding [Jensen 1995; Vorba et al. 2019] can instead provide more efficient importance sampling without introducing any bias. Recent guiding work [Reibold et al. 2018] can further selectively adapts to sampling problematic regions and complements the unguided strategy. Our novel photon-driven path guiding approach can also reconstruct high-quality sampling distributions that well approximate the complex incident light fields, thus leading to faster rendering.

Photon-based rendering. Particle density estimation has also been applied in computer graphics to evaluate the rendering equation, which introduces photon mapping and many other particle- or photon- based rendering methods [Shirley et al. 1995; Jensen 1996; Hachisuka et al. 2008; Knaus and Zwicker 2011]. These methods focus on photon density estimation at any given shading point, which avoids the high-frequency noise in MC rendering and is effective for computing complex global illumination. However, photon density estimation can only provide biased radiance estimates, since it blurs the photon contributions within a certain kernel bandwidth (though the bias can vanish to zero by progressive photon mapping approaches [Hachisuka et al. 2008; Hachisuka and Jensen 2009; Knaus and Zwicker 2011]). Our goal is not to compute photon density but to approximate incident light field for a local region (in a 3D voxel) as sampling distributions. Therefore, we consider the integral of irradiance over the surface area (i.e., the incident flux), which can be effectively evaluated using photons in an unbiased way.

Recently, Zhu et al. [2020] introduce a deep learning based method for photon density estimation in photon mapping. They leverage a PointNet [Qi et al. 2017] style neural network to process individual photons. However, the complexity grows linearly with the number of photons. We instead leverage a CNN [Ronneberger et al. 2015] style network and consider a noisy photon histogram map, composed by binning photons [Jensen 1995] as input; therefore, our complexity is independent to the photon count and runs in constant time, and we can reconstruct better sampling distributions with more input photons.

Path guiding. In general, path guiding aims to estimate the incoming light fields and draw samples accordingly to accelerate the convergence of Monte-Carlo rendering. The traditional path guiding technique is based on photons [Jensen 1995]; it traces light paths from the light sources, distributes photons in the scene, and constructs local photon histograms as sampling distributions for the importance sampling in path tracing. Though very simple to compute, such histogram-based sampling maps are only of high quality when accumulating dense enough photons. We extend this classical histogram-based technique to a new learning-based path guiding framework; our method regresses high-quality sampling maps from sparse photons, avoiding expensively tracing a large number of photons (though still supported).

Vorba et al. [2014] present a bidirectional guiding method, where both camera paths and light paths are guided using online fitted parametric distributions of Gaussian mixtures at spatial cache points. This technique was further extended to product sampling [Herholz et al. 2016]. Recently, [Ruppert et al. 2020] propose to account for parallax-awareness in parametric distribution modeling, leading to state-of-the-art guiding results. However, such an online fitting process can sometimes be unstable, thus refining mixture components is necessary to ensure robustness. Also, the mixture model makes it difficult to express complex incoming lights. Our approach leverages histograms as input (easily computed online) and an offline trained compact neural network to accelerate reconstructing distributions with high-frequency details, although currently we do not have flexibility to handle the parallax which is left for future work.

Recently, unidirectional guiding methods have become more effective and practical, thanks to the efficient adaptive caching structure introduced by Müller et al. [2017]. Many works extend this framework to achieve sampling in primary space [Guo et al. 2018], product sampling [Diolatzis et al. 2020], and variance-aware sampling [Rath et al. 2020]. These methods iteratively trace camera paths to adaptively reconstruct the incident light fields; this relies on early iteration paths to discover the light sources, in order to reconstruct reliable sampling distributions to guide the following iteration paths. However, the light discovery can be slow for a scene with dominant indirect lighting, and outliers in the early-iteration sampling distributions can bias the path sampling in later iterations and become hard to get fixed. In contrast, we leverage photons that are efficient in exploring indirect light transport; our learning based approach can recover high-quality sampling distributions from sparse photons at an early stage, effectively relieving the problem of a slow start in the guiding and rendering. Moreover, our photons are traced independently in each iteration, which does not accumulate the sampling errors through multiple iterations. Also, we are aware that path tracing is still the prevalent rendering algorithm due to its simplicity and flexibility compared to bidirectional methods, but the merit from using photons cannot be overlooked and ours can be an alternate guiding approach when unidirectional tracing fails.

Neural path guiding. Recently, deep neural networks have been applied to path guiding. Müller et al. [2019] train an online neural network to perform importance sampling directly through neural inference. This method can reproduce accurate ground-truth sampling functions, but using a neural network for direct sampling can be relatively expensive because of the repeated online inference and optimization steps. Some recent works leverage offline trained neural networks [Bako et al. 2019; Huo et al. 2020]; however, they only guide the path sampling at the primary bounce. While we also leverage a deep neural network, our method instead leverages photons and supports guiding at any bounces, enabling better rendering results than the first-bounce guiding approach [Bako et al. 2019] in complex scenes.

3 BACKGROUND

Physically-based rendering can be expressed by the Rendering Equation [Kajiya 1986] that describes the radiance leaving an intersection point $\boldsymbol{x}$ in direction ω_o:

$$L(\boldsymbol{x},\omega_o) = L_e(\boldsymbol{x},\omega_o) + \int_\Omega L_i(\boldsymbol{x},\omega_i) f_r(\boldsymbol{x},\omega_i,\omega_o)\cos\theta_i d\omega_i, \quad (1)$$

where $L_e(\boldsymbol{x},\omega_o)$ denotes the emitted radiance, $L_i(\boldsymbol{x},\omega_i)$ is the incident radiance from direction ω_i, f_r is the bidirectional scattering distribution function (BSDF), and Ω corresponds to the full sphere. The key component in the equation is the integral that computes the reflected radiance $L_r(\boldsymbol{x},\omega_o) = \int_\Omega L_i(\boldsymbol{x},\omega_i) f_r(\boldsymbol{x},\omega_i,\omega_o)\cos\theta_i d\omega_i$ over all directions in the sphere.

The integral can be numerically evaluated using Monte Carlo estimation [Veach 1997]:

$$L_r(\boldsymbol{x},\omega_o) = \frac{1}{N}\sum_{i=1}^{N} \frac{L_i(\boldsymbol{x},\omega_i) f_r(\boldsymbol{x},\omega_i,\omega_o)\cos\theta_i}{p(\omega_i)} \quad (2)$$

where N Monte Carlo path samples in various directions ω_i are drawn from the probability density function (PDF) $p(\omega_i)$. Considering global illumination with multiple bounces, $L_i(\boldsymbol{x},\omega_i)$ is computed by recursively evaluating integrals using Eqn. 1. In path tracing, rays are sampled from each intersection point to compute the radiance that contributes to the pixel color at multiple bounces.

The variance of the Monte Carlo estimate $L_r(\boldsymbol{x},\omega_o)$ can be reduced by sampling ω_i from a density function $p(\omega_i)$ that resembles the numerator $L_i(\boldsymbol{x},\omega_i) f_r(\boldsymbol{x},\omega_i,\omega_o)\cos\theta_i$. Ideally, if $p(\omega_i)$ and the numerator only differ by a constant scale, the variance is reduced to zero. However, this numerator is unknown and is as difficult as the integral to compute, due to complex visibility and indirect lighting in L_i; therefore, standard path tracing often proceeds with BSDF importance sampling for indirect lighting plus a direct light sampling technique (e.g., next-event estimation).

Path guiding aims to reconstruct a density function that matches the shape of the numerator as closely as possible. In particular, since the standard BSDF importance sampling satisfies [Veach 1997]:

$$p_{\mathrm{BSDF}}(\omega_i) \propto f_r(\boldsymbol{x},\omega_i,\omega_o) \quad (3)$$

recent path-guiding methods often set the target probability density to be proportional to the incident light [Vorba et al. 2014; Müller et al. 2017; Ruppert et al. 2020] (the following cosine term is sometimes included in BSDF sampling in Equ. 3 instead of guiding):

$$p_{\mathrm{guide}}(\omega_i) \propto L_i(\boldsymbol{x},\omega_i)\cos\theta_i. \quad (4)$$

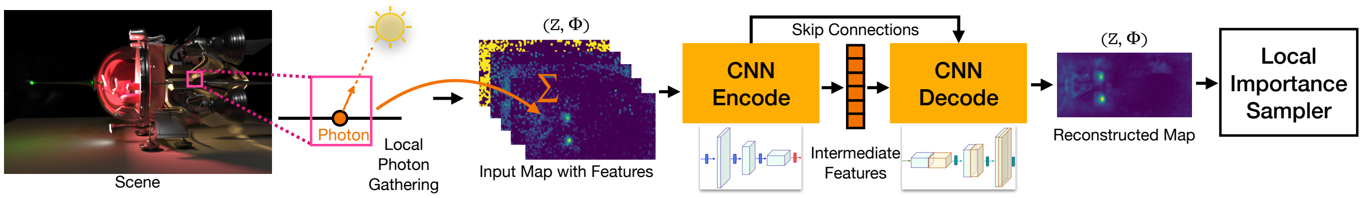

Fig. 2. Illustration of the entire system workflow. We partition the scene into multiple local regions using a spatial structure, where each voxel gathers neighboring photons locally. The gathered photon statistics are accumulated in the directional space represented by a noisy histogram map with additional features, and photons are deleted after such splatting is completed during particle tracing. Next, we use a pre-trained compact CNN to encode the input map to neural features, followed by a decoder to reconstruct a target sampling map. The output map is reused by a local importance sampler to decide the next sampled direction at any bounce in path tracing.

The final sampling strategy is achieved by combining the guiding and BSDF sampling using either the product sampling (i.e., $p_{\text{guide}}(\omega_i) \cdot p_{\text{BSDF}}(\omega_i)$ [Herholz et al. 2016]) or the one-sample Multiple Importance Sampling (MIS): [Veach and Guibas 1995b]

$$p(\omega_i) = \alpha p_{\text{BSDF}}(\omega_i) + (1-\alpha) p_{\text{guide}}(\omega_i), \tag{5}$$

where α is the mixture coefficient that determines the probability of choosing BSDF sampling or guided sampling.

Many recent works rely on early path samples in path tracing to approximate the incident light field (Eqn. 4), which is insufficient for challenging scenes with strong indirect lighting (Fig. 1). We instead leverage photons traced from the lights to compute the sampling density functions, which effectively explores the challenging light transport. Our novel approach advances the traditional path guiding with powerful deep learning techniques and an adaptive spatial structure, enabling effective path guiding from sparse photons.

4 OVERVIEW

Our path guiding approach uses a compact pre-trained neural network to regress high-quality sampling maps that can be used to guide path sampling. Meanwhile, we utilize an adaptive hierarchical grid for spatially storing the reconstructed distributions, enabling effective path guiding at multiple bounces. The whole system is illustrated in Fig. 2.

In the following sections, we first introduce our sampling map parameterization, target sampling density, and how to use photons to compute the histograms in Sec. 5. We then introduce our deep neural network that can regress better sampling maps given noisy low-quality histograms in Sec. 6. We present our full path guiding framework in Sec. 7, which describes our iterative sample gathering and rendering process, adaptive spatial structure, and how paths, photons, and the neural network are incorporated in the system.

5 COMPUTING SAMPLING MAPS FROM PHOTONS

Previous methods [Jensen 1995; Vorba et al. 2014] usually compute hemispherical distributions at sampled surface points to approximate incident light fields. However, such hemispherical functions approximate light fields at locally flat 2D surface regions, and are hard to interpolate on surfaces with complex normal variations. Inspired by the recent unidirectional path-guiding methods [Müller et al. 2017; Rath et al. 2020; Bako et al. 2019], we utilize a full spherical sampling distribution that models the incident light distribution in a local 3D region. In particular, we build a hierarchical grid (Sec. 7.1) in the scene space, and compute a spherical sampling distribution stored in a discrete data structure for each local 3D voxel of the adaptive grid. In this section, we discuss the representation of our sampling function and the computation of it from photons during particle tracing from light sources.

Spherical function representation. We use a regular directional grid that represents the sampling density function as a 2D sampling map (similar to [Bako et al. 2019]). We leverage the cylindrical mapping to parameterize the spherical domain for better area preservation (similar to [Müller et al. 2017; Rath et al. 2020]). In particular, a vector $r = (x, y, z)$ on a unit sphere is mapped to a 2D location $(u, v) = (z, \phi)$ on the sampling map, where $\phi = \arctan(y/x)$. This sampling map is similar to a standard environment map or radiance map in traditional lighting representation, but ours is monochromatic and uses cylindrical parameterization.

Target sampling density. As discussed in Sec. 3 (Eqn. 4), the goal of path guiding is to compute the sampling density at any position, making it proportional to the incident light $L_i(\boldsymbol{x}, \omega_i)$ or $L_i(\boldsymbol{x}, \omega_i) \cos\theta_i$. For our discrete case where we consider a 3D voxel region and a certain footprint (representing a solid angle bin) of a sampling map, it is in fact the expected incident light that is of our interest. In particular, given a voxel j and a solid angle footprint $\Delta\Omega_k$ of pixel k in the sampling map, the expected $L_i(\boldsymbol{x}, \omega_i) \cos\theta_i$ coming from the solid angle over the local surface area ΔA_j (that is of the scene geometry covered by the voxel) is expressed by:

$$\mathbb{E}[L_i(\boldsymbol{x}, \omega_i)\cos\theta_i] = \frac{\int_{\Delta A_j}\int_{\Delta\Omega_k} L_i(\boldsymbol{x}, \omega_i)\cos\theta_i d\omega_i d\boldsymbol{x}}{\Delta\Omega_k \Delta A_j} \tag{6}$$

$$= \frac{\Phi_{j,k}}{\Delta\Omega_k \Delta A_j}, \tag{7}$$

where $\Phi_{j,k}$ represents the total incident power in the spatial and directional range. Therefore, it is the total power (radiant flux)

$$\Phi_{j,k} = \int_{\Delta A_j}\int_{\Delta\Omega_k} L_i(\boldsymbol{x}, \omega_i)\cos\theta_i d\omega_i d\boldsymbol{x}, \tag{8}$$

that governs our sampling map distribution. Essentially, $\Phi_{j,k}$ models the integrated incident radiance. Note that the irradiance ($E(\boldsymbol{x}, \Delta\Omega_k) = \int_{\Delta\Omega_k} L_i(\boldsymbol{x}, \omega_i)\cos\theta_i d\omega_i$) is a standard radiometry term and widely used in previous works [Jensen 1995; Rath et al. 2020]; when divided by the surface area, $\Phi_{j,k}$ also describes the expected irradiance

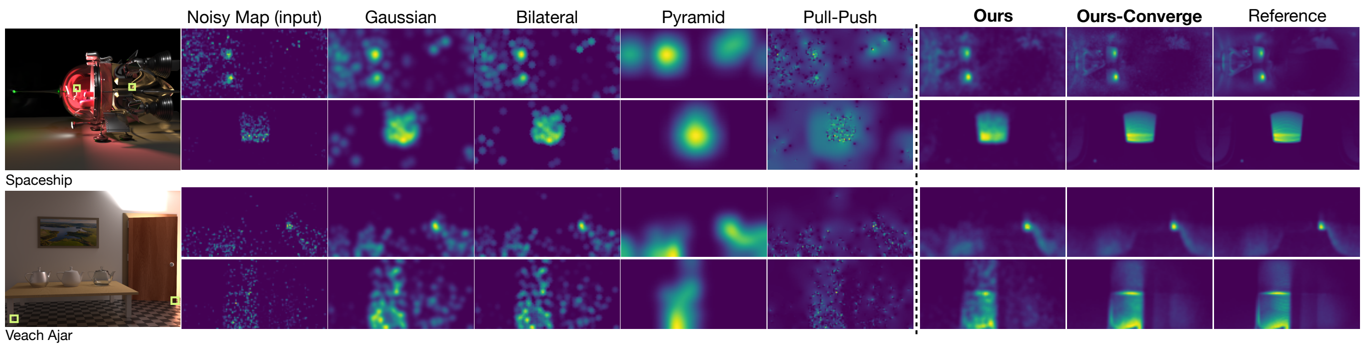

Fig. 3. Example reconstructed sampling maps (gamma transformed for better visualization purpose). With more iterations of path and photon tracing, the reconstructed sampling map can converge to the reference (after 8 iterations in this example). We also compare to other traditional image interpolation techniques (3 hierarchical levels for pyramid and pull-push filters). Although they can also improve the quality of the noisy histogram maps through blurring in general, our neural-based reconstruction is designed specifically for this task thus can produce maps closer to the targets.

$(\Phi_{j,k}/\Delta A_j)$ in the voxel. Therefore, we seek to obtain sampling densities that are proportional to the expected incident light:

$$p_{\text{guide}}(\omega_i) \propto \Phi_{j,k}/\Delta\Omega_k, \tag{9}$$

where we ignore the ΔA_j in Eqn. 7 since it is constant for all solid angles in the same voxel. This sampling density corresponds to a sampling map, each pixel value of which is proportional to $\Phi_{j,k}$. We thus reconstruct a sampling map by normalizing a power map that records the power $\Phi_{j,k}$ in each pixel.

Computing incident power. In this work, we leverage particle tracing to effectively evaluate the $\Phi_{j,k}$ (Eqn. 8). We trace light paths from the light sources to distribute photons in the scene, where each photon carries a portion of flux; $\Phi_{j,k}$ can then be estimated by simply binning the photons similar to [Jensen 1995]:

$$\Phi_{j,k} = \sum_{\omega_p \in \Delta\Omega_k, \boldsymbol{x}_p \in \Delta A_j} \Delta\Phi_p, \tag{10}$$

where p denotes a photon arriving at the surface point $\boldsymbol{x}_p$ from direction ω_p, and $\Delta\Phi_p$ is the power carried by it. Equation 10 essentially accumulates all the photon power inside a voxel and directional bin.

Note that Müller et al. [2017] leverages path tracing to accumulate the radiance samples inside a local voxel; this can be seen as an integral of the radiance over an area and a solid angle, similar to our power expression Eqn. 8. Our particle-based approach provides an unbiased estimate for the power integral $\Phi_{j,k}$ when the photon count goes to infinity.

Since the evaluation is governed by accumulating splatted values to a histogram map, we can progressively trace as many photons as needed without storing the entire photon point cloud (required by traditional photon mapping, leading to memory bottleneck from more traced particles). Once a photon is accumulated to a directional bin of the map inside a voxel, it is then deleted, except for the initialization phase (Sec. 7.1). Note that an accurate power map requires tracing a large number of photons, but in practice, we can only allow for tracing a small number of photons at rendering time, which by themselves cannot directly lead to high-quality sampling.

6 HIGH-QUALITY DISTRIBUTION RECONSTRUCTION

If directly computing sampling distributions by binning photons, neither using dense photons (slow) nor sparse photons (low-quality) is suitable for efficient path guiding. To overcome this, our central idea is to obtain accurate sampling maps offline as ground truth using dense photons, and leverage supervised learning to regress such maps from low-quality histograms that can be computed efficiently from sparse photons. Specifically, we propose to train a deep Convolutional Neural Network (CNN) that learns to reconstruct a high-quality sampling distribution from sparse photons.

Our sampling maps are reconstructed and updated repeatedly through multiple iterations in our path guiding framework (Sec. 7). We consider a normalized noisy sampling map S_t as input, acquired by accumulating a sparse set of photons from iteration 1 to t using Eqn. 10, where t denotes the iteration number. We also supply the noisy sampling map S_{t-1} from the previous iteration to ease the learning of where to in-paint. In addition, we record the number of photons per solid angle bin in S_t and S_{t-1}, resulting in maps P_t and P_{t-1}, and include the normalized buffers in the input. Inspired by the image inpainting techniques [Liu et al. 2018; Yu et al. 2019; Yi et al. 2020], we also concatenate a binary mask B_t indicating whether a solid angle bin contains photon data or not, and use light-weight masked convolutions to process the input maps. As a result, our full input is an image map with 5 separately-normalized (by the GPU) channels and our neural network $\mathbb{F}$ can be expressed by:

$$S_d = \mathbb{F}(S_t, S_{t-1}, P_t, P_{t-1}, B_t). \tag{11}$$

The output is a one-channel sampling map S_d (which is then converted to CDF for importance sampling), supervised by the ground-truth map $\tilde{S_d}$ computed from dense enough photons.

6.1 Neural architecture and loss computation

Our network is essentially designed to solve an image-to-image reconstruction task. Many existing 2D neural networks for image-to-image denoising, translation, and inpainting ([Chaitanya et al. 2017; Bako et al. 2017; Vogels et al. 2018; Liu et al. 2018]) can thus be potentially applied to address the problem. However, our neural network is applied on a large number of voxels, while our end goal is to speed up the overall rendering process. Therefore, we balance the inference speed and reconstruction quality in the network design.

We propose to use a compact U-Net [Ronneberger et al. 2015] style architecture with residual links and skip connections to achieve the sampling map reconstruction as illustrated in Fig. 2 (the detailed architecture is provided in supplementary material). It contains multiple downsampling and upsampling convolutional layers to extract high-level neural features from the input map S_t and output a better version S_d. The input noisy sampling maps are computed from sparse photons, which contain many empty bins. Therefore, we use the light-weight masked convolutions inspired by the recent image inpainting works [Liu et al. 2018; Yi et al. 2020], which ensures that valid (non-empty) and invalid (empty) bins are treated differently and only valid bins can contribute to convolutions. Note that the designed architecture is relatively compact, compared to the previous deep networks ([Chaitanya et al. 2017; Bako et al. 2017; Vogels et al. 2018]) used in denoising. Although the extra computational overhead introduced by the neural network is inevitable, the compactness allows sampling map reconstruction to finish in reasonable time on powerful GPUs. Due to limited system memory, we reconstruct low-resolution maps (64×128 or 32×64), which are already adequate for path guiding in common scenes. We believe our architecture can be further improved by advanced compression techniques [Cheng et al. 2018; Deng et al. 2020] and novel neural components, and we leave this as future work.

We utilize the standard L_1 loss to supervise the output sampling map:

$$\mathbb{L}_S = |\hat{S}_d - S_d| \tag{12}$$

where $\hat{S}_d$ is the ground-truth sampling map computed by accumulating dense photons. Inspired by the deep supervision [Xie and Tu 2015; Lee et al. 2015], we also provide the ground-truth signal on each decoding level to ease the loss back-propagation. To prevent over-blurring, we leverage an asymmetric function inspired by Vogels et al. [2018]; this leads to our full loss

$$\mathbb{L}_{\text{rec}} = \mathbb{L}_S \cdot (1 + (\lambda - 1) \cdot \mathbb{H}) \tag{13}$$

where $\mathbb{H} = 0$ if the output and the input values are both larger or smaller than the ground-truth value and $\mathbb{H} = 1$ if they are not on the same side. Specifically, when there are two equally-good output values, the function prefers the one that is closer to the input. This allows the reconstruction to retain some noise but also prevent details from being blurred out.

In Fig. 3, we present some examples of the reconstructed sampling maps along with the comparison with other traditional image inpainting techniques. We can clearly see the advantage of a group of learned filters represented by a data-driven neural network that is trained specifically under the context of path guiding, over the traditional hard-coded filters designed for general image processing (not specifically for path guiding) based on human knowledge.

ALGORITHM 1: Our neural path guiding framework. Through multiple iterations of path and light tracing, we construct a hierarchical grid (Sec. 7.1, in green), reconstruct and update the sampling map in each valid voxel (Sec. 7.2, in blue), and guide the path tracing using the reconstructed distributions (Sec. 7.3, in red). We also apply a final guided path tracing pass (Sec. 7.4, in purple).

```
Initialize 1 SPP path samples and 1 SPP photons ;
Initialize a spatial grid ;
for each iteration $t < T$ do
    Initiate $2^t$ SPP path samples;
    for each path do
        for each bounce $b$ do
            Locate voxel $j$ ($\boldsymbol{x}_b \in \Delta A_j$) ;
            if not isValid($j$) (no sampling map) then
                Sample($p_{\text{BSDF}}$) $\to \omega_b$ ;
            else
                Sample($p_{\text{MIS}}$) $\to \omega_b$ (Eqn. 15);
            end
            markValid($j$) ;
        end
        Compute path throughput and $L(\boldsymbol{x}_b, \omega_b)$ ;
        for each bounce at $\boldsymbol{x}_b \in \Delta A_j$ do
            if isValid($j$) then
                $L_b = L(\boldsymbol{x}_b, \omega_b) \cos\theta_b f_r(\boldsymbol{x}, \omega_b, \omega_o)$ ;
                if $\omega_b \leftarrow p_{\text{guide}}$ then $L_{j,\text{Guide}}$ += $L_b$ else $L_{j,\text{BSDF}}$ += $L_b$ ;
                if $\omega_b \leftarrow p_{\text{guide}}$ then $Q_{j,\text{Guide}}$ += 1 else $Q_{j,\text{BSDF}}$ += 1 ;
                if $Q_{j,\text{Guide}} \geq Q_{\text{Thr}}$ & $Q_{j,\text{BSDF}} \geq Q_{\text{Thr}}$ then Update $\alpha_j$ (Eqn. 14);
            end
        end
        Update the output image ;
    end
    Trace $2^t N_p$ light paths for photons;
    for each photon $p$ do
        Locate voxel $j$, solid angle $k$ ($\boldsymbol{x}_p \in \Delta A_j$, $\omega_p \in \Delta\Omega_k$) ;
        if isValid($j$) then
            Update power map: $\Phi_{j,k}$ += $\Delta\Phi_p$ (for Eqn.10);
            $M_j$ += 1 ;
            if $M_j > M_{thr}$ then
                Subdivide voxel $j$ into two sub-voxels (Sec. 7.1);
            end
        end
    end
    for each valid voxel $j$ do
        Reconstruct sampling maps ($p_{\text{Guide}}$) with neural net $\mathbb{F}$ ;
    end
end
Trace $N_f$ paths for final output (Sec. 7.4);
```

6.2 Discussion

The proposed neural network focuses on reconstructing high-quality distributions for local path sampling. This is a central sub-problem in many path guiding frameworks. Note that the problem of sampling map regression is independent of other modules in path guiding. We thus train our neural network independently without relying on any specific guiding frameworks. Therefore, our learning-based sampling map reconstruction module can potentially be extended to other existing path guiding frameworks by applying a proper variant of our neural architecture (e.g., mixture models [Vorba et al.

2014; Ruppert et al. 2020] and hierarchical maps [Müller et al. 2017; Rath et al. 2020]), and improves the traditional sampling distribution reconstruction modules.

7 PATH GUIDING WITH ADAPTIVE CACHING

In this section, we introduce our path guiding framework that leverages the presented deep network to reconstruct high-quality sampling maps in an adaptive and hierarchical spatial structure. The whole framework is illustrated in Algorithm 1.

We first fire some initial path and light rays (Line 1), initialize a grid (Line 2), and then utilize an iterative process (Line 3∼40) to adaptively build a hierarchical grid with per-voxel sampling maps for path guiding and rendering.

In each iteration, we trace camera paths (Line 4∼25); these paths can be guided (Line 8∼12) when tracing, and they are used to detect valid (i.e., containing path vertices) voxels (Line 13) and compute the mixture weight of one-sample MIS (Line 18∼21). We also trace photons (Line 26∼36) per iteration; in each valid voxel, we accumulate photon power (Line 31) that is required by our neural module and collect photon statistics for subdividing the hierarchical grid (Line 31∼34). We then reconstruct the sampling map in each valid voxel using our pre-trained deep neural network at the end of each iteration (Line 38∼40); these sampling maps are used to guide the path tracing in next iteration. After the iterative process, we apply a final-pass guided path tracing and compute the final beauty image (Line 41).

We adaptively partitioning the scene space to a hierarchical grid (Sec. 7.1). Meanwhile, the photons are collected for computing the noisy sampling maps in each valid voxel (Sec. 7.2); the path samples are used for rendering and computing the weight α for one-sample MIS (Sec. 7.3). After T iterations, we run a final path tracing pass (Sec. 7.4) with N_f spp. The final rendering result is computed by combining all iterations (except for the initialization phase) and the final pass, weighted by the inverse of their estimated variances [Müller 2019]. Note that we double the number of path and photon rays ($2^t N_c$ and $2^t N_p$ spp for iteration t, where N_c and N_p are initial values) after each iteration [Müller et al. 2017], so that both the quality of the input noisy histograms and per-pass rendering can be progressively improved.

7.1 Adaptive hierarchical grid

Recent work often utilize a binary KD-Tree [Müller et al. 2017; Rath et al. 2020] to adaptively partition the space, starting from a single root node that covers the entire scene. This coarse-to-fine spatial structure is effective and also necessary for these online learning approaches, since they need to acquire many samples in a large spatial region at an early stage. In contrast, our deep learning based approach can reconstruct a high-quality sampling map from a sparse set of photons; consequently, starting from a single root node is unnecessary and inefficient for our approach. Therefore, we propose to use a hierarchical grid for spatial partitioning, which combines uniform and adaptive spatial partitioning (Fig. 4).

Detecting valid voxels. While we can compute a sampling map for every voxel for path guiding, this is usually costly and unnecessary, since many voxels may not be reached by any camera path from the viewpoint. Therefore, we leverage camera paths to detect which voxels are involved for rendering this viewpoint. Specifically, when tracing $2^t N_c$ spp path samples in each iteration, we mark a voxel as valid, if there is at least one bounce point of the paths located in the voxel (Fig. 4). Once a voxel is marked as valid, we then start accumulating photons in the voxel for sampling map reconstruction and further subdivision of the voxel. This avoids the waste of caching redundant distributions and local KD-trees.

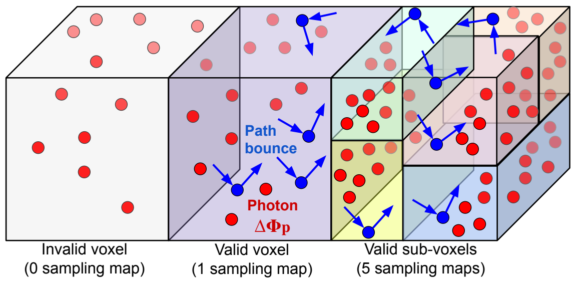

Fig. 4. The proposed hierarchical grid spatial caching structure. Path samples detect valid voxels to store sampling maps. A voxel is subdivided into a binary tree based on the local photon statistics. We split through the median to prevent skew or distorted distributions reconstructed in the initialization phase, and switch to the middle plane splitting in the iterative update phase when the photon point cloud is no longer stored. We alternate the splitting dimension with respect to the tree depth.

Initialization phase. We start the process by firing 1 spp path and photon rays. Path samples and photon samples are stored as point clouds in this particular phase to initialize our spatial grid; basically, we first build a regular grid given the spatial extent discovered by path samples, and then subdivide the grid given the initial per-voxel photons. Specifically, we use the collected path samples (after multiple bounces) to determine the bounding box of our spatial grid, which covers the *visible* part of the scene. We construct a regular grid by uniformly dividing the bounding box at a relatively coarse level (Fig. 4). Next, for each voxel in this uniform grid, we iteratively sub-partition the voxel into a local binary tree through the median photon which allows both sub-voxels to have a decent number of samples for reconstruction to start with, based on the number of photons that arrive at the voxel; this is also repetitively done in the following iterative update phase in a similar way but with a different splitting plane. Note that this initialization phase produces an initial hierarchical spatial grid – a coarse regular grid with local shallow KD-Trees. The initial grid is still coarse but relatively denser than a single shallow KD-Tree used in early stages in previous work [Müller et al. 2017]. This enables reconstructing more locally representative sampling maps, leading to better path guiding at early iterations in our framework, and better utilizes the benefits of our pre-trained deep neural network. The local KD-Trees are relatively shallow at this phase because photons are sparse, which are further subdivided in the iterative phase.

Iterative update phase. The spatial structure after the initialization phase can be still too coarse for a later time in rendering. Therefore, we iteratively subdivide it into a finer hierarchical grid. Our hierarchical grid is built to adapt to the complexity of incident light fields. We leverage the statistics of accumulated photons in each valid voxel for possible subdivisions. In particular, we consider M_j – the total number of photons hitting the valid voxel j. A voxel is split into two sub-voxels if $M_j > M_{\text{thr}}$, where M_{thr} is a predefined threshold. We recursively apply our subdivision criterion to sub-voxels. Unlike the initialization phase, we always use middle planes (instead of median) for splitting in this phase (same as [Müller et al. 2017]), since we do not store any photon point cloud anymore for the sake of memory, and this strategy is generally sufficient based on our observation. Once a voxel is subdivided, its two sub-voxels are reset as invalid waiting to be re-evaluated, inheriting the original noisy map with power values halved, and continue accumulating photons from the subsequent iterations once marked valid again. This photon-based subdivision process allows these complex voxels to utilize more local and accurate sampling maps, thus leading to more efficient renderings.

7.2 Sampling map reconstruction

Apart from determining the subdivision in the hierarchical grid, the main goal of tracing the per-iteration photons is to reconstruct the per-voxel sampling maps for path guiding. For any valid voxel (detected by camera paths), we accumulate photon power to compute the noisy power map of the voxel, as expressed by Eqn. 10. The power map accumulates all hitting photons $\Delta\Phi_p$ in the voxel through the current and previous iterations, which gets normalized to a noisy sampling map S_t as the input to the neural reconstruction module (Sec. 6) in iteration t. Once sampling maps are reconstructed and updated in one iteration, they are used in the next iteration to guide the path sampling.

7.3 Path guiding and one-sample MIS

In any iteration, if a path hits a voxel that does not have a sampling map, we use standard BSDF sampling at the bounce point; the voxel is then marked as valid and start accumulating photons immediately in the same iteration, enabling guiding in the subsequent iterations. On the other hand, once a path ray hits a valid voxel that has a reconstructed sampling map, path guiding can be achieved by importance sampling on the map (where CDF is built by GPUs). Since our sampling map only considers the incident radiance, we apply a one-sample MIS similar to previous works to combine guided sampling and BSDF sampling, as discussed in Eqn. 5. The combined sampling strategy however requires a parameter α that determines how often either sample strategy is selected. Usually, $\alpha = 0.5$ is a simple choice and performs reasonably well. An α that is learned via online optimization [Müller 2019] is also presented for better performance. Here we propose a simpler alternate method to achieve a similar goal, which can also serve as a better initialization for those approaches that try to search for an optimal α.

We present a heuristic α computation technique, based on collected path statistics; though simple, it results in effective per-voxel α_j in practice. In particular, we initialize $\alpha_j = 0.5$ in each valid voxel. Once a full path is constructed in rendering (either connect to or miss the light), we collect the reflected radiance contribution for every bounce point b on the path as $L_b = L(\boldsymbol{x}_b, \omega_b) \cos\theta_b f_r(\boldsymbol{x}, \omega_b, \omega_o)$. Meanwhile, for each voxel j, we accumulate all bounce contributions L_b (where $\boldsymbol{x}_b \in \Delta A_j$) in $L_{j,\text{BSDF}}$ and $L_{j,\text{Guide}}$, according to from which distribution ω_b is sampled. We also record the number of bounces sampled by two strategies as $Q_{j,\text{BSDF}}$ and $Q_{j,\text{Guide}}$. Once $Q_{j,\text{BSDF}} \geq Q_{\text{Thr}}$ and $Q_{j,\text{Guide}} \geq Q_{\text{Thr}}$ sub-paths have been collected in the voxel, we use the ratio of the averaged $L_{j,\text{BSDF}}$ and $L_{j,\text{Guide}}$ to update the mixing weight α_j:

$$\alpha_j = \frac{\overline{L}_{j,\text{BSDF}}}{\overline{L}_{j,\text{BSDF}} + \overline{L}_{j,\text{Guide}}}, \tag{14}$$

where $\overline{L}_{j,\text{BSDF}} = L_{j,\text{BSDF}}/Q_{j,\text{BSDF}}$ and $\overline{L}_{j,\text{Guide}} = L_{j,\text{Guide}}/Q_{j,\text{Guide}}$. Correspondingly, our one-sample MIS is expressed by:

$$p_{\text{MIS}}(\omega_i) = \frac{\overline{L}_{j,\text{BSDF}}}{\overline{L}_{j,\text{BSDF}} + \overline{L}_{j,\text{Guide}}} p_{\text{BSDF}}(\omega_i) + \frac{\overline{L}_{j,\text{Guide}}}{\overline{L}_{j,\text{BSDF}} + \overline{L}_{j,\text{Guide}}} p_{\text{guide}}(\omega_i). \tag{15}$$

We set $\alpha_j = 1$ if BSDF is a delta function and clamp α_j between 0.2 and 0.8 otherwise to handle statistical instability. This heuristic mixing weight considers the data that reflects the actual performance of BSDF sampling and guiding sampling, leading to effective mixed sampling in path guiding.

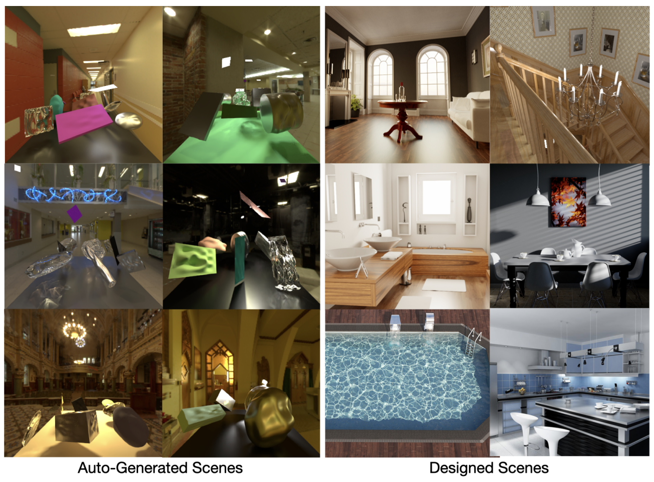

Fig. 5. Example scenes used for training our proposed neural network.

7.4 Rendering

Our learning based approach can reconstruct high-quality sampling maps from sparse photons in early iterations. We therefore leverage all path samples after the initialization phase for rendering the final image. While we can keep iteratively tracing more rays and refining the sampling maps, our reconstruction are often of sufficient quality after $T = 4{\sim}10$ iterations. Continuing tracing more photons afterwards merely leads to marginal sampling improvement, and the extra overhead from running the neural network cannot pay off. Therefore, we choose to stop iterating after $T = 4{\sim}10$ depending on light transport complexity, fix the per-voxel sampling maps, and run a final path tracing pass (with N_f spp) guided by the latest maps.

Scene/Method	PT	[Bako et al. 2019]	[Vorba et al. 2014]	[Müller et al. 2017]	[Rath et al. 2020]	[Ruppert et al. 2020]	**Ours**	Ours-NoFeat	Ours-Vanilla	Gauss-Best	Bilat-Best	Pyrmd-Vanilla-Best	Pyrmd-PullPush-Best
Comparison	SOTA rMSE ↓							Ablation rMSE ↓					
Caustics Egg	0.3187	0.1353	0.0462	0.0311	0.0121	0.0074	0.0052	0.0064	0.0091	0.0305	0.0157	0.4562	0.0168
Veach Ajar	0.3684	0.2585	0.0154	0.0073	0.0047	0.0033	0.0011	0.0024	0.0030	0.0057	0.0038	0.6331	0.0073
Bathroom	0.0610	0.0403	0.0204	0.0249	0.0142	0.0028	0.0050	0.0066	0.0076	0.0225	0.0208	0.1095	0.0156
Hotel	0.4176	0.2607	0.2838	0.0812	0.0792	0.0295	0.0276	0.0332	0.0421	0.0649	0.0543	0.7533	0.0645
Staircase	0.0176	0.0183	0.0110	0.0045	0.0038	0.0055	0.0013	0.0017	0.0018	0.0041	0.0037	0.0977	0.0040
Living Room	0.1928	0.1553	0.0235	0.0468	0.0416	0.0359	0.0060	0.0094	0.0134	0.0362	0.0341	0.6201	0.0299
Spaceship	0.2212	0.0914	0.0198	0.0716	0.0389	0.0192	0.0137	0.0230	0.0384	0.0871	0.0819	0.5023	0.0792
Classroom	0.0733	0.0514	0.0124	0.0085	0.0038	0.0040	0.0021	0.0026	0.0065	0.0181	0.0156	0.1718	0.0124
Wild Creek	0.1425	0.1100	0.0560	0.0618	0.0549	0.0423	0.0382	0.0393	0.0407	0.0595	0.0478	0.1551	0.0490
Kitchen	0.0644	0.0578	0.0249	0.0063	0.0035	0.0043	0.0030	0.0036	0.0063	0.0168	0.0110	0.0907	0.0100
Pool	0.1175	0.0528	0.0026	0.0025	0.0016	0.0015	0.0011	0.0013	0.0019	0.0081	0.0034	0.1531	0.0036

Table 1. Quantitative comparison. We compare to SOTA methods [Bako et al. 2019; Vorba et al. 2014; Müller et al. 2017; Rath et al. 2020; Ruppert et al. 2020] with equal rendering time. We show the corresponding rMSEs of the rendered full images of our testing scenes. Red, orange, and yellow denote the best, the second-best, and the third-best method in terms of rMSE (lower is better). We also present the ablation results with multiple neural setups (without additional features, or using a single vanilla U-Net) and traditional interpolation methods (Gaussian, bilateral, 5-level pyramid, and 5-level pull-push filters). We try multiple adaptive strategies for parameters of each traditional filter and report the best result.

8 IMPLEMENTATION

Dataset generation and neural network training. We create a large-scale dataset to train our sampling map reconstruction network. The dataset consists of both designed scenes and auto-generated scenes as shown in Fig. 5. We collect available scenes designed by researchers and artists from previous work and several websites [Bitterli 2016; Jakob 2010; Evermotion 2012; Trader 2020; Squid 2020; Blend Swap 2016]. This leads to 28 designed scenes, including multiple realistic indoor and outdoor scenes; we use 20 from them in our training set and the rest for testing our algorithm. To enhance the generalizability of our neural network, we further enlarge our training set by procedurally generating 500 scenes using randomized shape primitives, materials, and area lights, similar to [Zhu et al. 2020; Xu et al. 2018]. We also leverage a complex lighting dataset [Gardner et al. 2017] and randomly select an environment map for each generated scene as its additional illumination. This auto-generation process increases the diversity and complexity of our training scenes, leading to better generalization on testing scenes.

We partition the space of each training scene uniformly using a regular grid with a random resolution. Each voxel is a cube with a side length $\mathbb{R}_B/r_{\text{init}}$ where $\mathbb{R}_B$ is the diagonal length of the estimated bounding box and r_{init} controls the resolution ranging from 10 to 200. This makes our pre-trained neural network generalize to various voxel sizes in a hierarchical grid. We support the resolution of 128×64 (default) or 64×32 for our sampling maps (normalized values in *half-float*) given a certain amount of system memory, which is sufficient for path guiding. For each training scene, we augment the training set by randomly selecting multiple iteration numbers t from 1 to 12, tracing photons and computing noisy sampling maps based on Eqn. 10. The ground-truth sampling map is computed by accumulating photons generated through 20 iterations.

During rendering, photons are often distributed unevenly in different voxels (from several to several thousand), leading to diverse input distributions; we therefore train multiple versions of neural network as a mixture of experts [Jacobs et al. 1991], and make each network focus on a certain range of input photon sparsity in a voxel. Specifically, we train five networks separately and the corresponding ranges of photon numbers are $[0, 100)$, $[100, 500)$, $[500, 1000)$, $[1000, 5000)$, $[5000, \infty)$. This enables better reconstruction quality compared to using a single compact (i.e., low-capacity) neural network. During training, we use mini-batches with a size of 50 and train each network using ADAM optimizer [Kingma and Ba 2014] with a learning rate of 1.0×10^{-4} until convergence.

Path guiding details. We use $N_c = 1$ (spp) for all our experiments, leading to 2^t spp paths for iteration t. We also correspondingly trace the same number of light paths (N_p) per iteration for distributing photons. The initial uniform grid is implemented as a hash grid that can be accessed in $O(1)$ time. Each sub binary tree is like a local KD-tree that can be accessed in $O(\log(n))$ time. The final hierarchical grid is a hybrid spatial structure and can be accessed with a fast speed at rendering time. Since our spatial structure is adaptively constructed, it is not very sensitive to the initial resolution, and we use a resolution of $r_{\text{init}} = 100$ for all the testing scenes without over-partitioning. For voxel subdivision, we use an iteration-dependent threshold for the photon count, given by $M_{\text{thr}} = c \cdot \sqrt{2^t}$ similar to [Müller et al. 2017], where c is a scalar parameter depending on the number of initial photon rays. We find $c = 400 \sim 800$ performs reasonably in practice, and we use $c = 500$ in the experiments. Note that validity test is skipped in later iterations for subdivided voxels that already had good enough locality, and resources are recycled for voxels that stay invalid for a long time. In each iteration, the maximum number of allowed new spatial levels is set to 3 to prevent

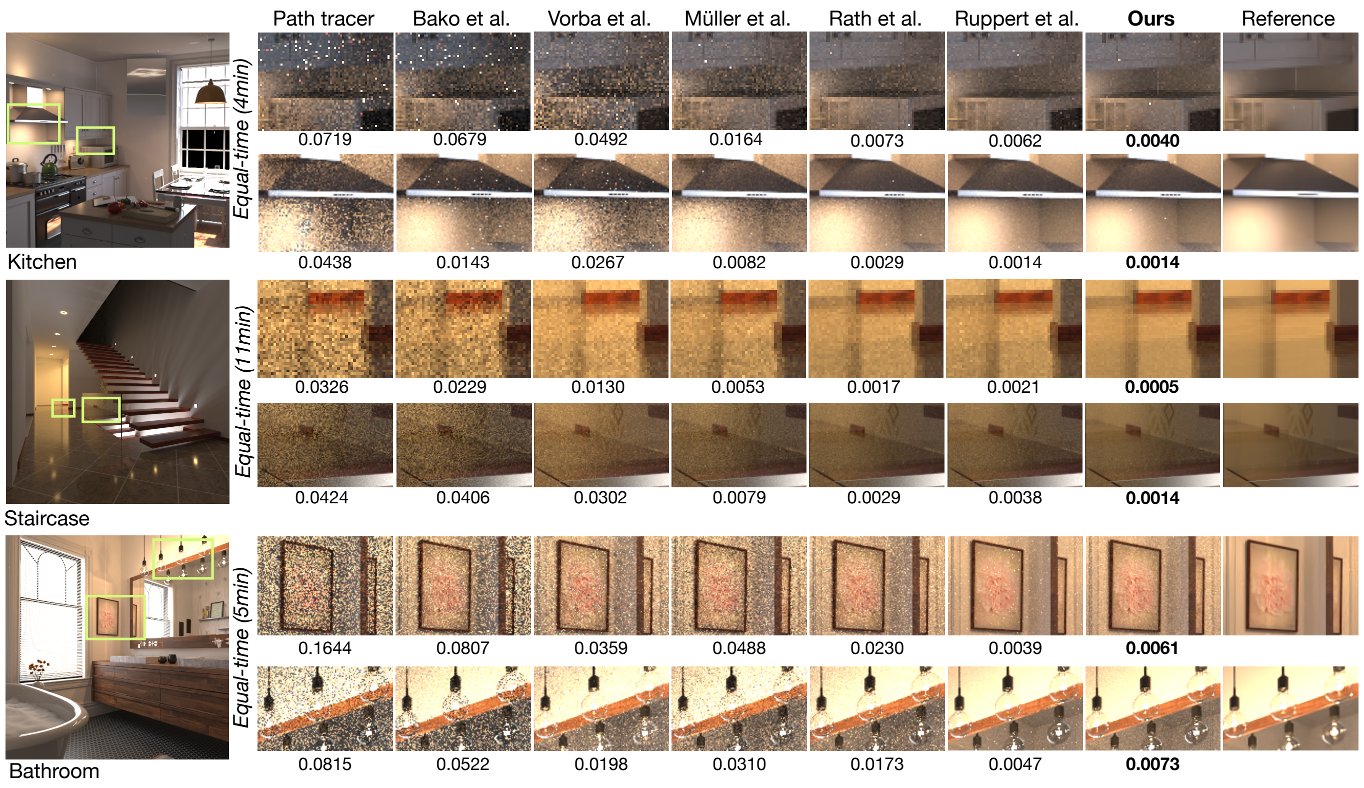

Fig. 6. Qualitative and quantitative comparison with baseline methods ([Müller et al. 2017; Bako et al. 2019; Rath et al. 2020; Vorba et al. 2014; Ruppert et al. 2020]) in equal rendering time (ours with GPU) on complex indoor scenes. Our deep learning based approach enables accurate sampling map reconstruction for the complex direct and indirect lighting, leading to efficient rendering. We show zoomed-in crops with rMSEs.

the cache from over densification. The number of spatial voxels in our approach ranges from ~30K to ~200K in the testing scenes.

We use a machine with a 32-thread Intel Core i9-7960X CPU, two Nvidia Titan RTX GPUs, and 128GB memory for rendering our testing scenes. Our framework is implemented in the Mitsuba engine [Jakob 2010], and we use the PyTorch C++ API [Paszke et al. 2019] at rendering time for sampling map reconstruction on GPUs. This ensures a straight comparison with previous methods, most of which are also implemented with Mitsuba. In particular, we only use GPUs to run neural network inference for sampling map reconstruction, while all other parts of the algorithm (including path and photon tracing, sampling, radiance computation, spatial grid construction, etc.) run on the CPU as in Mitsuba. Detailed running time breakdowns are provided in the supplementary material. Note that we indeed rely on extra GPU resources, and the data copying overhead is non-negligible although we carefully handle the data streaming (we only exchange non-empty valid values) and parallelization between CPU and GPU. A pure GPU-based framework leveraging specialized processor cores (e.g., [Parker et al. 2010]) could be implemented to mitigate these limitations in the future, but one may face some challenges since algorithmic changes and proper memory handling may be needed to make it feasible.

9 EVALUATION

We now present extensive experiments to evaluate our path guiding approach. We first evaluate the rendering quality of our method by comparing against various state-of-the-art path-guiding methods quantitatively and qualitatively. We then investigate sub-components in our system to justify their effectiveness. Many additional evaluation results can be found in the supplementary material.

Configuration. We evaluate our method comprehensively on 8 realistic testing scenes; the corresponding images of these scenes can be found in Figs. 6, 7, 8, 9 and 11. These testing scenes include diverse challenging indoor and outdoor cases with complex global illumination. For indoor scenes containing an environment map illumination (e.g., sunlight), we provide the window geometry for sampling light paths from the environment map, facilitating the photon tracing process in these scenes to increase the indoor photon visibility. For scenes where only a small part gets rendered from the camera viewpoint, we clamp the scenes by removing the invisible geometries. These strategies are incorporated for compromising the well-known photon visibility issue without the guided light tracing for photons, which is discussed later (Fig. 13).

For our method, the required time to achieve good rendering quality ranges roughly from 3 to 20 minutes (depending on scene complexity) on these testing scenes. We demonstrate equal-time

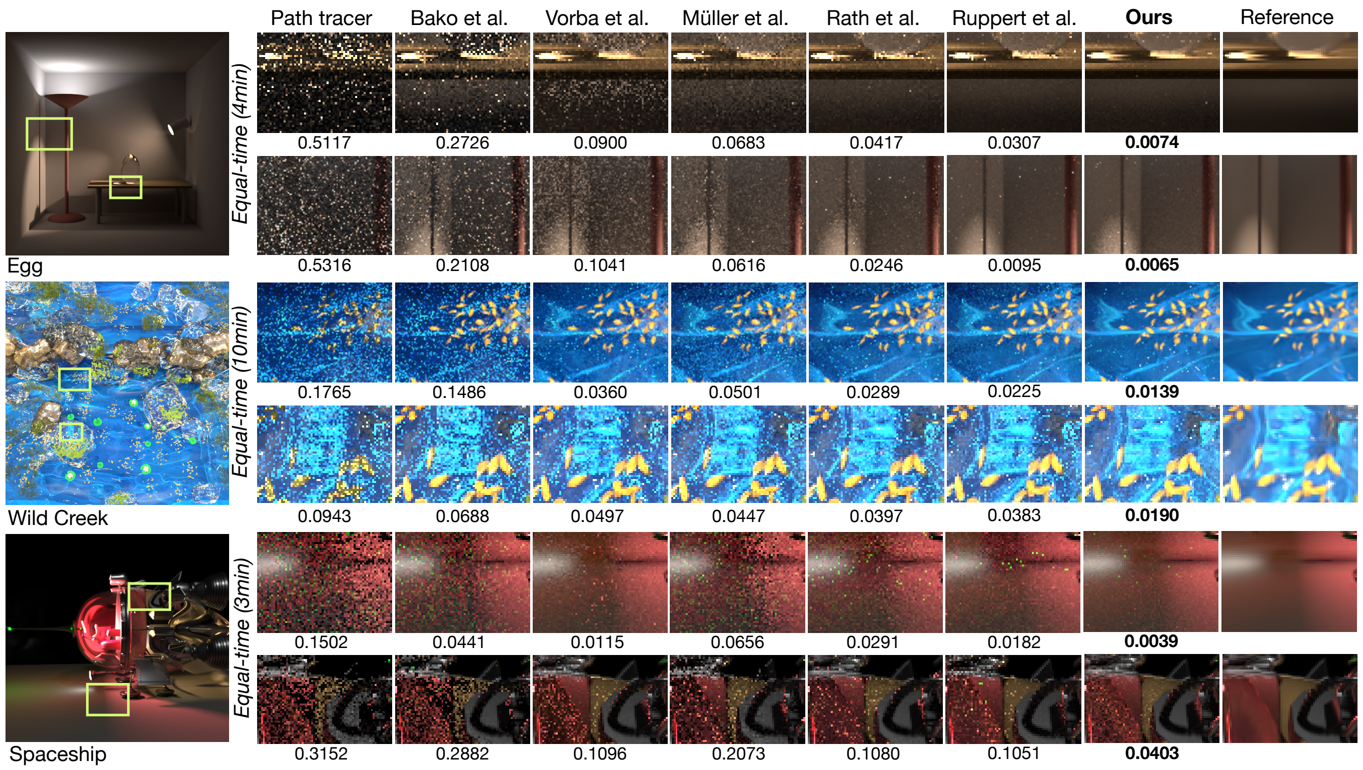

Fig. 7. Qualitative and quantitative comparison with baseline methods ([Müller et al. 2017; Bako et al. 2019; Rath et al. 2020; Vorba et al. 2014; Ruppert et al. 2020]) in equal rendering time (ours with GPU) on scenes containing transparent surfaces. We show zoomed-in crops with rMSEs. Our method achieves better visual quality and lower rMSEs in these challenging cases.

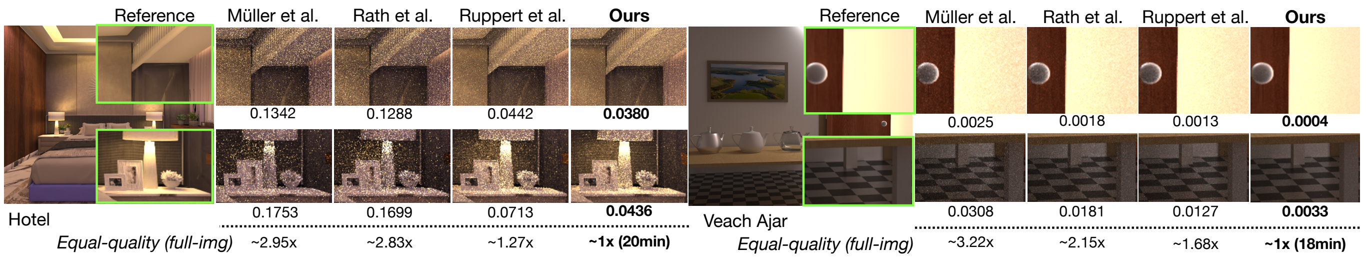

Fig. 8. Equal-time and equal-quality comparison. Similar to Fig. 6 and 7, we do equal-time comparisons with these best-performing methods ([Müller et al. 2017; Rath et al. 2020; Ruppert et al. 2020]). In addition, we also show equal-quality rendering time comparison. We list the corresponding rendering time (expressed by the scale to our time) of each method for achieving the same rMSE of the full image.

comparisons (while ours requires GPU) by comparing with four state-of-the-art CPU-based path-guiding methods [Müller et al. 2017; Vorba et al. 2014; Rath et al. 2020; Ruppert et al. 2020] and one GPU-based work [Bako et al. 2019]. To better illustrate the effectiveness of path guiding, we turn off the Next-Event Estimation (NEE) for ours and comparison methods as done in previous work [Vorba et al. 2014; Müller et al. 2017]. Comparison results with the standard NEE turned on are presented in the supplementary material.

Equal-time comparison. We now present the results of our method and compare against other methods with equal rendering time. For quantitative evaluation, we use the relative Mean Squared Error (rMSE, as used in [Rath et al. 2020]) as the metric. We show the averaged rMSE of the full-resolution images in Tab. 1. We also show qualitative comparisons of rendered images in Fig. 6 and Fig. 7; Fig. 6 show results of complex indoor scenes and Fig. 7 include scenes containing complex transparent objects.

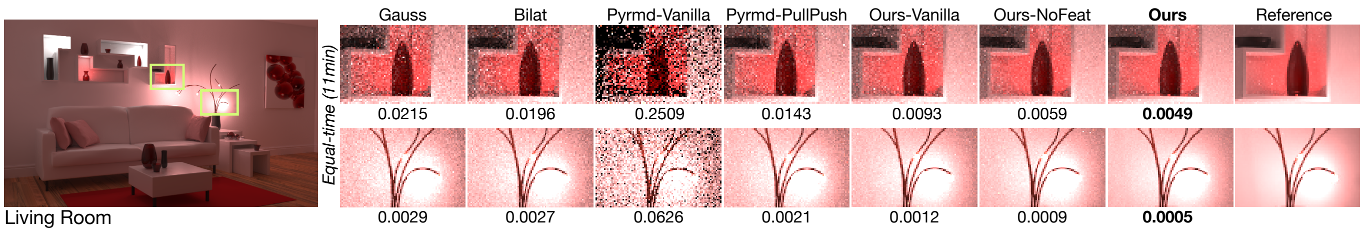

Fig. 9. Comparison to multiple traditional image interpolation methods and some variants of our proposed full neural architecture. Traditional 2D filtering algorithms can also inpaint the missing sampling map values, but can result in over-blurring or residual noise in different places (we adaptively tune the parameters for each filter and report the best result). Data-driven neural methods like ours are often better at image reconstruction and recovering missing details. We incorporate additional features and allow multiple versions of neural network to handle different input sparsity, leading to better result than a single vanilla U-Net.

In particular, we compare with [Müller et al. 2017; Rath et al. 2020] that use quadtrees to represent sampling distributions and reconstruct the trees by accumulating path samples online. Because of the pure online learning, it can be more challenging for these unidirectional methods to discover high-energy paths in early iterations of the rendering, especially for scenes that involve complex specular-diffuse interactions or other strong global illumination effects. In contrast, we leverage pre-trained neural networks that can achieve neural reconstruction of high-quality sampling maps with sparse photons at early iterations; our approach leverages photons from light paths that ease the process of light discovery for scenes with complex indirect illumination, resulting in more variance reduction.

[Bako et al. 2019] is a recent deep learning approach that first leverages an offline trained neural network for unidirectional path guiding; yet the method can only guide the primary bounces and produce unsatisfactory results in most testing cases. Their technique can be effective for lowering the initial severe noise with sparse path samples, especially on scenes with strong direct illumination. However, such a first-bounce technique is less effective for scenes with strong indirect illumination. Our approach extends the offline training idea to multiple bounces and successfully models the incident light field at any local regions in a scene, enabling better rendering quality as a result.

We also compare with previous work that use parametric mixture models (GMM in [Vorba et al. 2014] and VMM in [Ruppert et al. 2020]) to represent sampling distributions, as shown in Tab. 1, Fig. 6, and Fig. 7. The recent work [Ruppert et al. 2020] combines the classic tree-based spatial structure (as used in [Müller et al. 2017; Rath et al. 2020]) with efficient and flexible parametric distribution fitting techniques that consider parallax awareness; this leads to reasonably high rendering quality and outperforms other comparison methods (and sometimes ours) on the testing scenes. Fortunately, thanks to the effectiveness of our neural reconstruction of sampling distributions, we can still achieve lower variance on many other testing scenes. Overall, our approach can produce the state-of-the-art path guiding results.

Equal-quality comparison. Besides the equal-time comparison, we also compare the time spent to achieve the results of similar quality on some challenging scenes in Fig. 8; the corresponding rendering time (compared to our time) of each method is listed, for achieving the same rMSE (with a threshold of 10^{-4} in difference) of the full image as our method. We can see that our method can achieve the equal rendering quality with less rendering time in these cases.

The benefit of photons. We have demonstrated that our photon-driven approach can achieve efficient path guiding and rendering on various challenging testing scenes. To further demonstrate the benefits of using photons for complex scenes, we show results of a special/extreme case, designed to contain mostly glossy surfaces and be illuminated by extremely narrow-band spotlights. This makes the path samples very difficult to connect to these lights, for which previous path-based methods [Müller et al. 2017; Rath et al. 2020; Ruppert et al. 2020] have lower sampling quality. In contrast, emitted photons can quickly provide valid samples for reconstruction and our photon-driven approach thus works reasonably well.

Sampling map reconstruction. The core of our path guiding approach is a novel deep learning based sampling map reconstruction. We show examples of our reconstructed sampling maps in Fig. 3, compared with the reference and the reconstructions from a few traditional image filtering and inpainting methods, including Gaussian filtering, bilateral filtering, image pyramid [OpenCV 2021] and pull-push [Gortler et al. 1996], using the same input. For each of these techniques, we have tested different adaptive heuristics for its parameters (e.g., standard deviation, bandwidth, number of levels) and report the best result.

In general, given a noisy sampling map as input, traditional filtering and inpainting techniques can smooth the input, but their reconstruction quality is low due to the hard-coded kernels. In contrast, our neural reconstruction method can produce more accurate sampling maps that are closer to the ground truth by allowing the kernels to be learned from the data. In addition, our approach can regress more accurate sampling maps, when the input uses more iterations of photons; we show one converged example in Fig. 3 and more sampling map reconstructions through multiple iterations in the supplementary material. We also show the final renderings using these traditional techniques in Fig. 11, where better reconstruction leads to lower variance in the rendering result.

In Fig. 9, we also compare with a few reduced versions of our approach for an ablation study. We compare with a naive neural reconstruction method using a single vanilla U-Net without the use of our proposed architecture or multi-version inference strategy, and a

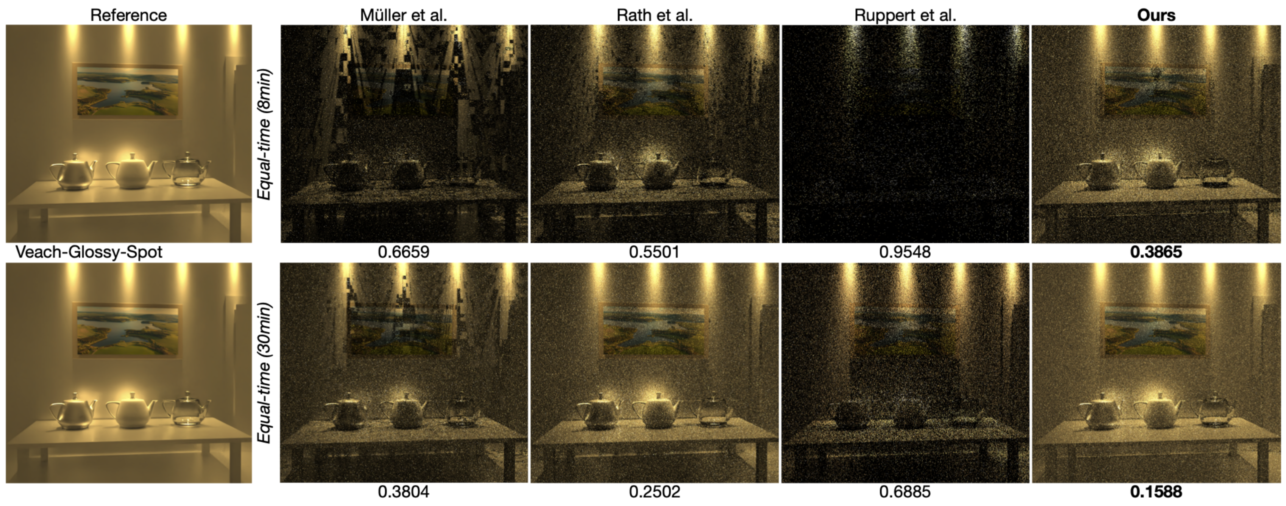

Fig. 10. Comparison to best-performing methods in two equal-times on a special scene where using photons is more beneficial than pure path samples. We modify the VEACH AJAR scene to have glossy surface materials and it is lit by narrow-band (1°) spotlights pointing upwards to the ceiling. We show the full images with rMSEs.

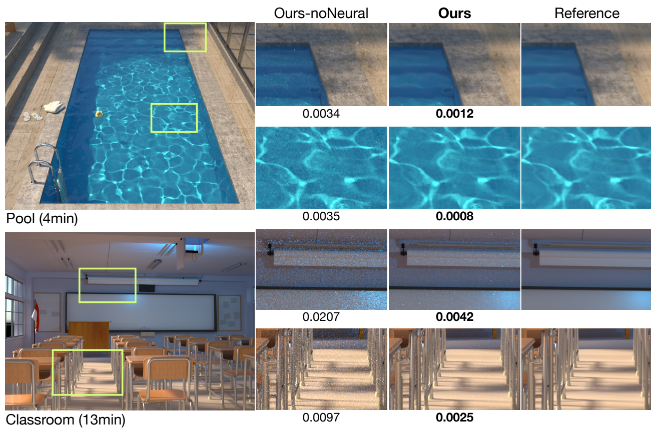

Fig. 11. We study the effectiveness of the proposed neural reconstruction module. We compare our full model with a downgraded version without the neural sampling map reconstruction step. We show crops with rMSEs given equal rendering time.

reduced variant without using the additional features (Eqn. 11).Note that our full model leads to better rendering quality compared with these ablated methods. To further justify the effectiveness of our neural module, we compare with only using the noisy input sampling map (without the neural reconstruction) for path guiding in Fig. 11. This example clearly demonstrates the benefit of leveraging the deep network in our framework. Directly using the noisy photon power histogram for sampling is not effective, since it does not sufficiently resemble the local incident radiance.

Light path guiding extension. In this work, we leverage photons to generate our sampling distributions. However, it is well-known that tracing photons can sometimes be inefficient, especially when only

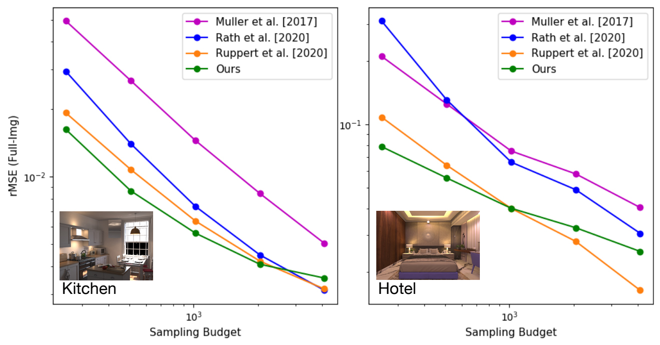

Fig. 12. The maximum resolution of our histogram map is limited by the available memory on the machine and the neural computational cost, leading to ours being overtaken by previous methods that leverage more memory-efficient representations such as quadtree ([Rath et al. 2020]) and parametric function ([Ruppert et al. 2020]) at a later time. Our speed of convergence slows down earlier due to such drawbacks. The average memory consumption on the KITCHEN and HOTEL scenes is 0.62GB and 0.68GB ([Müller et al. 2017]), 0.81GB and 2.77GB ([Rath et al. 2020]), 0.69GB and 0.74GB ([Ruppert et al. 2020]), 9.26GB and 12.38GB (Ours).

a small region of a large scene is visible to the camera, many traced photons may never reach any valid voxels, leading to expensive photon tracing and undesirably overly-dense spatial structure. Guiding the tracing of photons can address this issue to some extent. We show a simple extension of photon guiding in Fig. 13, by applying the light path guiding technique similar to [Vorba et al. 2014], which improves the rendering quality of scenes that are difficult for the standard photon tracing.

Limitations. The proposed approach is mainly designed for offline rendering, as previous path guiding work; it accelerates the

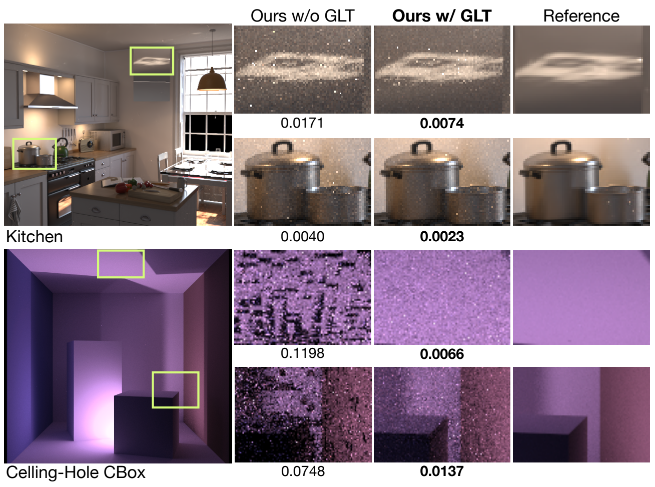

Fig. 13. Our method purely relies on local photons; therefore the photon visibility needs to be at a reasonable level for an acceptable reconstruction quality. Here are examples in the KITCHEN scene where most emitted photons from sunlight cannot appear inside the room and another extreme case in a modified CORNELL Box scene where many photons cannot enter the box. The reconstructed sampling maps have lower quality when *visible* photons are too sparse to rebuild the incident radiance. In contrast, it is sometimes beneficial to guide traced photons into visually important regions via the guided light tracing (GLT).

convergence of path tracing but still requires a moderate number of path samples. Combining our approach with modern denoising techniques can further reduce the number of samples (as shown in the supplementary material). Similar to previous work [Müller et al. 2017; Rath et al. 2020] that also use spatial voxels to store local sampling distributions, a structured artifact can appear in the rendered image when photons are not dense enough (the bottom of Fig. 13). Such artifact disappears with more photons; combining our method with parallax-aware techniques [Ruppert et al. 2020] through warping or transformation could potentially address it more effectively in the future, but may also expose new challenges in computational cost of histograms over mixture lobes.

We use standard 2D images as sampling distributions for deep CNN based reconstruction. However, this consumes more memory than the quadtrees in [Müller et al. 2017; Rath et al. 2020] and parametric models in [Vorba et al. 2014; Ruppert et al. 2020], and we can only adopt low-resolution histogram maps due to limited system memory. The other more compact representations can in fact fit more detailed sampling distributions due to their adaptive nature, although this also requires many more samples. We demonstrate this in Fig. 12, where previous methods start to overtake ours with a very large sampling budget (more than $10^3 \sim 10^4$ rays per pixel). However, our approach is still effective with a moderate sampling budget, which is often how path guiding is expected to be applied, especially when in practice it can be effectively combined with denoising techniques as presented in the supplementary material. Extension to more compact directional representations is possible as shown in a recent concurrent work [Zhu et al. 2021].

10 CONCLUSION AND FUTURE WORK

In this paper, we present a new deep learning-based photon-driven path guiding approach. Our approach leverages photons to reconstruct sampling distributions, which is sometimes more effective than pure unidirectional (path-driven) methods for challenging scenes that are dominated by indirect lighting; we propose to use a deep neural network to regress high-quality sampling maps from low-quality photon histograms, enabling effective path guiding as a result. To better utilize the benefits of such neural framework, we introduce an adaptive hierarchical grid to cache the reconstructed sampling maps spatially in the scene, allowing for path guiding at any bounce. The proposed approach achieves reasonably better rendering results than previous state-of-the-art path-guiding methods on many of our challenging test scenes.

We also observe some drawbacks of our image-based neural reconstruction approach including the excessive memory usage, limited resolution of our histogram maps, and uneven photon visibility. We believe this work can inspire future research on searching for a more flexible neural representation for sampling distributions and reducing the dependency on additional GPU resources.

We take a step towards making the neural path guiding more effective, thus also opening up many appealing future directions. Our approach leverages local photon statistics for sampling map reconstruction; an interesting extension is to also consider some global context and even achieve guiding in primary space (e.g., [Müller et al. 2019; Guo et al. 2018]). In addition, our target sampling density function can potentially be extended to advanced distributions such as variance-aware [Rath et al. 2020] or product sampling [Herholz et al. 2016] (avoiding the MIS, possibly by caching and reusing a discretized BSDF representation and leveraging GPU for computing the product) for better sampling efficiency. Combining our deep learning based local sampling reconstruction with reinforcement learning techniques [Huo et al. 2020] to achieve sampling with a proper reward function could provide more benefits.

Meanwhile, we leverage heuristic criterion to achieve voxel subdivision in the hierarchical grid; this spatial partitioning process could also be potentially learned via another deep neural network in the future. While we purely leverage photons in our method, combining photons and path samples in a holistic neural path guiding framework is an interesting future direction to explore. More importantly, we leverage regular 2D images as sampling distributions in the neural reconstruction; this inspires combining deep learning with other distribution representations such as hierarchical quadtrees [Müller et al. 2017; Rath et al. 2020] and parametric mixture models [Vorba et al. 2014; Ruppert et al. 2020] to reduce the excessive cost of GPU resources. Zhu et al. [2021] presents an example of such extension, although it still heavily relies on powerful GPUs.

ACKNOWLEDGMENTS

We greatly appreciate all anonymous reviewers for their constructive comments in the revision process. This work was supported in part by NSF grants 1703957 and 1764078, Google Ph.D. Fellowships, the Ronald L. Graham Chair and the UC San Diego Center for Visual Computing.

REFERENCES

Steve Bako, Mark Meyer, Tony DeRose, and Pradeep Sen. 2019. Offline Deep Importance Sampling for Monte Carlo Path Tracing. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 527–542.

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.

Benedikt Bitterli. 2016. Rendering resources. https://benedikt-bitterli.me/resources/.

LLC Blend Swap. 2016. Blend swap.

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 Transactions on Graphics (TOG)* 36, 4 (2017), 1–12.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. 2018. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine* 35, 1 (2018), 126–136.

Lei Deng, Guoqi Li, Song Han, Luping Shi, and Yuan Xie. 2020. Model Compression and Hardware Acceleration for Neural Networks: A Comprehensive Survey. *Proc. IEEE* 108, 4 (2020), 485–532.

Stavros Diolatzis, Adrien Gruson, Wenzel Jakob, Derek Nowrouzezahrai, and George Drettakis. 2020. Practical Product Path Guiding Using Linearly Transformed Cosines. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 23–33.

TM Evermotion. 2012. Evermotion 3d models.

Marc-André Gardner, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagné, and Jean-François Lalonde. 2017. Learning to predict indoor illumination from a single image. *arXiv preprint arXiv:1704.00090* (2017).

Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. 1996. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques.* 43–54.

Jerry Guo, Pablo Bauszat, Jacco Bikker, and Elmar Eisemann. 2018. Primary sample space path guiding. In *Eurographics Symposium on Rendering*, Vol. 2018. The Eurographics Association, 73–82.

Toshiya Hachisuka and Henrik Wann Jensen. 2009. Stochastic progressive photon mapping. In *ACM SIGGRAPH Asia 2009 papers.* 1–8.

Toshiya Hachisuka, Shinji Ogaki, and Henrik Wann Jensen. 2008. Progressive photon mapping. In *ACM SIGGRAPH Asia 2008 papers.* 1–8.

Sebastian Herholz, Oskar Elek, Jiří Vorba, Hendrik Lensch, and Jaroslav Křivánek. 2016. Product importance sampling for light transport path guiding. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 67–77.

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.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation* 3, 1 (1991), 79–87.

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

Henrik Wann Jensen. 1995. Importance driven path tracing using the photon map. In *Eurographics Workshop on Rendering Techniques.* Springer, 326–335.

Henrik Wann Jensen. 1996. Global illumination using photon maps. In *Rendering Techniques' 96.* Springer, 21–30.

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. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques.* 143–150.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Claude Knaus and Matthias Zwicker. 2011. Progressive photon mapping: A probabilistic approach. *ACM Transactions on Graphics (TOG)* 30, 3 (2011), 25.

Eric P Lafortune and Yves D Willems. 1993. Bi-directional path tracing. (1993).

Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. 2015. Deeply-supervised nets. In *Artificial intelligence and statistics.* 562–570.

Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. 2018. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV).* 85–100.

Thomas Müller. 2019. "Practical Path Guiding" in Production. In *ACM SIGGRAPH Courses: Path Guiding in Production, Chapter 10.* ACM, New York, NY, USA, 18:35–18:48. https://doi.org/10.1145/3305366.3328091

Thomas Müller, Markus Gross, and Jan Novák. 2017. Practical path guiding for efficient light-transport simulation. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 91–100.

Thomas Müller, Brian McWilliams, Fabrice Rousselle, Markus Gross, and Jan Novák. 2019. Neural importance sampling. *ACM Transactions on Graphics (TOG)* 38, 5 (2019), 1–19.

OpenCV. 2021. Image Pyramids. https://docs.opencv.org.

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

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 32.* Curran Associates, Inc., 8024–8035.

Mark Pauly, Thomas Kollig, and Alexander Keller. 2000. Metropolis light transport for participating media. In *Rendering Techniques 2000.* Springer, 11–22.

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.

Alexander Rath, Pascal Grittmann, Sebastian Herholz, Petr Vévoda, Philipp Slusallek, and Jaroslav Křivánek. 2020. Variance-Aware Path Guiding. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2020)* 39, 4 (July 2020), 151:1–151:12. https://doi.org/10.1145/3386569.3392441

Florian Reibold, Johannes Hanika, Alisa Jung, and Carsten Dachsbacher. 2018. Selective guided sampling with complete light transport paths. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–14.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention.* Springer, 234–241.

Lukas Ruppert, Sebastian Herholz, and Hendrik P. A. Lensch. 2020. Robust Fitting of Parallax-Aware Mixtures for Path Guiding. *ACM Transactions on Graphics (TOG)* (2020).

Peter Shirley, Bretton Wade, Philip M Hubbard, David Zareski, Bruce Walter, and Donald P Greenberg. 1995. Global illumination via density-estimation. In *Rendering Techniques' 95.* Springer, 219–230.

Turbo Squid. 2020. 3D Models, Plugins, Textures, and more at Turbo Squid.

CG Trader. 2020. Cg trader. *URL http://www. cgtrader. com* 4 (2020).

Eric Veach. 1997. *Robust Monte Carlo methods for light transport simulation.* Vol. 1610. Stanford University PhD thesis.

Eric Veach and Leonidas Guibas. 1995a. Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques.* Springer, 145–167.

Eric Veach and Leonidas J Guibas. 1995b. Optimally combining sampling techniques for Monte Carlo rendering. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques.* 419–428.

Eric Veach and Leonidas J Guibas. 1997. Metropolis light transport. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques.* 65–76.

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 Transactions on Graphics (TOG)* 37, 4 (2018), 1–15.

Jiří Vorba, Johannes Hanika, Sebastian Herholz, Thomas Müller, Jaroslav Křivánek, and Alexander Keller. 2019. Path Guiding in Production. In *ACM SIGGRAPH Courses.* ACM, New York, NY, USA, 18:1–18:77. https://doi.org/10.1145/3305366.3328091

Jiří Vorba, Ondřej Karlík, Martin Šik, Tobias Ritschel, and Jaroslav Křivánek. 2014. On-line learning of parametric mixture models for light transport simulation. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–11.

Saining Xie and Zhuowen Tu. 2015. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision.* 1395–1403.

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.

Zili Yi, Qiang Tang, Shekoofeh Azizi, Daesik Jang, and Zhan Xu. 2020. Contextual Residual Aggregation for Ultra High-Resolution Image Inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 7508–7517.

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. 2019. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE International Conference on Computer Vision.* 4471–4480.

Shilin Zhu, Zexiang Xu, Henrik Wann Jensen, Hao Su, and Ravi Ramamoorthi. 2020. Deep Kernel Density Estimation for Photon Mapping. In *Computer Graphics Forum*, Vol. 39. Wiley-Blackwell.

Shilin Zhu, Zexiang Xu, Tiancheng Sun, Alexandr Kuznetsov, Mark Meyer, Henrik Jensen, Hao Su, and Ravi Ramamoorthi. 2021. Hierarchical Neural Reconstruction for Path Guiding Using Hybrid Path and Photon Samples. In *ACM Transactions on Graphics.*