DOI: 10.1111/cgf.14675
Pacific Graphics 2022
N. Umetani, E. Vouga, and C. Wojtan
(Guest Editors)

Volume 41 (2022), Number 7

Real-time Deep Radiance Reconstruction from Imperfect Caches

Tao Huang[1], Yadong Song[1], Jie Guo[1†], Chengzhi Tao[1], Zijing Zong[1], Xihao Fu[1], Hongshan Li[2] and Yanwen Guo[1]

[1]State Key Lab for Novel Software Technology, Nanjing University, China
[2] Huawei Cloud Computing Technologies Co., Ltd.

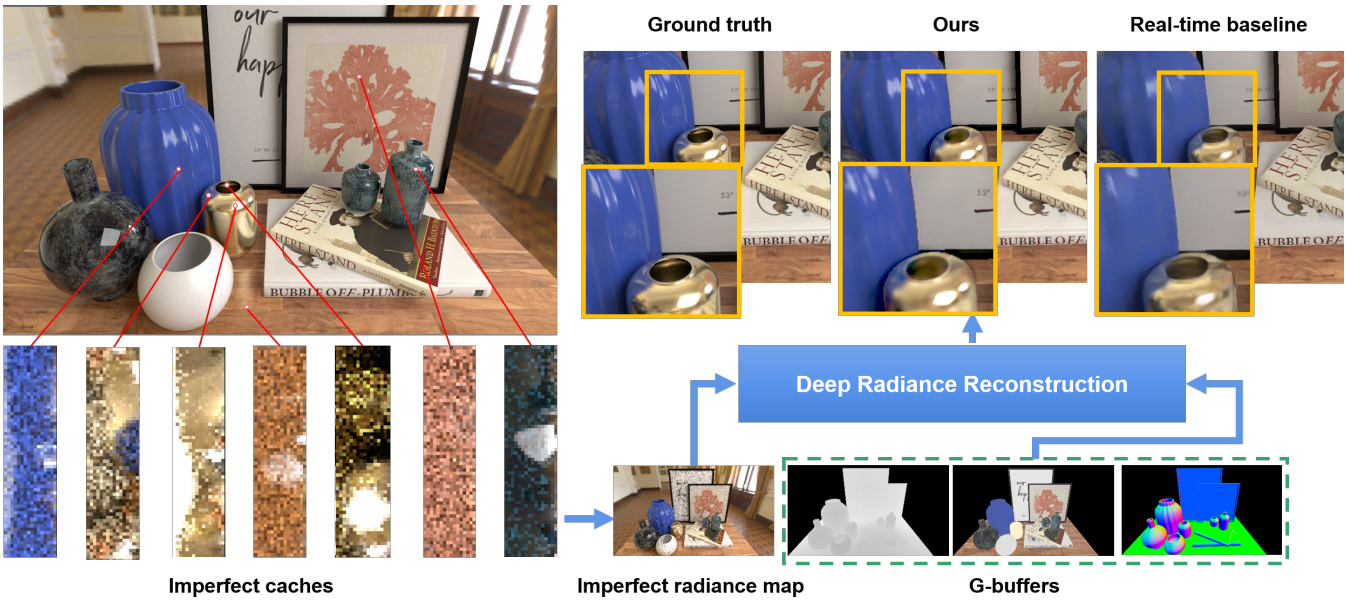

Figure 1: *Built upon imperfect caches precomputed at the barycentre of every triangle in a 3D scene with a low sampling rate (e.g., 32 samples per-pixel) and a low image resolution (e.g., 64×16), we propose a neural rendering pipeline that can reproduce a wide range of global illumination effects in real-time (> 90 frames per-second). The quality of reconstructed images are close to the path-traced ground truth and better than a real-time baseline (real-time path tracing+denoising), thanks to a deep learning-based radiance reconstruction method.*

Abstract

Real-time global illumination is a highly desirable yet challenging task in computer graphics. Existing works well solving this problem are mostly based on some kind of precomputed data (caches), while the final results depend significantly on the quality of the caches. In this paper, we propose a learning-based pipeline that can reproduce a wide range of complex light transport phenomena, including high-frequency glossy interreflection, at any viewpoint in real time (> 90 frames per-second), using information from imperfect caches stored at the barycentre of every triangle in a 3D scene. These caches are generated at a precomputation stage by a physically-based offline renderer at a low sampling rate (e.g., 32 samples per-pixel) and a low image resolution (e.g., 64×16). At runtime, a deep radiance reconstruction method based on a dedicated neural network is then involved to reconstruct a high-quality radiance map of full global illumination at any viewpoint from these imperfect caches, without introducing noise and aliasing artifacts. To further improve the reconstruction accuracy, a new feature fusion strategy is designed in the network to better exploit useful contents from cheap G-buffers generated at runtime. The proposed framework ensures high-quality rendering of images for moderate-sized scenes with full global illumination effects, at the cost of reasonable precomputation time. We demonstrate the effectiveness and efficiency of the proposed pipeline by comparing it with alternative strategies, including real-time path tracing and precomputed radiance transfer.

CCS Concepts

• ***Computing methodologies*** → ***Ray tracing;*** *Neural networks;*

† Corresponding author: guojie@nju.edu.cn

1. Introduction

Reproducing photo-realistic images from complex 3D scenes in real time has been a long-standing problem in computer graphics. Despite recent advances in hardware-accelerated ray tracing in modern GPUs [SABB18, Bur20, Har20] and efficient denoising techniques [SKW*17, CKS*17, ZSWL21, FWHB21, KIM*19], Monte Carlo ray/path tracing, as a flexible and general solution for capturing every physically-based rendering effect, is still out of reach for time-critical applications, especially for low-end devices with a very limited computational budget [CJ16, KKW*13, KVBB*19].

Currently, numerous rendering engines rely on some form of precomputation or baking to enable real-time global illumination. This technique computes parts of the lighting at a precompuation stage, and the resulting information is cached in specially-designed data structures e.g., some form of vertex attributes [SKS02, SHHS03, KBS11], textures [SSDS12, SSS*20] or probes [SL17, RLP*20]. At runtime, a query process is typically involved to reconstruct the final images from these precomputed caches. To guarantee the high accuracy of the reconstructed images, the precomputed caches should be generated at a very high sampling rate to suppress any noise or artifact. Baked textures or probes are also need a high resolution to avoid aliasing. Unfortunately, the high sampling rate and high image resolution require a large amount of precompuation time which is not affordable for some rendering systems running on a low-end devices.

In this paper, we investigate the possibility to generate caches for a static scene with a low sampling rate and a low image resolution. We name these caches as *imperfect caches* as they only capture incomplete information of the scene. Undoubtedly, this is beneficial for lowering the time consumption required at the precomputation stage, but will result in annoying noise and aliasing artifacts in the final images after query or interpolation based on traditional methods. With the advent of deep learning [LBH15], as well as its increasing and successful adoption in rendering [TFT*20], it is possible to reconstruct accurate radiance map from any viewpoint in the scene, just using these imperfect caches and some cheap G-buffers (e.g., depth, normal and albedo) generated at runtime.

To achieve the above goal, we develop a new neural rendering framework based on imperfect caches. The imperfect caches are generated by a physically-based path tracer with 32 samples per-pixel (spp), a fairly low sampling rate, and stored at the barycentre of every triangle in a 3D scene. The resolution of the cache is also set to a low level (64×16 in our current implementation), to further reduce the precomputation time. At runtime, a ray differential based query method is first involved to generate imperfect radiance (IR) maps from the imperfect caches close to the primary intersection points. These IR maps will have noise and aliasing artifacts due to the limited sampling rate and cache resolution. Considering the success of data-driven models in Monte Carlo image denoising and reconstruction [GLL*19, KHL19, BVM*17, YNL*21] in recent years, we design a *deep radiance reconstruction* method based on a dedicated convolutional neural network to reconstruct high-quality radiance maps of full global illumination from the IR maps. In this network, a new feature fusion strategy is adopted to better exploit useful contents from cheap G-buffers. Thanks to the light-weight design of the network, the whole pipeline, including cache query and network inference, runs in real time.

In summary, we make the following contributions:

- We introduce imperfect caches to bake outgoing radiance with full global illumination effects at the barycentre of every triangle of a 3D scene. The low sampling rate and low resolution of the caches reduce the time consumption at the precomputation stage.
- We design a deep neural network to infer high-quality images from low-quality inputs with the guidance of G-buffers. The light-weight design of the network enables real-time global illumination for scenes with a moderate scale.
- We adopt a novel feature fusion module in our network architecture to better fuse features from both low-quality inputs and clear G-buffers. This allows us to better exploit useful information from the G-buffers which can serve as the guidance for the final image reconstruction.

It should be noted that the proposed framework is a general way towards lowering the computational cost required at the precomputation stage, while still reconstructing accurate radiance maps from baked imperfect caches. Either vertex/triangle-based baking or texture space baking can be adopted in this framework.

2. Related Work

2.1. Precomputation for Real-time Rendering

In industry, it is very common to use precomputed scene information to achieve real-time global illumination. For instance, light maps have been widely adopted in real-time rendering of purely diffuse surfaces. They usually precompute and bake the global lighting effects as vertex attributes or textures. Schäfer et al. [SSDS12] proposed a memory efficient approach that combines texture- and vertex-based baking. Seyb et al. [SSS*20] described a new method based on light maps to handle low-level light source changes.

Another general attempt to handle complex illumination with precomputation is precomputered radiance transfer (PRT), which was first proposed by Sloan et al. [SKS02, SHHS03]. The rationale of PRT is to represent the lighting and the light transport function with a certain linear basis, such as spherical harmonics (SH), thus making precomputation storage affordable and, meanwhile, approximating the computationally expensive rendering integral at each vertex of a scene with simple operations of the basis. There have been many subsequent developments about basis functions in PRT. Sloan et al. [SLS05] extended it to arbitrarily deformable models by using zonal harmonics (ZH) which approximate spherical functions as sums of circularly symmetric Legendre polynomials around different axes. Tsai et al. [TS06] allowed real-time rendering with comparable quality under high-frequency lighting using spherical radial basis functions and clustered tensor approximation. Xu et al. [XJF*08] presented spherical piecewise constant basis function (SPCBF) for PRT, enabling new effects such as object rotation in all-frequency rendering of dynamic scenes and on-the-fly BRDF editing under rotating environment lighting. Currius et al. [CDAS20] approximated the precomputed local light field in a scene using spherical Gaussians (SG) and adopted a convolutional neural network to predict their parameters, so as to lower

the cost both in memory and performance. Besides, for non-distant illumination, it is not sufficient to sample the lighting only once. Kristensen et al. [KAMJ05] introduced the concept of unstructured light clouds for real-time relighting of scenes illuminated by local light sources. Wang et al. [WR18] developed a novel analytic formula for the spatial gradients of the SH coefficients for uniform polygonal area lights, further adding area light to the PRT framework. However, PRT still has many tough problems. For example, it cannot handle complex light paths, and indirect illumination from area light cannot be processed easily. In contrast, our method can handle complex light paths well, and has no restriction to the light sources in the scene.

2.2. Radiance Caching

Our work is also closely related to irradiance/radiance caching which can be traced back to the seminal work of Ward et al. [WRC88]. This method computes and caches indirect irradiance only at a sparse set of points in the scene, and extrapolates or interpolates these values whenever possible from cache points deemed to be sufficiently close by. However, glossy surfaces would invalidate the Lambertian assumption for irradiance caching algorithms. To handle moderately glossy and non-Lambertian surfaces, Křivánek et al. [KGPB05] suggested using radiance cache, representing the directional domain with spherical harmonics. A wealth of recent works further explored the use of radiance caching in offline [DBN17, MJJG18, ZBN19] and real-time rendering [MRNK21]. Müller et al. [MRNK21] presented a real-time neural radiance caching (NRC) method for path-traced global illumination. This method is designed to handle fully dynamic scenes, and makes no assumptions about the lighting, geometry, and materials. They employed online self-training to provide low-noise training targets and simulate infinite-bounce transport by merely iterating few-bounce training updates. Its high efficiency stems from path truncation and a low sampling rate. Therefore, the results are noisy and always require a denoiser for post-processing. In addition, due to the network capacity and positional encoding, NRC easily suffers from cross-shaped artifacts and over-blurriness of highlights.

The interpolation strategy of various cache records is an inevitable consideration no matter for radiance caching or irradiance caching. Even robust and principled solutions has been proposed [KG09, JDZJ08], they have strict restrictions and requirements for the scene. In our method, we also propose a well designed interpolation strategy and a neural reconstruction strategy to generate high-quality images.

2.3. Probe-based Real-time Rendering

Precomputed environment maps are widely adopted in production to achieve real-time rendering of specular surfaces. A reflected ray can estimate the incident radiance by querying these maps. These ideas originate with the irradiance volume [GSHG98], which precomputes probes to light diffuse objects. Recently, Silvennoinen et al. [SL17] used a sparse set of radiance probes combined with a local reconstruction step to estimate indirect radiance on diffuse surfaces. Different from the probe just saving radiance function, Majercik et al. [ZJPDM19, MMK*21] added the depth information to the probes, so as to avoid light leakage. This method relies on probes at a rather low resolution to estimate irradiance, but updates them at runtime.

Handling glossy reflection paths in these methods mentioned above requires significantly increasing the sample rate. In order to handle glossy reflection, Rodriguez et al. [RLP*20] proposed glossy light probes in which glossy components of radiance are precomputed and stored in sparsely located light probes. Combined with traditional light maps for diffuse lighting, they can interactively renders all light paths in static scenes with opaque objects. However the high time consumption of the precomputation is the main drawback of this approach. In contrast, our method can handle glossy reflections well and effectively reduce the preprocessing time by using low resolution radiance caches generated by a low sampling rate.

2.4. Neural Rendering

Recently, there is a huge interest in applying deep learning techniques to some rendering-related tasks, forming a new field named neural rendering [TFT*20]. For instance, deep learning techniques have been used to facilitate the learning of local light distributions and importance sampling of light paths [ZZ19, MMR*19, BMDS19, ZXS*21]. The networks involved in this task can be trained either offline [BMDS19] or online [MMR*19]. In Monte Carlo (MC) denoising, many learning-based methods also outperform their traditional counterparts. Kalantari et al. [KBS15] adopted a multi-layer perceptron to learn the filter weights for cross-bilateral and cross non-local means filters from training data. Bako et al. [BVM*17] utilized a convolutional neural network to predict filtering kernels for each pixel. Yu et al. [YNL*21] recently presented a self-attention based MC denoising deep learning network, achieving state-of-the-art MC denoising quality. Besides importance sampling and MC denoising, deep learning techniques have also been successfully used in supersampling [XNC*20], gradient domain rendering [KHL19, GLL*19] and deferred rendering [TZN19, GCD*20]. In this paper, we also design a dedicated neural network to recover high-quality images from low-quality imperfect caches.

Our method also bears some similarity with Neural Radiance Field (NeRF) [MST*20] which also relies on neural networks for novel view synthesis of static scenes. However, NeRF fails to represent signals with fine details at moderate resolutions due to its volumetric representation. It is also challenge to handle specular (or highly glossy) materials that our method is adept at and generate images at resolutions greater than a megapixel.

3. Motivation and Challenges

Before we describe our method, we would like to illustrate the motivation behind our work, and the challenges we face.

Arbitrary light paths. For majority of the real-time global illumination techniques, complex light paths with multiple glossy bounces are always expensive to compute. Consequently, a lot of existing approaches, such as light probes [SL17], light maps [SSDS12, SSS*20] and PRT [SKS02], prefer to take the diffuse

reflection into account. For PRT, it is still a challenge to handle indirect illumination stemming from local area lights [WR18]. The goal of our method is to handle light transport with arbitrary complexity, based on some form of precomputed caches. There is also no restriction on the types of light sources used in the scene.

Time consumption of precomputation. In precomputation-based real-time rendering methods, there is always a tradeoff between the precomputation time cost and the reconstructed image quality. Previous methods based on analytical query or interpolation mostly require high-quality caches generated with a long time, to avoid noise and aliasing artifacts. For instance, the glossy probe reprojection method proposed by Rodriguez et al. [RLP*20] takes roughly 30 hours on a cluster to precompute light maps and glossy probes with full global illumination. However, the neural reconstruction strategy which is adopted in our pipeline allows us to generate imperfect caches with a low sampling rate and a low resolution, thus significantly reducing the precomputation time cost.

High runtime frame rate. A high frame rate at runtime is the key to many rendering applications, including video games and virtual reality systems. To achieve this goal, traditional methods make many compromises of geometries, materials or luminaries in the scenes, limiting the lighting effects that can be reproduced. We shift the computational burden of simulating complex global illumination effects to the precomputation stage. During the runtime, a high frame rate can be achieved by an efficient query algorithm and a light-weight neural network for final image reconstruction. Recent developments on specialized performance optimization (e.g., TensorRT) for deep learning further improve the frame rate. Our current implementation allows us to generate a noise-free and aliasing-free image of 1080P (1920×1280) within 11 ms.

4. Method

To address the above challenges and achieve global illumination in real time without noticeable artifacts, we propose a neural rendering pipeline (see Fig.1) that can reproduce a wide range of global illumination effects, including high-frequency glossy interreflection, at any viewpoint in real time. In this section, we start with problem formulation and overview in Section 4.1, and then describe the generation of imperfect caches in Section 4.2. A ray differential based cache query and interpolation strategy is detailed in Section 4.3, while a deep radiance reconstruction method based on a dedicated neural network is exposed in Section 4.4.

4.1. Problem Formulation and Overview

Before we introduce the details of our method, we would like to formalize the problem and also give a high-level overview of the proposed framework.

Our real-time rendering framework comprises two stages: the precomputation stage and the runtime stage. In the first stage, we generate per-triangle imperfect caches of outgoing radiance for a given scene containing K triangles. We denote the set of imperfect caches generate by a low sampling rate path tracer as

$$\mathbb{R}_c = \{\mathbf{C}^{(1)}, \mathbf{C}^{(2)}, ..., \mathbf{C}^{(K)}\} \tag{1}$$

where the k-th cache $\mathbf{C}^{(k)}$ contains the outgoing radiance in the upper hemisphere of the k-th triangle. Some imperfect caches are visualized in Fig.1.

At runtime, we cast primary rays from a given viewpoint $\mathbf{o}$ in order to obtain the relevant information of primary intersection for each pixel. With the necessary information, we generate an imperfect radiance (IR) map $\mathbf{I}_{imp}$ for the viewpoint $\mathbf{o}$ by querying the set of imperfect caches $\mathbb{R}_c$. Here, we employ a ray differential based query and interpolation strategy $\mathcal{Q}$, according to the arrangement of our caches:

$$\mathbf{I}_{imp} = \mathcal{Q}(\mathbb{R}_c, \mathbf{o}). \tag{2}$$

Meanwhile, we also record the G-buffers, including depth $\mathbf{I}_{depth}$, albedo $\mathbf{I}_{albedo}$, normal $\mathbf{I}_{normal}$, at runtime.

The IR map $\mathbf{I}_{imp}$ has noise and aliasing artifacts due to the limited sampling rate and cache resolution. To reconstruct a high-quality image $\hat{\mathbf{I}}$ from the IR map, we resort to a deep radiance reconstruction method in which a light-weight convolutional neural network Φ with trainable parameters Θ is designed to infer $\hat{\mathbf{I}}$ from $\mathbf{I}_{imp}$, as well as some G-buffers, i.e.,

$$\hat{\mathbf{I}} = \Phi_\Theta(\mathbf{I}_{imp}, \mathbf{I}_{depth}, \mathbf{I}_{normal}, \mathbf{I}_{albedo}). \tag{3}$$

This network is trained on multiple examples with ground truth (synthesized by path tracing with a high sampling rate) from each 3D scene. Each training example contains an IR map, a depth map, a normal map, an albedo map and a corresponding ground truth, all of which are captured from a randomly chosen viewpoint in the scene. Our network training minimizes the following error:

$$\varepsilon(\Theta) = \sum_{m=1}^{M} \mathcal{L}(\mathbf{I}_{GT}^{(m)}, \hat{\mathbf{I}}^{(m)}) \tag{4}$$

where $\mathcal{L}$ is the loss function that is used in the image reconstruction network and $\mathbf{I}_{GT}^{(m)}$ is the ground truth for the m-th example.

4.2. Imperfect Caches

One of the key contribution of our real-time rendering framework is the introduction of imperfect caches which are generated at the barycentre of every triangle in the scene. Each imperfect caches is generated as follows. For each sampling point located at a 3D barycentre position $\mathbf{c}$, we render a 180° panorama storing the radiance collected from the center of the probe. Different from some typical radiance caching methods which record the incident radiance, we choose to cache exitant radiance $L_o(\mathbf{c}, \omega_o)$, the radiative energy leaving point $\mathbf{c}$ in direction ω_o in the panoramas. The exitant radiance is computed by path tracing the scene through a hemispherical camera. For best results, large triangles in the scene should be tessellated before generating the caches.

Our neural radiance reconstruction method allows the caches to be imperfect, meaning that we can generate them at a low sampling rate. This differs from some prior methods [RLP*20] which require noise-free caches, and is beneficial for reducing the time cost in the precomputation stage. Besides, we set the resolution of the cache to 64×16, so that the time and memory consumption of preprocessing can be further reduced.

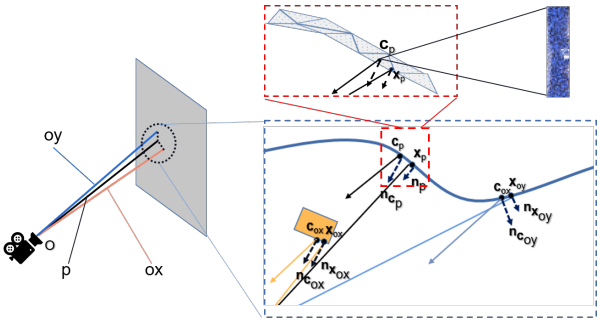

Figure 2: *Ray differential based cache query and interpolation. For each pixel in the image plane, we shoot three rays from the viewpoint* **o***: a primary ray (the black ray) and two offset rays (the orange and blue rays). We then find and query the imperfect cache on the same triangle with each intersection point. The queried radiance values from the caches are blended with customized weights according to the position and normal of the intersection points.*

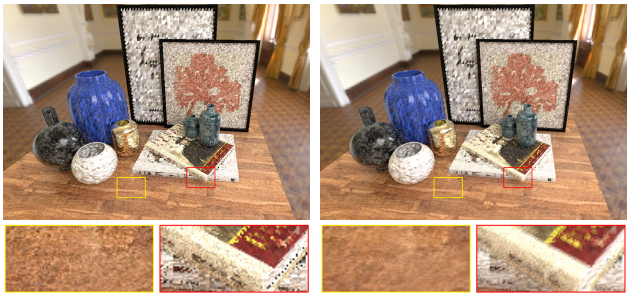

Figure 3: *Impact of the ray differential structure in cache query and interpolation. Here, we compare the query results without (left) and with (right) the ray differential structure.*

4.3. Cache Query and Interpolation

At runtime, we query the imperfect caches from a given viewpoint **o**. This will result in an IR map $\mathbf{I}_{imp}$ which inevitably contains noise and aliasing artifacts due to the imperfect nature of the caches. To suppress noise and aliasing artifacts as much as possible and ease the burden of the network in the following subsection, we leverage a ray differential [Ige99] based query strategy in this step, as shown in Fig.2.

For each image pixel, we cast three rays from the viewpoint **o**: a primary ray (p) and two offset rays (ox and oy). The offset rays are generated by offsetting the primary ray by a pixel along the X and Y axes, respectively. When these rays starting from the camera hit the mesh in the scene, the caches that are located on the same triangle with the intersection points ($\mathbf{x}_p$, $\mathbf{x}_{ox}$ and $\mathbf{x}_{oy}$) are selected to query the radiance information. These caches have the barycentres of $\mathbf{c}_p$, $\mathbf{c}_{ox}$ and $\mathbf{c}_{oy}$ as shown in Fig.2. It should be noted that we do not require the intersection points to be non-specular. Even if a ray hits a transmissive or mirror-reflected material, we do not need to bounce the ray until it hits a non-specular surface. Since we store outgoing radiance in the caches, our framework is free from BRDF sampling and Monte Carlo integration. Therefore, the time consumption in this step is very low.

Finally, we query and interpolate the caches' data based on the information of the barycentric coordinates, distances and surface normals. Specifically, for each ray we obtain the exitant radiance by mapping the incident direction to the cache's local space according to direction of the ray and the normal of the cache's position **c**. After obtaining three radiance values, we average them using a customized weighting scheme to obtain the final pixel value. The weight for each ray is determined by the following heuristic:

$$W = W_d \cdot W_n + \delta \tag{5}$$

where W_d and W_n denote weights calculated according to world positions and surface normals, respectively. δ is a small term to avoid computation error when all other terms are zero. It is set as 0.01 in our currently implementation. Note that world positions and surface normals both have impacts on the weight but are relatively independent. Therefore, they are simply multiplied in our weighting formula. The position weight is computed by

$$W_{d,i\in\{p,ox,oy\}} = e^{-\mathrm{dist}(\mathbf{c}_i,\mathbf{x}_i)} \tag{6}$$

where dist is a function returning the distance between the cache's position $\mathbf{c}_i$ and the intersection point $\mathbf{x}_i$ of the corresponding ray in the scene. This weight penalizes the case in which the distance between the cache's position and the intersection point is too large. The normal weight is computed according to

$$W_{n,i\in\{p,ox,oy\}} = \max(0, (\mathbf{n}_{\mathbf{c}_i} \cdot \mathbf{n}_{\mathbf{x}_i})). \tag{7}$$

The normal weight is large when the normals at the intersection point and the cache's position are very similar, and approaches zero when their dot product is negative.

The effect of employing the ray differential structure in cache query is demonstrated in Fig.3. As seen, if we query the caches using a single ray for each pixel, the IR map will be very noisy and contains severe artifacts such as aliasing due to the limited resolution of caches. Our ray differential based method can alleviate this problem to some extent, resulting in more smooth structures.

4.4. Deep Radiance Reconstruction

Even adopting a ray differential structure in the cache query stage, the imperfect nature of the caches still makes the resulting image low quality. To further remove the noise and aliasing artifacts, a straightforward way is to increase the sampling rate and the cache resolution. However, this will cause a large amount of precomputation time. In our framework, we resort to a deep radiance reconstruction method based on a dedicated neural network to achieve the same effect without introducing too much time consumption in the precomputation stage.

4.4.1. Network Architecture

Our network is a typical encoder-decoder architecture, shown in Fig.4, which extracts multi-scale feature maps from an IR map and the corresponding G-buffers, respectively. The IP map has 3 channels while the G-buffers form 7 channels (3 for normal, 3 for albedo and 1 for depth). Notably, the IR map and the G-buffers have quite

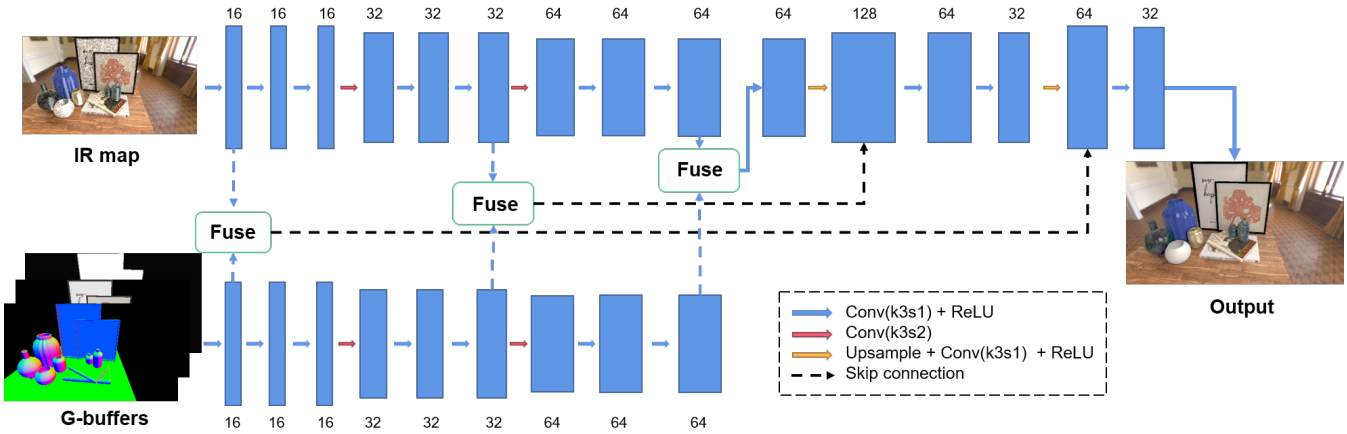

Figure 4: *Detailed architecture of the network used in deep radiance reconstruction. We extract features from IR map and G-buffers respectively using two different branches. The extracted feature maps are fused at multiple scales and then concatenated to the corresponding layers in the decoder using skip connection.*

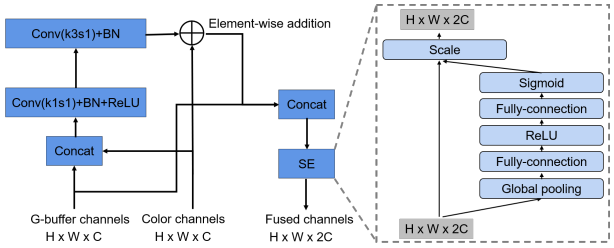

Figure 5: *The details of the fusion module (Fuse in Fig.4) used in the network. Here, SE means a Squeeze-and-Excitation (SE) [HSA*20] block.*

different behaviors: the IR map comprises the same contents with the final image but is plagued with artifacts; the G-buffers have high image quality but can only serve as the guidance. Considering this, we design two branches in the encoder of our network to ensure that different branches extract features from the IR map and the G-buffers properly. Features are extracted at three different scales in the encoder. We downsample the resolution of feature maps using stride-2 convolutions as shown in Fig.4. To better exploit features from both branch, we employ a new feature fusion module which will be explained later. Feature fusion is conducted hierarchically and the fused features are skip-connected to the corresponding layers in the decoder. The skip connections allow the network to preserve more details after passing through a deep neural network. A noise-free and aliasing free image is produced as the output of the network.

4.4.2. Feature Fusion

Considering the different behaviors of IR maps and G-buffers, we develop a new fusion module to fuse features from different branches to combine the advantages of IR maps and G-buffers while compensating for the deficiency of each other. The details are visually explained in Fig.5. With this new feature fusion module, the features from different branches are successively integrated at multiple scales to ensure global smoothness while preserving visually salient details.

In this fusion module, we first generate a residual feature map from G-buffers to perform element-wise addition with color channels. The residual block contains a 1×1 convolution layer to squeeze the channels and a 3×3 convolution layer to generate the residual feature maps. The remained part is similar to concatenation. As we have dual channels of features from both branches, we add a Squeeze-and-Excitation (SE) [HSA*20] block, which can adaptive recalibrate channel-wise feature responses. In this way, the information from G-buffers is merged into the radiance map to enhance sharp edges and correct erroneous estimation.

4.4.3. Loss Function

The loss function we used to train the network in the deep radiance reconstruction stage has two terms:

$$\mathcal{L} = \mathcal{L}_{L1} + \alpha\mathcal{L}_{perceptual} \tag{8}$$

where α is a weight to balance the influence of different terms and is currently set as 0.03.

To ensure pixel-level accuracy, a standard L1 loss is used to evaluate the difference between our reconstructed image $\hat{\mathbf{I}}$ the the ground-truth image $\mathbf{I}_{GT}$. Moreover, a perceptual loss $\mathcal{L}_{perceptual}$ is adopted to improve the perceptual quality of the reconstructed image. The perceptual loss is calculated via the learned perceptual image patch similarity (LPIPS) [ZIE*18], according to the features of $\hat{\mathbf{I}}$ and $\mathbf{I}_{GT}$ obtained by a pretrained VGG19 network [SZ15].

4.4.4. Dataset Preparation and Implementation

We currently train a separate network for each 3D scene in the experiments. We collect several scenes and design two camera paths for each scene. One camera path contains 100 frames, including

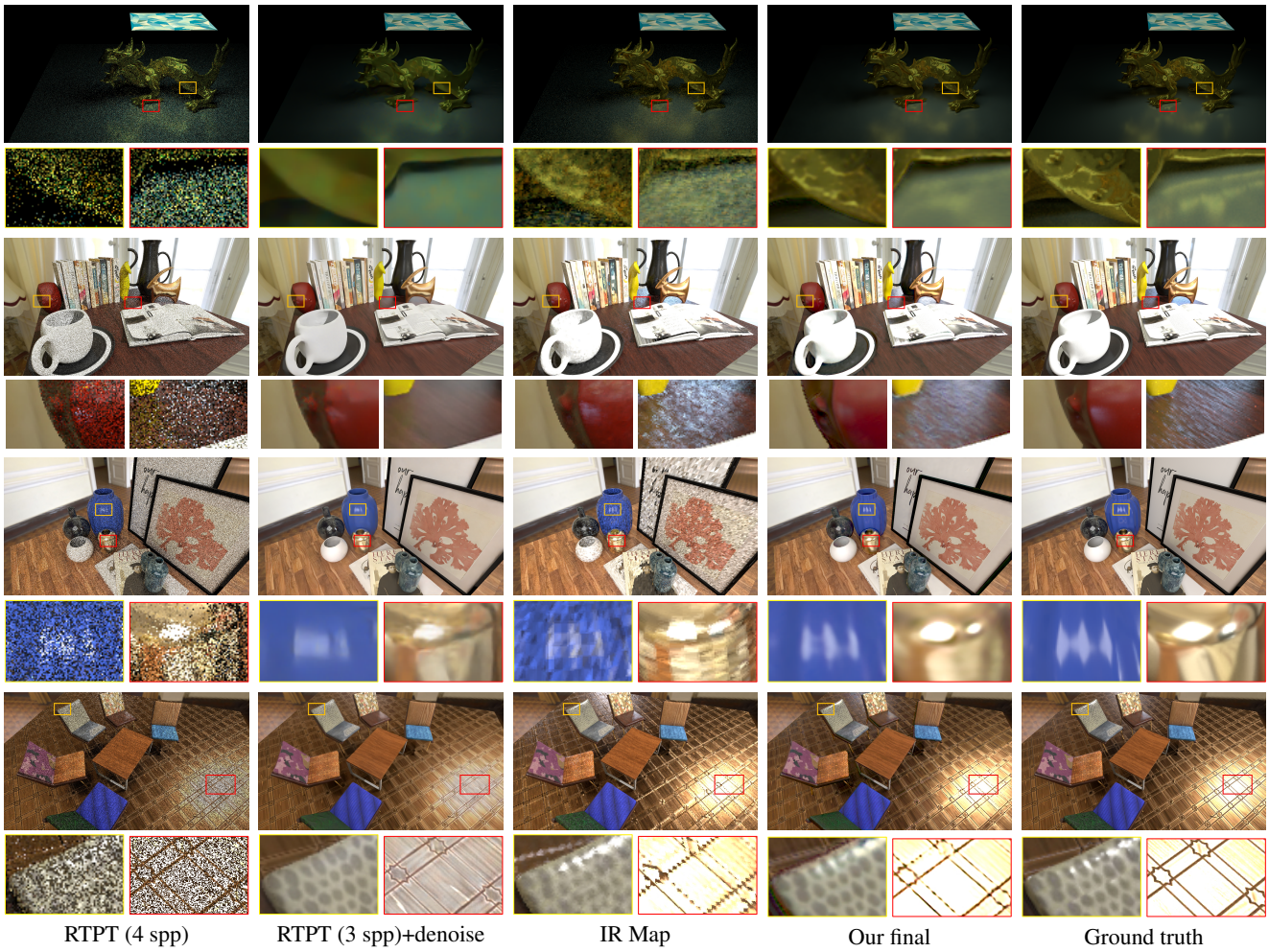

Figure 6: *Equal frame rate comparisons against two real-time baselines: RTPT (4 spp) implemented on top of the Falcor rendering framework [BYC*20] and RTPT (3 spp) with an AI-accelerated denoiser based on [CKS*17]. Here, we provide both IR maps and our reconstructed final images for four test scenes: dragon, tabletop1, tabletop2 and chairs.*

90 frames for training and 10 frames for validation. Another camera path contains 200 frames which are only used for testing. The ground-truth images are generated by an offline path tracer sampled at 4000 spp.

Our neural network is trained using the PyTorch framework [PGM*19]. Mini-batch SGD and Adam optimizer [KB14] are used for optimization. Specifically, we set the minibatch size as 2, β_1 in Adam optimizer as 0.9 and β_2 as 0.999. Exponential learning rate decay is used, with a decay rate of 0.95 per epoch. The initialization of the network follows the default setting in PyTorch.

Because the range of the color and depth in the input is relatively large, logarithm transformation $y = \log(1+x)$ is applied to HDR images to avoid large values before feeding images into our network. The depth is normalized by $y = x/d_{max}$ where d_{max} is the maximum depth in the scene.

5. Results

We have implemented our neural reconstruction network in Pytorch, and network inference is accelerated by TensorRT. The rest, including precomputation of caches, real-time interpolation and query, is all implemented in Direct3D 12. All results are rendered at 1920×1080 on an AMD Ryzen 7 5800X CPU PC with 32 GB memory and NVIDIA RTX 3090 GPU.

To show the advantages of our method, we first compare our method to some real-time rendering methods that also support global illumination. We then analyze the performance of our algorithm. Finally, we evaluate each part of our algorithm on four test scenes (dragon, tabletop1, tabletop2, chairs) to emphasize their value. The first scene comes from Wang et al.'s work [WR18]. This scene contains some complex area light sources to test the effect of our method on different types of light source. The other three also add some elements with different textures and materials with varying levels of glossiness to showcase all kinds of reflections.

Figure 7: *Visually comparing our method against the glossy probe reprojection method proposed by Rodriguez et al. [RLP*20], with equal precomputation time.*

5.1. Qualitative Comparison

Fig. 6 shows selected frames from our supplementary video's camera paths, together with our comparisons and the corresponding ground truth. We accurately capture glossy light paths at real-time frame rate (>90 FPS), including complex secondary glossy effects (e.g., the glossy highlights on the reflected dragon on the floor — right closeup of the first row).

In Fig.6, we compare to two baselines, along with path-traced ground truth. All these results are rendered using the Falcor rendering framework [BYC*20]. Both of the baselines use real-time path tracing (RTPT) implemented on top of the NVIDIA's Falcor framework. The first baseline is a real-time path tracer, denoised with an AI-accelerated denoiser based on [CKS*17]. We give this path tracer the same compute budget as our prototype, resulting in 3 paths per pixel. This method captures the general structure of light paths and provides good results. However, some highlights (e.g., in the reflections on the vases — right closeup of the third row) and texture details (e.g., details of squama on the dragon — left closeup of the first row) are missing due to blurriness caused by the denoiser. The second baseline removes the denoising component and increases the sampling rate for 1 more path per pixel to achieve the same frame rate as the first baseline. This method has strong noise due to insufficient count of samples. In addition, we compare the temporal stability of each method by collecting several consecutive frames generated. In Fig.8, the flickering highlights along the edge of the vase are absent in our method, but are obvious in other real-time baselines. It indicates that our method is temporally stable and does not suffer from flickering artifacts. Due to the low quality of the IR maps, our method occasionally suffers from some artifacts along the boundaries, especially at some places without sufficient G-buffer information (e.g., on the golden pot in Fig. 1). A finer tessellation of the scene could alleviate this problem. Nevertheless, our solution has overall good image quality. The quality is best appreciated over the entire path in the supplement video.

In Fig. 7, we compare our method to the glossy probe reprojection method proposed by Rodriguez et al. [RLP*20]. Like ours, this method can also handle arbitrary paths in real time. However, it's precomputation time cost is very large since high-quality probes captured at a very large sampling rate (2048 spp) and a high resolution (1024×512) are required. To ensure the same precomputation time as ours, probes in the glossy probe reprojection method should be captured at a low sampling rate and a low resolution. However, this leads to obvious noise and aliasing artifacts as highlighted in Fig. 7.

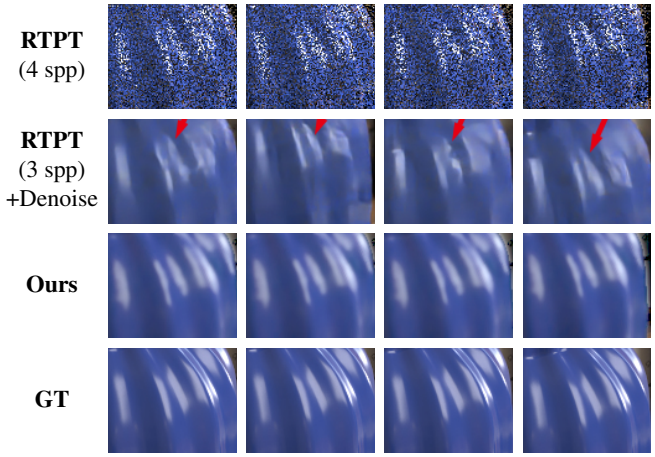

Figure 8: *We compare closeups from four consecutive frames generated by our method, RTPT (4 spp), RTPT (3 spp)+denoiser and GT. Note that the flickering highlights along the edge of the vase are absent in our method, but are obvious in other real-time baselines (marked by red arrows).*

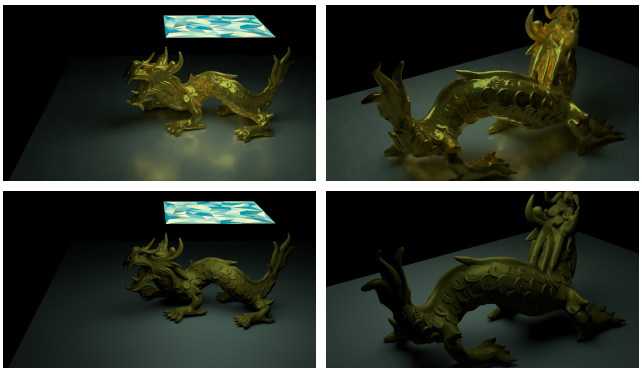

Figure 9: *Visually comparing our method against the Precomputed Radiance Transfer (PRT) method supporting hundreds of area lights [WR18]. The first row is our method's result, while the bottom is PRT's result. In our method, the specular reflection on the floor is well displayed and the details on the dragon are clearer.*

PRT is another famous real-time rendering method based on precomputed data. In Fig.9, we also compare to Wang et al.'s PRT method for polygonal area lights [WR18]. Again, the result is plausible, but glossy effects are missing (e.g., the highlight on the dragon). Besides, their PRT method fails to handle indirect illumination of area lights (e.g., the mirror image on the floor). Our method generates more convincing results that are close to the ground truth.

		Ours		RTPT	
		IR Map	Final	+Denoise	-Denoise
PSNR↑	dragon	30.12	34.14	31.94	23.72
	tabletop	22.57	33.38	27.85	16.47
	tabletop2	26.14	30.81	27.90	18.24
	chairs	20.40	26.30	23.95	16.41
SSIM↑	dragon	0.92	0.98	0.96	0.81
	tabletop	0.80	0.98	0.95	0.63
	tabletop2	0.94	0.97	0.90	0.75
	chairs	0.73	0.89	0.84	0.58
LPIPS↓	dragon	0.257	0.068	0.118	0.386
	tabletop	0.249	0.047	0.062	0.374
	tabletop2	0.076	0.036	0.105	0.257
	chairs	0.369	0.126	0.203	0.485

Table 1: *Quantitative error metrics, using learned perceptual image patch similarity (LPIPS) [ZIE*18], peak signal to noise ratio (PSNR) and structural similarity (SSIM) [WBSS04], comparing RTPT (4 spp), RTPT (3 spp)+denoiser, IR maps, and our final results.*

5.2. Quantitative Evaluation

We perform a quantitative evaluation using learned perceptual image patch similarity (LPIPS) [ZIE*18], peak signal to noise ratio (PSNR) and structural similarity (SSIM) [WBSS04]. We compute the error between the generated and ground-truth images. Error is averaged over 200 frames along the path recorded in each scene. Table 1 summarizes the error for baselines and our method. The error for our method after deep radiance reconstruction is consistently lower than baselines.

5.3. Performance Analysis

The timings and memory consumption for our method, including preprocessing, are shown in Table 2. Rendering times are also averaged over 200 frames. Since our network is trained scene by scene, the time consumption of the preprocessing step consists of three parts: cache precomputation, network training and the generation of ground truth. Time for cache precomputation is approximately linear with the resolution and sample rate. If we want to obtain equivalent high-quality results directly through query and interpolation, cache resolution should be at least 1024×256 (similar to that in [RLP*20]), which means that we need 768 hours for preprocessing. With deep radiance reconstruction, time consumption is significantly reduced to an average of 6.5 hours, which is nearly a 120x speedup in preprocessing. At runtime, we first perform ray casting, which uses only <1 ms of GPU time for query and interpolation. Then, thanks to our light-weight network architecture and TensorRT for the acceleration of network inferring, it only costs roughly 10 ms for the reconstruction of a 1080P (1920×1080) image.

	dragon	chairs	tabletop2	tabletop
Preprocess				
Cache precomput.	2.8 h	4 h	3.2 h	3.6 h
Network training	1.5 h	2 h	2 h	2 h
Ground truth	1.2 h	1.3 h	1.6 h	1.5 h
Total	5.5 h	7.3 h	6.8 h	7.1 h
Runtime				
Query/Interpolation	0.6 ms	0.7 ms	0.8 ms	0.7 ms
Network infer	10.1 ms	10.1 ms	10.1 ms	10.1 ms
Total	10.7 ms	10.8 ms	10.9 ms	10.8 ms
VRAM for caches	3.9 GB	4.2 GB	3.2 GB	5.7 GB
VRAM for network	2.9 GB	2.9 GB	2.9 GB	2.9 GB

Table 2: *Timings and memory consumption of our method. Rendering timings average costs over frames in our videos.*

Figure 10: *Ablation experiment for using higher-resolution caches calculated with equivalent preprocessing time. Left: image generated by interpolation with higher-resolution caches.*

5.4. Ablation Study

We analyze our method with various ablation experiments in this section.

Effectiveness of deep radiance reconstruction. In Fig.10, we show a visual comparison to the results generated by querying and interpolating higher-resolution (64 × 32) caches, which are calculated with equivalent preprocessing time as our method. The experiment demonstrates the quality gained from our reconstruction network.

Impact of the loss term. As aforementioned, our method adopts a customized loss function, which is a weighted sum of L1 loss and perceptual loss. To understand its impact on the reconstruction quality, we conduct several ablation experiments with alternative ways for it, i.e., using L1 or perceptual loss only. The results are reported in Table 3. We observe about 4dB improvement in PSNR by using our customized loss function compared to other losses. And the other two quantitative indicators (SSIM/LPIPS) are also significantly better. This indicates the superiority of the proposed loss function. Meanwhile, we also display the impact of different loss terms through the closeups of final results, shown in Fig.11.

Effectiveness of the fusion module. We also conduct an ablation experiment to analyze the quality improvements stemming from

	Perceptual Loss	L1 Loss	Our Loss
PSNR	13.66	29.41	33.38
SSIM	0.72	0.92	0.98
LPIPS	0.132	0.091	0.047

Table 3: *Ablation experiment quantitative result for our loss function, tested on the tabletop scene.*

Figure 11: *Ablation experiment for our customized loss function. Left: the highlights on the outer surface and the interior of the bottle reconstructed by the well trained network using L1 loss. Unreasonable color on the inner wall and ghost appears at the edge of the glossy reflections area. Middle: the reconstructed result by the network trained by our customized Loss function. Right: ground truth image.*

our feature fusion module. We compare our fuse strategy with the one utilizing the G-buffer channels by directly concatenating. In Fig. 12, we provide a visual comparison to demonstrate the contribution of our feature fusion module. When the network is trained without the fusion module, obvious artifacts occur.

5.5. Limitations

Our method achieves plausible results with a satisfactory accuracy in many cases; however it has some limitations.

Generalization ability of the model. Since we choose to train a network for each scene to maximize its quality, we need additional network training and ground truth generation for preprocessing stage. If the generalization ability of the reconstruction network is improved, the preprocessing time consumption of our method can be further reduced.

Memory consumption of the precomputation. Our caches are located on the barycentre of each triangle mesh instance in the scene, therefore the complexity of the scene will become an important factor affecting the memory cost of our method. Besides, in order to retain both high and low frequency information, we still store the panorama directly as HDR images. We believe that if a reasonable way can be found to compress the caches and keep all the information as much as possible, the application space of this method will be greatly improved.

Restriction on scene and lighting. Since outgoing radiance from each scene is already computed in the precomputation stage, we can handle complex light paths and generate high-quality images

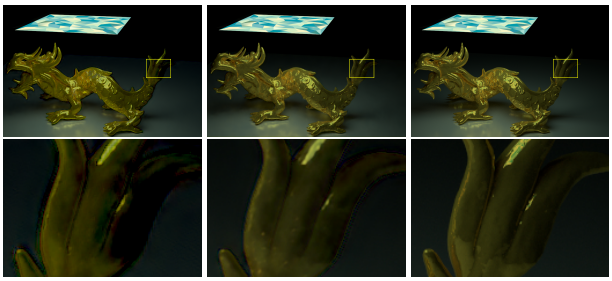

Figure 12: *Ablation experiment for our feature fusion module. Left: the reconstructed result by the network without our fusion module. Middle: the reconstructed result by the network with our fusion module. Right: ground truth image.*

in real-time frame rate. However, this comes at the limitation to static scenes and lighting.

Tessellation of the triangles. Our approach generates caches in the center of each triangle mesh. Hence, the quality of our results relies on the tessellation of the scene. Even the scene is tessellated sufficiently from a far view, large triangles may still appear after zoom-in of the camera. It would be possible to use some mipmap strategies to improve the image quality for zoom-in cases.

6. Conclusions

In this paper, we have presented a new approach for real-time global illumination using information from imperfect caches stored at the barycentre of every triangle in a 3D scene and a light-weight neural network. Our work includes three main technical contributions: the introduction of imperfect caches to lower the time consumption at the precomputation stage due to the low sampling rate and low resolution of the caches, a light-weight deep neural network to infer high-quality images from low-quality inputs with high efficiency and a new feature fusion module to better fuse features from both low-quality inputs and clear G-buffers. Our solution allows real-time walk-through with global illumination for opaque scenes based on precomputation: the merit is that diffuse and glossy light paths are both precomputed, such that rendering at runtime can be completed at a fast speed through our query, interpolation and deep reconstruction.

Acknowledgements

We would like to thank the reviewers for their valuable feedback. 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

[BMDS19] BAKO S., MEYER M., DEROSE T., SEN P.: Offline deep importance sampling for monte carlo path tracing. *Computer Graphics Forum 38*, 7 (2019), 527–542. 3

[Bur20] BURGESS J.: Rtx on the nvidia turing gpu. *IEEE Micro 40*, 2 (2020), 36–44. 2

[BVM*17] BAKO S., VOGELS T., MCWILLIAMS B., MEYER M., NOVÁK J., HARVILL A., SEN P., DEROSE T., ROUSSELLE F.: Kernel-predicting convolutional networks for denoising monte carlo renderings. *ACM TOG 36*, 4 (Aug. 2017). 2, 3

[BYC*20] BENTY N., YAO K.-H., CLARBERG P., CHEN L., KALLWEIT S., FOLEY T., OAKES M., LAVELLE C., WYMAN C.: The Falcor rendering framework, 08 2020. https://github.com/NVIDIAGameWorks/Falcor. URL: https://github.com/NVIDIAGameWorks/Falcor. 7, 8

[CDAS20] CURRIUS R. R., DOLONIUS D., ASSARSSON U., SINTORN E.: Spherical gaussian light-field textures for fast precomputed global illumination. *Comput. Graph. Forum 39*, 2 (2020), 133–146. 2

[CJ16] CHRISTENSEN P. H., JAROSZ W.: The path to path-traced movies. *Foundations and Trends in Computer Graphics and Vision 10*, 2 (2016), 103–175. 2

[CKS*17] CHAITANYA C. R. A., KAPLANYAN A. S., SCHIED C., SALVI M., LEFOHN A., NOWROUZEZAHRAI D., AILA T.: Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder. *ACM Trans. Graph. 36*, 4 (jul 2017). 2, 7, 8

[DBN17] DUBOUCHET R. A., BELCOUR L., NOWROUZEZAHRAI D.: Frequency based radiance cache for rendering animations. In *28th Eurographics Symposium on Rendering, Rendering - Experimental Ideas & Implementations, EGSR 2017, EI&I Track, Helsinki, Finland, 19-21 June 2017* (2017), Zwicker M., Sander P. V., (Eds.), Eurographics Association, pp. 41–53. 3

[FWHB21] FAN H., WANG R., HUO Y., BAO H.: Real-time monte carlo denoising with weight sharing kernel prediction network. *Computer Graphics Forum 40*, 4 (2021), 15–27. 2

[GCD*20] GAO D., CHEN G., DONG Y., PEERS P., XU K., TONG X.: Deferred neural lighting: free-viewpoint relighting from unstructured photographs. *ACM Trans. Graph. 39*, 6 (2020), 258:1–258:15. 3

[GLL*19] GUO J., LI M., LI Q., QIANG Y., HU B., GUO Y., YAN L.-Q.: Gradnet: Unsupervised deep screened poisson reconstruction for gradient-domain rendering. *ACM TOG 38*, 6 (Dec. 2019). 2, 3

[GSHG98] GREGER G., SHIRLEY P., HUBBARD P. M., GREENBERG D. P.: The irradiance volume. *IEEE Computer Graphics and Applications 18*, 2 (1998), 32–43. 3

[Har20] HARADA T.: Hardware-accelerated ray tracing in amd radeon prorender 2.0. https://gpuopen.com/learn/radeon-prorender-2-0/, 2020. 2

[HSA*20] HU J., SHEN L., ALBANIE S., SUN G., WU E.: Squeeze-and-excitation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence 42*, 8 (2020), 2011–2023. 6

[Ige99] IGEHY H.: Tracing ray differentials. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., p. 179–186. URL: https://doi.org/10.1145/311535.311555, doi:10.1145/311535.311555. 5

[JDZJ08] JAROSZ W., DONNER C., ZWICKER M., JENSEN H. W.: Radiance caching for participating media. *ACM Trans. Graph. 27*, 1 (2008), 7:1–7:11. 3

[KAMJ05] KRISTENSEN A. W., AKENINE-MÖLLER T., JENSEN H. W.: Precomputed local radiance transfer for real-time lighting design. *ACM Trans. Graph. 24*, 3 (jul 2005), 1208–1215. 3

[KB14] KINGMA D., BA J.: Adam: A method for stochastic optimization. *Computer Science* (2014). 7

[KBS11] KAVAN L., BARGTEIL A. W., SLOAN P.-P.: Least Squares Vertex Baking. *Computer Graphics Forum* (2011). 2

[KBS15] KALANTARI N. K., BAKO S., SEN P.: A machine learning approach for filtering monte carlo noise. *ACM Trans. Graph. 34*, 4 (jul 2015). 3

[KG09] KRIVÁNEK J., GAUTRON P.: *Practical Global Illumination with Irradiance Caching*. Synthesis Lectures on Computer Graphics and Animation. Morgan & Claypool Publishers, 2009. 3

[KGPB05] KRIVÁNEK J., GAUTRON P., PATTANAIK S. N., BOUATOUCH K.: Radiance caching for efficient global illumination computation. *IEEE Trans. Vis. Comput. Graph. 11*, 5 (2005), 550–561. 3

[KHL19] KETTUNEN M., HÄRKÖNEN E., LEHTINEN J.: Deep convolutional reconstruction for gradient-domain rendering. *ACM TOG 38*, 4 (Aug. 2019). 2, 3

[KIM*19] KOSKELA M., IMMONEN K., MÄKITALO M., FOI A., VIITANEN T., JÄÄSKELÄINEN P., KULTALA H., TAKALA J.: Blockwise multi-order feature regression for real-time path-tracing reconstruction. *ACM Trans. Graph. 38*, 5 (June 2019). 2

[KKW*13] KELLER A., KARRAS T., WALD I., AILA T., LAINE S., BIKKER J., GRIBBLE C., LEE W.-J., MCCOMBE J.: Ray tracing is the future and ever will be... In *ACM SIGGRAPH 2013 Courses* (New York, NY, USA, 2013), SIGGRAPH '13, Association for Computing Machinery. 2

[KVBB*19] KELLER A., VIITANEN T., BARRÉ-BRISEBOIS C., SCHIED C., MCGUIRE M.: Are we done with ray tracing? In *ACM SIGGRAPH 2019 Courses* (New York, NY, USA, 2019), SIGGRAPH '19, Association for Computing Machinery. 2

[LBH15] LECUN Y., BENGIO Y., HINTON G.: Deep learning. *Nature 521* (2015), 436. doi:10.1038/nature14539. 2

[MJJG18] MARCO J., JARABO A., JAROSZ W., GUTIERREZ D.: Second-order occlusion-aware volumetric radiance caching. *ACM Trans. Graph. 37*, 2 (2018), 20:1–20:14. 3

[MMK*21] MAJERCIK Z., MUELLER T., KELLER A., NOWROUZEZAHRAI D., MCGUIRE M.: Dynamic diffuse global illumination resampling. In *ACM SIGGRAPH 2021 Talks* (2021). 3

[MMR*19] MÜLLER T., MCWILLIAMS B., ROUSSELLE F., GROSS M., NOVÁK J.: Neural importance sampling. *ACM Trans. Graph. 38*, 5 (oct 2019). 3

[MRNK21] MÜLLER T., ROUSSELLE F., NOVÁK J., KELLER A.: Real-time neural radiance caching for path tracing. *ACM TOG 40*, 4 (Aug. 2021). 3

[MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV* (2020). 3

[PGM*19] PASZKE A., GROSS S., MASSA F., LERER A., BRADBURY J., CHANAN G., KILLEEN T., LIN Z., GIMELSHEIN N., ANTIGA L., DESMAISON A., KÖPF A., YANG E. Z., DEVITO Z., RAISON M., TEJANI A., CHILAMKURTHY S., STEINER B., FANG L., BAI J., CHINTALA S.: Pytorch: An imperative style, high-performance deep learning library. *CoRR abs/1912.01703* (2019). 7

[RLP*20] RODRIGUEZ S., LEIMKÜHLER T., PRAKASH S., WYMAN C., SHIRLEY P., DRETTAKIS G.: Glossy probe reprojection for interactive global illumination. *ACM Trans. Graph. 39*, 6 (nov 2020). 2, 3, 4, 8, 9

[SABB18] SANDY M., ANDERSSON J., BARRÉ-BRISEBOIS C.: Directx: Evolving microsoft's graphics platform. Game Developers Conference 2018, 2018. 2

[SHHS03] SLOAN P.-P., HALL J., HART J., SNYDER J.: Clustered principal components for precomputed radiance transfer. *ACM Trans. Graph. 22*, 3 (jul 2003), 382–391. 2

[SKS02] SLOAN P.-P., KAUTZ J., SNYDER J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques* (2002). 2, 3

[SKW*17] SCHIED C., KAPLANYAN A., WYMAN C., PATNEY A., CHAITANYA C. R. A., BURGESS J., LIU S., DACHSBACHER C., LEFOHN A., SALVI M.: Spatiotemporal variance-guided filtering: Real-time reconstruction for path-traced global illumination. In *Proceedings of High Performance Graphics* (New York, NY, USA, 2017), HPG '17, Association for Computing Machinery. 2

[SL17] SILVENNOINEN A., LEHTINEN J.: Real-time global illumination by precomputed local reconstruction from sparse radiance probes. *ACM Trans. Graph. 36*, 6 (2017), 230:1–230:13. 2, 3

[SLS05] SLOAN P. J., LUNA B., SNYDER J. M.: Local, deformable precomputed radiance transfer. *ACM Trans. Graph. 24*, 3 (2005), 1216–1224. 2

[SSDS12] SCHÄFER H., SÜSSMUTH J., DENK C., STAMMINGER M.: Memory efficient light baking. *Computers & Graphics 36*, 3 (2012), 193–200. Novel Applications of VR. 2, 3

[SSS*20] SEYB D., SLOAN P., SILVENNOINEN A., IWANICKI M., JAROSZ W.: The design and evolution of the uberbake light baking system. *ACM Trans. Graph. 39*, 4 (2020), 150. 2, 3

[SZ15] SIMONYAN K., ZISSERMAN A.: Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (2015), Bengio Y., LeCun Y., (Eds.). URL: http://arxiv.org/abs/1409.1556. 6

[TFT*20] TEWARI A., FRIED O., THIES J., SITZMANN V., LOMBARDI S., SUNKAVALLI K., MARTIN-BRUALLA R., SIMON T., SARAGIH J., NIESSNER M., PANDEY R., FANELLO S., WETZSTEIN G., ZHU J.-Y., THEOBALT C., AGRAWALA M., SHECHTMAN E., GOLDMAN D. B., ZOLLHÖFER M.: State of the art on neural rendering. *Computer Graphics Forum 39*, 2 (2020), 701–727. 2, 3

[TS06] TSAI Y.-T., SHIH Z.-C.: All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. *ACM Trans. Graph. 25*, 3 (jul 2006), 967–976. 2

[TZN19] THIES J., ZOLLHÖFER M., NIESSNER M.: Deferred neural rendering: image synthesis using neural textures. *ACM Trans. Graph. 38*, 4 (2019), 66:1–66:12. 3

[WBSS04] WANG Z., BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process. 13*, 4 (2004), 600–612. 9

[WR18] WANG J., RAMAMOORTHI R.: Analytic spherical harmonic coefficients for polygonal area lights. *ACM Trans. Graph. 37*, 4 (2018), 54:1–54:11. 3, 4, 7, 8

[WRC88] WARD G. J., RUBINSTEIN F. M., CLEAR R. D.: A ray tracing solution for diffuse interreflection. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1988, Atlanta, Georgia, USA, August 1-5, 1988* (1988), Beach R. J., (Ed.), ACM, pp. 85–92. 3

[XJF*08] XU K., JIA Y., FU H., HU S., TAI C.: Spherical piecewise constant basis functions for all-frequency precomputed radiance transfer. *IEEE Trans. Vis. Comput. Graph. 14*, 2 (2008), 454–467. 2

[XNC*20] XIAO L., NOURI S., CHAPMAN M., FIX A., LANMAN D., KAPLANYAN A.: Neural supersampling for real-time rendering. *ACM TOG 39*, 4 (July 2020). 3

[YNL*21] YU J., NIE Y., LONG C., XU W., ZHANG Q., LI G.: Monte carlo denoising via auxiliary feature guided self-attention. *ACM Trans. Graph. 40*, 6 (dec 2021). 2, 3

[ZBN19] ZHAO Y., BELCOUR L., NOWROUZEZAHRAI D.: View-dependent radiance caching. In *Proceedings of the 45th Graphics Interface Conference 2019, Kingston, Ontario, Canada, May 28-31, 2019* (2019), Tagliasacchi A., Teather R. J., (Eds.), Canadian Human-Computer Communications Society / ACM, pp. 22:1–22:9. 3

[ZIE*18] ZHANG R., ISOLA P., EFROS A. A., SHECHTMAN E., WANG O.: The unreasonable effectiveness of deep features as a perceptual metric, 2018. 6, 9

[ZJPDM19] ZANDER M., JEAN-PHILIPPE G., DEREK N., MORGAN M.: Dynamic diffuse global illumination with ray-traced irradiance fields. *Journal of Computer Graphics Techniques (JCGT), vol. 8, no. 2, 1-30, 2019* (2019). URL: http://jcgt.org/published/0008/02/01/. 3

[ZSWL21] ZHUANG T., SHEN P., WANG B., LIU L.: Real-time denoising using brdf pre-integration factorization. *Computer Graphics Forum 40*, 7 (2021), 173–180. 2

[ZXS*21] ZHU S., XU Z., SUN T., KUZNETSOV A., MEYER M., JENSEN H. W., SU H., RAMAMOORTHI R.: Photon-driven neural reconstruction for path guiding. *ACM Trans. Graph. 41*, 1 (nov 2021). 3

[ZZ19] ZHENG Q., ZWICKER M.: Learning to importance sample in primary sample space. *Computer Graphics Forum 38*, 2 (2019), 169–179. 3