Screen Space Global Illumination with Dynamic Light Probes

Beibei Wang[1], Huw Bowles[2†], AJ Weeks[2]

[1]Nanjing University of Science and Technology
[2]Electric Square Ltd

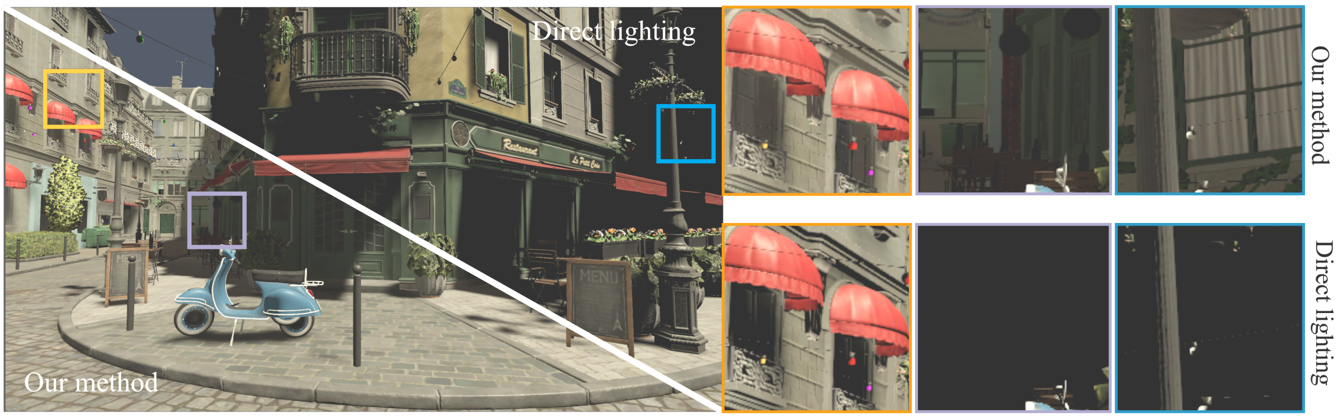

Figure 1: *Our method enables a fast computation of global illumination in the BistroExterior Scene, with extra 0.34 ms.*

Abstract

Global Illumination (GI) plays a crucial role in rendering realistic results for video game production. Computing indirect illumination is costly due to the complex light transport. Existing real-time GI approaches in video games typically require heavy precomputation that imposes a significant burden on production, and limits the range of lighting conditions achievable at run-time. Recently, raytracing based approaches avoid the expensive precomputation and support dynamic light sources and geometries. However, they are still expensive at run-time. In this paper, we propose a lightweight global illumination computation approach, which is much more efficient than raytracing based approaches. We derive a progressive spherical harmonics projection model to update light probes efficiently at run time by spreading the calculation over multiple frames. We demonstrate screen space global illumination (SSGI) in this framework by updating the probes using radiance samples from the frame buffer. This allows dynamic changes in scene and lighting and progressively accumulate multiple bounces of indirect light. Finally, we demonstrate our method can also be used for direct illumination for area lights or environment maps, via injecting direct light sources into the probes. Compared to existing approaches, our method is suitable for device with low-performance. Since our method inherits the drawbacks of screen space approaches, our method assumes adequate direct illumination in the screens.

CCS Concepts
• ***Computing methodologies*** $\rightarrow$ ***Rendering;***

1. Introduction

Global illumination enhances the realism significantly in real-time rendering applications, however, it's computational costly even for diffuse surfaces, due to the complex light transport. In this paper, we focus on diffuse GI only and propose a lightweight GI approach.

[†] Beibei Wang and Huw Bowles contribute equally.

For this reason, the predominant methods for achieving realistic lighting are to either use a set of probes [GSHG98] in the world that sample the illumination, or to bake the incident irradiance on surfaces of the scene into lightmaps. These approaches are efficient and yield high quality results, but are so time-consuming that a significant portion of development budget must be invested in maintaining the precomputed data. This slows down iteration and hampers the creative process. These methods also impose constraints on the dynamic changes to the scene geometry, materials, and lighting conditions, which must remain close to what was used at baking time.

To overcome these challenges, a number of directions have been explored to compute GI at run-time instead of baking it offline. Majercik et al. [MGNM19] proposed a dynamic solution for diffuse global illumination (DDGI). They represent the light field with irradiance probes and update both the irradiance and visibility with ray tracing at run-time. Their method allows geometry and lighting changes at run-time and is able to produce high quality images. However, this method relies heavily on ray tracing, which makes it prohibitively expensive. Furthermore, their probes use an octahedral representation, which is less compact than SH representations, especially when there are large number of probes in the scene.

To solve these issues, we propose a lightweight dynamic diffuse global illumination based on irradiance probes. We first derive a progressive projection model to update spherical harmonic (SH) representations, which enables the dynamic changing of targeting functions. Then we introduce the progressive SH projection model for light probes, and update the irradiance of light probes by accumulating lighting from screen space buffers, which is much cheaper than raytracing. As a result, our method is able to produce diffuse GI with much less rendering cost, compared to DDGI. We also apply our method for directing illumination (DI) computation of area light sources and environment maps, by injecting the lighting into the probes. Thanks to the high performance and lightweight implementation, our method has been used in a mobile game.

To summarize, our main contributions include:

- an improved SH projection model which allows the per-light probe irradiance distribution function to be updated progressively,
- a dynamic light probes method, using screen-space buffers to compute multi-bounce lighting, and
- a simple solution for direct lighting of area light sources and environment maps.

2. Previous Work

In this section, we first review the light probes based real-time global illumination approaches and then briefly review the other real-time GI approaches.

2.1. Light probe based approaches.

The irradiance volume [GSHG98, NPG05] subdivides scenes into discretized points and precomputes the irradiance distribution function at these points, called light probes. During the run-time, these light probes are interpolated across shading points to compute the indirect illumination of static objects or moving objects. Spherical harmonics (SH) is suitable to represent low frequency effects [RH01]. The irradiance volume can provide convincing global illumination with low cost, so it's widely used in games [Tat05], [HHE11].

The placement of light probes can be regular or irregular [Tat05]. A more advanced probe placement method was proposed by Wang et al. [WKKN19]. The regular three-dimensional grid allows automatic interpolation at the cost of extra storage cost. An irregular probe placement requires less storage, but also requires manual interpolation, thereby forfeiting hardware acceleration. Silvennoinen et al. [SL17] proposed to place sparse radiance probes in the scene and reconstruct the incident radiance field from them. Furthermore, gradients [AKDS04] along the three dimensions of the irradiance can be used for interpolation, for higher quality. In our paper, we use a cascaded grids for the probe placement, similar to Halen et al. [HBHB21] which requires less storage cost for open scenes.

Several later works further improved upon irradiance volumes. Kontkanen et al. [KL06] proposed a new sampling strategy for precomputation stage to reduce aliasing caused by the low resolution of the grid. Vendler [Ven07] introduced a dynamic solution by transferring from volume to volume between frames to solve dynamic lighting and scenes. Jendersie et al [JKG16] proposed a hybrid method for multiple bounce indirect illumination that combines hierarchical radiosity and (ir)radiance volumes under dynamic lighting conditions. The main difference between these works and ours is that we use the screen frame buffer to update the volumes instead of an extra structure. The screen space data, e.g. GBuffer and radiance buffer, is obtained almost for free.

Glossy GI. Silvennoinen and Timonen [ST15] introduced reflection probes to simulate glossy reflections and they also extended global illumination volumes to store reflection probe visibility. Manson and Sloan [MS16] proposed a fast filter method for glossy lobes. Rodriguez et al. [RLP*20] introduced an approach for handling complex light paths with multiple glossy interactions, but it cannot be used in real-time rendering. Our work targets at diffuse GI, and leave glossy reflection for the future work.

Light leaking is a problem for all these probe methods, which is caused by interpolation of probes across occluders. This issue is alleviated by McGuire et al. [MMNL17], which caches a variance shadow map. The visibility problem can also be solved using Imperfect Shadow Maps (ISM) [RGK*08]. In our paper, we do not consider the visibility, since our method targets at the low-performance devices.

More recently, Majercik et al. [MGNM19] combined light probes with ray tracing to support fully dynamic diffuse global illumination (DDGI). Compared with DDGI, our method targets the same problem, dynamic diffuse GI, but with different solutions. We use screen space buffers to gather lighting information, rather than raytracing. This leads to much higher performance than DDGI, although our method inherits the drawbacks of other screen-space methods, namely view dependent behavior.

2.2. Other GI approaches

We briefly review the real-time GI approaches. A wider review of other GI approaches could be found in Müller et al. [MRNK21] and Ritschel et al. [RDGK12].

Besides light probes, there are several groups of real-time diffuse GI approaches, including virtual point lights (VPL) [Kel97] and point-based approaches [Chr08, HREB11], Light propagation volumes (LPV) [KD10], Precomputed radiance transfer (PRT) [SKS02, NRH03].

VPLs [Kel97] can also be used to compute indirect illumination, by treating the interactions at the surface as virtual point lights. The visibility of these VPLs can be computed by reflective shadowing maps (RSM) [DS05] or imperfect shadowing maps [RGK*08, REH*11]. VPL approaches support fully dynamic scenes, however, they have sever issues, including the singularity issue and the lack of temporal coherence.

Point based approaches [Chr08] was first proposed for offline rendering, by caching surfels with direct lighting. These surfels are treated as a coarse geometry representation, which makes the visibility computation efficient. Later, it is accerated to reach interactive frame rate [REG*09] or real-time frame rate [HREB11]. Point based GI do not support dynamic lighting and geometry, which constrains its application.

Light propagation volumes [KD10] use RSM to sample directly lit surfaces and propagate the radiance in the volume for indirect illumination. This approach support fully-dynamic scenes. Our method is similar to theirs. Our method uses the screen space buffer to inject the light probes and propagate the contribution with more iterations, while they use the RSM to inject the radiance volume, and propagate the contribution with diffusion.

Precomputed radiance transfer (PRT) approaches precompute visibility, bidirectional reflectance distribution function (BRDF) and the lighting with some basis functions, like SH [SKS02, KSS02] or wavelets [NRH03] and compute their product at run time. These methods support dynamic lighting and high-frequency materials, at the cost of fixed geometries.

Recently, thanks to advanced sampling strategies [OLK*21], like resampling importance sampling (RIS), the Monte Carlo based path tracing with low sample rate can also provide less-noise results. By combining RIS with DDGI [MMK*21], the light leaking and artifact issues can be alleviated. These methods are able to provide high-quality results and are full-dynamic scenes, but their computation is too heavy for real-time applications for now.

(Ir)radiance caching [WRC88, KGPB05] have been proposed to improve GI performance, by caching the irradiance/radiance distributions at some shading points and reuse them for rendering. Recently, radiance caching is combined with deep learning approaches by Müller et al. [MRNK21] and enables real-time rendering.

Screen space based approaches [NRS14, MMNL16] usually use a deep frame buffer to reconstruct a coarse representation of the geometry around a pixel. In our approach, we use use the screen-space framebuffer as the indirect lighting source to inject the light probes. Compared to other sources (RSM or world space representation), the screen space frame buffer is simple and cheap, although inherits the drawbacks of all screen-space approaches. Thus, our method assumes adequate direct illumination in the screens.

3. Background

In this section, we first briefly review the theory of spherical harmonics and then recap the general idea of light probes.

3.1. Spherical Harmonics

The Spherical Harmonics (SH) basis is well suited to representing low-frequency functions, such as the distribution of irradiance incident to a point in space. We give a brief review of SH here and direct the reader to [Gre03] for further details.

The basis functions are defined in spherical coordinates and given by

$$y_l^m = \begin{cases} \sqrt{2}K_l^m \cos(m\varphi)P_l^m(\cos\theta) & m > 0, \\ \sqrt{2}K_l^m \sin(-m\varphi)P_l^{-m}(\cos\theta) & m < 0, \\ K_l^0 P_l^0(\cos\theta) & m = 0. \end{cases} \tag{1}$$

Where l is the band, which is a positive integer starting from 0, and m takes signed integer values from $-l$ to l. Each band corresponds to a different frequency. P is the associated Legendre polynomial and K is a scaling factor to normalize the functions:

$$K_l^m = \sqrt{\frac{(2l+1)}{4\pi}\frac{(l-|m|)!}{(l+|m|)!}}. \tag{2}$$

Projecting a spherical function f onto this basis yields a set of coefficients

$$c_l^m = \int_s f(s) y_l^m(s) \mathrm{d}s. \tag{3}$$

The approximated function $\hat{f}$ of the original function is then reconstructed by

$$\hat{f}(s) = \sum_{l=0}^{n-1} \sum_{m=-l}^{l} c_l^m y_l^m(s). \tag{4}$$

In the following sections, we omit the band indices and use the shorthand for clarity

$$\hat{f}(s) = cy(s). \tag{5}$$

3.2. Light Probe Method

Greger et al. [GSHG98] first proposed light probes to simulate indirect diffuse illumination for real-time applications.

Their method has two steps: an offline precomputation step, and a run-time rendering step. In the precomputation step, a large number of light probes are placed in the scene. Each light probe then evaluates the irradiance distribution function at its given position, and then stores the result using SH coefficients (see Equation 3). Since the irradiance distribution function of a diffuse surface has

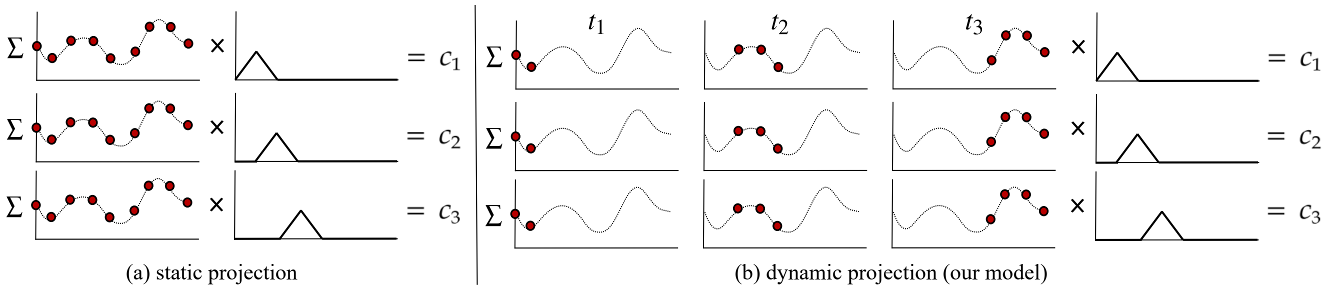

Figure 2: *In the static projection model, the samples from the functions are projected into the basis functions, resulting in the coefficients to represent the input functions. In our model (dynamic projection model), we only project subset of the samples at one time and then process this partial projection multiple times. We accumulate the coefficients during the entire process.*

low frequency, only three bands of SH coefficients [RH01] are required for a reasonable result. During rendering, the SH coefficients of light probes around the shading point are interpolated to reconstruct the irradiance distribution function at that point. The surface normal is then used to compute the final indirect illumination at that point (see Equation 5).

Despite being used widely in many real-time applications, the irradiance volumes method has limited flexibility. Since the lighting is precomputed in an (often slow) offline step, geometry and lighting cannot be changed at run-time. Furthermore, during the precomputation step, only the first bounce of indirect illumination is precomputed, additional bounces are ignored. Another limitation of this technique is that probes must be placed at regular intervals along a dense axis-aligned grid.

In this paper, we enhance the original light probe method by removing the precomputation step, to support scene changes at run-time. We also seamlessly incorporate multiple-bounce lighting.

4. Dynamic Spherical Harmonics Projection Model

In the classical SH projection, the SH coefficients c are evaluated by projecting the function onto the SH basis functions (Equation 3). During the projection, a large number of samples are taken in the spherical coordinates of function, with the function remaining constant during the projection process. We refer to this approach as the *static projection model.*

We propose a novel progressive spherical harmonics projection model, or *dynamic projection model*, to replace the static projection model. In the dynamic projection model, f is projected progressively and is allowed to change over time, as shown in Figure 2. The cost of the projection can then be amortized over many iterations.

As illustrated in Figure 3, the key point of our method is to calculate the error from the reconstructed function $c_t y(s)$ and the updating function $f(s)$, and to use the projection of this error to update the probe coefficients:

$$c_{t+1} = c_t - \int_s (c_t y(s) - f(s))y(s)\mathrm{d}s, \tag{6}$$

where the indices of the coefficient $t+1$ and t indicate different frames / time.

When we evaluate this integral with M samples, Equation 6 becomes:

$$\begin{aligned} c_{t+1} = &\quad c_t - \sum_{s=0}^{s=M} w_s(c_t y(s) - f(s))y(s), \\ = &\quad c_t - \sum_{s=0}^{s=M} w_s c_t - w_s f(s)y(s), \\ = &\quad (1 - \sum_{s=0}^{s=M} w_s)c_t + \sum_{s=0}^{s=M} w_s f(s)y(s), \end{aligned} \tag{7}$$

where w is the differential solid angle of the sample. One special case is considering one sample at a time

$$c_{t+1} = c_t - w(c_t y - f)y, \tag{8}$$

The projection can be moved into the brackets, canceling out the reconstruction and leading to the following form after collecting terms:

$$c_{t+1} = (1-w)c_t + wfy. \tag{9}$$

In this form, it is apparent that we are applying a recursive exponential filter and c_t will provide an unbiased estimate of the SH coefficients. The filter weight trades off convergence rate and variance. We multiply w with a user-facing hysteresis parameter to give control over this trade off.

The choice of the sample count is determined by the time budget. A large sample count results in quick convergence (less frames) with low frame rate, while a small sample count leads to slow convergence (more frames) with high frame rate.

In Figure 4, we compare our progressive probe update against the classic projection of a spherical function for validation.

5. Dynamic Light Probes

Based on the dynamic projection model of SH, we propose a dynamic light probe method. In our method, the SH coefficients which represent the irradiance distribution function at each probe location are computed at run-time progressively, which avoids expensive precomputation and allows multi-bounce lighting. Then, our

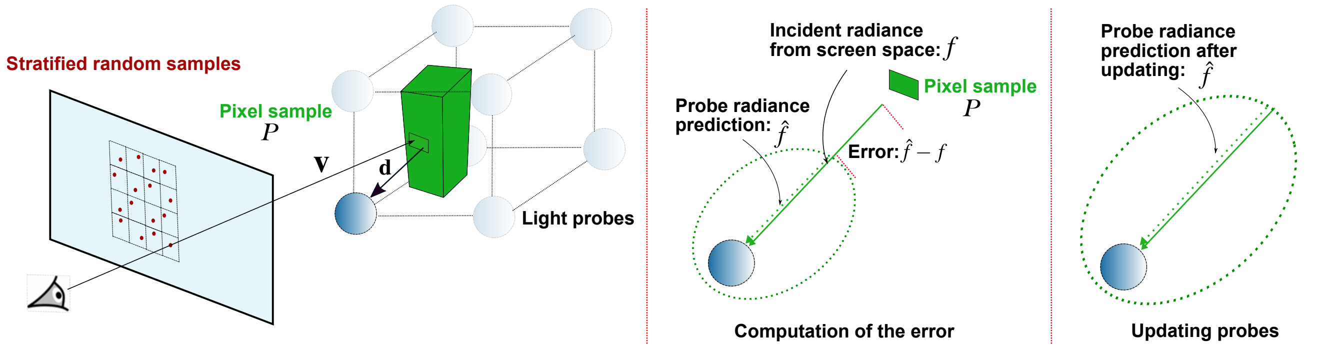
Stratified random samples
Pixel sample P
$\mathbf{v}$
$\mathbf{d}$
Light probes
Incident radiance from screen space: f
Probe radiance prediction: $\hat{f}$
Pixel sample P
Error: $\hat{f} - f$
Computation of the error
Probe radiance prediction after updating: $\hat{f}$
Updating probes

Figure 3: *Overview of our method. a) We take stratified random samples of pixels from the frame after the lighting pass. b) For each sample we compute the incident radiance to every light probe. c) We compare the predicted incident radiance with that predicted by the probe, and update the probe coefficients with the projection of the prediction error.*

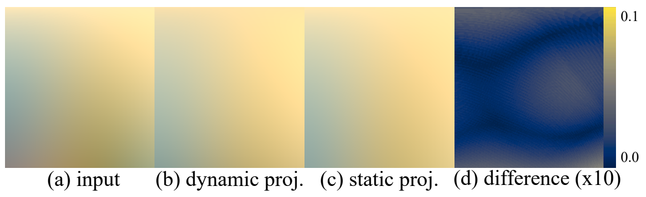

Figure 4: *Comparison of a full static projection (c) of a spherical function onto the 2nd order spherical harmonics basis, versus our progressive projection (b), demonstrating that our model converges to the ground truth. The difference image (d) shows the difference between the statics projection approach and our dynamic model.*

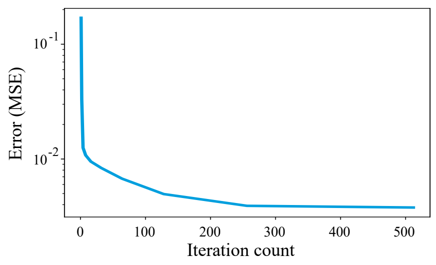

Figure 5: *The error between the dynamic SH projection model and the static SH projection model converges over varying iteration count.*

method is applied for direct lighting of area lights and environment map, via injecting them into the light probes.

5.1. Screen Space Irradiance Accumulating

In this section, we show light probes updating with the screen space frame buffers, as shown in Figure 3.

At each frame, we take a set of radiance samples from the frame buffer after the opaque surface lighting pass, which include the radiance, position, and normal. A frame buffer is subdivided into regular tiles, and one sample is taken from each tile using stratified random sampling. In practice, we use 20×20 samples per frame. The sample count can be adjusted based on performance budget.

Then we update all the probes with these samples. For each probe, we first reconstruct the radiance in the view direction (from the probe to the pixel) with its current SH coefficients c_t using Equation 5, and then compute the difference between the reconstructed radiance and the actual radiance of the pixel f. We then update the probe coefficients by projecting the error to the basis functions. To maximize efficiency and parallelism, we sum up the sample projections and projection weights of all screen samples in the opaque surface lighting pass, and then use these two summed quantities to obtain updated probe coefficients, which matches Equation 7 exactly.

The differential solid angles w in Equation 7 are the weights of a sample. To compute w, we first approximate the area of the surface represented by a pixel A by its solid angle Ω, view depth z and its angle w.r.t. the view direction $\mathbf{v}$:

$$A = \frac{\Omega z^2}{\mathbf{n} \cdot \bar{\mathbf{v}}} \tag{10}$$

This can overestimate the area for shallow view angles relative to the surface, so we clamp $\mathbf{n} \cdot \bar{\mathbf{v}}$ to be no less than 0.01. We then compute the differential solid angle by multiplying by a cosine factor to account for orientation relative to probe and dividing by the squared distance:

$$w = A \frac{\mathbf{n} \cdot \hat{\mathbf{d}}}{\mathbf{d} \cdot \mathbf{d}}, \tag{11}$$

where $\mathbf{d}$ is the vector from the pixel sample to the probe position.

Here we discuss several properties of our method.

Natural multiple-bounce computation. The updated light probes will be used to reconstruct the radiance for the next frame. As more frames accumulate, the indirect illumination of the multiple bounces are obtained naturally.

Fully-dynamic scenes. Note that our method supports fully-dynamic scenes, including changing light sources or geometries.

When the light / geometry change, our method does not need any special treating, since the convergence is fast.

Convergence. Our method does not check converge to stop the light probe updating, since it has low cost and the detecting of the convergence itself is costly and has low robust.

5.2. Application for Area Lights and Environment map

Our method can be directly extended to handle lighting from area light or environment map. These light sources are efficiently sampled and integrated in the light probe updating loop. After updating *cy*, each probe takes a number of jittered samples from the nearby light sources. A spatial data structure is employed to ensure probes only sample light sources that are close enough to have an impact on them. The integration of these samples must be considered carefully, because the light sources may be overlapping texels of the framebuffer, and therefore acting like one-way surfaces since they don't prevent the pixels they cover from being sampled, but on the other hand they may be outside the view frustum.

In practice, we could find a reasonable "magic number" to control the fight between light sources and the scene samples, and the "double sampling" when light sources covered screen pixels didn't matter in practice.

6. Implementation

6.1. Rendering pipeline

In our implementation, our method includes two passes: an opaque surface lighting pass and the light probe updating pass:

- **Opaque surface lighting pass**: compute direct illumination of point or directional light sources directly and then compute indirect illumination with light probes, by interpolating the contribution of the nearby light probes, which is the same as Sec. 3.2. The initial light probe does not have any energy.
- **Light probe updating pass:** We run a compute shader over the probe volume. We obtain 20×20 dynamic irradiance samples using stratified sampling of the frame buffer. The tiles are processed sequentially, with one representative pixel from a tile from each iteration. We splat the samples to light probes as mentioned in Sec. 5.1.

6.2. Probe placement

We have cascaded grids that are placed around the camera at quantized locations, which is similar to Halen et al. [HBHB21], except we don't do any copying after shifting the grids. During our progressive updating, we jitter the probe position randomly within its grid cell, similar to Lumen [Wri21]. This integrates the lighting over the volume of the cell and eliminates aliasing across light probes. In the following, $\mathbf{d}$ is the vector from the pixel sample to the *jittered* probe position.

6.3. Data structure

We store irradiance per-probe using three bands of SH coefficients, totaling 27 coefficients. The probes are organized in a dense axis-aligned grid and stored in a set of seven FP16 precision 3D textures. We refer to one grid of probes as a *cascade* and support multiple nested cascades.

7. Results

We implemented our algorithm inside the High Definition Render Pipeline of the Unity game engine and we will release our source code. All timings in this section are measured on a NVIDIA GeForce RTX 3090 GPU. All the materials in our test scenes are diffuse.

7.1. Progressive SH Splatting Validation

In Figure 4, we compare our progressive SH splatting model with the classical SH splatting model on a blurry environment map. We map the input environment map to a spherical distribution with spherical coordinates, represent it with the two models, and then reconstruct the images (Figure 4 (b) and (c)) with these two representations. For both methods, we sample the input distribution regularly. We update the SH coefficients using Equation 9 for our model and using Equation 3 for the classical model. We found that our method produces almost identical results as the classical model. Figure 5 shows a curve for the error between our model and the static projection model w.r.t the iteration count. As the iteration count increases, the error decreases and tends to converge.

7.2. Quality Validation

In Figure 6, we show more results with our method and direct illumination only. In Figure 9, we show the results of our method for area lights. Overall, our method is able to provide reasonable GI results with high performance. We provide the probe updating cost, which is affordable for low-performance devices. The probe updating time varies, depending on the light probes counts.

In Figure 7, we compare our method with DDGI, with the same light probe count ($16 \times 8 \times 16$). We also provide the cost for our method and DDGI. More specifically, the cost for our method is for light probe updating (0.07 ms), and the cost of DDGI is for light probe ray shooting (0.79 ms) and light probe updating (0.19). Compared to DDGI, as expected, our method is less accurate, while our method is $14\times$ faster.

7.3. Discussion and Limitations

By introducing the visibility between the samples and the light probes will improve the quality of our rendered results, while increasing the computational cost. In Figure 8, we compare our method (with occlusion), our method (without occlusion) with path tracing. Our method (with occlusion) produces closer result to the reference, compared to our method, with much more expensive cost. Thus, we suggest the users to use our method (without occlusion) for low-performance device, and the version with occlusion for next-gen device.

In our method, the light probes are updated using screen samples, thus, any off-screen geometry will not contribute to the final result. The lighting will tend to converge to lighting from the surfaces on

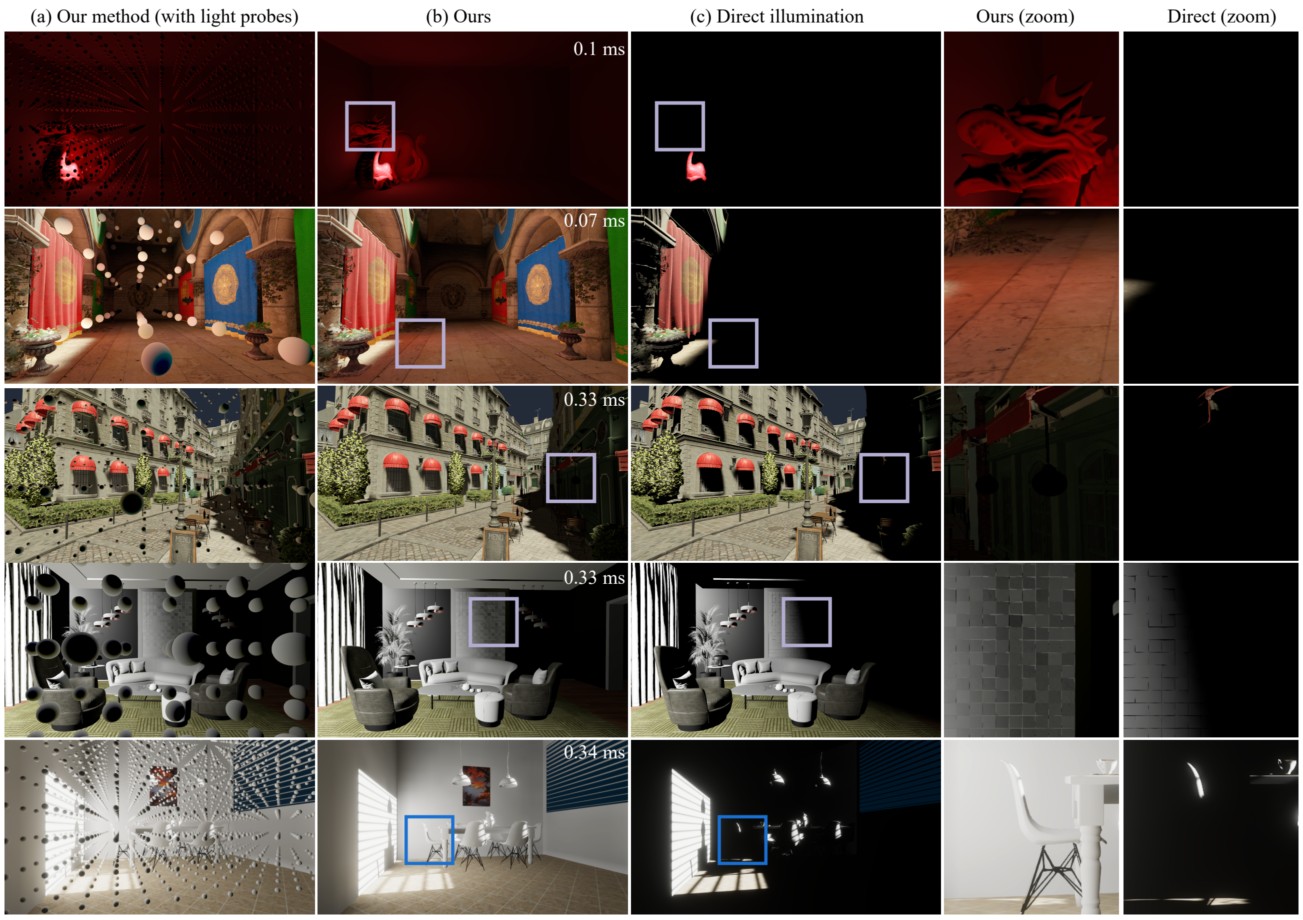

Figure 6: *Comparison between our method and direct illumination on set of scenes. The extra time cost for our method compared to the direct illumination only is shown. Depending on the light probe count, the cost differs.*

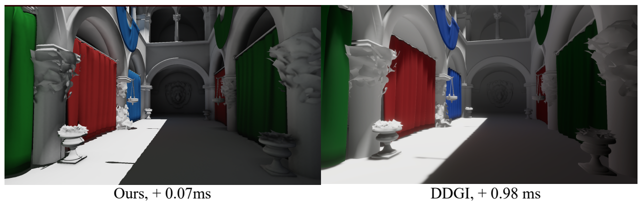

Figure 7: *Comparison between our method and DDGI on the Sponza Scene. Our method costs 0.07 ms to update the light probes, while DDGI costs about 1 ms to shoot rays and update light probes.*

screen, and off screen emitters will fade. If a brightly emitting object becomes occluded or goes off-screen, its contribution will fade out. Specific applications may be able to apply filtering heuristics to preserve lighting information. Other solutions may be to use Reflective Shadow Maps (RSMs) or bake out reflection depth images. We leave these as interesting future work directions.

8. Conclusion

We have presented a novel progressive spherical harmonics projection model, which enables functions being splatted to SH basis progressively. Based on this model, we introduced a dynamic light probe approach for multiple bounce indirect illumination, and lighting for area light and environment map. Our method avoids precomputation, computes multiple bounces of indirect illumination, and exhibits similar performance to previous light probe methods. Our method is much faster than the state of the art methods, and is suitable for low-performance device. We believe our algorithm will be a useful tool for real-time global illumination computation in game or movie development industry.

References

[AKDS04] Annen T., Kautz J., Durand F., Seidel H.-P.: Spherical Harmonic Gradients for Mid-Range Illumination. In *Eurographics Workshop on Rendering* (2004), The Eurographics Association. 2

[Chr08] Christensen P.: *Point-based approximate color bleeding*. Tech. Rep. 08-01, Pixar Technical Notes, 2008. 3

[DS05] Dachsbacher C., Stamminger M.: Reflective shadow maps.

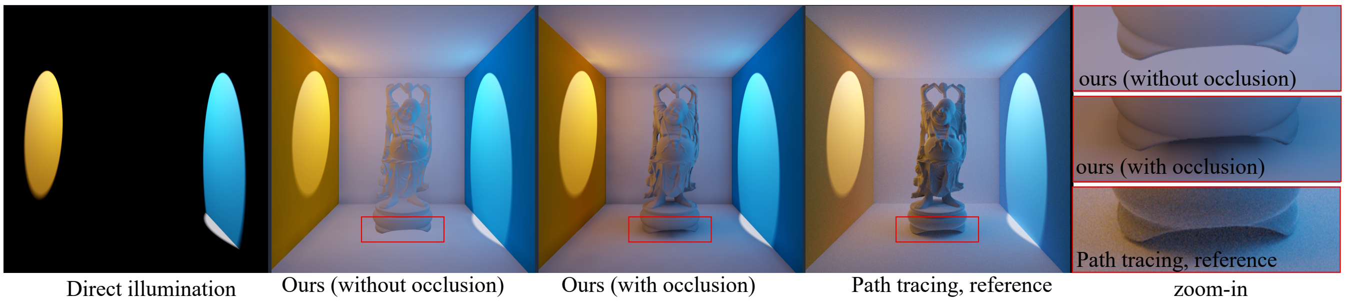

Figure 8: *Comparison between our method (without occlusion), our method (with occlusion) and path tracing on the Buddha scene. By introducing raytracing between the samples and the light probes, the quality of our method are improved, however, much longer time is required.*

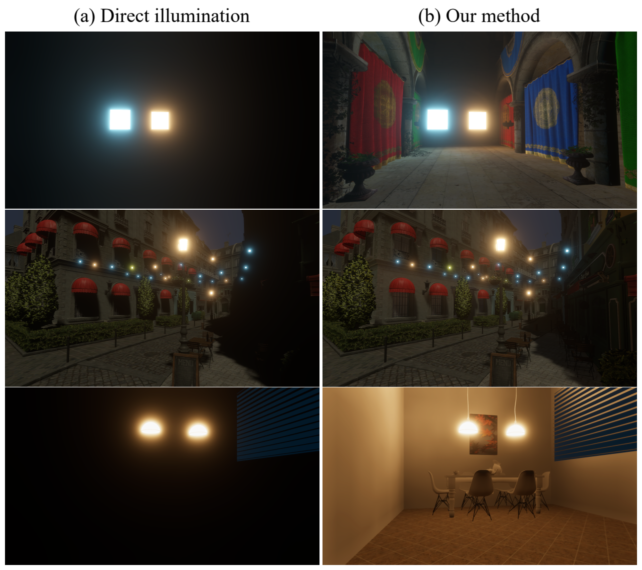

Figure 9: *Comparison between our method and direct illumination on set of scenes with area lights.*

In *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2005), I3D '05, Association for Computing Machinery, p. 203–231. URL: https://doi.org/10.1145/1053427.1053460, doi:10.1145/1053427.1053460. 3

[Gre03] GREEN R.: Spherical harmonic lighting: The gritty details. http://www.research.scea.com/gdc2003/spherical-harmonic-lighting.html, 2003. 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. 2, 3

[HBHB21] HALEN H., BRINCK A., HAYWARD K., BEI X.: Global illumination based on surfels. ACM SIGGRAPH 2021 courses, 2021. 2, 6

[HHE11] HALL C., HALL R., EDWARDS D.: Rendering in cars 2. ACM SIGGRAPH 2011 courses, 2011. 2

[HREB11] HOLLANDER M., RITSCHEL T., EISEMANN E., BOUBEKEUR T.: Manylods: Parallel many-view level-of-detail selection for real-time global illumination. *Computer Graphics Forum 30*, 4 (2011), 1233–1240. 3

[JKG16] JENDERSIE J., KURI D., GROSCH T.: Precomputed illuminance composition for real-time global illumination. In *I3D 2016* (2016), I3D '16, pp. 129–137. 2

[KD10] KAPLANYAN A., DACHSBACHER C.: Cascaded light propagation volumes for real-time indirect illumination. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2010), I3D '10, Association for Computing Machinery, p. 99–107. URL: https://doi.org/10.1145/1730804.1730821, doi:10.1145/1730804.1730821. 3

[Kel97] KELLER A.: Instant radiosity. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., p. 49–56. URL: https://doi.org/10.1145/258734.258769, doi:10.1145/258734.258769. 3

[KGPB05] KRIVANEK J., GAUTRON P., PATTANAIK S., BOUATOUCH K.: Radiance caching for efficient global illumination computation. *IEEE Transactions on Visualization and Computer Graphics 11*, 5 (2005), 550–561. doi:10.1109/TVCG.2005.83. 3

[KL06] KONTKANEN J., LAINE S.: Sampling precomputed volumetric lighting. *journal of graphics, gpu, and game tools 11*, 3 (2006), 1–16. 2

[KSS02] KAUTZ J., SLOAN P.-P., SNYDER J.: Fast, arbitrary brdf shading for low-frequency lighting using spherical harmonics. In *Proceedings of the 13th Eurographics Workshop on Rendering* (Goslar, DEU, 2002), EGRW '02, Eurographics Association, p. 291–296. 3

[MGNM19] MAJERCIK Z., GUERTIN J.-P., NOWROUZEZAHRAI D., MCGUIRE M.: Dynamic diffuse global illumination with ray-traced irradiance fields. *Journal of Computer Graphics Techniques (JCGT) 8*, 2 (June 2019), 1–30. 2

[MMK*21] MAJERCIK Z., MUELLER T., KELLER A., NOWROUZEZAHRAI D., MCGUIRE M.: Dynamic diffuse global illumination resampling. In *ACM SIGGRAPH 2021 Talks* (New York, NY, USA, 2021), SIGGRAPH '21, Association for Computing Machinery. URL: https://doi.org/10.1145/3450623.3464635, doi:10.1145/3450623.3464635. 3

[MMNL16] MARA M., MCGUIRE M., NOWROUZEZAHRAI D., LUEBKE D.: Deep G-Buffers for Stable Global Illumination Approximation. In *Eurographics/ ACM SIGGRAPH Symposium on High Performance Graphics* (2016), Assarsson U., Hunt W., (Eds.), The Eurographics Association. doi:10.2312/hpg.20161195. 3

[MMNL17] MCGUIRE M., MARA M., NOWROUZEZAHRAI D., LUEBKE D.: Real-time global illumination using precomputed light field probes. In *I3D 2017* (February 2017), p. 11. 2

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

[MS16] MANSON J., SLOAN P.-P.: Fast filtering of reflection probes. *Computer Graphics Forum* (2016). 2

[NPG05] NIJASURE M., PATTANAIK S., GOEL V.: Real-time global illumination on gpus. *Journal of Graphics Tools 10*, 2 (2005), 55–71. URL: https://doi.org/10.1080/2151237X.2005.10129194, arXiv:https://doi.org/10.1080/2151237X.2005.10129194, doi:10.1080/2151237X.2005.10129194. 2

[NRH03] NG R., RAMAMOORTHI R., HANRAHAN P.: All-frequency shadows using non-linear wavelet lighting approximation. *ACM Trans. Graph. 22*, 3 (jul 2003), 376–381. URL: https://doi.org/10.1145/882262.882280, doi:10.1145/882262.882280. 3

[NRS14] NALBACH O., RITSCHEL T., SEIDEL H.-P.: Deep screen space. In *Proceedings of the 18th Meeting of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2014), I3D '14, Association for Computing Machinery, p. 79–86. URL: https://doi.org/10.1145/2556700.2556708, doi:10.1145/2556700.2556708. 3

[OLK*21] OUYANG Y., LIU S., KETTUNEN M., PHARR M., PANTALEONI J.: Restir gi: Path resampling for real-time path tracing. *Computer Graphics Forum 40*, 8 (2021), 17–29. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14378, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14378, doi:https://doi.org/10.1111/cgf.14378. 3

[RDGK12] RITSCHEL T., DACHSBACHER C., GROSCH T., KAUTZ J.: The state of the art in interactive global illumination. *Comput. Graph. Forum 31*, 1 (feb 2012), 160–188. URL: https://doi.org/10.1111/j.1467-8659.2012.02093.x, doi:10.1111/j.1467-8659.2012.02093.x. 3

[REG*09] RITSCHEL T., ENGELHARDT T., GROSCH T., SEIDEL H.-P., KAUTZ J., DACHSBACHER C.: Micro-rendering for scalable, parallel final gathering. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 28*, 5 (2009).

[REH*11] RITSCHEL T., EISEMANN E., HA I., KIM J. D. K., SEIDEL H.-P.: Making imperfect shadow maps view-adaptive: High-quality global illumination in large dynamic scenes. *Computer Graphics Forum 30*, 8 (2011), 2258–2269. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2011.01998.x, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2011.01998.x, doi:https://doi.org/10.1111/j.1467-8659.2011.01998.x. 3

[RGK*08] RITSCHEL T., GROSCH T., KIM M. H., SEIDEL H.-P., DACHSBACHER C., KAUTZ J.: Imperfect Shadow Maps for Efficient Computation of Indirect Illumination. *ACM Trans. Graph. (Proc. of SIGGRAPH ASIA 2008) 27*, 5 (2008). 2, 3

[RH01] RAMAMOORTHI R., HANRAHAN P.: An efficient representation for irradiance environment maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (2001), SIGGRAPH '01, pp. 497–500. 2, 4

[RLP*20] RODRIGUEZ S., LEIMKÜHLER T., PRAKASH S., WYMAN C., SHIRLEY P., DRETTAKIS G.: Glossy probe reprojection for interactive global illumination. *ACM Transactions on Graphics (SIGGRAPH Asia Conference Proceedings) 39*, 6 (December 2020). 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* (New York, NY, USA, 2002), SIGGRAPH '02, Association for Computing Machinery, p. 527–536. URL: https://doi.org/10.1145/566570.566612, doi:10.1145/566570.566612. 3

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

[ST15] SILVENNOINEN A., TIMONEN V.: Multi-scale global illumination in quantum break. ACM SIGGRAPH 2015 courses, 2015. 2

[Tat05] TATARCHUK N.: Irradiance volumes for games. Game Developer's Conference, 2005. 2

[Ven07] VENDLER R.: Cheap and dirty irradiance volumes for real-time dynamic scenes. VGC, 2007. 2

[WKKN19] WANG Y., KHIAT S., KRY P. G., NOWROUZEZAHRAI D.: Fast non-uniform radiance probe placement and tracing. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2019), I3D '19, pp. 5:1–5:9. 2

[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* (New York, NY, USA, 1988), SIGGRAPH '88, Association for Computing Machinery, p. 85–92. URL: https://doi.org/10.1145/54852.378490, doi:10.1145/54852.378490. 3

[Wri21] WRIGHT D.: Radiance caching for real-time global illumination. ACM SIGGRAPH 2021 courses, 2021. 6