Analytic Spherical Harmonic Gradients for Real-Time Rendering with Many Polygonal Area Lights

LIFAN WU, University of California, San Diego
GUANGYAN CAI, University of California, San Diego
SHUANG ZHAO, University of California, Irvine
RAVI RAMAMOORTHI, University of California, San Diego

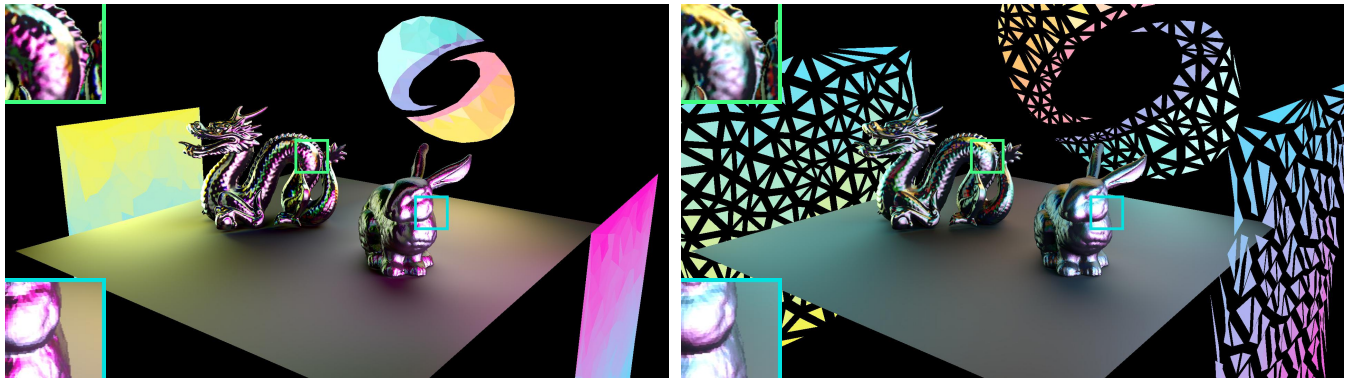


Fig. 1. We present an analytic formula for spherical harmonic (SH) gradients from uniform polygonal area lights, and show how this new theoretical result enables scaling Precomputed Radiance Transfer (PRT) to hundreds of area lights. We first compute the lighting SH coefficients and gradients on a sparse 3D grid. To evaluate SH coefficients for any intermediate point (vertex), we exploit SH gradients and use accurate Hermite interpolation. Here we render a glossy scene with 713 polygonal (triangular) lights and 1.3M polygons at 36fps. Each light transforms independently (in terms of color, location, orientation etc.), enabling the appearance of textured lights or more complex patterns, and causing significant changes in glossy highlights (compare left and right images).

Recent work has developed analytic formulae for spherical harmonic (SH) coefficients from uniform polygonal lights, enabling near-field area lights to be included in Precomputed Radiance Transfer (PRT) systems, and in offline rendering. However, the method is inefficient since coefficients need to be recomputed at each vertex or shading point, for each light, even though the SH coefficients vary smoothly in space. The complexity scales linearly with the number of lights, making many-light rendering difficult. In this paper, we develop a *novel analytic formula for the spatial gradients of the spherical harmonic coefficients for uniform polygonal area lights*. The result is a significant generalization, involving the Reynolds transport theorem to reduce the problem to a boundary integral for which we derive a new analytic formula, showing how to reduce a key term to an earlier recurrence for SH coefficients. The implementation requires only minor additions to existing code for SH coefficients. The results also hold implications for recent efforts on differentiable rendering. We show that SH gradients enable very sparse spatial sampling, followed by accurate Hermite interpolation. This enables *scaling PRT to hundreds of area lights* with minimal overhead and real-time frame rates. Moreover, the SH gradient formula is a new mathematical result that potentially enables many other graphics applications.

CCS Concepts: • **Computing methodologies** → **Rendering**.

Additional Key Words and Phrases: analytic gradients, spherical harmonics, area lighting, differentiable rendering

ACM Reference Format:
Lifan Wu, Guangyan Cai, Shuang Zhao, and Ravi Ramamoorthi. 2020. Analytic Spherical Harmonic Gradients for Real-Time Rendering with Many Polygonal Area Lights. *ACM Trans. Graph.* 39, 4, Article 1 (July 2020), 14 pages. https://doi.org/10.1145/3386569.3392373

1 INTRODUCTION

In this paper, we address a fundamental mathematical question, deriving an analytic formula for the spatial gradients of spherical harmonic (SH) coefficients from a uniform polygonal area light. While both area lights and spherical harmonics are widely used in rendering, to our knowledge, there has been no previous work on finding analytic SH gradients for them. We believe the result has implications for many problems in rendering and beyond.

Our immediate practical motivation is for real-time rendering with precomputed radiance transfer (PRT) [Sloan et al. 2002]. PRT and SH lighting enable dynamic low-frequency environments with realistic highlights and real-time shading, including soft shadows. Hence, they are widely used in real-time applications like games and even in offline rendering [Pantaleoni et al. 2010]. However, the

Authors' addresses: Lifan Wu, University of California, San Diego, liw086@eng.ucsd.edu; Guangyan Cai, University of California, San Diego, g5cai@ucsd.edu; Shuang Zhao, University of California, Irvine, shz@ics.uci.edu; Ravi Ramamoorthi, University of California, San Diego, ravir@cs.ucsd.edu.

0730-0301/2020/7-ART1 $15.00
https://doi.org/10.1145/3386569.3392373

PRT method has often been limited to distant environment maps, since area light SH coefficients differ at each vertex on the object.

Recent work [Wang and Ramamoorthi 2018] has derived an analytic formula for SH coefficients for a uniform polygonal area light (building on earlier work on irradiance tensors [Arvo 1995; Snyder 1996]), demonstrating area lights in PRT. A similar approach has also been applied to SH integrals for both offline and real-time rendering [Belcour et al. 2018]. However, those methods require computing the SH coefficients at each integration point (each vertex in PRT), for each light, which precludes easily scaling to large numbers of lights.

An important observation is that the light field from area lights, and hence its SH coefficients, is smooth spatially. This suggests that the *spatial gradients of SH coefficients* are a critical quantity. Although gradients and differential methods have recently received significant attention [Li et al. 2018; Zhang et al. 2019], there has so far been little previous work in analytically computing SH gradients (as opposed to SH coefficients). One reason may be the challenge of generalizing the SH coefficient derivation, which is already complex, with additional complexity from computing derivatives along each spatial direction. In this paper, we address this long-standing challenge and show how gradient-based interpolation can enable very sparse spatial sampling, and easy scaling to multiple light sources with minimal overhead (see Fig. 1). Our specific contributions include:

Derivation of Analytic SH Gradient Formula. Our main contribution is the derivation of a novel analytic formula for SH gradients. This is a new result, which is a significant generalization of that for SH coefficients. In particular, we show how to reduce the problem to a boundary integral (§4), where a key term can be reduced to an earlier SH coefficient recurrence (§5). Our practical Algorithm 2 in §6.1 is simple, requiring only a few simple modifications to existing code for computing analytic SH coefficients.

Gradient-Based Interpolation. We demonstrate gradient-based interpolation from a sparse set of samples. An overview of our method is in Algorithm 1 in §6. We develop a Hermite cubic interpolant that is consistent with the analytic gradients (§6.2 and Algorithm 4), and more accurate than previous Taylor-series based numerical approaches [Annen et al. 2004]. Even for rendering with one area light source, we demonstrate a 2× speedup over explicit analytic computation of SH coefficients at each vertex. Our major benefit is for handling multiple area lights, where we can linearly accumulate SH coefficients and gradients for all lights on a sparse grid with minimal overhead, followed by gradient-based interpolation.

Efficient Real-Time Rendering with Multiple Area Lights. We implement our approach within a real-time PRT system. We can handle *hundreds of uniform polygonal area lights in real-time*, which was not previously possible (§7, Figs. 1,10,11). This approach also enables easily breaking a high-resolution textured light source into uniform polygonal luminaires, which can be handled with our method.

2 RELATED WORK

PRT and Spherical Harmonics. SH have been widely used in both real-time and offline rendering, going back to the work by Cabral et al. [1987]. In particular, they have widely been used in practice for PRT [Sloan et al. 2002], enabling soft shadows and other light transport effects. Approaches based on all-frequency relighting [Ng et al. 2003] can be more accurate but have not gained widespread practical adoption because of the high precomputation and storage costs. Other analytic methods for area lights, such as the work by Heitz et al. [2016], do not consider shadows. There have been many subsequent developments in PRT; we describe only the closest previous work and refer readers to a survey [Lehtinen 2007; Ramamoorthi 2009] for a more thorough introduction.

Annen et al. [2004] proposed spherical harmonic gradients for mid-range illumination, improving on simply interpolating a small number of locations for incident illumination [Sloan et al. 2002]. However, they did not derive an analytic gradient formula, and required 2D numerical integration over the area light source, with complexity still scaling linearly with the number of lights. In §4 and Appendix C, we show that their result is essentially a different parameterization; our approach enables reduction to a boundary integral with an analytic form.

Zhou et al. [2005] introduced dynamic scenes and near-field lights for all-frequency relighting. Using gradients enables sparser representations, which in turn enable scaling with only a small overhead to hundreds of lights, which was not previously possible. Since our algorithm only pertains to the lighting, other benefits such as dynamic scenes, or any other PRT transport algorithm, can easily be included. Other all-frequency area light methods include the wavelet propagation approach of [Sun and Ramamoorthi 2009] which also handles texture, but the results require approximations and some operations like out-of-plane light rotation are not permitted. Their method is again limited to a single or small number of lights.

Ren et al. [2006] break lights and geometry into spheres and use spherical harmonic exponentiation to enable real-time soft shadows in more complex dynamic scenes. However, the spherical approximation is not generally suitable for planar area lights (even using multiple spheres is a poor approximation of a planar surface from all directions). Moreover, analytic formulae are provided only for sphere lights. Note that none of the above-mentioned papers compute analytic spherical harmonic gradients, and this computation may also benefit these approaches in the future.

Illumination Gradients. Irradiance and radiance caching [Ward et al. 1988; Křivánek et al. 2005b, 2008] are important techniques for global illumination. Greger et al. [1998] introduced the irradiance volume to precompute and store the irradiance distribution function in a volumetric grid. Shading at arbitrary points is computed by trilinearly interpolating from the irradiance volume. In our work, we precompute and store SH coefficients and gradients in the grid, and perform more accurate Hermite interpolation using the gradients.

A series of works [Ward and Heckbert 1992; Křivánek et al. 2005a, 2006; Jarosz et al. 2008a,b] exploit irradiance and radiance gradients to enable sparse sampling of (ir)radiance caching points and accurate value interpolation. They use Monte Carlo integration to evaluate those gradients numerically. In contrast, we derive analytic formulae for SH gradients. More recently, second-order derivatives (Hessians) have been used for error control of the first-order approximation using gradients [Jarosz et al. 2012; Schwarzhaupt et al. 2012; Marco et al. 2018]. It is worth exploring analytic forms of higher order derivatives in the future.

Holzschuch and Sillion [1995] developed analytic radiosity gradients with constant and linear emitters. They used Stokes' theorem to separate the radiosity gradient into two parts: a contour integral and a surface integral. We achieve similar results using the Reynolds transport theorem [Leal 2007]. Their follow-up work [Holzschuch and Sillion 1998] presented analytic second-order derivatives of the radiosity function, enabling error bounding for hierarchical radiosity algorithms. However, their analysis of the radiosity method only handles diffuse surface patches, while our SH gradients apply to incident lighting and can also be used for glossy surfaces.

Differentiable Rendering. Building on early approaches to gradient light transport [Arvo 1994; Ramamoorthi et al. 2007], recent efforts enable differentiating paths including shadows [Li et al. 2018] and radiative transport [Zhang et al. 2019], with different parameterizations [Loubet et al. 2019]. Our method can also be viewed as a differentiable rendering technique, and indeed makes use of the Reynolds transport theorem [Leal 2007], also used in the work by Zhang et al. [2019]. However our goals are very different: we compute SH gradients for PRT, deriving a novel analytic formula. Insights from our paper can be relevant in the future for differentiable rendering and machine learning applications [Liu et al. 2019].

Numerical/Automatic Differentiation. For gradient computations, it is possible to use numerical finite differencing. However, this requires finding the appropriate step size, and can be noisy and inefficient, as noted in the papers above. It is also possible to use automatic differentiation (for example, as used in the work by Li et al. [2015]). However, the results are not optimized, and are less efficient than our analytic gradients. Moreover, we present an explicit analytical derivation, with novel insights, which cannot be achieved with automatic differentiation.

3 PRELIMINARIES

We first introduce basic background on PRT, spherical harmonics, area lights and differentiating integrals.

3.1 Reflection Equation and PRT

The simplest version of precomputed radiance transfer (PRT) tries to solve the reflection equation at spatial position $\boldsymbol{x}$,[1]

$$B(\boldsymbol{x}) = \int_{\mathbb{S}^2} L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) T(\boldsymbol{x}, \boldsymbol{\omega}_i) \, \mathrm{d}\boldsymbol{\omega}_i, \tag{1}$$

where $B(\boldsymbol{x})$ is the reflected radiance or image intensity, $L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)$ is the incoming lighting at $\boldsymbol{x}$ from direction $\boldsymbol{\omega}_i$, and we integrate over the sphere of incoming directions. $T(\boldsymbol{x}, \boldsymbol{\omega}_i)$ is the transport function which is precomputed at each vertex in PRT, and encapsulates the BRDF, cosine term and visibility.

In this paper, our focus is on lighting, i.e., efficient SH projections of L_i, rather than the transport T. Any general PRT method can be used to handle the transport function without modifications to the relevant code. In general, L_i and T are expanded in (real) spherical harmonics $Y_{lm}(\boldsymbol{x})$, which are orthonormal basis functions on the unit sphere. Therefore, the integral reduces to a simple summation,

$$B(\boldsymbol{x}) = \sum_{l=0}^{l_{\max}} \sum_{m=-l}^{l} L_{lm}(\boldsymbol{x}) T_{lm}(\boldsymbol{x}), \tag{2}$$

where L_{lm} and T_{lm} are spherical harmonic coefficients for the lighting and transport respectively (with $L = \sum_{l,m} L_{lm} Y_{lm}$ and $T = \sum_{l,m} T_{lm} Y_{lm}$). We typically consider $l_{\max} = 8$ in this paper, involving 81 SH terms; the summation can be computed at each vertex $\boldsymbol{x}$ as a dot-product of lighting and transport coefficient vectors in graphics hardware for each color channel.

In the original PRT formulation for distant environment maps, lighting is independent of spatial position $\boldsymbol{x}$ and the lighting coefficients L_{lm} can be computed once for each frame, while transport coefficients $T_{lm}(\boldsymbol{x})$ are precomputed and stored. In our case, for near-field area lighting, $L_{lm}(\boldsymbol{x})$ changes at each spatial location.

3.2 Spherical Harmonics

Spherical harmonics (SH) are a set of orthogonal functions Y_{lm} defined on the unit sphere $\mathbb{S}^2$. Let $\boldsymbol{\omega} = (\theta, \phi) = (x, y, z)$ be a unit direction on $\mathbb{S}^2$. The forms of the real SH basic functions are summarized in Appendix A. In particular, zonal harmonics (ZH) $Y_{l0}(\boldsymbol{\omega}) = K_l P_l(\cos\theta)$ are a subset of SH basis functions for $m = 0$, where the normalization $K_l = \sqrt{\frac{2l+1}{4\pi}}$ and P_l is the Legendre polynomial of degree l. Zonal harmonics are radially symmetric around the z-axis. To represent a ZH basis function that is centered around an arbitrary axis $\boldsymbol{\omega}_c$, we write the rotated ZH using dot products: $Y_{l0}(\boldsymbol{\omega} \cdot \boldsymbol{\omega}_c) = K_l P_l(\boldsymbol{\omega} \cdot \boldsymbol{\omega}_c)$.

As pointed out by Nowrouzezahrai et al. [2012], any SH basis function Y_{lm} can be sparsely factorized into a sum of rotated ZH basis functions,

$$Y_{lm}(\boldsymbol{\omega}) = \sum_j \alpha_{l,j}^m Y_{l0}(\boldsymbol{\omega} \cdot \boldsymbol{\omega}_{l,j}), \tag{3}$$

where $\boldsymbol{\omega}_{l,j}$ represents the central direction of each rotated ZH lobe and $\alpha_{l,j}^m$ is its corresponding weight. Theoretically, to represent each band-l SH basis function, $2l + 1$ rotated ZH lobes are required. However in practice, the ZH lobes can be shared across all bands (Appendix B), resulting in a sparse weight matrix $\alpha_{l,j}^m$ and faster SH rotation.

3.3 Analytic Spherical Harmonic Coefficients

Given a polygonal light source A with unit intensity, we denote its projection onto a unit sphere centered at the shading point $\boldsymbol{x}$ as $Q(\boldsymbol{x})$. The SH coefficients of this area light source with respect to $\boldsymbol{x}$ are given by integrating the SH basis functions over Q,

$$L_{lm}(\boldsymbol{x}) = \int_{Q(\boldsymbol{x})} Y_{lm}(\boldsymbol{\omega}) \, \mathrm{d}\boldsymbol{\omega}. \tag{4}$$

By applying the zonal harmonic factorization (Eq. (3)) to Y_{lm}, the SH coefficients can be rewritten as

$$\begin{aligned} L_{lm}(\boldsymbol{x}) &= \int_{Q(\boldsymbol{x})} \left(\sum_j \alpha_{l,j}^m Y_{l0}(\boldsymbol{\omega} \cdot \boldsymbol{\omega}_{l,j}) \right) \mathrm{d}\boldsymbol{\omega} \\ &= \sum_j \alpha_{l,j}^m \underbrace{\int_{Q(\boldsymbol{x})} Y_{l0}(\boldsymbol{\omega} \cdot \boldsymbol{\omega}_{l,j}) \, \mathrm{d}\boldsymbol{\omega}}_{=: L_{l,j}(\boldsymbol{x})}. \end{aligned} \tag{5}$$

[1] For notational simplicity, we assume diffuse reflections and drop dependence of reflected direction $\boldsymbol{\omega}_o$ in B and T. In general, we can also handle non-diffuse reflections $B(\boldsymbol{x}, \boldsymbol{\omega}_o)$ using a glossy PRT extension (see Fig. 1). Since we focus only on lighting L_i, any PRT framework can be supported, including interreflections and dynamic scenes.

Therefore, computing $L_{lm}(\boldsymbol{x})$ boils down to computing the ZH coefficients $L_{l,j}(\boldsymbol{x})$. Belcour et al. [2018] further represented the ZH coefficient as the sum of cosine-power integrals and found analytic solutions for these integrals over spherical polygons. Wang and Ramamoorthi [2018] derived analytic ZH coefficients by applying Stokes' Theorem to convert surface integrals to boundary integrals. The boundary integrals are further solved using the recurrence relations of Legendre polynomials, which are also used in our derivation. However, our approach is different, in terms of finding the SH gradients, which we reduce to a novel boundary integral by differentiating the surface integral for ZH coefficients. A crucial insight helping us to find the analytic formula is in reducing a key term to the earlier ZH coefficient recurrence, which involves minimal overhead to evaluate coefficients and gradients jointly.

3.4 Differentiating Integrals

In this paper, our goal is to find the SH spatial gradients $\nabla_{\boldsymbol{x}} L_{lm}(\boldsymbol{x})$, which we will often denote simply as $\nabla L_{lm}(\boldsymbol{x})$. This reduces to differentiating the integral on the right-hand side (RHS) of Eq. (4). To this end, we leverage the Reynolds transport theorem which originates in fluid mechanics [Leal 2007] and generalizes the Leibniz integral rule for differentiation under the integral operator.

Let $\Omega(\pi)$ be an n-dimensional manifold parameterized by π. We are interested in differentiating the integration of a function f over the region $\Omega(\pi)$. The partial derivative with respect to π can be expressed as

$$\partial_\pi \left(\int_{\Omega(\pi)} f \, \mathrm{d}\Omega(\pi) \right) = \int_{\Omega(\pi)} \dot{f} \, \mathrm{d}\Omega(\pi) + \int_{\partial\Omega(\pi)} \langle \boldsymbol{n}, \dot{\boldsymbol{x}} \rangle f \, \mathrm{d}\partial\Omega(\pi), \tag{6}$$

where $\partial_\pi := \frac{\partial}{\partial_\pi}$ and $\dot{f} := \partial_\pi f$. The differentiation result has two parts. The first one is an integral on the original domain $\Omega(\pi)$. The other one is a boundary integral on the $(n-1)$-dimensional region $\partial\Omega(\pi)$. For its integrand, we define $\dot{\boldsymbol{x}} := \partial_\pi \boldsymbol{x}$, $\boldsymbol{n}$ is the normal direction at each $\boldsymbol{x} \in \partial\Omega(\pi)$ and points towards the exterior by convention, and $\langle \cdot, \cdot \rangle$ is the dot product of two vectors.

Recent work in differentiable rendering [Li et al. 2018; Zhang et al. 2019] have already shown the significance of the boundary integral for correctly evaluating geometric derivatives. Later in §4 and §5, we will see the boundary integral also leads to significant formula simplification and enables the analytic derivation.

4 DIFFERENTIATING SPHERICAL HARMONIC COEFFICIENTS

In this section, we will use the Reynolds transport theorem to differentiate the SH coefficients for area lights, showing that SH gradients can be computed from boundary integrals. In §5 we will further derive our key result, an analytic formula. In §6 we develop our SH gradient (jointly with SH coefficients) evaluation algorithm and the gradient-based interpolation method, showing how to handle multiple area light sources. While the derivation is somewhat involved, the actual algorithm involves only a few simple modifications to previous code for SH coefficients. Readers interested primarily in implementation may wish to first browse §6 and Algorithms 1 and 2.

Let $A = (\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N)$ be a uniform polygonal light source with N points in $\mathbb{R}^3$ and $\boldsymbol{x}$ be the shading point where we want to evaluate the incident radiance (see Fig. 2-a). Note that $Q(\boldsymbol{x}) = (\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_N)$ is a spherical polygon obtained by projecting A onto the unit sphere $\mathbb{S}^2$ centered at $\boldsymbol{x}$ (see Fig. 2-a), where $\boldsymbol{\omega}_i = \frac{\boldsymbol{p}_i - \boldsymbol{x}}{\|\boldsymbol{p}_i - \boldsymbol{x}\|}$. Both $Q(\boldsymbol{x})$ and its boundary (which consists of arcs on $\mathbb{S}^2$) $\partial Q(\boldsymbol{x}) = \{\widehat{\boldsymbol{\omega}_i \boldsymbol{\omega}_{i+1}} \mid i = 1, \ldots, N\}$ may vary with $\boldsymbol{x}$.

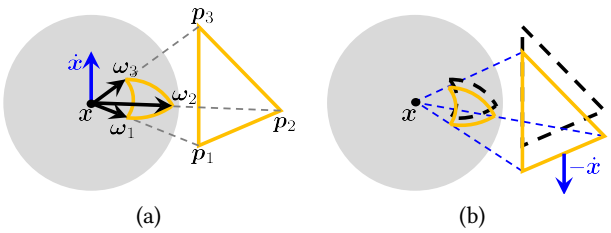


Fig. 2. (a) Illustration of the spherical projection for a triangle light. (b) When the shading point $\boldsymbol{x}$ moves by $\dot{\boldsymbol{x}}$ (equivalent to the area light moves by $-\dot{\boldsymbol{x}}$), the projected spherical polygon also changes accordingly.

4.1 Spherical Harmonic Gradient

Given a shading point $\boldsymbol{x} = (x, y, z)$, we define the SH gradient as the spatial gradients with respect to $\boldsymbol{x}$,

$$\nabla L_{lm}(\boldsymbol{x}) = (\partial_x L_{lm}(\boldsymbol{x}), \partial_y L_{lm}(\boldsymbol{x}), \partial_z L_{lm}(\boldsymbol{x})). \tag{7}$$

Without loss of generality, we will focus on one of the spatial partial derivatives $\partial_z L_{lm}(\boldsymbol{x})$ in the following derivations. The other gradient components can be evaluated in the same way.

To evaluate the SH coefficient in Eq. (4), the SH basis function is integrated over a varying domain $Q(\boldsymbol{x})$ as the shading point $\boldsymbol{x}$ changes. By applying the Reynolds transport theorem to the right-hand side (RHS) of Eq. (4), we can write the partial derivative as

$$\partial_z \int_{Q(\boldsymbol{x})} Y_{lm}(\boldsymbol{\omega}) \, \mathrm{d}\boldsymbol{\omega} = \int_{Q(\boldsymbol{x})} \partial_z [Y_{lm}(\boldsymbol{\omega})] \, \mathrm{d}\boldsymbol{\omega} + \int_{\partial Q(\boldsymbol{x})} \langle \boldsymbol{n}_\perp, \dot{\boldsymbol{\omega}} \rangle Y_{lm}(\boldsymbol{\omega}) \, \mathrm{d}\ell(\boldsymbol{\omega}). \tag{8}$$

The first integral on the RHS vanishes, because the SH basis function is independent of the shading point position so $\partial_z [Y_{lm}(\boldsymbol{\omega})] = 0$.

The second integral is due to the moving boundary $\partial Q(\boldsymbol{x})$ as $\boldsymbol{x}$ varies (see Fig. 2-b). For every $\boldsymbol{\omega} \in \partial Q(\boldsymbol{x})$, $\mathrm{d}\ell(\boldsymbol{\omega})$ represents the arc length measure. The normal vector $\boldsymbol{n}_\perp$ is in the tangent space of $\boldsymbol{\omega} \in \mathbb{S}^2$ and perpendicular to the boundary curve. We denote $\dot{\boldsymbol{\omega}}$ as the change rate of the boundary location $\boldsymbol{\omega}$ with respect to z, i.e. $\dot{\boldsymbol{\omega}} = \partial_z \boldsymbol{\omega}$. Let $\boldsymbol{y} \in \partial A$ be a point on the polygonal light boundary. Further, the shading point $\boldsymbol{x}$ has a change rate $\dot{\boldsymbol{x}} = (0, 0, 1)$. We can establish the following relation between the change rates (see Fig. 3),

$$\dot{\boldsymbol{\omega}} = \partial_z \left(\frac{\boldsymbol{y} - \boldsymbol{x}}{\|\boldsymbol{y} - \boldsymbol{x}\|} \right) = \frac{-\dot{\boldsymbol{x}}}{\|\boldsymbol{y} - \boldsymbol{x}\|} - \boldsymbol{\omega} \left\langle \boldsymbol{\omega}, \frac{-\dot{\boldsymbol{x}}}{\|\boldsymbol{y} - \boldsymbol{x}\|} \right\rangle. \tag{9}$$

4.2 Reduction to Edge/Arc Integrals

The previous subsection shows that the spatial partial derivative can be simplified as

$$\partial_z L_{lm}(\boldsymbol{x}) = \int_{\partial Q(\boldsymbol{x})} \langle \boldsymbol{n}_\perp, \dot{\boldsymbol{\omega}} \rangle Y_{lm}(\boldsymbol{\omega}) \, \mathrm{d}\ell(\boldsymbol{\omega}), \tag{10}$$

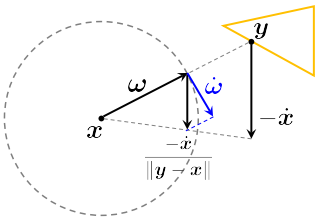


Fig. 3. Evaluating the change rate of $\boldsymbol{\omega}$. We first project $-\dot{\boldsymbol{x}}$ to the unit sphere, then obtain $\dot{\boldsymbol{\omega}}$ by extracting the component of $\frac{-\dot{\boldsymbol{x}}}{\|\boldsymbol{y}-\boldsymbol{x}\|}$ that is perpendicular to $\boldsymbol{\omega}$.

which is a 1D integral on the spherical polygon edges.

We now decompose the boundary integral into a sum of arc integrals for each arc $\widehat{\boldsymbol{\omega}_i\boldsymbol{\omega}_{i+1}} \in \partial Q(\boldsymbol{x})$,

$$\partial_z L_{lm}(\boldsymbol{x}) = \sum_{i=1}^{N} \underbrace{\int_{\widehat{\boldsymbol{\omega}_i\boldsymbol{\omega}_{i+1}}} \langle \boldsymbol{n}_i, \dot{\boldsymbol{\omega}} \rangle Y_{lm}(\boldsymbol{\omega}) \, \mathrm{d}\ell(\boldsymbol{\omega})}_{=: G_{lm}^{(i)}} . \tag{11}$$

For every $\boldsymbol{\omega} \in \widehat{\boldsymbol{\omega}_i\boldsymbol{\omega}_{i+1}}$, it has the same normal direction

$$\boldsymbol{n}_i = \frac{\boldsymbol{\omega}_i \times \boldsymbol{\omega}_{i+1}}{\|\boldsymbol{\omega}_i \times \boldsymbol{\omega}_{i+1}\|} . \tag{12}$$

Further, the edge integral can be parameterized with the radian angle $t \in [0, T_i]$, $T_i = \arccos(\boldsymbol{\omega}_i \cdot \boldsymbol{\omega}_{i+1})$ as

$$\boldsymbol{\omega}(t) = \boldsymbol{\omega}_i \cos t + \boldsymbol{\lambda} \sin t, \tag{13}$$

where $\boldsymbol{\lambda} = \boldsymbol{n}_i \times \boldsymbol{\omega}_i$. The direction change rate $\dot{\boldsymbol{\omega}}(t)$ can be evaluated using Eq. (9). The arc length measure $\mathrm{d}\ell(\boldsymbol{\omega})$ is equal to $\mathrm{d}t$ since the sphere radius is one. In summary, we can rewrite the arc integrals in Eq. (11) as simple 1D integrals,

$$\boxed{G_{lm}^{(i)} = \int_0^{T_i} \langle \boldsymbol{n}_i, \dot{\boldsymbol{\omega}}(t) \rangle Y_{lm}(\boldsymbol{\omega}(t)) \, \mathrm{d}t .} \tag{14}$$

At this point, it would be possibly to simply evaluate $G_{lm}^{(i)}$ efficiently using 1D numerical quadrature rules.[2] However, we will go even further, deriving a fully analytic recurrence formula in §5.

Discussion. The reduction of area light computations to edge integrals over the bounding arcs is common, but previous work has used Stokes' theorem for polynomial or spherical harmonic coefficients [Snyder 1996; Wang and Ramamoorthi 2018]. Our use of the Reynolds transport theorem to reduce the *gradients* to boundary integrals is a novel approach, to the best of our knowledge.

Annen et al. [2004] developed a semi-analytic solution to SH gradients. We show in Appendix C that their results can also be derived from the Reynolds transport theorem, but using a different parameterization. In their case, the integration domain is independent of the varying parameters. As a result, the boundary integral (second integral on the RHS of Eq. (8)) becomes zero after differentiating with the Reynolds transport theorem. On the other hand, we use a parameterization such that the integration domain varies while the integrand is independent. Therefore, the integral of the differentiated quantity (first integral on the RHS of Eq. (8)) is zero.

Although both methods result in equivalent solutions, they have different algorithmic implications. In the work by Annen et al. [2004], they need to numerically integrate functions in 2D, even though the integrand can be evaluated analytically or by automatic differentiation. On the other hand, our parameterization results in dimension reduction. The 1D integrals can not only be evaluated numerically in a more efficient way, but also lead to analytic solutions. Different applications may prefer specific parameterizations. For example, in differentiable rendering [Li et al. 2018; Zhang et al. 2019; Loubet et al. 2019], one wants to avoid the boundary integration as much as possible, because the gradient estimation requires additional effort for edge sampling. Further, the edge integrals they evaluated are too complex to have analytic solutions. In contrast, we prefer to reduce SH gradients to edge integrals. The dimension reduction on the integral domain makes the analytic derivation much easier.

5 ANALYTIC FORMULA

In this section, we will show how to solve SH gradients analytically. First, we use ZH factorization [Nowrouzezahrai et al. 2012], see Eqs. (3, 5), to rewrite Eq. (14) in terms of ZH integrals (we focus on one edge in the following derivations),

$$G_{lm}^{(i)} = \sum_j \alpha_{l,j}^m \underbrace{\int_0^{T_i} \langle \boldsymbol{n}_i, \dot{\boldsymbol{\omega}}(t) \rangle Y_{l0}(\boldsymbol{\omega}(t) \cdot \boldsymbol{\omega}_{l,j}) \, \mathrm{d}t}_{=: G_{l,j}^{(i)}} . \tag{15}$$

The weights $\alpha_{l,j}^m$ and central directions $\boldsymbol{\omega}_{l,j}$ of the ZH lobes can be precomputed. Now, the problem reduces to how to evaluate the integral $G_{l,j}^{(i)}$ analytically.

The input of our method includes the edge endpoints $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$, the shading point $\boldsymbol{x}$, its change rate $\dot{\boldsymbol{x}}$ that equals $(0, 0, 1)$ when differentiating with respect to z, and the central direction $\boldsymbol{\omega}_{l,j}$ of the j-th ZH lobe. Briefly, the derivation involves simplifying the integrand, rearranging terms and reducing to the known recurrence relations of Legendre polynomials.

5.1 Solving for $G_{l,j}^{(i)}$

Transforming to Local Frame. We seek to represent the integrand of $G_{l,j}^{(i)}$ by a function of t, i.e., $g(t) = \langle \boldsymbol{n}_i, \dot{\boldsymbol{\omega}}(t) \rangle Y_{l0}(\boldsymbol{\omega}(t) \cdot \boldsymbol{\omega}_{l,j})$, and simplify it as much as possible. First, we translate the edge $\overline{\boldsymbol{p}_i\boldsymbol{p}_{i+1}}$ by $-\boldsymbol{x}$ so that the shading point is at the origin. We denote the distances from the shading point to the edge endpoints as $\ell_i = \|\boldsymbol{p}_i - \boldsymbol{x}\|$ and $\ell_{i+1} = \|\boldsymbol{p}_{i+1} - \boldsymbol{x}\|$. The arc $\widehat{\boldsymbol{\omega}_i\boldsymbol{\omega}_{i+1}}$ is represented by two unit vectors $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_{i+1}$. Then, we build a local frame $(\boldsymbol{\omega}_i, \boldsymbol{\lambda}_i, \boldsymbol{n}_i)$, where $\boldsymbol{n}_i$ and $\boldsymbol{\lambda}_i$ are defined in Eqs. (12, 13). We transform all the related vectors $\boldsymbol{\omega}_i, \boldsymbol{\omega}_{i+1}$, and $\boldsymbol{\omega}_{l,j}$ into this local frame by a rotation operator $\mathcal{R}(\boldsymbol{u}) = (\boldsymbol{u} \cdot \boldsymbol{\omega}_i, \boldsymbol{u} \cdot \boldsymbol{\lambda}_i, \boldsymbol{u} \cdot \boldsymbol{n}_i)$. One benefit is that expressions of $\boldsymbol{\omega}(t)$ are simpler after rotation: $\mathcal{R}(\boldsymbol{\omega}(t)) = (\cos t, \sin t, 0)$, because the edge is completely in the xy-plane and $\boldsymbol{\omega}_i$ aligns with the x-axis. Moreover, the function value $g(t)$ is unchanged since the dot product is invariant under rotation.

[2] Note that the integrand is smooth and low-dimensional (1D), so Monte Carlo sampling is not required; a standard integration rule like Simpson's or a higher-order quadrature scheme could be employed. This may be desirable in some applications.

Simplifying $\langle \boldsymbol{n}_i, \dot{\boldsymbol{\omega}}(t)\rangle$. To evaluate this term, we can expand and simplify it using Eq. (9):

$$\begin{aligned}\langle \boldsymbol{n}_i, \dot{\boldsymbol{\omega}}(t)\rangle &= \left\langle \boldsymbol{n}_i, \frac{-\dot{\boldsymbol{x}}}{\|\boldsymbol{y}(t)-\boldsymbol{x}\|}\right\rangle - \langle \boldsymbol{n}_i, \boldsymbol{\omega}(t)\rangle \left\langle \boldsymbol{\omega}(t), \frac{-\dot{\boldsymbol{x}}}{\|\boldsymbol{y}(t)-\boldsymbol{x}\|}\right\rangle \\ &= -\frac{1}{\|\boldsymbol{y}(t)-\boldsymbol{x}\|}\langle \boldsymbol{n}_i, \dot{\boldsymbol{x}}\rangle, \end{aligned} \tag{16}$$

because $\boldsymbol{n}_i$ is perpendicular to $\boldsymbol{\omega}(t)$. Assuming $\boldsymbol{n}_i = (n_x, n_y, n_z)$, we know that $\langle \boldsymbol{n}_i, \dot{\boldsymbol{x}}\rangle = n_z$. Then, it remains to find $\ell(t) = \|\boldsymbol{y}(t) - \boldsymbol{x}\|$. We denote $\boldsymbol{y}(t) = \boldsymbol{x} + \ell(t)\boldsymbol{\omega}(t)$ as the intersection point of $\overline{\boldsymbol{p}_i\boldsymbol{p}_{i+1}}$ and the ray with direction $\boldsymbol{\omega}(t)$ starting at $\boldsymbol{x}$. The solution can be found by solving a linear system,

$$\ell(t) = \left(\frac{\sin t}{\ell_{i+1}} - \frac{\sin(t-T_i)}{\ell_i}\right)^{-1} \sin T_i. \tag{17}$$

Recall that ℓ_i and ℓ_{i+1} are the distances to area light vertices $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$ respectively, from the shading point $\boldsymbol{x}$ (corresponding to $\ell(0)$ and $\ell(T_i)$). We provide the detailed derivation in Appendix D.1.

To sum up, Eq. (16) can be written as

$$\langle \boldsymbol{n}_i, \dot{\boldsymbol{\omega}}(t)\rangle = -\frac{n_z}{\sin T_i}\left(\frac{\sin t}{\ell_{i+1}} - \frac{\sin(t-T_i)}{\ell_i}\right). \tag{18}$$

Simplifying $Y_{l0}(\boldsymbol{\omega}(t)\cdot\boldsymbol{\omega}_{l,j})$. Given a precomputed central direction $\boldsymbol{\omega}_{l,j}$, we denote the vector after rotation as $\mathcal{R}(\boldsymbol{\omega}_{l,j}) = (c_x, c_y, c_z)$. The ZH basis function can be written as

$$Y_{l0}(\mathcal{R}(\boldsymbol{\omega}(t))\cdot\mathcal{R}(\boldsymbol{\omega}_{l,j})) = K_l P_l(c_x\cos t + c_y \sin t). \tag{19}$$

Finally, plugging Eqs. (18, 19) back in Eq. (15), we have

$$\begin{aligned}G_{l,j}^{(i)} =& \frac{n_z K_l}{\ell_i \sin T_i}\int_0^{T_i} \sin(t-T_i)P_l(c_x\cos t + c_y\sin t)\,\mathrm{d}t - \\ & \frac{n_z K_l}{\ell_{i+1}\sin T_i}\int_0^{T_i}\sin(t)P_l(c_x\cos t + c_y\sin t)\,\mathrm{d}t. \end{aligned} \tag{20}$$

Rearranging Terms. Despite our simplifications to the integrand, it remains nontrivial to evaluate the integrals in Eq. (20) analytically. Fortunately, for $h(t) = c_x\cos t + c_y\sin t$, the integral $\int hP_l(h)\,\mathrm{d}t$ does have a closed-form solution, which can be derived using integration by parts [Wang and Ramamoorthi 2018]. Therefore, our goal is to rearrange the integrand so that it reduces to this integral.

We first combine the linear combination of sine and cosine waves to a single sine wave with a scaled amplitude A and a phase shift T',

$$h(t) = c_x\cos t + c_y\sin t = A\sin(t+T'), \tag{21}$$

where $A = \sqrt{c_x^2 + c_y^2}$ and $T' = \arctan(c_x/c_y)$. Using trigonometric identities, the first integral on the RHS of Eq. (20) can be reformulated as

$$\frac{\cos(T_i+T')}{A}C_l - \frac{\sin(T_i+T')}{A}E_l, \tag{22}$$

where $C_l = \int_0^{T_i} hP_l(h)\,\mathrm{d}t$ and $E_l = \int_0^{T_i}(\frac{\mathrm{d}}{\mathrm{d}t}h)P_l(h)\,\mathrm{d}t$. We provide the derivation details in Appendix D.2. The second integral can be solved in the same way. Finally, Eq. (20) can be simplified as

$$\boxed{\begin{aligned}G_{l,j}^{(i)} =& \frac{n_z K_l}{A\ell_i\sin T_i}(C_l\cos(T_i+T') - E_l\sin(T_i+T')) - \\ & \frac{n_z K_l}{A\ell_{i+1}\sin T_i}(C_l\cos T' - E_l\sin T').\end{aligned}} \tag{23}$$

Analytic Formula for E_l. Notice that the analytic solution to E_l can be directly derived using a change of variable $\mathrm{d}h = (\frac{\mathrm{d}}{\mathrm{d}t}h)\,\mathrm{d}t$,

$$\begin{aligned}E_l &= \int_0^{T_i}(\frac{\mathrm{d}}{\mathrm{d}t}h)P_l(h)\,\mathrm{d}t = \int_{h(0)}^{h(T_i)} P_l(h)\,\mathrm{d}h \\ &= \frac{1}{2l+1}[P_{l+1}(h) - P_{l-1}(h)]\Big|_{c_x}^{c_x\cos T_i + c_y\sin T_i}. \end{aligned} \tag{24}$$

Here we use the following recurrence relation of the Legendre polynomials: $(2l+1)P_l(h) = \frac{\mathrm{d}}{\mathrm{d}h}(P_{l+1}(h) - P_{l-1}(h))$. [3]

Recurrence Formula for C_l. Unlike E_l, it is difficult, if not impossible, to derive a direct representation for C_l. However, our key insight is in reducing the integral expressions to this specific form. Indeed, Wang and Ramamoorthi [2018] have developed a recurrence formula for C_l and associated edge integrals,

$$C_l = \frac{1}{l+1}\left[(c_x\sin T_i - c_y\cos T_i)P_l(h(T_i)) + c_y P_l(c_x) + (c_x^2 + c_y^2 - 1)D_l + lB_{l-1}\right], \tag{25}$$

where the edge integrals B_l and D_l are given by $B_l = \int_0^{T_i} P_l(h)\,\mathrm{d}t$ and $D_l = \int_0^{T_i}\frac{\mathrm{d}}{\mathrm{d}h}P_l(h)\,\mathrm{d}t$. Their associated recurrence formulae are,

$$B_l = \frac{2l-1}{l}C_{l-1} - \frac{l-1}{l}B_{l-2}, \tag{26}$$

$$D_l = (2l-1)B_{l-1} + D_{l-2}. \tag{27}$$

The base cases[4] for $l = 0$ are

$$B_0 = T_i,\ D_0 = 0. \tag{28}$$

5.2 Summary

Based on Eqs. (11, 15), the SH gradient evaluated at one point $\boldsymbol{x}$ can be expressed as

$$\partial_z L_{lm} = \sum_i\sum_j \alpha_{l,j}^m G_{l,j}^{(i)} = \sum_j \alpha_{l,j}^m \underbrace{\left(\sum_i G_{l,j}^{(i)}\right)}_{=:G_{l,j}}. \tag{29}$$

We have just derived an analytic formula for $G_{l,j}^{(i)}$ (Eq. (23)), reducing it to simpler integrals of the Legendre polynomials which are easier to solve.

Analytic SH Coefficients. The edge integrals B_l, C_l and D_l are not only used for evaluating SH gradients, but also building blocks for computing SH coefficients [Wang and Ramamoorthi 2018]. Therefore, we can simultaneously compute SH coefficients and gradients without much overhead. For completeness, we provide analytic formulae for SH coefficients, which are rewritten and simplified from previous work with respect to our notation.

Similar to Eq. (29), we can decompose one SH coefficient into the contributions from each individual ZH lobe as $L_{lm} = \sum_j \alpha_{l,j}^m L_{l,j}$. Denoting $H_{l,j}^{(i)}$ as the intermediate quantity for the individual contribution from the j-th ZH lobe and the i-th edge,

$$H_{l,j}^{(i)} = c_z K_l B_l, \tag{30}$$

[3] The identity still holds for $l = 0$ if we define $P_{-1}(h) \equiv 0$.

[4] Additionally, we define $B_{-1} \equiv 0$ and $D_{-1} \equiv 0$.

Algorithm 1 Evaluation of lighting coefficients for every vertex

1: **function** LightCoeffForPRT
2: // SH Evaluation on a 3D grid
3: Build a uniform 3D grid of resolution M^3
4: **for** each grid point $\boldsymbol{x}$ **do**
5: **for** each area light in the scene **do**
6: Accumulate $L_{lm}(\boldsymbol{x})$ and $\nabla L_{lm}(\boldsymbol{x})$ ▷ Algorithm 2
7: **end for**
8: **end for**
9: // Gradient-based Interpolation for PRT vertices
10: **for** each vertex $\boldsymbol{v}$ **do**
11: Find its eight adjacent grid points
12: **for** each SH basis (l, m) **do**
13: Fetch L_{lm} and ∇L_{lm} at the eight grid points
14: Hermite interpolate $L_{lm}(\boldsymbol{v})$ ▷ Algorithm 4
15: **end for**
16: **end for**
17: **end function**

the ZH coefficient $L_{l,j}$ (Eq. (5)) is given by the following recurrence formula,

$$L_{l,j} = \frac{2l-1}{l(l+1)} \underbrace{\sum_i H^{(i)}_{l-1,j}}_{=: H_{l-1,j}} + \frac{(l-2)(l-1)}{l(l+1)} L_{l-2,j}. \tag{31}$$

The base case for $l = 0$ is basically the solid angle subtended by the polygon [Arvo 1995], scaled by K_0,

$$L_{0,j} = K_0 \left[\sum_{i=1}^{N} \arccos\left(\frac{\boldsymbol{\omega}_i \times \boldsymbol{\omega}_{i-1}}{\|\boldsymbol{\omega}_i \times \boldsymbol{\omega}_{i-1}\|} \cdot \frac{\boldsymbol{\omega}_i \times \boldsymbol{\omega}_{i+1}}{\|\boldsymbol{\omega}_i \times \boldsymbol{\omega}_{i+1}\|} \right) - (N-2)\pi \right]. \tag{32}$$

We also define $L_{-1,j} \equiv 0$ for completeness.

Although previous work and ours both reduce to the same set of edge integrals, the derivation techniques are quite different. Previous work (Eqs. (30, 31)) uses Stokes' Theorem to convert the surface integrals for SH coefficients to the edge integrals. On the other hand, our reduction is based on differentiation of surface integrals under a specific parameterization, followed by term rearrangements with algebraic identities.

6 ALGORITHM

Based on the analytic formulae presented in §5, we demonstrate a practical algorithm to evaluate SH coefficients and gradients simultaneously, given a shading point and one polygonal light (§6.1). However, it is still challenging to handle a scene with many area lights, since the method scales linearly in the number of lights.

Fortunately, SH coefficients vary smoothly as the shading point moves. Based on this observation, we develop an efficient algorithm to evaluate the lighting coefficients in the PRT framework, especially when there are multiple uniform polygonal area lights. Our method is outlined in Algorithm 1. We evaluate SH coefficients and gradients for all lights on a sparse grid (Lines 2–8 of Algorithm 1), followed by interpolating SH coefficients of PRT vertices in between (Lines 9–16). In terms of interpolation, we present a gradient-based, tricubic Hermite interpolation method within a 3D grid (§6.2), which is more accurate than the trilinear interpolation and previous Taylor-series based interpolation [Annen et al. 2004]. The computation time of interpolation is independent of the number of lights, allowing us to render a scene with hundreds of area lights in real-time.

Algorithm 2 SH coefficients and gradients for one polygonal light

1: **function** SHCoeffAndGrad($\boldsymbol{x}$, N, $\{\boldsymbol{p}_i\}$, $l_{\max}$, $\{\alpha^m_{l,j}\}$, $\{\boldsymbol{\omega}_{l,j}\}$)
2: **for** $i = 0$ to $N-1$ **do** ▷ Precomputation for each vertex
3: $\boldsymbol{p}_i = \boldsymbol{p}_i - \boldsymbol{x}$, $\ell_i = \|\boldsymbol{p}_i\|$, $\boldsymbol{\omega}_i = \boldsymbol{p}_i / \ell_i$
4: **end for**
5: **for** $i = 0$ to $N-1$ **do** ▷ Precomputation for each edge
6: $\boldsymbol{n}_i = \frac{\boldsymbol{\omega}_i \times \boldsymbol{\omega}_{i+1}}{\|\boldsymbol{\omega}_i \times \boldsymbol{\omega}_{i+1}\|}$, $\boldsymbol{\lambda}_i = \boldsymbol{n}_i \times \boldsymbol{\omega}_i$
7: $T_i = \arccos(\boldsymbol{\omega}_i \cdot \boldsymbol{\omega}_{i+1})$
8: **end for**
9: **for** $j = 0$ to $2l_{\max}$ **do** ▷ Iterate over $2l_{\max}+1$ ZH lobes
10: $\boldsymbol{\omega}_c = \boldsymbol{\omega}_{l_{\max},j}$ ▷ Share ZH lobes, Appendix B
11: $\{H_{l,j}\} = 0$ ▷ Initialization for ZH coefficients
12: $\{G_{l,j}\} = 0$ ▷ Initialization for ZH gradients
13: **for** $i = 0$ to $N-1$ **do** ▷ Iterate over N edges
14: $c_x = \boldsymbol{\omega}_c \cdot \boldsymbol{\omega}_i$, $c_y = \boldsymbol{\omega}_c \cdot \boldsymbol{\lambda}_i$, $c_z = \boldsymbol{\omega}_c \cdot \boldsymbol{n}_i$ ▷ Rotation
15: $n_z = \boldsymbol{n}_i[z]$ ▷ $\langle \boldsymbol{n}_i, \dot{\boldsymbol{x}} \rangle$
16: $A = (c_x^2 + c_y^2)^{1/2}$, $T' = \arctan(c_x / c_y)$ ▷ Eq. (21)
17: $(\{B_l\}, \{C_l\}, \{E_l\}) = \text{Recurrence}(c_x, c_y, T_i, l_{\max})$
▷ Algorithm 3
18: **for** $l = 0$ to $l_{\max}$ **do**
19: $K_l = \sqrt{\frac{2l+1}{4\pi}}$ ▷ SH Normalization factor
20: $H_{l,j}$ += $c_z K_l B_l$ ▷ Eq. (30)
21: $G_{l,j}$ += $\mathrm{G}(n_z, K_l, A, \ell_i, \ell_{i+1}, T_i, T', C_l, E_l)$
▷ Eq. (23)
22: **end for**
23: **end for**
24: $L_{0,j} = K_0 \times \text{solidAngle}(N, \{\boldsymbol{\omega}_i\})$ ▷ Base case, Eq. (32)
25: **for** $l = 1$ to $l_{\max}$ **do**
26: $L_{l,j} = \frac{2l-1}{l(l+1)} H_{l-1,j} + \frac{(l-2)(l-1)}{l(l+1)} L_{l-2,j}$ ▷ Eq. (31)
27: **end for**
28: **end for**
29: **for** $l = 0$ to $l_{\max}$ **do** ▷ ZH factorization
30: **for** $m = -l$ to l **do**
31: $L_{lm} = 0$
32: $\partial_z L_{lm} = 0$
33: **for** $j \in \{j \mid \alpha^m_{l,j} \neq 0\}$ **do** ▷ Sparse weights
34: L_{lm} += $\alpha^m_{l,j} L_{l,j}$
35: $\partial_z L_{lm}$ += $\alpha^m_{l,j} G_{l,j}$ ▷ Eq. (29)
36: **end for**
37: **end for**
38: **end for**
39: **return** $(\{L_{lm}\}, \{\partial_z L_{lm}\})$
40: **end function**

6.1 Iterative Evaluation of SH Coefficients and Gradients

We demonstrate the iterative evaluation of *both* SH coefficients and gradients in Algorithm 2. The algorithm takes a shading point $\boldsymbol{x}$, a polygon $\{\boldsymbol{p}_i\}$ with N points, the weights $\{\alpha^m_{l,j}\}$ and central directions $\{\boldsymbol{\omega}_{l,j}\}$ of the precomputed ZH lobes up to degree $l_{\max}$ as

Algorithm 3 Iterative evaluation of edge integral recurrences

1: **function** RECURRENCE(c_x, c_y, T_i, $l_{\max}$)
2: $B_0 = T_i$, $C_0 = c_x \sin T_i - c_y \cos T_i + c_y$ ▷ Base cases
3: $D_0 = 0$, $E_0 = c_x \cos T_i + c_y \sin T_i - c_x$
4: **for** $l = 1$ to $l_{\max}$ **do**
5: $B_l = \frac{2l-1}{l} C_{l-1} + \frac{l-1}{l} B_{l-2}$ ▷ Eq. (26)
6: $D_l = (2l-1) B_{l-1} + D_{l-2}$ ▷ Eq. (27)
7: $C_l = \mathrm{C}(c_x, c_y, T_i, l, D_l, B_{l-1})$ ▷ Eq. (25)
8: $E_l = \mathrm{E}(c_x, c_y, T_i, l)$ ▷ Eq. (24)
9: **end for**
10: **return** $(\{B_l\}, \{C_l\}, \{E_l\})$
11: **end function**

input. It outputs $(l_{\max}+1)^2$ SH coefficients $\{L_{lm}\}$ and SH gradients (spatial partial derivatives) $\{\partial_x L_{lm}\}$, $\{\partial_y L_{lm}\}$, $\{\partial_z L_{lm}\}$. [5]

Precomputation. First, we precompute the required quantities for each polygon vertex and edge. Note that the order of polygon vertices does matter. Specifically, the dot product of the face normal $(\boldsymbol{p}_1 - \boldsymbol{p}_0) \times (\boldsymbol{p}_2 - \boldsymbol{p}_0)$ and $\boldsymbol{p}_0 - \boldsymbol{x}$ should be positive. In Lines 2–4 of Algorithm 2, we translate the polygon vertices $\boldsymbol{p}_i$ by $-\boldsymbol{x}$, compute the distances ℓ_i between the vertices and the shading point, and obtain $\boldsymbol{\omega}_i$ by projecting the vertices onto the unit sphere. Then, we build a local frame $(\boldsymbol{\omega}_i, \boldsymbol{\lambda}_i, \boldsymbol{n}_i)$ for each polygon edge (Line 6) and compute its subtended angle T_i (Line 7).

Evaluation of Individual ZH Coefficients and Gradients. Starting from Line 9 of Algorithm 2, we evaluate the ZH coefficients (indicated in the orange background) and gradients (indicated in the purple background) for up to $(2l_{\max}+1)$ ZH lobes. We use $\boldsymbol{\omega}_c$ to indicate the central direction of the j-th ZH lobe (Line 10), given the lobe sharing strategy in Appendix B. The contributions to ZH coefficients and gradients will be accumulated for every polygon edge (Lines 13–25).

For each edge, the local coordinates (c_x, c_y, c_z) of $\boldsymbol{\omega}_c$ in the edge's local frame are calculated in Line 14. We calculate the values n_z, A and T' in Lines 15 and 16, which are required for the gradient evaluation. The edge integrals B_l, C_l and E_l are evaluated iteratively (Line 17) from $l = 0$ to $l_{\max}$. We demonstrate the computation of the recurrence formulae (Eqs. (24–27)) in Algorithm 3. For each band l, the values $H_{l,j}$ (Line 21) and the ZH gradients $G_{l,j}$ (Line 22) are updated based on Eqs. (30) and (23) respectively.

Finally, the ZH coefficients $L_{l,j}$ are evaluated in Lines 26–29, based on another recurrence formula related to $H_{l,j}$ (Eq. (31, 32)).

ZH Factorization. The final step is to reconstruct SH coefficients and gradients from the evaluated ZH coefficients $L_{l,j}$ and gradients $G_{l,j}$ (Lines 31–40). The ZH factorization weights $\alpha^m_{l,j}$ are precomputed according to [Nowrouzezahrai et al. 2012]. The sparsity of weights is maximized, so the SH reconstruction is efficient.

Summary. Evaluating ZH coefficients and gradients (Lines 9–30 of Algorithm 2) takes $O(Nl^2_{\max})$ time and reconstructing SH values with the ZH factorization (Lines 31–40) takes $O(l^3_{\max})$ time. Note that the last ZH factorization step is quite fast, since the weights $\alpha^m_{l,j}$ are sparse. The overall storage required is $O(l^2_{\max})$. Both the time and space complexity are the same as in previous work [Wang and Ramamoorthi 2018]. In terms of implementation, computing SH gradients along with SH coefficients only requires minimal effort (lines in Algorithm 2 with the purple background).

[5] For conciseness, we only show one of the partial derivatives in Algorithm 2. To obtain the other two partial derivatives, we only need to replace n_z by n_x/n_y (lines 15 and 22) and ∂_z by ∂_x/∂_y (lines 34 and 37).

6.2 Gradient-Based Interpolation

Given SH coefficients and gradients evaluated on a 3D grid, we can interpolate for any inside point (vertex in PRT) according to its eight adjacent grid points. Note that the interpolation time only depends on the highest SH degree $l_{\max}$ and is independent of the number of lights, making our method scalable to many lights.

In terms of interpolation methods, one can interpolate SH coefficients trilinearly, without using SH gradients at all. Previous work [Annen et al. 2004] uses an interpolation method based on Taylor series. They approximate the coefficients by the first-order Taylor polynomial from each adjacent grid point, then combine the results using the inverse distance weighting [Ward and Heckbert 1992]. Both interpolation methods result in limited accuracy (see Figs. 4 and 7). Instead, we use a more principled Hermite interpolation [Van Loan 1996], approximating the interpolant as a tricubic polynomial. We first provide details of the cubic Hermite interpolation in 1D. Then, we discuss its extension to 3D.

1D Cubic Hermite Interpolation. Suppose the function we are going to interpolate $f(x)$ is continuous on $[x_L, x_R]$ and we know its values f_L, f_R and the first-order derivatives f'_L, f'_R at the endpoints, respectively. The cubic Hermite interpolant $q(x)$ is a cubic polynomial with four unknown coefficients a, b, c and d [Van Loan 1996],

$$q(x) = a + b(x - x_L) + c(x - x_L)^2 + d(x - x_L)^2(x - x_R), \quad (33)$$

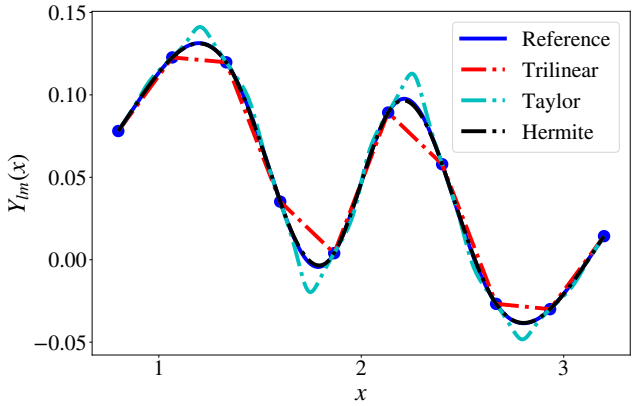


Fig. 4. Given a rectangular area light and a shading point, we plot the SH coefficients for $(l, m) = (7, 2)$ as the shading point moves along the x-axis. The reference curve (blue solid line) is densely sampled at 2000 points. Alternatively, we evaluate SH coefficients and gradients at 10 points (blue dots) and interpolate the coefficients in between. The trilinear interpolation (red dashed line) and the Taylor-series based interpolation (cyan dashed line) [Annen et al. 2004] result in insufficient accuracy, while the cubic Hermite interpolation result (black dashed line) matches the reference almost perfectly.

Algorithm 4 Tricubic Hermite interpolation from function values f_i and gradients $\nabla f_i = (\partial_x f_i, \partial_y f_i, \partial_z f_i)$ of the eight adjacent grid points $i = 0, 1, \ldots, 7$

1: **function** TricubicHermite($\boldsymbol{p}$, *gridSize*, $\{f_i\}$, $\{\nabla f_i\}$)
2: $(x_R, y_R, z_R) = gridSize$ ▷ Size of a grid voxel
3: $(x, y, z) = \boldsymbol{p}$ ▷ Vertex coordinates inside the grid voxel
4: **for** $i = 0$ to 3 **do** ▷ Interpolate along the x-axis
5: g_i = Hermite1D$(x, \{0, f_{2i}, \partial_x f_{2i}\}, \{x_R, f_{2i+1}, \partial_x f_{2i+1}\})$
6: Interpolate ∇g_i linearly based on ∇f_{2i} and ∇f_{2i+1}
7: **end for**
8: **for** $i = 0$ to 1 **do** ▷ Interpolate along the y-axis
9: h_i = Hermite1D$(y, \{0, g_{2i}, \partial_y g_{2i}\}, \{y_R, g_{2i+1}, \partial_y g_{2i+1}\})$
10: Interpolate ∇h_i linearly based on ∇g_{2i} and ∇g_{2i+1}
11: **end for**
12: $q(x, y, z)$ = Hermite1D$(z, \{0, h_0, \partial_z h_0\}, \{z_R, h_1, \partial_z h_1\})$
13: **return** $q(x, y, z)$ ▷ Interpolate along the z-axis
14: **end function**
15: **function** Hermite1D$(x, \{x_L, f_L, f'_L\}, \{x_R, f_R, f'_R\})$
16: $x_\Delta = x_R - x_L$, $s = (f_R - f_L)/x_\Delta$
17: $a = f_L$, $b = f'_L$, $c = \frac{s - f'_L}{x_\Delta}$, $d = \frac{f'_L + f'_R - 2s}{x_\Delta^2}$ ▷ Eq. (35)
18: **return** $a + b(x - x_L) + c(x - x_L)^2 + d(x - x_L)^2(x - x_R)$
19: ▷ Eq. (33)
20: **end function**

satisfying

$$q(x_L) = f_L,\ q(x_R) = f_R,\ q'(x_L) = f'_L,\ q'(x_R) = f'_R. \tag{34}$$

We can obtain the four unknown coefficients by solving this linear system (Eq. (34)),

$$a = f_L,\ b = f'_L,\ c = (s - f'_L)/x_\Delta,\ d = (f'_L + f'_R - 2s)/x_\Delta^2, \tag{35}$$

where $x_\Delta = x_R - x_L$ and $s = (f_R - f_L)/x_\Delta$. We demonstrate it in Lines 15–20 of Algorithm 4. The 1D curve plot in Fig. 4 indicates the accuracy benefit comparing to other interpolation methods.

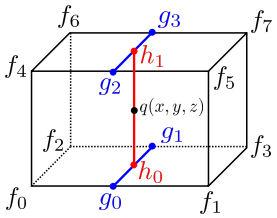
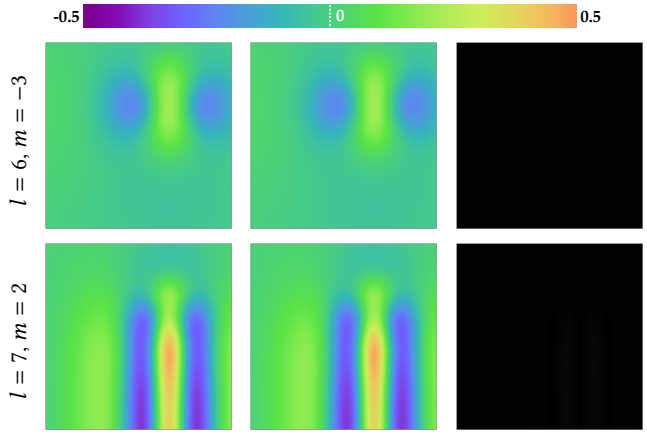


Fig. 5. Illustration of 3D Hermite interpolation.

3D Tricubic Hermite Interpolation. Tricubic Hermite interpolation can be done by performing the 1D cubic Hermite interpolation along the three axes progressively (see Fig. 5). For any intermediate points, the SH coefficients are Hermite interpolated but we need to know the SH gradients (first-order derivatives) as well. In theory, applying Hermite interpolation to the first-order derivatives requires the second-order derivatives, which will involve extra cost to evaluate. For efficiency, we interpolate SH gradients trilinearly and get satisfactory results.

We provide the pseudocode of the 3D tricubic Hermite interpolation in Algorithm 4. For a point $\boldsymbol{p}$ in a grid voxel with size (x_R, y_R, z_R), we translate it so that the bottom-left corner is at $(0, 0, 0)$. First, we interpolate along the x-axis (Lines 4–7), calculating the function values $g_0, \ldots, g_3$ by the 1D Hermite interpolation and gradients $\nabla g_0, \ldots, \nabla g_3$ by the linear interpolation (see the blue points in Fig. 5). We then interpolate along the y-axis (Lines 8–11) and obtain the values h_0, h_1 and $\nabla h_0, \nabla h_1$ (see the red points in Fig. 5). Finally, we compute the interpolant value $q(x, y, z)$ by interpolating along the z-axis (Line 12).

Note that theoretically, the interpolation result depends on the order of the individual 1D steps. However in practice, we do not observe significant differences when we change the order.



(a) FD (b) Ours (c) 20× Abs. diff.

Fig. 6. We show visualization plots of SH gradients (∂_x-component) for $(l, m) = (6, -3)$ in the top row and $(l, m) = (7, 2)$ in the bottom row. Images in column (a) are computed using finite differences (FD) based on [Wang and Ramamoorthi 2018], and images in column (b) are computed with our method (Algorithm 2). In these images, each pixel stores a partial derivative value encoded in false colors; The (20×) absolute differences between FD results and our results are given in column (c), indicating the correctness of our derivation and algorithm.

7 RESULTS

We implement Algorithms 1–4 in GPU shaders and compute SH lighting coefficients for each vertex in a scene. The lighting coefficients are used in a PRT system, which is implemented within the Falcor open-source real-time rendering framework [Benty et al. 2019]. We run our algorithm on a few scenes with multiple polygonal area lights using an NVIDIA RTX 2080 Ti GPU. The scene configurations and performance statistics are summarized in Table 1. We release our code and data in the supplementary material.

7.1 Validation and Evaluation

Validation of Analytic SH Gradients. To validate our derivation of analytic SH gradients, we compare SH gradients evaluated using our method (Algorithm 2) and finite differences (FD) based on the work by Wang and Ramamoorthi [2018]. Given a rectangular area light with the bottom-left corner at $(-5, -5, 1)$ and the top-right corner at $(5, 5, 1)$, the ∂_x-component of SH gradients is evaluated in another square region whose bottom-left corner is at $(3, 3, 0)$ and top-right corner is at $(6, 6, 0)$. We visualize the derivatives as 2D false-colored images in Fig. 6. Our analytic formulae agree with the numerical FD results, except for some negligible differences caused by the FD step size δ (we choose $\delta = 10^{-3}$). Note that computing derivatives with the central FD requires SH coefficient evaluation at multiple points

Table 1. Scene configurations and performance statistics. We compare running time and image mean absolute errors (MAE) to those by Wang and Ramamoorthi [2018], which we treat as the Reference. Note that the total running time of our method also includes other necessary operations such as rasterization. We also provide the running time of evaluating *only* SH coefficients to highlight the low overhead of SH gradient evaluation.

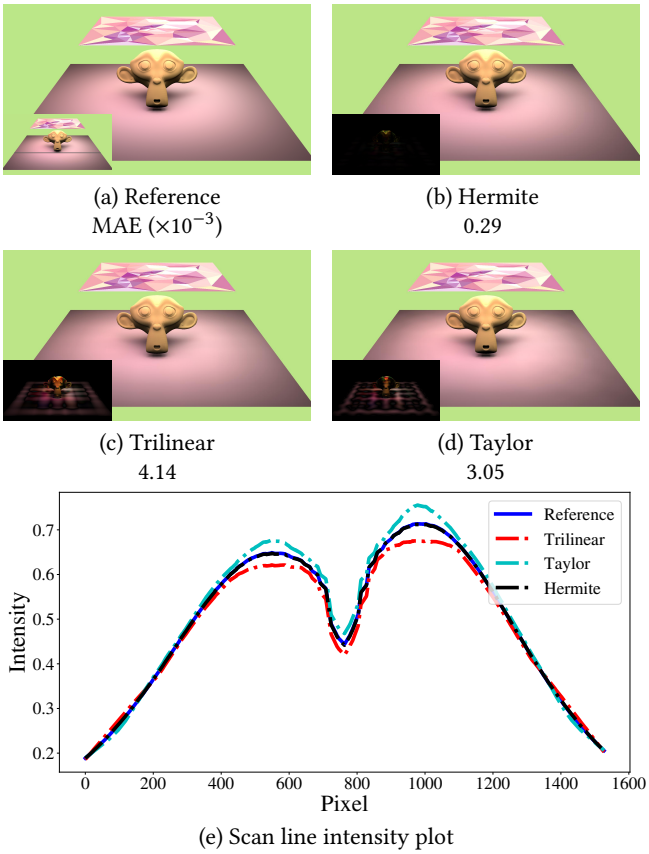
Scene	Figure	Triangles	Lights	Grid Reso.	Eval. SH coeff. (ms)	Eval. coeff. & grad. (ms)	Interp. (ms)	**Total (ms)**	FPS	MAE ($\times 10^{-3}$)	Reference (ms)	Speed up
Dragon & Bunny	Fig. 1	1.28M	723	8^3	16.5	20.2	6.6	**27.9**	35.8	0.18	110K	3943×
Monkey	Fig. 7	71.2K	118	8^3	1.3	1.8	0.4	**3.3**	303	0.29	177	35×
Plants	Fig. 8	190K	2	8^3	0.08	0.1	2.1	**3.1**	323	0.17	9.4	3×
Asian Dragon	Fig. 9	1.42M	181	8^3	2.3	3.0	8.1	**12.7**	78.7	1.35	3.22K	254×
Room	Fig. 10	1.71M	344	8^3	4.3	5.8	10.3	**17.7**	56.5	0.34	6.85K	387×
Buddha	Fig. 11	422K	210	8^3	4.8	5.7	6.0	**12.7**	78.7	0.07	28.6K	2252×

(two for each axis), while our SH gradient evaluation can come along with a single SH coefficient evaluation, causing minimal overhead. We compare the performance numbers of these two methods with a CPU-based C++ implementation; our method is 3× faster than FD.

Interpolation Methods. We have already demonstrated in Fig. 4 that the Hermite interpolation is most accurate in 1D cases. In terms of the extension to 3D, we compare rendering results using different interpolation methods in Fig. 7. Hermite interpolation results in an order of magnitude smaller error than that of trilinear interpolation and Taylor-series based interpolation. Note that Taylor-series based interpolation [Annen et al. 2004] requires SH gradients, thus can also benefit from our analytic SH gradient evaluation. Although the mean absolute error (MAE) numbers are relatively small, there are regions with significant inaccuracies in other methods. Checking the error maps in Fig. 7(b–d), the result using Hermite interpolation has significantly fewer pixels with large differences from the reference. Further, we check and plot the image intensity values along a scan line in Fig. 7(e). The black curve representing Hermite interpolation is almost identical to the reference blue curve, while the red and cyan curves, representing trilinear and Taylor-series based interpolation respectively, deviate from the reference.

Scalability with Multiple lights. In Fig. 8(a), we compare the total lighting coefficient computation time for all vertices (time for other stages in the rendering pipeline such as rasterization is excluded) with the previous method [Wang and Ramamoorthi 2018]. The running time of the previous method goes up linearly with the number of lights as expected, while our method has only a small time cost even with 512 lights. This is because the expensive computation for every light is done on a sparse grid in our method. Our technique can render a scene with hundreds of area lights in real-time, which was not previously possible. We further analyze the performance of our method and plot the running time of each step in Fig. 8(b). The SH coefficient and gradient evaluation time scales linearly with the increasing number of lights, but the performance impact is mitigated since we use a sparse grid. The Hermite interpolation for each vertex requires essentially constant time.

Grid Resolution. We demonstrate how the grid resolution influences the rendering performance and accuracy in Fig. 9. Even though the result with resolution 4^3 already yields good visual quality (inspection of error images shows subtle differences in shading on the dragon and floor), the image accuracy improves as the grid resolution becomes higher. But using a finer grid also requires longer



(a) Reference
MAE ($\times 10^{-3}$)

(b) Hermite
0.29

(c) Trilinear
4.14

(d) Taylor
3.05

(e) Scan line intensity plot

Fig. 7. Accuracy comparison of different interpolation methods: (b) tricubic Hermite interpolation, (c) trilinear interpolation, and (d) Taylor-series based interpolation [Annen et al. 2004]. The reference image (a) is rendered by computing the lighting SH coefficients at every vertex. We achieve almost the same image quality (b) by computing SH coefficients and gradients in a 3D grid with resolution 8^3, and Hermite interpolating the light coefficients for each vertex. The (10×) absolute error images are given in the bottom-left insets, as well as the corresponding mean absolute error (MAE) numbers. We also plot image intensity curves (e) for different interpolation methods along a scan line (illustrated in the inset of (a)).

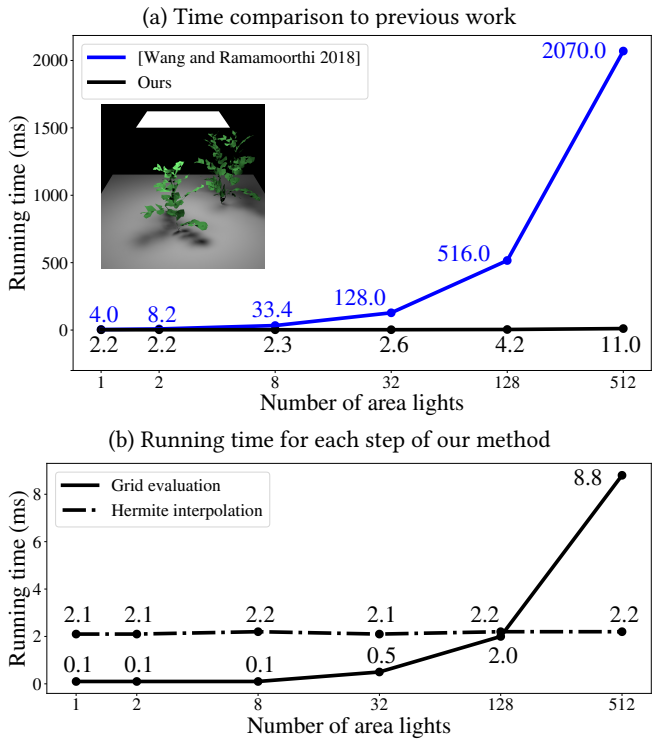


Fig. 8. Plots of SH coefficient computation time at all vertices with increasing numbers of area lights. The scene is shown in the inset of (a); we gradually subdivide the rectangular area light up to 512 triangles. (a) Previous work scales linearly in the number of area lights, so they cannot handle many lights in real-time. On the other hand, our method is insensitive to the increase in the number of lights, since we only need to compute the light coefficients and gradients on a sparse grid. (b) Breaking down the running time of our method, we can see the time for grid evaluation is also linear in the number of lights (but still efficient because of the sparse grid), while Hermite interpolation costs essentially constant time.

computation time and larger storage overhead. To balance performance and accuracy, we use a resolution of 8^3 in all the results.

Relation to Source Radiance Fields. In previous work [Zhou et al. 2005], source radiance fields (SRF) are precomputed and stored for efficient incident radiance evaluation. To support dynamic lighting, a 5D SRF is required for each area light, causing additional storage overhead that scales linearly in the number of lights. However, our method only stores SH coefficients and gradients at sparse grid points, which is independent of the number of lights. Moreover, the source radiance fields at intermediate points are interpolated, which can also benefit from our gradient-based interpolation.

7.2 Main Results

We now present additional results of rendering more complex scenes. Please see the supplementary video for animated versions of Figs. 1, 10, and 11, rendered at real-time frame rates (35-80 fps)

Textured Lights. Given a textured area light, we break the light source into smaller polygons (triangles) that are each uniformly

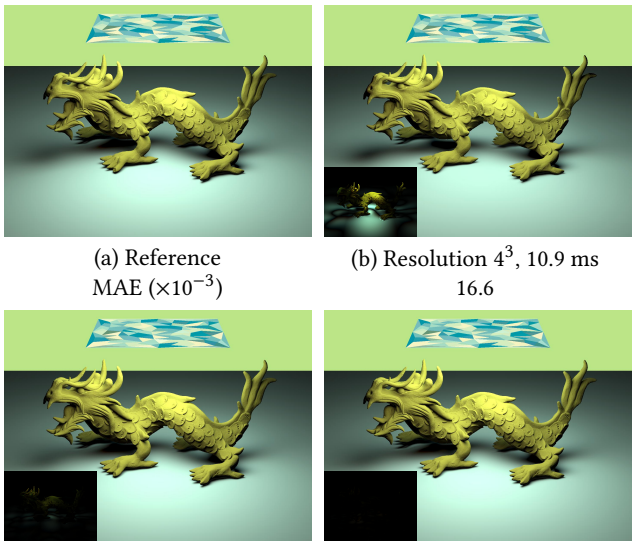


Fig. 9. Performance and accuracy comparison with increasing resolutions of 3D grids. The bottom-left insets show (5×) absolute error images.

Fig. 10. A living room illuminated by two textured lights. Light sources and scene layout are illustrated in the inset figure.

emissive. We show an example in Fig. 10, in which a room is illuminated by a blue and a pink textured light (see the bottom-left inset of Fig. 10). Compared to the previous method [Wang and Ramamoorthi 2018], we can render this scene with hundreds of lights (and 1.7M polygons) in real-time, achieving a more than two orders of magnitude speed up.

Glossy Reflection. We also show two examples with glossy materials in Figs. 1 and 11. We compute the glossy reflection by extending Eq. (1) to a triple product SH integral of lighting, BRDF and precomputed cosine-weighted visibility [Sloan et al. 2002; Ng et al. 2004]. Since the time complexity of the triple product integral computation is $O(l_{\max}^5)$ [Ng et al. 2004], we bandlimit the lighting and visibility SH coefficients with $l_{\max} = 4$. Phong BRDFs are used in all these examples, represented by SH coefficients with $l_{\max} = 8$. Our method focuses on evaluating the lighting SH coefficients only, and is orthogonal to the glossy PRT framework.

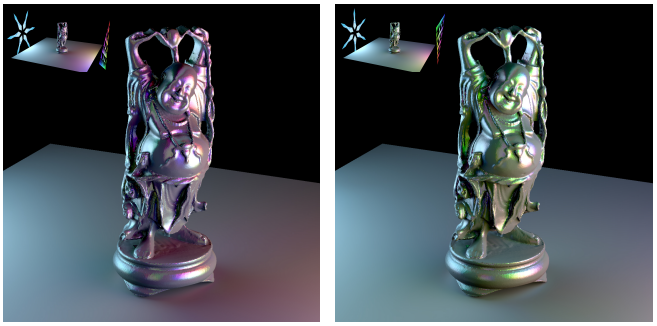


Fig. 11. Glossy reflections caused by more complex light sources. Light sources and scene layout are illustrated in the inset figure.

In Fig. 1, we show images rendered with three textured lights of gradually changing colors. These textured lights are made up of more than seven hundred uniform triangular light sources in total, and each of them is allowed to move independently, ultimately forming a pattern which is not even strictly a textured light source(Fig. 1(right)). Colors of the glossy highlights change significantly as we transform the lights. Even though there are hundreds of independent dynamic lights, we are still able to render this scene in real-time, which might be challenging for source radiance fields [Zhou et al. 2005], since it requires precomputation for each light source.

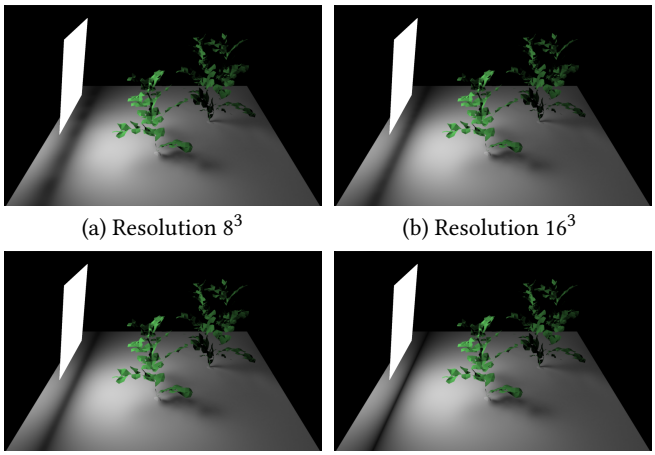
In Fig. 11, we illuminate a Buddha model on the ground with two lights of irregular shapes (see the insets). The left light is a blue snow flake and the right one is a colorful fractal triangle. As we rotate the lights, the glossy highlights on the buddha change from purple to green. The highlights on the ground also vary according to the light transformation.

7.3 Limitations and Future Work

On regions close to the light source or at grazing angles, there can be high-frequency lighting variations that require a fine grid for accurate SH interpolation. Figure 12 shows a scene with a double-sided area light inside its bounding box. The shading that is on the ground and close to the light's grazing angle looks blurry when the grid resolution is 8^3. We will have more accurate interpolation results given grids with higher resolutions. Using multi-level adaptive grids [Greger et al. 1998] may further improve the performance and accuracy in such cases.

Our method is based on PRT using spherical harmonics, which approximates path tracing and captures only low-frequency effects due to the band-limited SH. In Fig. 13, we compare our result to the image rendered using path tracing. There are some subtle differences at the shadows on the ground and the plant leaves near the light source. Nevertheless, we match the result generated by Wang and Ramamoorthi [2018] almost perfectly, which also uses PRT. How to improve the accuracy of all-frequency effects using spherical harmonic PRT is orthogonal to our work and beyond the scope of this paper. We also hope our work can inspire future research on real-time path tracing.

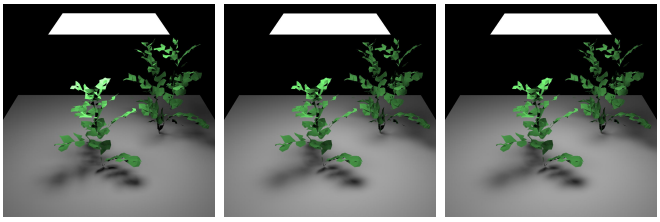
In this paper, we focus on accurate analytic formulae for SH gradients. In certain cases, e.g., the light source is far away, one sample per pixel for numerical integration could be sufficient and faster than analytic evaluation. Switching between numerical and analytic evaluations according to some heuristics [Yuan et al. 2012] might be potentially helpful. In addition, applying analytical approximations [Lecocq et al. 2017] might make the gradient evaluations more efficient.



(a) Resolution 8^3 (b) Resolution 16^3

(c) Resolution 32^3 (d) Reference

Fig. 12. An example with a double-sided area light source inside the scene. Due to the high-frequency light variations, we need finer grids for accurate SH interpolation.



(a) Path tracing ≈ (b) PRT (Ours) = (c) PRT (Wang and Ramamoorthi 2018)

Fig. 13. Comparison against path tracing. Our result (b) is close to the image rendered using path tracing (a) in spite of subtle differences at the shadows on the ground and the plant leaves near the light source, while it matches the result of Wang and Ramamoorthi [2018] almost perfectly.

Note that non-uniform or textured lights can be handled by our method, simply by breaking the light source into smaller uniform components. With techniques demonstrated in prior analytic methods [Arvo 1995; Chen and Arvo 2000, 2001], it might be possible to extend our analytic SH gradients for piecewise linear area light sources. Moreover, we are currently limited to uniform angular emission rather than, for example, spotlights. We seek to lift this limitation in the future, perhaps by developing fast boundary numerical integration schemes.

Finally, note that we do not currently address multiple lights shadowing each other (although PRT methods that support dynamic shadows can partially address this situation). We believe the boundary integral formulation of this paper could also generalize occluded irradiance gradients [Arvo 1994] to SH gradients by incorporating polygon depth clipping.

8 CONCLUSIONS

We have presented a novel analytic derivation for SH gradients from uniform polygonal area lights, showing how to reduce the calculations to a boundary integral and ultimately to an earlier recurrence for SH coefficients. The derivation fills an important gap in both SH and PRT methods, as well as more recent differential rendering techniques. While the derivation is complicated, the actual implementation is simple, requiring only a few additional lines of code beyond those for SH coefficients. We show how gradients can be used for Hermite interpolation with very high accuracy and sparse grid sizes. Crucially, we can accumulate the contributions of hundreds of lights with only minor overhead, enabling PRT to scale to hundreds of independent area lights in real-time.

PRT represents only one possible application. SH gradients could also be used for importance sampling the radiance field from multiple area lights in offline rendering, and for extensions such as path guiding. We simply need to sample the Hermite-interpolated SH lighting at each shading point or pixel. Given that SH gradients are a fundamental mathematical quantity, we believe there are many other interesting possibilities in rendering and beyond.

ACKNOWLEDGMENTS

We thank the reviewers for their constructive comments. We are grateful to Jingwen Wang for implementation suggestions. This work was funded in part by NSF grant 1900927, an NVIDIA Fellowship, the Ronald L. Graham Chair, and the UC San Diego Center for Visual Computing.

REFERENCES

T Annen, J Kautz, F Durand, and H Seidel. 2004. Spherical Harmonic Gradients for Mid-range Illumination. *Proceedings of the 15th Eurographics Conference on Rendering Techniques* (2004), 331–336.

J Arvo. 1994. The irradiance Jacobian for partially occluded polyhedral scenes. In *SIGGRAPH 94*. 343–350.

J Arvo. 1995. Applications of irradiance tensors to the simulation of non-Lambertian phenomena. In *SIGGRAPH 95*. 335–342.

L Belcour, G Xie, C Hery, M Meyer, W Jarosz, and D Nowrouzezahrai. 2018. Integrating clipped spherical harmonics expansions. *ACM Transactions on Graphics* 37, 2 (2018), 19:1–19:12.

N Benty, K Yao, L Chen, T Foley, M Oakes, C Lavelle, and C Wyman. 2019. The Falcor Rendering Framework. https://github.com/NVIDIAGameWorks/Falcor

B Cabral, N Max, and R Springmeyer. 1987. Bidirectional Reflection functions from surface bump maps. In *SIGGRAPH 87*. 273–281.

M Chen and J Arvo. 2000. A Closed-Form Solution for the Irradiance Due to Linearly-Varying Luminaires. *Proceedings of the 11th Eurographics Workshop on Rendering* (2000), 137–148.

M Chen and J Arvo. 2001. Simulating Non-Lambertian Phenomena Involving Linearly-Varying Luminaires. *Proceedings of the 12th Eurographics Workshop on Rendering* (2001), 25–38.

G Greger, P Shirley, P Hubbard, and D Greenberg. 1998. The Irradiance Volume. *IEEE Computer Graphics & Applications* 18, 2 (1998), 32–43.

E. Heitz, J. Dupuy, S. Hill, and D. Neubelt. 2016. Real-Time Polygonal-Light Shading with Linearly Transformed Cosines. *ACM Transactions on Graphics* 35, 4 (2016), 41:1–41:8.

N Holzschuch and F Sillion. 1995. Accurate Computation of the Radiosity Gradient with Constant and Linear Emitters. *Rendering Techniques 1995, Eurographics Symposium on Rendering* (1995), 186–195.

N Holzschuch and F Sillion. 1998. An Exhaustive Error-Bounding Algorithm for Hierarchical Radiosity. *Computer Graphics Forum* 17 (1998), 197–218.

W Jarosz, C Donner, M Zwicker, and H Jensen. 2008a. Radiance Caching for Participating Media. *ACM Transactions on Graphics* 27, 1 (2008), 7:1–7:11.

W Jarosz, V Schönefeld, L Kobbelt, and H Jensen. 2012. Theory, Analysis and Applications of 2D Global Illumination. *ACM Transactions on Graphics* 31, 5 (2012), 125:1–125:21.

W Jarosz, M Zwicker, and H Jensen. 2008b. Irradiance Gradients in the Presence of Participating Media and Occlusions. *Computer Graphics Forum (Proceedings of EGSR)* 27, 4 (2008), 1087–1096.

J Křivánek, K Bouatouch, S Pattanaik, and J Žára. 2006. Making Radiance and Irradiance Caching Practical: Adaptive Caching and Neighbor Clamping. *Rendering Techniques 2006, Eurographics Symposium on Rendering* (2006), 127–138.

J Křivánek, P Gautron, K Bouatouch, and S Pattanaik. 2005a. Improved Radiance Gradient Computation. *SCCG '05: Proceedings of the 21th spring conference on computer graphics* (2005), 155–159.

J Křivánek, P Gautron, S Pattanaik, and K Bouatouch. 2005b. Radiance Caching for Efficient Global Illumination Computation. *IEEE Transactions on Visualization and Computer Graphics* 11, 5 (2005), 550–561.

J Křivánek, P Gautron, G Ward, H Jensen, E Tabellion, and P Christensen. 2008. Practical Global Illumination with Irradiance Caching. In *SIGGRAPH Courses*.

L. Gary Leal. 2007. *Advanced Transport Phenomena: fluid mechanics and convective transport processes*. Vol. 7. Cambridge University Press.

P Lecocq, A Dufay, G Sourimant, and J Marvie. 2017. Analytic Approximations for Real-Time Area Light Shading. *IEEE Transactions on Visualization and Computer Graphics* 99 (2017).

J Lehtinen. 2007. A Framework for Precomputed and Captured Light Transport. *ACM Transactions on Graphics* 26, 4 (2007), 13:1–13:22.

T. Li, M. Aittala, F. Durand, and J. Lehtinen. 2018. Monte Carlo ray tracing through edge sampling. *ACM Transactions on Graphics* 37, 6 (2018), 222:1–222:11.

T. Li, J. Lehtinen, R. Ramamoorthi, W. Jakob, and F. Durand. 2015. Anisotropic Gaussian mutations for Metropolis Light Transport Through Hessian-Hamiltonian Dynamics. *ACM Transactions on Graphics* 34, 6 (2015), 209:1–209:13.

H Liu, M Tao, C Li, D Nowrouzezahrai, and A Jacobson. 2019. Beyond Pixel Norm-Balls: Parametric Adversaries using an Analytically Differentiable Renderer. *International Conference on Learning Representations* (2019).

G Loubet, N Holzschuch, and W Jakob. 2019. Reparameterizing discontinuous integrands for differentiable rendering. *ACM Transactions on Graphics* 38, 6 (2019), 228:1–228:14.

T MacRobert. 1948. *Spherical harmonics: an elementary treatise on harmonic functions with applications*. Dover Publications.

J Marco, A Jarabo, W Jarosz, and D Gutierrez. 2018. Second-Order Occlusion-Aware Volumetric Radiance Caching. *ACM Transactions on Graphics* 37, 2 (2018), 20:1–20:14.

R Ng, R Ramamoorthi, and P Hanrahan. 2003. All-Frequency Shadows using Non-Linear Wavelet Lighting Approximation. *ACM Transactions on Graphics* 22, 3 (2003), 376–381.

R Ng, R Ramamoorthi, and P Hanrahan. 2004. Triple Product Wavelet Integrals for All-Frequency Relighting. *ACM Transactions on Graphics (Proc. SIGGRAPH 04)* 23, 3 (2004), 475–485.

D Nowrouzezahrai, P Simari, and E Fiume. 2012. Sparse Zonal Harmonic Factorization for Efficient SH Rotation. *ACM Transactions on Graphics* 31, 3 (2012), 23:1–23:9.

J Pantaleoni, L Fascione, M Hill, and T Aila. 2010. PantaRay: fast ray-traced occlusion caching of massive scenes. *ACM Transactions on Graphics* 29, 4 (2010).

R Ramamoorthi. 2009. Precomputation-Based Rendering. *Foundations and Trends in Computer Graphics and Vision* 3, 4 (2009), 281–369.

R Ramamoorthi, D Mahajan, and P Belhumeur. 2007. A First Order Analysis of Lighting, Shading, and Shadows. *ACM Transactions on Graphics* 26, 1 (2007).

Z Ren, R Wang, J Snyder, K Zhou, X Liu, B Sun, P Sloan, H Bao, Q Peng, and B Guo. 2006. Real-time Soft Shadows in Dynamic Scenes using Spherical Harmonic Exponentiation. *ACM Transactions on Graphics* 25, 3 (2006), 977–986.

J Schwarzhaupt, H Jensen, and W Jarosz. 2012. Practical Hessian-Based Error Control for Irradiance Caching. *ACM Transactions on Graphics* 31, 6 (2012), 193:1–193:10.

P Sloan, J Kautz, and J Snyder. 2002. Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments. *ACM Transactions on Graphics* 21, 3 (2002), 527–536.

J. Snyder. 1996. *Area Light Sources for Real-Time Graphics*. Technical Report MSR-TR-96-11. Microsoft Research.

B Sun and R Ramamoorthi. 2009. Affine double and triple product wavelet integrals for rendering. *ACM Transactions on Graphics* 28, 2 (2009).

C. F. Van Loan. 1996. *Introduction to Scientific Computing: A Matrix-Vector Approach Using MATLAB*. Prentice-Hall, Inc.

J Wang and R Ramamoorthi. 2018. Analytic Spherical Harmonic Coefficients for Polygonal Area Lights. *ACM Transactions on Graphics* 37, 4 (2018), 54:1–54:11.

G Ward and P Heckbert. 1992. Irradiance Gradients. In *Eurographics Rendering Workshop 92*. 85–98.

G Ward, F Rubinstein, and R Clear. 1988. A Ray Tracing Solution for Diffuse Interreflection. *SIGGRAPH* 22, 4 (1988), 85–92.

H Yuan, D Nowrouzezahrai, and P Sloan. 2012. Irradiance Rigs. *Journal of Graphics, GPU, and Game Tools* 16, 1 (2012).

C. Zhang, L. Wu, C. Zheng, I. Gkioulekas, R. Ramamoorthi, and S. Zhao. 2019. A Differential Theory of Radiative Transfer. *ACM Transactions on Graphics* 38, 6 (2019).

K Zhou, Y Hu, S Lin, B Guo, and H Shum. 2005. Precomputed shadow fields for dynamic scenes. *ACM Transactions on Graphics* 24, 3 (2005), 1196–1201.

A SPHERICAL HARMONICS

The real SH functions for $l \geq 0, -l \leq m \leq l$ are [MacRobert 1948]

$$Y_{lm}(\boldsymbol{\omega}) = \begin{cases} \sqrt{2}K_{lm}\sin(|m|\phi)P_l^{|m|}(\cos\theta), & m < 0, \\ K_{l0}P_l^0(\cos\theta), & m = 0, \\ \sqrt{2}K_{lm}\cos(m\phi)P_l^m(\cos\theta), & m > 0, \end{cases} \tag{36}$$

where P_l^m are the associated Legendre polynomials and K_{lm} is the normalization term, given by $K_{lm} = \sqrt{\frac{2l+1}{4\pi}\frac{(l-|m|)!}{(l+|m|)!}}$. Setting $m = 0$, we obtain the ZH basis form with $K_l = K_{l0}$ and the Legendre polynomial $P_l = P_l^0$.

B SHARING ZH LOBES ACROSS BANDS

To enable efficient ZH factorization, Nowrouzezahrai et al. [2012] presented a ZH lobe sharing strategy. For SH basis functions up to degree $l_{\max}$, the central directions of ZH lobes are

$$\begin{aligned} \{\boldsymbol{\omega}_{0,0} = \boldsymbol{\omega}_{1,0} = \boldsymbol{\omega}_{2,0} = \boldsymbol{\omega}_{3,0} = \cdots = \boldsymbol{\omega}_{l_{\max},0}, \\ \boldsymbol{\omega}_{1,1} = \boldsymbol{\omega}_{2,1} = \boldsymbol{\omega}_{3,1} = \cdots = \boldsymbol{\omega}_{l_{\max},1}, \\ \boldsymbol{\omega}_{1,2} = \boldsymbol{\omega}_{2,2} = \boldsymbol{\omega}_{3,2} = \cdots = \boldsymbol{\omega}_{l_{\max},2}, \\ \cdots\cdots, \boldsymbol{\omega}_{l_{\max},2l_{\max}}\}. \end{aligned} \tag{37}$$

Specifically, the band-l SH basis functions will use a set of $2l+1$ central directions $\{\boldsymbol{\omega}_{l,0}, \ldots, \boldsymbol{\omega}_{l,2l}\}$.

C RELATION TO [Annen et al. 2004]

Annen et al. [2004] developed a semi-analytic solution to SH gradients. They integrate the SH basis functions over the area domain A of the polygonal light source, which is independent of the varying shading point $\boldsymbol{x}$. By changing the solid angle measure to the area measure in Eq. (4), the SH coefficients can be rewritten as

$$L_{lm}(\boldsymbol{x}) = \int_{[0,1]^2} Y_{lm}(\boldsymbol{s}(\boldsymbol{u}))\frac{\langle \boldsymbol{n}(\boldsymbol{y}(\boldsymbol{u})), -\boldsymbol{s}(\boldsymbol{u})\rangle}{\|\boldsymbol{y}(\boldsymbol{u}) - \boldsymbol{x}\|^2}|\det \mathcal{J}_{\boldsymbol{y}}|\,\mathrm{d}\boldsymbol{u}. \tag{38}$$

Note that $\boldsymbol{y} : [0,1]^2 \to A$ is a transformation that warps a unit square to a polygon and $\mathcal{J}_{\boldsymbol{y}}$ denotes the corresponding Jacobian matrix. The normalized direction vector $\boldsymbol{s}(\boldsymbol{u}) = \frac{\boldsymbol{y}(\boldsymbol{u})-\boldsymbol{x}}{\|\boldsymbol{y}(\boldsymbol{u})-\boldsymbol{x}\|}$ is from the shading point $\boldsymbol{x}$ to a point $\boldsymbol{y}(\boldsymbol{u})$ on the area light. In the change-of-measure term $\frac{\langle \boldsymbol{n}(\boldsymbol{y}(\boldsymbol{u})), -\boldsymbol{s}(\boldsymbol{u})\rangle}{\|\boldsymbol{y}(\boldsymbol{u})-\boldsymbol{x}\|^2}$, we indicate $\boldsymbol{n}(\boldsymbol{y}(\boldsymbol{u}))$ as the surface normal.

To differentiate the integral in Eq. (38), we can directly move the differentiation operator into the integration,

$$\partial_z L_{lm}(\boldsymbol{x}) = \int_{[0,1]^2} \partial_z \left[Y_{lm}(\boldsymbol{s}(\boldsymbol{u}))\frac{\langle \boldsymbol{n}(\boldsymbol{y}(\boldsymbol{u})), -\boldsymbol{s}(\boldsymbol{u})\rangle}{\|\boldsymbol{y}(\boldsymbol{u}) - \boldsymbol{x}\|^2}|\det \mathcal{J}_{\boldsymbol{y}}| \right] \mathrm{d}\boldsymbol{u}. \tag{39}$$

This is a special case of the Reynolds transport theorem. The boundary integral vanishes since the integration domain is static. The integrand in Eq. (39) can be computed either analytically or using automatic differentiation. But the 2D integral needs to be evaluated numerically, and has no known analytic form.

D DETAILED DERIVATIONS

D.1 Deriving $\ell(t)$ in Eq. (17)

After transforming the edge $\overline{\boldsymbol{p}_i\boldsymbol{p}_{i+1}}$ into the local frame $(\boldsymbol{\omega}_i, \boldsymbol{\lambda}_i, \boldsymbol{n}_i)$, the edge is in the xy-plane so we can omit the z-coordinate and solve for $\ell(t)$ in 2D. The 2D local coordinates of the edge endpoints are $\boldsymbol{p}_i = (\ell_i, 0)$ and $\boldsymbol{p}_{i+1} = (\ell_{i+1}\cos T_i, \ell_{i+1}\sin T_i)$. We want to know the travel distance ℓ of the ray with direction $(\cos t, \sin t)$ starting from $(0,0)$, before hitting the edge. The geometric relation can be described in the following linear system,

$$\begin{cases} \ell\cos t = (1-k)\ell_i + k\ell_{i+1}\cos T_i, \\ \ell\sin t = k\ell_{i+1}\sin T_i. \end{cases} \tag{40}$$

After eliminating k, we have

$$\begin{aligned} &\ell(\ell_i\sin t - \ell_{i+1}\sin(t - T_i)) = \ell_i\ell_{i+1}\sin T_i \\ \Rightarrow\ &\ell(t) = \left(\frac{\sin t}{\ell_{i+1}} - \frac{\sin(t-T_i)}{\ell_i}\right)^{-1}\sin T_i. \end{aligned} \tag{41}$$

D.2 Deriving Eq. (22)

Let $h = c_x\cos t + c_y\sin t = A\sin(t+T')$, where $A = \sqrt{c_x^2 + c_y^2}$ and $T' = \arctan(c_x/c_y)$. We use a change of variable $u = t + T'$. Then, the first integral on the RHS of Eq. (20) can be rewritten as

$$\begin{aligned} &\int_0^{T_i} \sin(t - T_i)P_l(c_x\cos t + c_y\sin t)\,\mathrm{d}t \\ &= \frac{1}{A}\int_{T'}^{T_i+T'} A\sin(u - (T_i + T'))P_l(A\sin u)\,\mathrm{d}u \end{aligned} \tag{42}$$

Using the angle difference identity for sine, the formula becomes

$$\begin{aligned} &\frac{1}{A}\int_{T'}^{T_i+T'} A\sin(u)\cos(T_i + T')P_l(A\sin u)\,\mathrm{d}u - \\ &\frac{1}{A}\int_{T'}^{T_i+T'} A\cos(u)\sin(T_i + T')P_l(A\sin u)\,\mathrm{d}u. \end{aligned} \tag{43}$$

Since $h = A\sin(u)$ and $u = t + T'$, we know that $\frac{\mathrm{d}}{\mathrm{d}u}h = A\cos(u)$ and $\frac{\mathrm{d}}{\mathrm{d}t}u = 1$. Therefore, the formula can be further simplified as

$$\begin{aligned} &\frac{\cos(T_i + T')}{A}\int_{T'}^{T_i+T'} hP_l(h)\,\mathrm{d}u - \frac{\sin(T_i + T')}{A}\int_{T'}^{T_i+T'} (\frac{\mathrm{d}}{\mathrm{d}u}h)P_l(h)\,\mathrm{d}u \\ &= \frac{\cos(T_i + T')}{A}\underbrace{\int_0^{T_i} hP_l(h)\,\mathrm{d}t}_{=:C_l} - \frac{\sin(T_i + T')}{A}\underbrace{\int_0^{T_i} (\frac{\mathrm{d}}{\mathrm{d}t}h)P_l(h)\,\mathrm{d}t}_{=:E_l}. \end{aligned} \tag{44}$$

Using the same technique, the second integral on the RHS of Eq. (20) can be expressed as

$$\begin{aligned} &\int_0^{T_i} \sin(t)P_l(c_x\cos t + c_y\sin t)\,\mathrm{d}t \\ &= \frac{1}{A}\int_{T'}^{T_i+T'} A\sin(u - T')P_l(A\sin u)\,\mathrm{d}u \\ &= \frac{1}{A}\int_{T'}^{T_i+T'} A\sin(u)\cos(T')P_l(A\sin u)\,\mathrm{d}u - \\ &\quad \frac{1}{A}\int_{T'}^{T_i+T'} A\cos(u)\sin(T')P_l(A\sin u)\,\mathrm{d}u \\ &= \frac{\cos(T')}{A}\underbrace{\int_0^{T_i} hP_l(h)\,\mathrm{d}t}_{=:C_l} - \frac{\sin(T')}{A}\underbrace{\int_0^{T_i} (\frac{\mathrm{d}}{\mathrm{d}t}h)P_l(h)\,\mathrm{d}t}_{=:E_l}. \end{aligned} \tag{45}$$