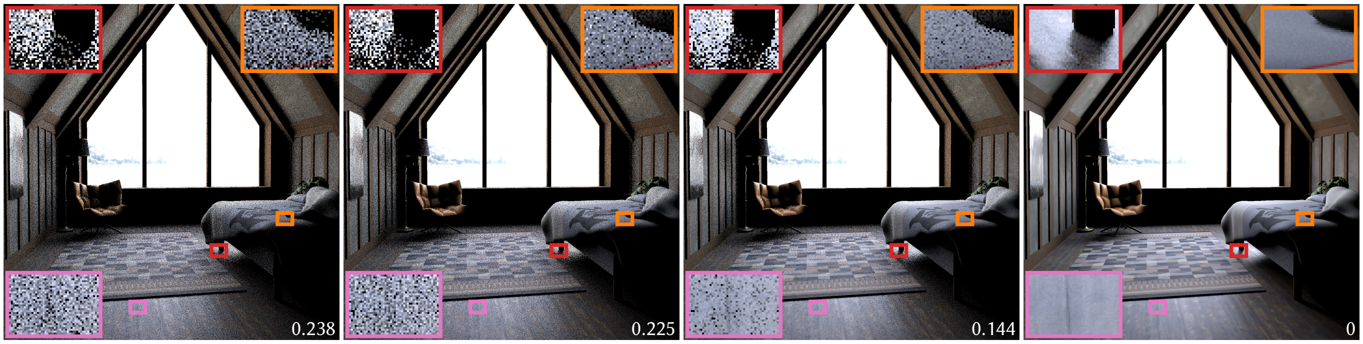
BRDF Importance Sampling for Polygonal Lights

CHRISTOPH PETERS, Karlsruhe Institute of Technology, Germany



Solid angle sampling and BRDF sampling [Dong et al., 2015] with balance heuristic, 0.74 ms | Our projected solid angle sampling and BRDF sampling [Dong et al., 2015] with balance heuristic, 0.91 ms | Our projected solid angle and LTC sampling with clamped optimal MIS, 1.47 ms | Reference

Fig. 1. An attic lit by a light probe, which is masked by a pentagon. Our projected solid angle sampling of this polygon provides better importance sampling for diffuse surfaces than solid angle sampling (orange inset). We also use it to sample the polygon proportional to an LTC [Heitz et al., 2016], thus reducing variance for specular shading. Timings are full frame times at a resolution of 1440^2 with two samples per pixel on an NVIDIA RTX 2080 Ti. Numbers are RMSEs.

With the advent of real-time ray tracing, there is an increasing interest in GPU-friendly importance sampling techniques. We present such methods to sample convex polygonal lights approximately proportional to diffuse and specular BRDFs times the cosine term. For diffuse surfaces, we sample the polygons proportional to projected solid angle. Our algorithm partitions the polygon suitably and employs inverse function sampling for each part. Inversion of the distribution function is challenging. Using algebraic geometry, we develop a special iterative procedure and an initialization scheme. Together, they achieve high accuracy in all possible situations with only two iterations. Our implementation is numerically stable and fast. For specular BRDFs, this method enables us to sample the polygon proportional to a linearly transformed cosine. We combine these diffuse and specular sampling strategies through novel variants of optimal multiple importance sampling. Our techniques render direct lighting from Lambertian polygonal lights with almost no variance outside of penumbrae and support shadows and textured emission. Additionally, we propose an algorithm for solid angle sampling of polygons. It is faster and more stable than existing methods.

CCS Concepts: • **Computing methodologies** → **Ray tracing**; • **Mathematics of computing** → *Nonlinear equations.*

Additional Key Words and Phrases: projected solid angle sampling, solid angle sampling, light sampling, next event estimation, spherical polygons, spherical triangles, polygonal lights, real-time ray tracing, rendering, linearly transformed cosines, LTC, Monte Carlo integration, optimal MIS

Author's address: Christoph Peters, christoph.peters@kit.edu, Karlsruhe Institute of Technology, Am Fasanengarten 5, 76131, Karlsruhe, Germany.

ACM Reference Format:
Christoph Peters. 2021. BRDF Importance Sampling for Polygonal Lights. *ACM Trans. Graph.* 40, 4, Article 140 (August 2021), 14 pages. https://doi.org/10.1145/3450626.3459672

1 INTRODUCTION

As GPUs with dedicated hardware units for ray tracing become widely available, interest in real-time ray tracing grows steadily. However, interactive applications still need to limit their budget to a few rays per pixel. Thus, GPU-friendly importance sampling methods that achieve low variance for particular light transport phenomena are in high demand. In this work, we introduce such methods for shading with convex polygonal area lights.

According to the reflection equation, the radiance reflected by a surface into direction $\omega_o \in \Omega$ is

$$L_o(\omega_o) = \int_\Omega L_i(\omega_i) V(\omega_i) f_r(\omega_i, \omega_o) n^\mathsf{T} \omega_i \,\mathrm{d}\omega_i,$$

where $\Omega \subset \mathbb{R}^3$ is the hemisphere around the surface normal $n \in \mathbb{R}^3$, f_r is the bidirectional reflectance distribution function (BRDF), L_i gives incoming radiance due to the area light, $V(\omega_i) \in \{0, 1\}$ indicates whether the light is visible and $n^\mathsf{T}\omega_i$ denotes a dot product. A Monte Carlo estimator takes a random sample ω_i from Ω proportional to a known density $p(\omega_i)$ and estimates the integral as

$$\frac{L_i(\omega_i) V(\omega_i) f_r(\omega_i, \omega_o) n^\mathsf{T} \omega_i}{p(\omega_i)}.$$

The randomness manifests as noise. Importance sampling reduces the variance by constructing a density p that approximates the integrand well.

With solid angle sampling, the density p is constant within the solid angle of the area light and zero elsewhere. Such methods are available for all common types of area lights [Arvo, 1995, Gamito, 2016, Guillén et al., 2017, Ureña et al., 2013, Wang, 1992]. If the area light is a Lambertian emitter, i.e. L_i is constant, the Monte Carlo estimator is proportional to $V(\omega_i)f_r(\omega_i,\omega_o)n^\mathsf{T}\omega_i$. Visibility V only varies for shading points in penumbrae. Elsewhere, variance is dominated by the BRDF times cosine $f_r(\omega_i,\omega_o)n^\mathsf{T}\omega_i$ and could be cancelled by a strategy sampling the solid angle proportional to these known terms. Efficient strategies for BRDF importance sampling exist [Heitz and d'Eon, 2014], but they produce samples in the whole hemisphere, which often miss the area light. Multiple importance sampling (MIS) [Veach and Guibas, 1995] offers a robust way to combine both approaches but considerable variance remains.

Our method produces samples nearly proportional to BRDF times cosine, but only within the solid angle of the polygonal light. It works for both diffuse and specular shading. If the light is Lambertian, we attain unbiased shading with almost no noise outside of penumbrae.

To this end, we first introduce a method to sample convex polygons proportional to the cosine term $n^\mathsf{T}\omega_i$, i.e. to sample the projected solid angle uniformly (Sec. 3). It is based on inverse function sampling. We propose a concise formulation of the relevant distribution function (Sec. 3.1). To be able to invert it in all cases, we design a special iterative procedure based on geometric constructions (Sec. 3.3). Coupled with a sophisticated initialization strategy, it has cubic convergence order, i.e. the number of correct digits roughly triples with each iteration (Sec. 3.4 and 3.5). We demonstrate that two iterations suffice to get a low error, even in the worst case (Sec. 3.6). Thus, the method as a whole achieves coherent execution and a great performance, making it suitable for real-time rendering on GPUs. We also ensure good numerical stability and address boundary cases without compromising efficiency (Sec. 3.7).

For diffuse BRDFs, which are nearly constant, this strategy achieves low variance on its own. However, most materials mix diffuse and specular components. To address specular components, we employ linearly transformed cosines (LTCs) [Heitz et al., 2016]. LTCs exploit that applying linear transforms to direction vectors distorts distributions on the sphere. With a table of suitably optimized transforms, the cosine distribution $\frac{1}{\pi}n^\mathsf{T}\omega_i$ turns into a good fit of specular BRDFs. Thus, our projected solid angle sampling also enables sampling of the polygon proportional to an LTC approximation of the specular BRDF (Sec. 4.1).

MIS [Veach and Guibas, 1995] is the established method to combine these diffuse and specular sampling strategies. However, standard MIS heuristics introduce variance. We design a weighted balance heuristic that achieves almost no variance in fully lit regions but has robustness issues in penumbrae (Sec. 4.2). To overcome these issues, we derive inexpensive variants of optimal MIS [Kondapaneni et al., 2019] (Sec. 4.3).

For surface shading, projected solid angle sampling is better but also more costly than solid angle sampling of polygons [Arvo, 1995]. Both options remain relevant. Therefore, we also revisit solid angle sampling and find a new formulation that is significantly faster and more stable than Arvo's method [1995] (Sec. 5).

With our techniques, we comfortably satisfy the performance constraints of real-time rendering (Sec. 6.6). At the same time, we achieve low variance outside of penumbrae for all BRDFs that can be approximated reasonably well by LTCs (Sec. 6.2 and 6.3). Unlike the closed-form computation of shading with LTCs [Heitz et al., 2016], our approach is unbiased. It is also more flexible and supports shadows, textured emitters, emission profiles and portals naturally, at the cost of increased variance (Sec. 6.5). Overall, we offer comprehensive solutions for sampling of convex polygonal lights and show the way forward for other types of area lights.

The supplemental includes the complete source code of our real-time renderer and all code needed to reproduce our results.

2 RELATED WORK

In this section, we discuss related work on sampling of area lights, shading with polygonal lights, LTCs and MIS.

Solid angle sampling of area lights is well-understood. Closed-form solutions are available for spheres [Wang, 1992], triangles [Arvo, 1995] and polygons [Arvo, 2001]. A special solution for rectangles improves stratification [Ureña et al., 2013]. Summed-area tables of re-sampled light probes enable sampling of rectangular portals [Bitterli et al., 2015]. Solid angle sampling of cylinders and disks is possible with rejection sampling [Gamito, 2016]. An iterative solution for ellipses avoids rejection sampling [Guillén et al., 2017]. For spherical lights, efficient projected solid angle sampling is available [Peters and Dachsbacher, 2019, Ureña and Georgiev, 2018].

Another branch of work considers integration over spherical polygons. Computation of the solid angle of a triangle requires only a single inverse trigonometric function [van Oosterom and Strackee, 1983]. More general methods integrate polynomials written in the spherical harmonics basis over spherical polygons [Belcour et al., 2018, Wang and Ramamoorthi, 2018]. Special quadrature rules approximate shading for polygonal lights with data-driven emission profiles [Luksch et al., 2020]. Uniform sampling of the area of a polygon is another option for Monte Carlo integration [Turk, 1992].

Closely related to our work, there are two techniques to sample the projected solid angle of triangles or polygons uniformly. Ureña [2000] recursively subdivides a triangle into four smaller triangles. He descends this tree stochastically in proportion to the projected solid angle of each triangle. Once the remaining triangle is small enough, he uses solid angle sampling. This method is reliable but far more expensive than our approach. Our approach is more similar to the one of Arvo [2001], which relies on inverse function sampling. For the inversion, it uses Newton's method with a cubic polynomial for the initialization. Although it resembles our method, stability issues make it difficult to use in practice and the cost is high [Hart et al., 2020]. Sec. 3.8 provides a detailed discussion.

Recently, sampling problems have been studied more fundamentally. Based on four or nine samples of the target function in 2D primary sample space, Hart et al. [2020] construct a bilinear or biquadratic approximation of the sought-after density. By sampling this approximation exactly, they warp samples in primary sample space. The triangle cut parametrization [Heitz, 2020] provides a class of exact alternatives to inverse function sampling. If sampling of a sufficiently good approximate density is possible, an area-preserving construction maps the area under this approximate density onto the area under the actual target density. We tried using this approach for projected solid angle sampling of polygons but were unable to find a family of approximate densities that is suitable for all cases.

LTCs apply precomputed linear transforms to directions in a cosine distribution [Heitz et al., 2016]. The resulting approximations to BRDFs enable closed-form shading with polygonal lights (without shadows). Sec. 4.1 provides more details. LTCs also work for linear lights and disk lights [Heitz and Hill, 2017a,b]. A method similar to LTCs enables closed-form shading and sampling for spherical lights but struggles with anisotropic highlight shapes [Dupuy et al., 2017].

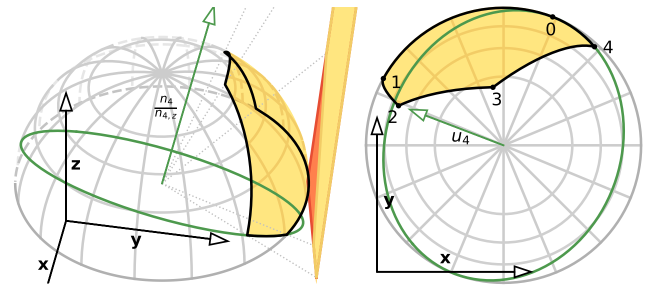
In principle, LTCs enable BRDF importance sampling but that requires projected solid angle sampling for the relevant domains [Heitz et al., 2016]. Li et al. [2018] use inverse function sampling with bisection and Newton's method for LTC importance sampling of edges in their differentiable renderer. Loubet et al. [2020] build upon Arvo's method [2001] to sample small triangles proportional to an LTC in their specular next event estimation. Our method should be immediately beneficial to the speed of this technique.

MIS [Veach and Guibas, 1995] provides a framework for combining multiple sampling techniques with heuristic weighting. For example, it is common to combine solid angle sampling [Arvo, 1995] with a strategy that samples the hemisphere proportional to the BRDF times cosine [Heitz and d'Eon, 2014]. The balance heuristic is provably never much worse than the best heuristic with non-negative weights. Though, heuristics with truly minimal variance may use negative weights and require additional knowledge about the integrand and the sampling techniques [Kondapaneni et al., 2019]. A more practical scheme only requires variance estimates for each technique [Grittmann et al., 2019]. If one of the sampling densities uses a tabulated density, this density can be optimized to minimize variance overall [Karlík et al., 2019]. MIS also generalizes to continuous families of sampling techniques [West et al., 2020].

Ratio estimators [Heitz et al., 2018] compute the quotient of Monte Carlo estimates for shadowed and unshadowed shading and denoise it. Multiplication by an LTC estimate of unshadowed shading gives a biased but consistent estimator. Our method is unbiased but implicitly multiplies by LTC estimates of unshadowed shading through division by the density (Sec. 4.2). We are concerned with a single area light but light hierarchies [Moreau et al., 2019] efficiently select important lights among thousands of candidates. Bitterli et al. [2020] store sampled positions on light sources per pixel. Through reservoir sampling and resampled importance sampling [Talbot et al., 2005], this simple screen space data structure gives rise to real-time importance sampling of huge numbers of area lights. However, the quality depends on spatiotemporal coherence, the overhead to make the method unbiased is substantial and generalizing beyond direct illumination is non-trivial. Our method does not rely on spatiotemporal coherence or learning at all.

3 PROJECTED SOLID ANGLE SAMPLING OF POLYGONS

In this section, we introduce our method to sample the projected solid angle of a convex polygon uniformly. On its own, this method provides excellent importance sampling for diffuse shading and Sec. 4 generalizes it to mixed diffuse and specular BRDFs.



(a) 3D view (b) View of the xy-plane

Fig. 2. We clip a polygon at the horizon and project it onto the unit sphere. Edge j becomes an arc of a great circle with normal vector n_j (green). Its projection to the xy-plane is an ellipse that is characterized by $u_j = \frac{n_{j,\mathrm{xy}}}{n_{j,\mathrm{z}}}$. This example uses $j = 4$, i.e. the edge connects vertex 4 to vertex 0.

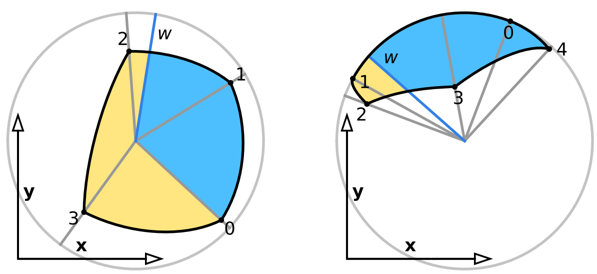
We begin with geometric constructions to compute this projected solid angle within a range of azimuthal angles (Sec. 3.1). In some cases, this distribution function is easy to invert (Sec. 3.2). Otherwise, our iterative procedure (Sec. 3.3) with a sophisticated initialization (Sec. 3.4) converges quickly (Sec. 3.5 and 3.6). Our implementation is designed to work well on GPUs and to be stable in single-precision arithmetic (Sec. 3.7). Since a prior work [Arvo, 2001] takes a similar approach, we discuss differences explicitly (Sec. 3.8).

3.1 Partitioning Polygons into Sectors

As input, our method expects vertices $v_0, \dots, v_{m-1} \in \mathbb{R}^3$ of a convex polygon. In particular, all vertices must be coplanar. Non-convex polygons have to be split into convex polygons. The vertices have to be given in a coordinate frame where the point being shaded is the origin and the shading normal is aligned with the z-axis. LTCs also require such a coordinate-frame and we use the same construction [Heitz et al., 2016]. Additionally, the winding of the vertices as seen from the shading point has to be clockwise. We ensure that by flipping the sign on all y-coordinates if the shading point is on the wrong side of the plane of the polygon. Finally, the polygon must be clipped to the hemisphere $\mathrm{z} \geq 0$. Supplement A.1 in the supplemental document describes our clipping procedure.

We characterize the projected solid angle of this polygon in terms of its edges. For convenience, let $v_m := v_0$ such that edge $j \in \{0, \dots, m-1\}$ connects vertices v_j and v_{j+1}. If we project the polygon onto the unit sphere, edge j turns into an arc of a great circle with normal $n_j := v_j \times v_{j+1}$ (Fig. 2a). According to the Nusselt analog, sampling proportional to the cosine-term is equivalent to sampling the projection of the solid angle to the xy-plane uniformly [Pharr et al., 2016, chapter 13.6.3]. This projection turns the great circle into an ellipse (Fig. 2b).

Let $q \in \mathbb{S}^2$ be a point on this great circle ($\mathbb{S}^2$ is the unit sphere). We use indices x, y, z to extract entries from vectors or matrices. For example, $q_{\mathrm{xy}} \in \mathbb{R}^2$ is the projection of q to the xy-plane. The point



(a) Central case (b) Decentral case

Fig. 3. We draw rays from the origin through the vertices to split the unit circle into sectors. (a) If the surface normal points towards the polygon, each sector contains exactly one edge. (b) Otherwise, one sector is empty and the others contain parts of two edges. For sampling, we seek a sector boundary w such that the blue area is $\xi_0 A_\Sigma$ (Sec. 3.2 and 3.3).

satisfies $\|q\| = 1$ and $n_j^\mathsf{T} q = n_{j,\mathrm{xy}}^\mathsf{T} q_\mathrm{xy} + n_{j,\mathrm{z}} q_\mathrm{z} = 0$, which implies

$$1 = q^\mathsf{T} q = q_\mathrm{xy}^\mathsf{T} q_\mathrm{xy} + q_\mathrm{z} q_\mathrm{z} = q_\mathrm{xy}^\mathsf{T} q_\mathrm{xy} + \left(-\frac{q_\mathrm{xy}^\mathsf{T} n_{j,\mathrm{xy}}}{n_{j,\mathrm{z}}}\right)\left(-\frac{n_{j,\mathrm{xy}}^\mathsf{T} q_\mathrm{xy}}{n_{j,\mathrm{z}}}\right).$$

For a more handy result, let

$$I := \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \in \mathbb{R}^{2\times 2}, \quad u_j := \frac{n_{j,\mathrm{xy}}}{n_{j,\mathrm{z}}} \in \mathbb{R}^2, \quad C_j := I + u_j u_j^\mathsf{T} \in \mathbb{R}^{2\times 2},$$

where $u_j u_j^\mathsf{T}$ is an outer product. Then the characterization of q_xy simplifies to

$$q_\mathrm{xy}^\mathsf{T} C_j q_\mathrm{xy} = 1.$$

This equation describes the ellipse in the xy-plane through the positive definite matrix C_j. We only need to store the vector $u_j \in \mathbb{R}^2$, which is essentially the edge normal n_j, scaled onto the plane $\mathrm{z} = 1$. The formulation above fails if $n_{j,\mathrm{z}} = 0$, i.e. when the great circle passes through the zenith. We disregard this case throughout the paper but Supplement A.2 describes how we handle it with just a few additional instructions.

Of course, what we actually care about is the area enclosed by these ellipses, i.e. the projected solid angle. To separate the different ellipses, we consider rays in the xy-plane from the origin through each vertex of the polygon (Fig. 3). These m rays subdivide the unit disk into m sectors. The structure of the resulting subdivision depends on whether the origin is part of the polygon. If so, each sector contains exactly one edge of the polygon (Fig. 3a). We call this case the central case because the center is part of the polygon. It occurs when the surface normal points towards the polygon.

The other case is the decentral case. In this case, the sector between the counterclockwise and clockwise ends of the polygon is completely empty (Fig. 3b). The other $m-1$ sectors contain parts of exactly two edges of the polygon. To understand why, recall that the polygon is convex. The sector, projected into 3D along the shading normal, defines another convex set. Thus, the part of the polygon within the sector is also convex. And by construction of the sector, vertices only lie at the sector boundaries.

Our sampling procedure first determines the azimuth of the sample and then takes care of the inclination. Consequently, the first step is to select one of the sectors with a probability proportional to its enclosed area. Computing these areas works as follows:

Proposition 1. *We consider the sector running at most 180° counterclockwise from the direction vector* $s_0 \in \mathbb{R}^2$ *to* $s_1 \in \mathbb{R}^2$ *(cf. Fig. 4). Its intersection with the ellipse characterized by* $C_j \in \mathbb{R}^{2\times 2}$ *has area*

$$\frac{1}{2\sqrt{|C_j|}} \operatorname{atan2}\begin{pmatrix} -s_0^\mathsf{T} R s_1 \\ \frac{1}{\sqrt{|C_j|}} s_0^\mathsf{T} C_j s_1 \end{pmatrix}, \tag{1}$$

where $|C_j|$ *denotes the determinant,* $\operatorname{atan2}(\rho\sin\alpha, \rho\cos\alpha)^\mathsf{T} := \alpha$ *for all* $\alpha \in [-\pi, \pi)$, $\rho > 0$ *and* $R := \left(\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right)$ *is the matrix that rotates 90° counterclockwise.*

Proof. Let $LL^\mathsf{T} := C_j$ be a Cholesky decomposition. For all points $q \in \mathbb{R}^2$ on the ellipse, we have

$$\|L^\mathsf{T} q\|^2 = q^\mathsf{T} L L^\mathsf{T} q = q^\mathsf{T} C_j q = 1.$$

Thus, L^T maps the ellipse onto the unit circle. The vectors $L^\mathsf{T} s_0$ and $RL^\mathsf{T} s_0$ have equal length and constitute an orthogonal frame. Therefore, the opening angle of the transformed sector $L^\mathsf{T} s_0, L^\mathsf{T} s_1$ is

$$\operatorname{atan2}\begin{pmatrix} (RL^\mathsf{T} s_0)^\mathsf{T} L^\mathsf{T} s_1 \\ (L^\mathsf{T} s_0)^\mathsf{T} L^\mathsf{T} s_1 \end{pmatrix} = \operatorname{atan2}\begin{pmatrix} -s_0^\mathsf{T} LRL^\mathsf{T} s_1 \\ s_0^\mathsf{T} C_j s_1 \end{pmatrix}.$$

Basic algebra shows $LRL^\mathsf{T} = |L| R$. Dividing both entries by $|L| = \sqrt{|C_j|}$ leaves the arctangent unchanged and gives the angle in Equation (1). Multiplying by 1/2 yields the area inside the unit circle and the determinant of the transformation $|L^{-\mathsf{T}}| = \sqrt{|C_j|}^{-1}$ converts this area back to the ellipse. □

Everything in Equation (1) except for the arctangent is inexpensive to evaluate because

$$|C_j| = 1 + \|u_j\|^2, \qquad C_j s_1 = s_1 + u_j (u_j^\mathsf{T} s_1).$$

In the central case, we compute the area A_j for sector j from ellipse u_j within the sector $s_0 := v_{j,\mathrm{xy}}, s_1 := v_{j+1,\mathrm{xy}}$. In the decentral case, we first have to sort the vertices counterclockwise around the zenith to obtain the sectors. Then, we identify the inner and outer ellipse for each sector, apply Equation (1) to both and subtract the results. Sec. 3.7 describes our GPU-friendly implementation of these steps. The sum of the areas for all sectors is the projected solid angle of the polygon $A_\Sigma \in \mathbb{R}$.

3.2 Sampling in the Central Case

Per sample, our sampling procedure takes exactly two random variables ξ_0, ξ_1 distributed uniformly on $[0, 1)$ as input and maps them continuously. Therefore, it is compatible with stratification. Sampling the azimuth in the central case means determining a direction $w \in \mathbb{R}^2$ such that the parts of the polygon in the sector $v_{0,\mathrm{xy}}, w$ have area $\xi_0 A_\Sigma$ (Fig. 3a).

If we have selected sector j for sampling, the area A in the sector from $s_0 := v_{j,\mathrm{xy}}$ to w must be

$$\frac{1}{2\sqrt{|C_j|}} \operatorname{atan2}\begin{pmatrix} -s_0^\mathsf{T} R w \\ \frac{1}{\sqrt{|C_j|}} s_0^\mathsf{T} C_j w \end{pmatrix} = A := \xi_0 A_\Sigma - \sum_{k=0}^{j-1} A_k \in [0, A_j).$$

For now, the length of w is irrelevant. Then a suitable solution is

$$\begin{pmatrix} -s_0^\mathsf{T} R w \\ \frac{1}{\sqrt{|C_j|}} s_0^\mathsf{T} C_j w \end{pmatrix} = \begin{pmatrix} \sin(2\sqrt{|C_j|}A) \\ \cos(2\sqrt{|C_j|}A) \end{pmatrix}.$$

That is a 2×2 system of linear equations. The solution for w is

$$w = \begin{pmatrix} -s_0^\mathsf{T} R \\ \frac{1}{\sqrt{|C_j|}} s_0^\mathsf{T} C_j \end{pmatrix}^{-1} \begin{pmatrix} \sin(2\sqrt{|C_j|}A) \\ \cos(2\sqrt{|C_j|}A) \end{pmatrix}.$$

To obtain the actual sample, we must scale the direction w by an appropriate factor $\rho \geq 0$. Since the sample must lie within the ellipse,

$$(\rho w)^\mathsf{T} C_j (\rho w) \leq 1 \qquad \Rightarrow \qquad \rho^2 \leq \frac{1}{w^\mathsf{T} C_j w}.$$

Sampling the area in the xy-plane uniformly requires us to sample the squared scaling factor ρ^2 uniformly, not ρ [Pharr et al., 2016, chapter 13.6.2]. Hence, we define

$$\rho = \sqrt{\xi_1 \frac{1}{w^\mathsf{T} C_j w}}.$$

Note that the original length of w cancels in ρw. Finally, we have to project upwards to the hemisphere. The normalized sample direction in the coordinate frame in which the vertices were given is

$$\left(\rho w_\mathrm{x},\ \rho w_\mathrm{y},\ \sqrt{1 - \|\rho w\|^2}\right)^\mathsf{T} \in \Omega. \tag{2}$$

3.3 Iterative Procedure for the Decentral Case

In the decentral case, we have not one but two ellipses to account for. We need to sample the area inside of the outer ellipse but outside of the inner ellipse within a sector. Let the indices of the inner and outer ellipses be $i, o \in \{0, \ldots, m-1\}$ and let the sector to sample begin at $s_0 := v_{j,\mathrm{xy}}$. Sec. 3.7 explains how to find the relevant pairings. We seek a direction $w \in \mathbb{R}^2$ such that the enclosed area in the sector from s_0 to w matches a prescribed area $A \in [0, A_j)$. Thus, we have to solve

$$\frac{1}{2\sqrt{|C_o|}} \operatorname{atan2}\begin{pmatrix} -s_0^\mathsf{T} R w \\ \frac{1}{\sqrt{|C_o|}} s_0^\mathsf{T} C_o w \end{pmatrix} - \frac{1}{2\sqrt{|C_i|}} \operatorname{atan2}\begin{pmatrix} -s_0^\mathsf{T} R w \\ \frac{1}{\sqrt{|C_i|}} s_0^\mathsf{T} C_i w \end{pmatrix} = A.$$

Such a linear combination of two arctangents is not trivial to invert. The core of our method is a fast and accurate iterative solution. We do not believe that a conceivable closed-form solution, e.g. using the triangle-cut parametrization [Heitz, 2020], could be substantially faster or even more accurate in single-precision arithmetic.

Fig. 4 illustrates our strategy for the polygon from Fig. 3b. Our initialization (Sec. 3.4) or the previous iteration provide a current direction $w_n \in \mathbb{R}^2$. We construct an approximation to the target function around this direction. To this end, we consider tangent lines of the two ellipses. Then instead of working with the area enclosed between the ellipses, we consider the area enclosed between these tangent lines.

We begin by constructing the tangent line for ellipse $l \in \{i, o\}$. The ray in direction w_n intersects the ellipse at

$$\frac{1}{\sqrt{w_n^\mathsf{T} C_l w_n}} w_n. \tag{3}$$

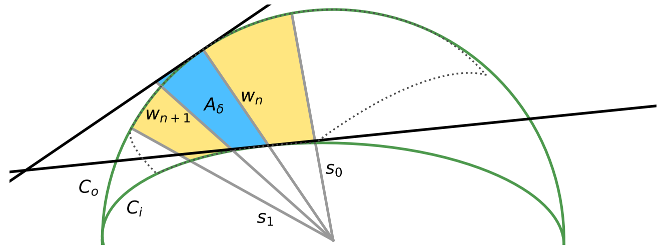


Fig. 4. We sample the area between two ellipses C_i, C_o. The pairs of direction vectors $s_0, s_1 \in \mathbb{R}^2$ and $w_n, w_{n+1} \in \mathbb{R}^2$ define two sectors. In each iteration, we compute the residual error A_δ of the current direction w_n. The next direction w_{n+1} is constructed such that the area between tangent lines to the ellipses at w_n within the sector w_n, w_{n+1} is A_δ.

The implicit function of the ellipse is $q^\mathsf{T} C_l q - 1$ where $q \in \mathbb{R}^2$. Its gradient is $2C_l q$. Therefore, $C_l w_n$ is a normal of the tangent line. By Equation (3), the point $q \in \mathbb{R}^2$ is located on the tangent line if

$$(C_l w_n)^\mathsf{T} q = (C_l w_n)^\mathsf{T} \frac{1}{\sqrt{w_n^\mathsf{T} C_l w_n}} w_n = \sqrt{w_n^\mathsf{T} C_l w_n}.$$

The intersection of the ray in the sought-after direction w with this tangent line is at

$$\frac{\sqrt{w_n^\mathsf{T} C_l w_n}}{w_n^\mathsf{T} C_l w} w. \tag{4}$$

Now we need to determine the area in the sector from w_n to w between these tangent lines. For each tangent line, we form a triangle from the origin and the two intersection points in Equations (3) and (4). Its signed area is given by the 2×2-determinant

$$\frac{1}{2} \left| \left(\frac{1}{\sqrt{w_n^\mathsf{T} C_l w_n}} w_n,\ \frac{\sqrt{w_n^\mathsf{T} C_l w_n}}{w_n^\mathsf{T} C_l w} w \right) \right| = \frac{1}{2} \frac{|(w_n,\ w)|}{w_n^\mathsf{T} C_l w} = \frac{1}{2} \frac{w^\mathsf{T} R w_n}{w_n^\mathsf{T} C_l w}.$$

Note how we rewrite the determinant using the 90°-rotation R.

With that, we are prepared to solve for a direction $w \in \mathbb{R}^2$ that attains a prescribed area A_δ between the tangent lines (Fig. 4):

$$\begin{aligned} &\frac{1}{2} \frac{w^\mathsf{T} R w_n}{w_n^\mathsf{T} C_o w} - \frac{1}{2} \frac{w^\mathsf{T} R w_n}{w_n^\mathsf{T} C_i w} = A_\delta \\ \Leftrightarrow\ & w^\mathsf{T} R w_n (w_n^\mathsf{T} C_i w - w_n^\mathsf{T} C_o w) = 2 A_\delta w_n^\mathsf{T} C_i w w_n^\mathsf{T} C_o w \\ \Leftrightarrow\ & w^\mathsf{T} \underbrace{(R w_n (w_n^\mathsf{T} C_i - w_n^\mathsf{T} C_o) - 2 A_\delta C_i w_n w_n^\mathsf{T} C_o)}_{=:T} w = 0 \end{aligned} \tag{5}$$

Written in terms of $T \in \mathbb{R}^{2\times 2}$, we have to solve $w^\mathsf{T} T w = 0$. It is a quadratic equation, written in homogeneous coordinates. We could fix $w_\mathrm{y} = 1$ to turn it into

$$T_{\mathrm{x,x}} w_\mathrm{x}^2 + (T_{\mathrm{x,y}} + T_{\mathrm{y,x}}) w_\mathrm{x} + T_{\mathrm{y,y}} = 0$$

and apply the quadratic formula. However, Blinn [2006] provides a more direct and stable solution. Only one of the two possible roots of this quadratic is relevant. On the basis that it depends on T continuously, Supplement A.3 specializes Blinn's solver to compute only this one. In the end, the output of our iterative procedure is

$$w_{n+1} := \begin{cases} \left(\frac{|T_{\mathrm{x,y}}+T_{\mathrm{y,x}}|}{2} + \sqrt{\Delta},\ -T_{\mathrm{x,x}}\right)^{\mathsf{T}} & \text{if } T_{\mathrm{x,y}} + T_{\mathrm{y,x}} \geq 0, \\ \left(T_{\mathrm{y,y}},\ \frac{|T_{\mathrm{x,y}}+T_{\mathrm{y,x}}|}{2} + \sqrt{\Delta}\right)^{\mathsf{T}} & \text{otherwise,} \end{cases} \tag{6}$$

where $\Delta := -\frac{1}{4}|T + T^{\mathsf{T}}|$ is the discriminant. This vector solves the quadratic equation $w_{n+1}^{\mathsf{T}} T w_{n+1} = 0$.

Summary of the Algorithm. Each iteration starts from a direction $w_n \in \mathbb{R}^2$. The residual error is

$$A_\delta := A_\delta(w_n) := A - \tag{7}$$

$$\frac{1}{2\sqrt{|C_o|}} \operatorname{atan2}\left(\frac{-s_0^{\mathsf{T}} R w_n}{\frac{1}{\sqrt{|C_o|}} s_0^{\mathsf{T}} C_o w_n}\right) + \frac{1}{2\sqrt{|C_i|}} \operatorname{atan2}\left(\frac{-s_0^{\mathsf{T}} R w_n}{\frac{1}{\sqrt{|C_i|}} s_0^{\mathsf{T}} C_i w_n}\right).$$

We use it to construct the homogeneous quadratic T according to Equation (5). Then we solve it using Equation (6). The sign on the resulting direction w_{n+1} may be wrong, so we check the dot product with the half-vector of the sector. If it is negative, we flip the sign.

Once all iterations are completed, we compute the scaling factor

$$\rho = \sqrt{(1 - \xi_1)\frac{1}{w_{n+1}^{\mathsf{T}} C_i w_{n+1}} + \xi_1 \frac{1}{w_{n+1}^{\mathsf{T}} C_o w_{n+1}}}.$$

The normalized sample direction arises by projecting ρw_{n+1} onto the upper hemisphere as in Equation (2).

3.4 Initialization for the Decentral Case

Locally, the tangent approximation in the iteration works well but when the direction w_n is far from the correct result, it may fail. We have to provide an initialization $w_0 \in \mathbb{R}^2$ that is always close enough to ensure fast convergence. We tried many possible solutions. The one presented here is the only one that withstood the rigorous search for failure cases in Sec. 3.6.

Fig. 5a illustrates our strategy: We approximate the relevant area by two quads and sample those uniformly. As before, i, o are the indices of the inner and outer ellipse. The sector ranges from a vertex at s_0 to a vertex at s_1. As a first step, we split the sector in half by defining the half-vector

$$s_h := \frac{s_0}{\|s_0\|} + \frac{s_1}{\|s_1\|}. \tag{8}$$

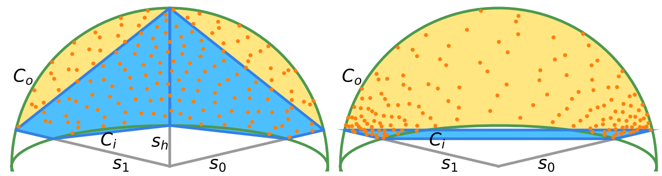
According to Equation (3), the ray through s_j, where $j \in \{0, h, 1\}$, intersects ellipse $l \in \{i, o\}$ at $\lambda_{l,j} s_j$, where

$$\lambda_{l,j} := \frac{1}{\sqrt{s_j^{\mathsf{T}} C_l s_j}}.$$

We get two quads with vertices $\lambda_{i,k}s_k, \lambda_{i,h}s_h, \lambda_{o,h}s_h, \lambda_{o,k}s_k$ for $k \in \{0, 1\}$ (Fig. 5a). After computing their areas through 2×2-determinants, we select one of them to sample it uniformly. It is bounded by inner and outer edges. If the point $q \in \mathbb{R}^2$ lies on the edge $l \in \{i, o\}$, it satisfies the line equation $r_l^{\mathsf{T}} q = D_l$, where

$$r_l := C_l(\lambda_{l,h}s_h + \lambda_{l,k}s_k) \in \mathbb{R}^2, \qquad D_l := \lambda_{l,h} r_l^{\mathsf{T}} s_h \in \mathbb{R},$$

as derived in Supplement A.4. This formula for the edge normal r_l is a generalization of the half-vector formula in Equation (8) and inherits excellent numerical stability for small sectors.



(a) Two quads (b) Single quad

Fig. 5. (a) We form two quads from intersections of the inner and outer ellipses C_i, C_o with the sector boundaries s_0, s_1 and their half vector s_h. Our initialization samples them uniformly. The shown samples are scaled to cover the area between the ellipses and are still nearly uniform. (b) An approximation with a single quad is inadequate for large sectors.

Now we seek a direction $w_0 \in \mathbb{R}^2$ such that the quad between these edges in the sector s_k, w_0 has an area $A_q \in \mathbb{R}$. Supplement A.4 proves that w_0 satisfies the quadratic equation $w_0^{\mathsf{T}} Q w_0 = 0$, where

$$Q := \lambda_{o,k} D_o R s_k r_i^{\mathsf{T}} - (\lambda_{i,k} D_i R s_k + 2A_q r_i) r_o^{\mathsf{T}} \in \mathbb{R}^{2\times 2}.$$

Furthermore, the solver in Equation (6) computes the correct root.

An obvious question is why we split the sector in half. Working with a single quad would be less costly. However, if the shading point gets close to the polygon, the projected solid angle within a sector may almost fill half the unit disk. A single quad approximates this situation poorly (Fig. 5b). Two quads fare relatively well. The overhead of selecting the correct quad is low and pays off in terms of robustness.

3.5 Theoretical Error Analysis

Our construction with tangent lines is reminiscent of Newton's method. However, our method actually achieves higher convergence order. When Newton's method is applied to sampling problems, the derivative is the sampled density. According to Supplement A.5, this density is proportional to

$$\frac{1}{w_n^{\mathsf{T}} C_o w_n} - \frac{1}{w_n^{\mathsf{T}} C_i w_n}. \tag{9}$$

It accounts for the distance at which the ray through w_n intersects the ellipses but ignores the angles of intersection. By incorporating this easily available information, our method attains local cubic convergence, as proven in Supplement A.5.

Local cubic convergence is incredibly fast. Once the error is small enough, the number of zero digits in the residual error approximately triples with each iteration. An error of 0.1 may turn into an error of 10^{-9} in only two iterations. Our analysis in the next section shows that our initialization is good enough to benefit from this cubic convergence in practice.

There is a corner case in which our guarantee of local cubic convergence does not hold. It occurs when the two ellipses meet exactly at the point to be sampled. We encounter this configuration at the clockwise and counterclockwise ends of the polygon. A still more challenging variant of this case occurs when the ellipses meet at the horizon. In that case, the tangent lines become parallel and coefficients of the homogeneous quadratic T become all zero.

In practice, we still observe good convergence near these cases. Though, we encounter rare numerical difficulties if we get too close. Thankfully, we do not need the iteration then. The direction to be sampled is close to one of the two sector boundaries s_0, s_1. The initialization samples these corners with high accuracy. Thus, we simply disable the iteration if the quotient of the target area A and the sector area is less than 10^{-5} or greater than $1-10^{-5}$.

3.6 Empirical Error Analysis

To verify that our initialization is good enough for cubic convergence, we set up a numerical experiment. We use 80-bit floats because our concern is theoretical convergence, not numerical stability. The objective is to find the worst possible failure case after two iterations. We treat this as optimization problem where the error must be maximized. A test case is characterized by inner and outer ellipses $u_i, u_o \in \mathbb{R}^2$, sector boundaries s_0, s_1 and a target area A. Exploiting rotational symmetry, this search space is six-dimensional.

In each step, we generate a random test case. Then we run 100 iterations of the Nelder-Mead-optimizer [1965] to maximize the error. The error that we consider is the area A_δ as defined in Equation (7), divided by the projected solid angle enclosed by the ellipses and the sector. This is a backward error that tells us what perturbation in the random number ξ_0 explains the error in the result. Since the objective is non-convex and high-frequent, we try $3 \cdot 10^{10}$ random initializations.

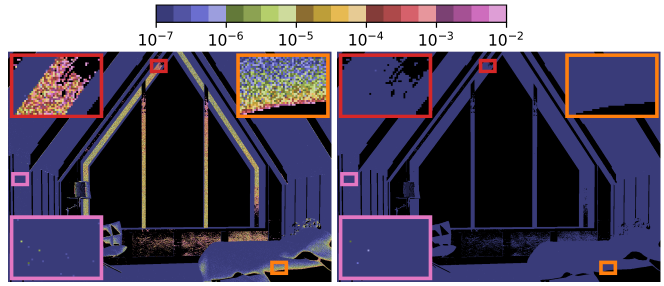
In spite of the 80-bit arithmetic, rounding error is still a concern. To compensate, we limit sectors to 179.999° and discard test cases where the area within the sector is less than 10^{-2}. With this configuration, we find a worst case with a backward error of $1.8 \cdot 10^{-5}$. When we allow minimal areas of 10^{-7}, the maximal backward error increases to $6.6 \cdot 10^{-5}$. This result is likely contaminated by rounding error but either way, it is still an acceptable error.

The average case is far better. Even the 99th percentile of the backward error among 10^8 random test cases is $4.6 \cdot 10^{-15}$. If we use the initialization only, i.e. we set the iteration count to zero, the 99th percentile increases to 0.02 and the worst-case error is 0.27. Clearly, our initialization is good enough to benefit from local cubic convergence in practice. On this basis, we consider our method with two iterations to be unbiased. Renderers using double precision arithmetic may want to run a third iteration but it is hard to conceive that it would make a visible difference.

In practice, rounding errors in single precision arithmetic are far more influential than these theoretical errors. Fig. 6 visualizes them in our renderer. The backward error sometimes exceeds 10^{-3} when the projected solid angle becomes small (i.e. in dark regions). However, the residual error A_δ after two iterations is below 10^{-7} almost everywhere.

3.7 Speed and Stability

As we implement our method on GPUs, we face several challenges. The first problem is that we must not access arrays of local variables with dynamically computed indices because that incurs costly register spilling. Thus, an upper bound for the vertex count m is provided at compile-time and most loops are unrolled. On the other hand, there are loops that do not cause spilling, e.g. in our iteration.



(a) Backward error (b) Backward error times A_Σ

Fig. 6. Errors in single-precision arithmetic for diffuse samples in Fig. 1. The backward error grows moderately large as the polygon vanishes below the horizon. On black pixels, it has vanished entirely. If we multiply by the projected solid angle A_Σ, errors are below 10^{-7} almost everywhere.

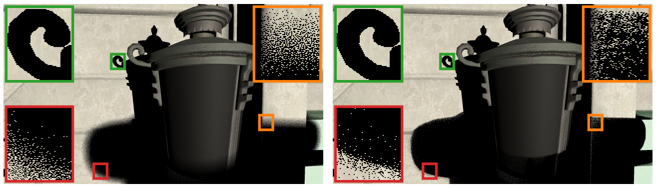
Since we also observed instruction cache misses, we avoid unrolling wherever possible to keep the binary small.

For the decentral case, we have to pair inner and outer ellipses with sector boundaries. Note that outer edges run clockwise, whereas inner edges run counterclockwise (Fig. 3b). The right-hand rule indicates that the z-coordinate of the edge normal $n_{j,z}$ is positive for outer ellipses and negative for inner ellipses. For the ellipse vector u_j, the sign is irrelevant and we repurpose the sign-bit of $u_{j,x}$ to flag inner ellipses. For all $j \in \{0, \dots, m-1\}$, we associate vertex v_j with the next ellipse in counterclockwise direction. It is ellipse u_j if that is an outer ellipse or ellipse u_{j-1} otherwise (u_{m-1} for $j=0$). The vertex at the clockwise end of the polygon is adjacent to an inner and an outer ellipse, so we store the inner ellipse separately. If there is no such ellipse, the central case is present.

Now we sort the vertices counterclockwise and apply the same permutation to the associated ellipses. Once we are done, it is easy to iterate over pairings of ellipses and sectors in counterclockwise order. The first inner ellipse has been stored separately and the first outer ellipse is at the start of the list. In each step, we read the next ellipse and overwrite the current inner or outer ellipse by it dependent on its flag. The corresponding sector boundaries are pairs of consecutive vertices in the sorted list.

Sorting on GPUs is commonly done using sorting networks to avoid register spilling. Since the polygon is convex and provided with sorted vertices, the permutation of our sequence is known to be bitonic. We design optimal sorting networks for bitonic sequences with up to eight vertices (see Supplement A.6). Vertex comparison utilizes the z-coordinate of the edge normal as explained above.

With these discrete problems out of the way, we turn our attention to numerical stability. Our algorithm mostly works in 2D projective space. It constructs vectors in $\mathbb{R}^2$ with no concern for their length. This design is key to its efficiency and stability. We avoid numerical under- or overflow with a fast, approximate L^1-normalization. For short edges, the cross products in the edge normals n_j are prone to cancellation. We avoid them for each entry through Kahan's algorithm, which cleverly exploits fused multiply-add instructions and guarantees a correct sign for $n_{j,z}$ [Jeannerod et al., 2013] (unless there is underflow). For consistency, comparisons during sorting work the same way.



(a) 1 cm square light (b) 1 mm square light

Fig. 7. A wall lit by a small light source that is 40 meters away. The nearby lantern casts a hard shadow, while a bollard near the light source casts a softer shadow. For a 1 cm² light source, our technique handles this situation faithfully except for a few mild outliers (red inset). Except for area sampling [Turk, 1992], all other techniques fail in this case. With a 1 mm² light source, more samples miss the light and shadow rays miss the occluder.

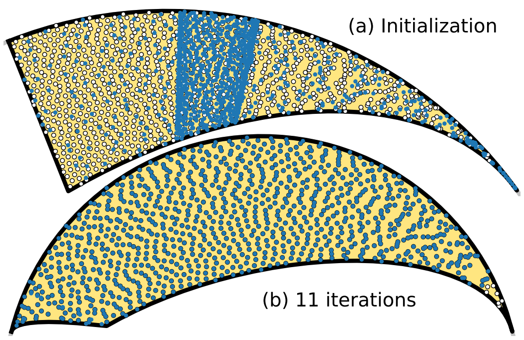


Fig. 8. Two failure cases of Arvo's method [2001]. We show the xy-plane, blue points are Arvo's samples and white circles with black outline are the ground truth. (a) When the cubic interpolation polynomial is not monotonic, the initialization is heavily non-uniform. (b) Seven samples that should converge to the right corner enter a divergent cycle instead.

Through these adjustments and a few more in Supplement A.7, our method is exceptionally resilient to rounding errors. Even with small, distant polygons, it still works (Fig. 7a). All other techniques in our renderer, except area sampling [Turk, 1992], already fail for this configuration. Though, when pushed too hard, our method with single-precision arithmetic also fails (Fig. 7b). These artifacts could be hidden by clamping outputs to the sector. As an unbiased remedy, we recommend sampling the area of the polygon uniformly for such extreme situations.

Supplement A.7 provides pseudocode for the full algorithm and the supplemental code contains reference implementations.

3.8 Relation to Arvo's Method

Arvo's method for projected solid angle sampling of polygons [Arvo, 2001] has many commonalities with ours. To produce a sample, it first samples the azimuth and then the inclination. By virtue of this design decision, it constructs the same distribution function as our method. However, the resulting formulas barely resemble ours. Arvo employs angles and trigonometric identities where we rely on directions and algebraic geometry.

Like our method, Arvo's method then produces an approximate initialization and refines it iteratively. The initialization uses a cubic interpolation polynomial constructed from four equidistant samples of the target function. As shown in Fig. 8a, this approach has serious failure cases. The derivative of the inverse target function may go to infinity and polynomials cannot reproduce that.

The iteration uses Newton's method, which offers local quadratic convergence (not cubic) as long as the derivative is non-zero. The derivative is the density given in Equation (9). Unfortunately, this density always becomes zero at both ends of the polygon. In some cases, the method does not converge at all (Fig. 8b).

Arvo discusses details of the implementation rather superficially, making it difficult to implement the method well. Our GPU implementation reuses the methods from Sec. 3.7 to pair edges and sectors. Other than that, we implement formulas from Arvo's writing verbatim. Certainly, some adjustments could make the method more stable but eventually this path leads back to our method.

As is, our implementation of Arvo's method has serious stability issues. Hart et al. [2020] write about similar problems as they describe their implementation: "we used six iterations of bisection, which we found to be faster and more stable than both Newton-Raphson and Arvo's cubic approximation." With six iterations of bisection, we expect backward errors around 10^{-2}, which is similar to the error of our method without iterations.

Sec. 6.6 shows that Arvo's method is considerably slower than ours, even when the iteration count is fixed to three. Overall, Arvo's method is a highly relevant related work but it is difficult to use and inferior to our method in almost every regard. Its only advantage is direct support for non-convex polygons, where a sector may contain more than two ellipses. However, the mapping from random numbers to samples is discontinuous in this case. Subdividing a non-convex polygon into convex polygons also introduces discontinuities but results in a more GPU-friendly implementation.

4 IMPORTANCE SAMPLING FOR MIXED BRDFS

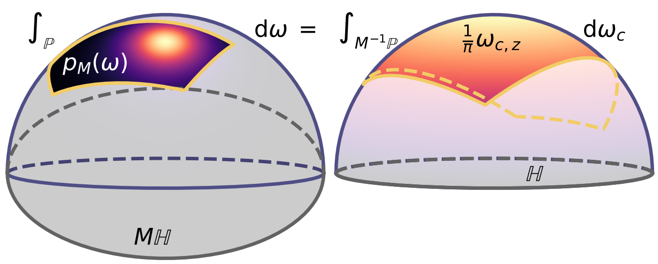
When we use our projected solid angle sampling on a surface with a Lambertian diffuse BRDF for shading with a Lambertian emitter, the Monte Carlo estimator is proportional to $V(\omega_i)$. In general, it provides low variance for diffuse BRDFs but not for specular BRDFs. We address this shortcoming by combining our technique with LTCs [Heitz et al., 2016] (Sec. 4.1). For mixed diffuse and specular BRDFs, we introduce a special MIS heuristic, which is optimal when the light is not occluded (Sec. 4.2). For the general case, we blend this heuristic with the balance heuristic (Sec. 4.3).

4.1 Sampling Linearly Transformed Cosines

Given a linear transform $M \in \mathbb{R}^{3\times3}$ with $|M| > 0$, the corresponding LTC at $\omega \in \Omega$ is the density

$$p_M(\omega) := \frac{1}{\pi} \max\left(0, \frac{(M^{-1}\omega)_z}{\|M^{-1}\omega\|}\right) \frac{|M^{-1}|}{\|M^{-1}\omega\|^3}.$$

Let $\mathbb{P} \subset \Omega$ be the solid angle of the polygon formed by the vertices $v_0, \ldots, v_{m-1}$. Likewise, let $M^{-1}\mathbb{P}$ be the solid angle of the polygon



(a) LTC integral in world space (b) Clamped cosine integral

Fig. 9. Integrating an LTC p_M over a solid angle $\mathbb{P}$ is equivalent to integrating a clamped cosine $\frac{1}{\pi}\max(0, \omega_{c,z})$ over the transformed solid angle $M^{-1}\mathbb{P}$.

with vertices $M^{-1}v_0, \ldots, M^{-1}v_{m-1}$. The LTC is constructed to ensure [Heitz et al., 2016]

$$\int_{\mathbb{P}} p_M(\omega)\,\mathrm{d}\omega = \int_{M^{-1}\mathbb{P}} \frac{1}{\pi}\max(0, \omega_{c,z})\,\mathrm{d}\omega_c. \tag{10}$$

Thus, the rather complicated integral over the LTC p_M reduces to computation of the projected solid angle for the transformed polygon (Fig. 9).

The key observation about LTCs is that they provide good fits to widely used specular BRDFs [Heitz et al., 2016]. A table of linear transforms is precomputed so that each LTC fits the BRDF times cosine $f_r(\omega_i, \omega_o)n^\mathsf{T}\omega_i$ as function of $\omega_i \in \Omega$. Our renderer uses a $64 \times 64 \times 51$ table with transforms for different isotropic roughness values, inclinations of ω_o and Fresnel F_0 parameters.

Once LTCs are available, it is straight forward to turn our projected solid angle sampling into importance sampling of LTCs [Heitz et al., 2016]. We clip the transformed polygon with vertices $M^{-1}v_0, \ldots, M^{-1}v_{m-1}$ to the half-space $z \geq 0$. If the clipped polygon is empty, we disable the specular sampling technique. Otherwise, we apply our projected solid angle sampling, transform the sampled directions using M and normalize. The transformed samples all fall into the polygon $\mathbb{P}$ and sample it proportional to the LTC p_M. To normalize the density, we divide by the projected solid angle in Equation (10), which we compute during sampling anyway.

Note that the transform M changes the horizon (Fig. 9a). In general, the LTC p_M is zero in parts of the upper hemisphere and non-zero in parts of the lower hemisphere. We can avoid samples below the horizon by clipping the polygon against the horizons in both spaces. However, that potentially introduces two additional vertices, which incur a non-negligible cost. Samples below the horizon are fairly rare, even for specular highlights at grazing angles, so our implementation only clips against the horizon after transforming vertices with M^{-1}.

4.2 Weighted Balance Heuristic

Now that we have suitable sampling techniques for diffuse and specular BRDFs, we direct our attention to mixed BRDFs combining both. In this section, our goal is a Monte Carlo estimator with zero variance under idealizing assumptions. Namely, we assume that the BRDF mixes a Lambertian diffuse BRDF and a specular BRDF that is approximated by an LTC perfectly. Besides, we assume a Lambertian emitter with emitted radiance L_e and no occlusion.

We consider a fixed outgoing light direction $\omega_o \in \Omega$. Let a_0, a_1 be the albedos of the diffuse and specular components. The specular albedo a_1 is tabulated alongside the LTC transform M_1. The Lambertian diffuse BRDF corresponds to an LTC p_{M_0} where $M_0 := I$ is the identity matrix. All of our derivations generalize to more than $N := 2$ components but we do not explore this possibility.

In our idealized setting, the BRDF times cosine is simply

$$f_r(\omega_o, \omega_i)n^\mathsf{T}\omega_i = \sum_{j=0}^{N-1} a_j p_{M_j}(\omega_i). \tag{11}$$

With projected solid angle sampling and LTC importance sampling, the normalized sampling densities for $\omega_i \in \mathbb{P}$ are

$$p_j(\omega_i) := \frac{p_{M_j}(\omega_i)}{\int_{\mathbb{P}} p_{M_j}(\omega)\,\mathrm{d}\omega}, \tag{12}$$

where $j \in \{0, \ldots, N-1\}$. If we combine these techniques using standard MIS heuristics, we get non-zero variance (Fig. 10b, pink inset).

This variance is entirely avoidable. By combining Equations (11) and (12), we write the integrand as linear combination of our sampling densities:

$$L_e f_r(\omega_o, \omega_i)n^\mathsf{T}\omega_i = \sum_{j=0}^{N-1} c_j p_j(\omega_i) \text{ where } c_j := L_e a_j \int_{\mathbb{P}} p_{M_j}(\omega)\,\mathrm{d}\omega.$$

Intuitively, the coefficient c_j is simply the unshadowed shading for the polygonal light provided by classic LTCs [Heitz et al., 2016]. Note that c_j must be computed per color channel.

Our weighted balance heuristic uses the MIS weights

$$w_j^w(\omega_i) := \frac{c_j p_j(\omega_i)}{\sum_{k=0}^{N-1} c_k p_k(\omega_i)}.$$

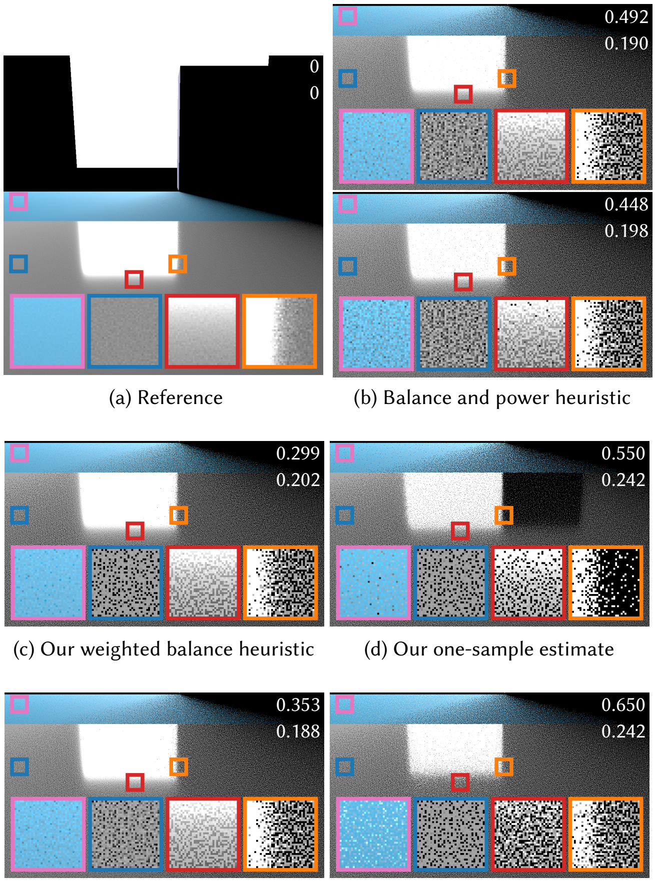
With these MIS weights, the MIS estimator becomes

$$\sum_{j=0}^{N-1} L_e f_r(\omega_o, \omega_i)n^\mathsf{T}\omega_i \frac{w_j^w(\omega_i)}{p_j(\omega_i)} = \sum_{j=0}^{N-1} c_j.$$

Independent of the random numbers, it gives us the unshadowed shading. Thus, it achieves zero variance in the idealized setting.

Of course, we could have computed this result without sampling. The point is that a strategy that gives zero variance for the idealized setting still gives low variance when the BRDF deviates from its LTC approximation. By using MIS, we get unbiased shading for arbitrary BRDFs and full support for shadows or textured emission. Note that only p_0 is guaranteed to be non-zero in all parts of the polygon above the horizon. Thus, we have to make sure that the diffuse albedo a_0 is non-zero in each color channel. Clamping to a minimum of 0.01 is sufficient to avoid fireflies or increased variance.

Our derivation assumes one sample per technique but generalizing to different sample counts is trivial. Our implementation also supports a one-sample estimator [Veach and Guibas, 1995]. It randomly chooses one technique proportional to its weight c_j (reduced to luminance). It works well in the idealized setting but allocates too few samples for the diffuse technique when the specular lobe is shadowed (Fig. 10d). Therefore, we do not recommend its use.



(a) Reference (b) Balance and power heuristic

(c) Our weighted balance heuristic (d) Our one-sample estimate

(e) Our clamped optimal MIS, $v = 1/2$ (f) BRDF sampling, balance heuristic

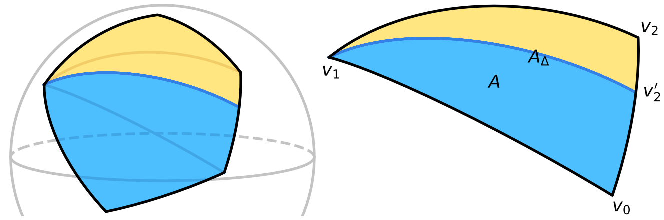
Fig. 10. A blue diffuse plane and a white glossy plane, both lit by a rectangular Lambertian emitter that is partially occluded by a wall. Standard MIS heuristics give noise in fully lit regions (pink inset). Clamped optimal MIS is our most robust heuristic. We report RMSEs of HDR frames without (top) and with (bottom) clamping of overexposed pixels.

4.3 Clamped Optimal Multiple Importance Sampling

By design, our weighted balance heuristic is optimal for unoccluded surfaces. Fig. 10 shows a large penumbra to illustrate a drawback of this design. Outside of the specular highlight, the specular shading estimate c_1 is close to zero. Thus, specular samples make almost no contribution to the diffuse penumbra (Fig. 10c, blue inset). In this regard, the balance heuristic is better because it weights diffuse and specular samples more similarly (Fig. 10b). Our weighted balance heuristic performs well in the specular highlight (hence the lower RMSE without clamping) but poorly in the diffuse shadows (hence the higher RMSE with clamping).

To overcome this issue, we recommend a simple blend between our weighted balance heuristic and the standard balance heuristic. The MIS weights are

$$w_j^c(\omega_j) := v w_j^w(\omega_j) + (1-v)\frac{p_j(\omega_j)}{\sum_{k=0}^{N-1} p_k(\omega_j)},$$



(a) Triangle fan (b) Triangles v_0, v_1, v_2 and v_0, v_1, v_2'

Fig. 11. (a) To sample the solid angle of a convex polygon, we treat it as triangle fan. (b) We seek a new vertex v_2' such that the triangle v_0, v_1, v_2' has solid angle $A = \xi_0 A_\Delta$.

where $v \in [0, 1]$ is a user-defined parameter. In our experiments, we use $v = 1/2$, which gives a good tradeoff between the strengths of both heuristics (Fig. 10e). This approach is less ad hoc than it may seem. Supplement B demonstrates connections to optimal MIS. In this interpretation, v is an estimate of light visibility. We refer to this heuristic as our clamped optimal MIS because it arises from our variant of optimal MIS [Kondapaneni et al., 2019] by clamping negative MIS weights.

5 SOLID ANGLE SAMPLING OF POLYGONS

Our projected solid angle sampling offers excellent importance sampling for surface shading but is more costly than solid angle sampling [Arvo, 1995, 2001] and less useful in volumes. Thus, solid angle sampling remains interesting. In this section, we briefly revisit this problem and find an algorithm that is more stable and faster than Arvo's method. Both methods map random numbers $\xi_0, \xi_1 \in [0, 1)$ to directions in exactly the same manner.

Our method partitions a convex polygon into a triangle fan (Fig. 11a). Triangles are selected in proportion to their solid angle. Then the core problem is to sample a single triangle. Let $v_0, v_1, v_2 \in \mathbb{S}^2$ be normalized direction vectors from the shading point to the vertices of the triangle ($\mathbb{S}^2$ denotes the unit sphere). We compute its solid angle with a single arctangent using van Oosterom and Strackee's formula [1983]:

$$A_\Delta := 2\,\mathrm{atan2}\left(\frac{|(v_0, v_1, v_2)|}{1 + v_0^\mathsf{T} v_1 + v_0^\mathsf{T} v_2 + v_1^\mathsf{T} v_2}\right). \tag{13}$$

To sample the triangle proportional to solid angle, we construct a new vertex $v_2' \in \mathbb{S}^2$ on the edge connecting v_0, v_2 such that the smaller triangle v_0, v_1, v_2' has solid angle $A := \xi_0 A_\Delta$ (Fig. 11b). To satisfy van Oosterom's formula for this subtriangle, we rotate the input to atan2 by 90° and obtain two vectors that must be orthogonal:

$$\begin{pmatrix} \cos\frac{A}{2} \\ \sin\frac{A}{2} \end{pmatrix}^\mathsf{T} \begin{pmatrix} -|(v_0, v_1, v_2')| \\ 1 + v_0^\mathsf{T} v_1 + v_0^\mathsf{T} v_2' + v_1^\mathsf{T} v_2' \end{pmatrix} = 0.$$

We rewrite the determinant as triple product $(v_0 \times v_1)^\mathsf{T} v_2'$ and separate terms with and without v_2', which gives

$$\underbrace{\left(\cos\frac{A}{2}(v_0 \times v_1) - \sin\frac{A}{2}(v_0 + v_1)\right)^\mathsf{T}}_{=:u} v_2' = \underbrace{\sin\frac{A}{2}(1 + v_0^\mathsf{T} v_1)}_{=:D}.$$

Since v_2' is on the edge connecting v_0, v_2, we get the system

$$u^\mathsf{T} v_2' = D, \qquad (v_0 \times v_2)^\mathsf{T} v_2' = 0, \qquad \|v_2'\| = 1. \tag{14}$$

The two linear equations constrain the vertex v_2' to a line, which must be orthogonal to u and $v_0 \times v_2$. The line direction is

$$r := u \times (v_0 \times v_2) \in \mathbb{R}^3.$$

We seek its intersections with the unit sphere. The point $v_2' = -v_0$ satisfies Equation (14) but gives a degenerate triangle. The other solution is offset by a multiple of the line direction r, namely

$$v_2' = -v_0 + 2\frac{v_0^\mathsf{T} r}{\|r\|^2} r.$$

Indeed, this point on the line satisfies $\|v_2'\| = 1$ because

$$\left\| -v_0 + 2\frac{v_0^\mathsf{T} r}{\|r\|^2} r \right\|^2 = \|v_0\|^2 - 4\frac{v_0^\mathsf{T} r}{\|r\|^2} v_0^\mathsf{T} r + 4\frac{(v_0^\mathsf{T} r)^2}{\|r\|^4}\|r\|^2 = 1.$$

To evaluate r, we have a shortcut. We store a few intermediate results from Equation (13), namely

$$G_0 := |(v_0, v_1, v_2)|, \qquad G_1 := v_0^\mathsf{T} v_2 + v_1^\mathsf{T} v_2, \qquad G_2 := 1 + v_0^\mathsf{T} v_1.$$

Supplement C proves

$$r = \left(G_0 \cos\frac{A}{2} - G_1 \sin\frac{A}{2}\right) v_0 + G_2 \sin\frac{A}{2} v_2.$$

With that, we have constructed the subtriangle v_0, v_1, v_2' with solid angle A at negligible cost. We skip computation of u, D and get r right away. It remains to sample the edge connecting v_1, v_2' using the random number ξ_1. Except for minor optimizations, this part of our method is identical to Arvo's method [1995]. Supplement C provides the full algorithm. It also explains how we address stability issues in the determinant computation for G_0 using a Householder reflection.

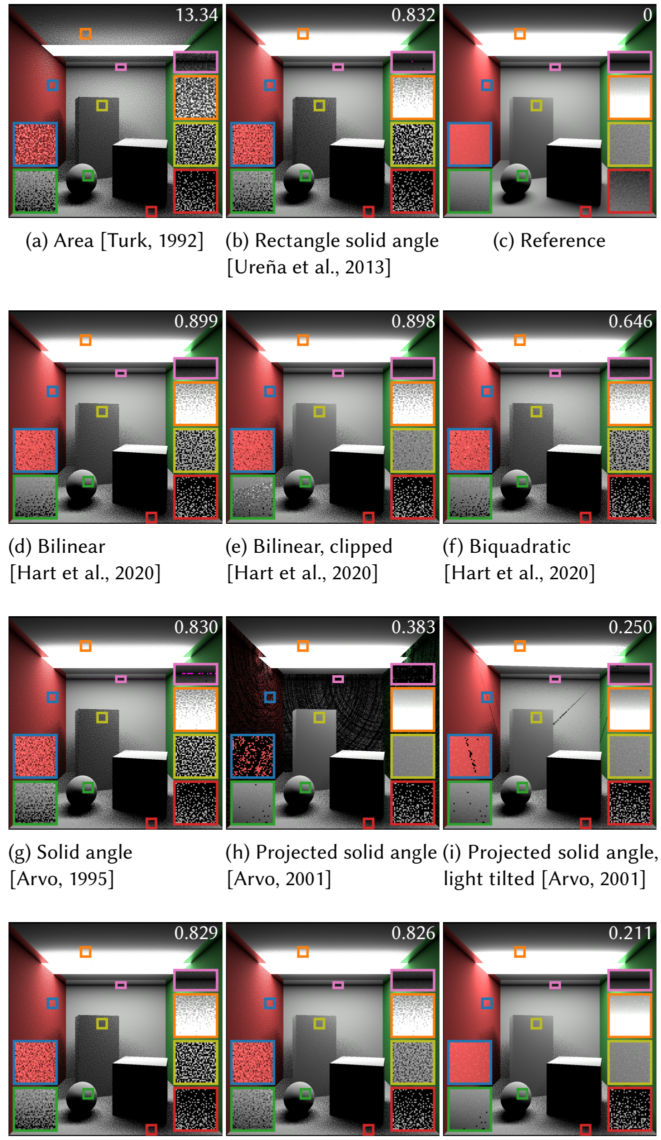
6 RESULTS

Since we have already demonstrated that our technique is accurate (Sec. 3.5 and 3.6), the following evaluation focuses on comparisons to related work and potential applications. After describing our renderer (Sec. 6.1), we compare to other approaches for light sampling on diffuse surfaces (Sec. 6.2) and analyze variance due to imperfect LTC fits (Sec. 6.3). We also propose a biased variant of our technique (Sec. 6.4) and use textured emission for portals, emission profiles and textured lights (Sec. 6.5). Finally, we report run times (Sec. 6.6).

6.1 Our Renderer

Our renderer uses Vulkan and casts shadow rays using the extension VK_KHR_ray_query. It is a deferred renderer with a 32-bit visibility buffer [Burns and Hunt, 2013] that renders direct lighting only. All surfaces use the Frostbite BRDF [Lagarde and de Rousiers, 2014]. Shading normals get clipped to the hemisphere of the outgoing light direction to avoid black pixels due to normal mapping.

Our pseudorandom numbers come from a 2D Sobol sequence that is assigned to pixels using hash-based permutations of a quadtree [Ahmed and Wonka, 2020]. We store them in precomputed textures. This approach maintains the good stratification but removes regular patterns in screen space.



(a) Area [Turk, 1992] (b) Rectangle solid angle [Ureña et al., 2013] (c) Reference

(d) Bilinear [Hart et al., 2020] (e) Bilinear, clipped [Hart et al., 2020] (f) Biquadratic [Hart et al., 2020]

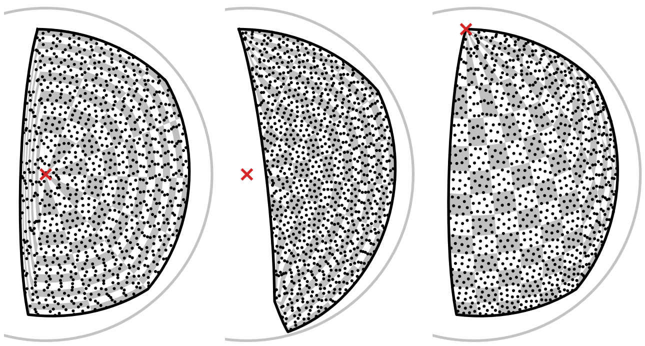
(g) Solid angle [Arvo, 1995] (h) Projected solid angle [Arvo, 2001] (i) Projected solid angle, light tilted [Arvo, 2001]

(j) Solid angle, ours (k) Solid angle, clipped, ours (l) Projected solid angle, ours

Fig. 12. A Cornell box with a rectangular, Lambertian area light stretching through it from left to right. All techniques use one sample per pixel. Our projected solid angle sampling has almost no variance outside of penumbrae. The reported RMSEs are dominated by the bright ceiling.

6.2 Diffuse Shading

Fig. 12 compares ten sampling strategies using a Cornell box. The materials have roughness one, i.e. they are diffuse but not Lambertian. Uniform area sampling [Turk, 1992] (Fig. 12a) suffers from considerable variance on all surfaces, especially near the light. Note that noisy image regions appear darker due to clamping of pixel values at one. Solid angle sampling is considerably better (Fig. 12b, 12g,



(a) Projected solid angle sampling (b) Solid angle sampling

Fig. 13. A 12×12 grid and a blue noise point set [de Goes et al., 2012] in primary sample space are used as input to our sampling methods. Our method for projected solid angle sampling samples radially around the zenith (×). The mapping is continuous and area preserving but non-conformal, especially within narrow sectors or at the clockwise and counterclockwise ends. Our solid angle sampling is strongly non-conformal near vertex 0 (×).

12j). Remaining variance is mostly due to the cosine term (blue and green insets) or visibility (yellow and red insets). Established methods [Arvo, 1995, Ureña et al., 2013] produce infinite or NaN sample coordinates near the plane of the light source, which we mark with pink pixels (pink inset). Our method is more robust (Fig. 12j).

Projected solid angle sampling offers nearly zero variance outside of penumbrae (Fig. 12l, 12h). Residual variance is due to the non-Lambertian BRDF. Our method is robust and gives the best result by far (Fig. 12l). Arvo's method [Arvo, 2001] fails entirely for this scene because light edges are parallel to shading normals, which triggers numerical issues (Fig. 12h). Rotating the light by 1° around each axis reduces these artifacts to a few lines (Fig. 12i, blue inset).

We use the method of Hart et al. [2020] on top of our solid angle sampling. Sampling a bilinear density in primary sample space reduces variance moderately (Fig. 12d). The biquadratic variant makes a minor improvement (Fig. 12f). With our solid angle sampling, we have the option to clip polygons before sampling, which eliminates some zero contributions when the polygon is partially below the horizon (Fig. 12k, yellow inset). Since the method of Hart et al. is not invariant under reordering of vertices, clipping introduces discontinuous changes in variance there (Fig. 12e, green inset).

Fig. 13 demonstrates how our techniques preserve stratification of samples. In certain situations, our area preserving mappings are far from being conformal. In these spots, stratification deteriorates but overall it is preserved well.

6.3 Specular Shading

Originally, LTCs are biased because the BRDF is replaced by its LTC approximation [Heitz et al., 2016]. In our case, LTCs only serve as density in importance sampling and thus our renderings are unbiased. As shown in Fig. 14, the variance due to approximation errors of LTCs is low. Our clamped optimal MIS increases this variance slightly but improves robustness (Fig. 10).

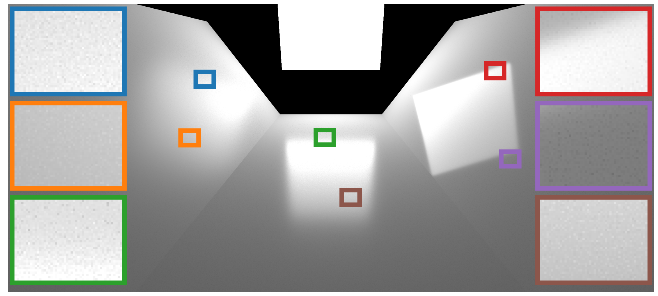
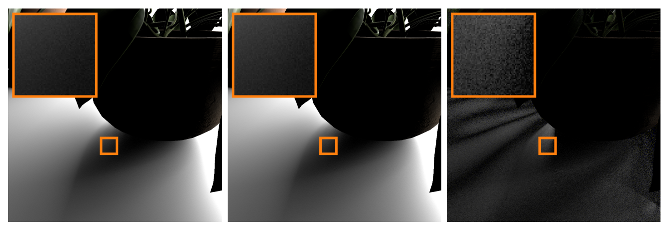


Fig. 14. Three planes of different roughness, lit by a square Lambertian emitter. We use projected solid angle and LTC sampling, combined by our weighted balance heuristic. Although the approximation of the BRDF is imperfect, variance is low using two samples per pixel.



(a) Unbiased (b) Biased (c) Difference times ten

Fig. 15. A plane with low roughness, a large light and partial occlusion provokes noticeable bias in our biased variant of projected solid angle sampling. These images use our clamped optimal MIS at 4096 samples per pixel.

Our techniques enable excellent rendition of shadowed specular highlights (Fig. 1, red inset, Fig. 10e). Compared to BRDF importance sampling [Dong et al., 2015, Heitz and d'Eon, 2014], the boundaries of specular highlights are much smoother because our samples always hit the light (Fig. 10f and 10e, red inset).

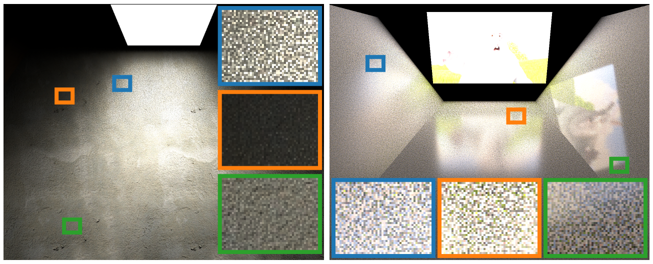
6.4 Biased Variant

In real-time rendering, it is common to sacrifice unbiasedness for efficiency. Thus, we also propose a biased variant of our technique for projected solid angle sampling. From Sec. 3.6, we know that the initialization without any iterations still gives small errors most of the time. Therefore, we disable the iteration in this biased variant. Additionally, we use a piecewise polynomial approximation to atan2 with a maximal forward error of $1.2 \cdot 10^{-5}$ radians.

Most of the time, results of this biased technique are indistinguishable from unbiased renderings. Fig. 15 shows converged renderings of a scene that provokes visible bias. In spite of the adversarial example, the error is only noticeable in direct comparison.

6.5 Textured Emission

Our Monte Carlo estimate naturally supports textured emission. For example, Fig. 1 uses ω_i for a lookup in a light probe, thus turning the light source into a portal [Bitterli et al., 2015]. Our sampling strategy is unaware of the radiance distribution in this light probe. Variance increases compared to a Lambertian emitter but since the



(a) IES profile (Zumtobel Mirel) (b) Textured light (Big Buck Bunny)

Fig. 16. Two kinds of textured emission, rendered using projected solid angle and LTC sampling with our clamped optimal MIS and two samples per pixel. (a) IES profiles contribute to variance in regions with transitions of brightness levels, which shrink at greater distance. (b) Low-dynamic range emission textures add moderate variance everywhere.

light probe used here has relatively low dynamic range, variance is still low. Light probes with higher contrast call for MIS with additional sampling strategies [Bitterli et al., 2015].

As shown in Fig. 16a, IES profiles are also suitable for our purposes. They typically only have a few hard transitions between brightness levels. There is significant variance in these transition regions but they shrink as we move away from the light source. Texturing the area of the light source is a compelling option for displays (Fig. 16b). In this case, every region in the shaded image reflects the whole color palette of the texture. Thus, variance increases everywhere but due to the limited dynamic range convergence is relatively fast.

6.6 Run Time

Our test system consists of an NVIDIA Geforce RTX 2080 Ti, an Intel Core i5-9600K and 16 GB RAM. Our supplemental presents results with frame times that include ray tracing. Here, we are more interested in the cost of different sampling techniques, independent of other aspects of the renderer. Therefore, we disable ray tracing and point the camera at a plane, which is lit from above by a polygonal light. To ensure that shading computations are the limiting factor for frame times, we use 128 samples per pixel. We are still able to deduce the cost for a single sample per pixel if we use these 128 samples for 128 duplicates of the same light source. Per sample, our renderer also reads random numbers and evaluates the BRDF. We measure timings for these steps separately and subtract them, by introducing a baseline sampling technique, which produces incorrect samples but executes only 13 instructions per sample.

Table 1 lists the results of this experiment. With solid angle sampling, the cost for the first sample is similar for all approaches. Only the method specialized to rectangles [Ureña et al., 2013] is significantly faster. In Arvo's method for solid angle sampling [1995] and ours, the main cost per light is computation of the solid angle and our renderer uses the optimized formula of van Oosterom and Strackee [1983] for both methods. However, our technique is 20 to 55% faster for many samples, dependent on the vertex count. For four vertices, it comes close to the method for rectangles [Ureña et al., 2013]. Area sampling [Turk, 1992] is not much faster.

Table 1. Timings in milliseconds for rendering a frame at 1920 × 1080 resolution using 128 samples per pixel. The samples are either taken from 128 different polygonal lights or all from the same light. The baseline timings in the last row, which include access to random numbers, BRDF evaluation, etc., have been subtracted from each timing in the rows above. Timings for six or seven vertices are given in Supplement A.7.

	128 lights			128 samples		
Polygon vertex count	3	4	5	3	4	5
Area, Turk	5.17	6.69	7.78	0.80	0.88	1.10
Rectangle, Ureña et al.	-	3.65	-	-	1.11	-
Solid angle, Arvo	5.36	8.81	10.2	1.35	2.05	2.48
Solid angle, ours	5.92	8.06	7.59	1.12	1.32	1.60
Solid angle, clipped, ours	4.54	7.12	10.1	1.65	2.06	2.35
Bilinear, Hart et al.	5.75	7.21	9.61	2.33	2.59	2.86
Biquadratic, Hart et al.	11.8	14.8	17.3	7.28	7.72	8.12
Proj., central, Arvo	25.7	37.2	54.5	10.7	11.1	11.4
Proj., central, ours	8.07	11.4	13.9	2.13	2.28	2.76
Biased, central, ours	7.65	11.9	15.2	1.91	2.26	2.49
Proj., decentral, Arvo	43.8	59.1	82.8	20.1	20.6	21.7
Proj., decentral, ours	20.0	28.9	39.3	11.1	11.5	11.9
Biased, decentral, ours	13.3	20.4	29.6	4.83	5.52	5.85
+ Baseline	6.15	6.46	6.25	3.54	3.61	3.63

The overhead of clipping is insignificant, in spite of the noteworthy quality improvement (Fig. 12k, yellow inset). Sampling a bilinear density in primary sample space is also inexpensive but the biquadratic version roughly doubles the timings [Hart et al., 2020]. Note that our implementation exploits that corners of primary sample space always map to the same vertex and that the biquadratic version uses a closed-form solver for the cubic.

For projected solid angle sampling techniques, we take separate measurements for the central and decentral cases (Fig. 3). We always use three iterations of Newton's method for Arvo's technique [2001]. This way, code execution is coherent but it does not always converge to adequate accuracy. Nonetheless, our more robust technique is twice as fast in the decentral case and roughly four times faster in the less frequently occurring central case.

Our biased technique (Sec. 6.4) offers an appreciable speedup. The cost per sample in the decentral case, where the iteration is disabled, halves. Compared to our solid angle sampling, this biased variant is two to four times slower, depending on the situation. Considering the significant reduction in variance, this cost is well-justified.

7 CONCLUSIONS AND FUTURE WORK

We have presented a comprehensive suite of solutions for sampling of polygonal lights, ranging from inexpensive solid angle sampling to LTC importance sampling. Our discussion emphasizes GPUs but the methods are equally compelling for CPU renderers, especially when SIMD instructions are used. We retain the benefits of LTCs but turn them into a more flexible and unbiased technique.

At the same time, our work reveals a path towards BRDF importance sampling for light sources of other shapes, e.g. cylinders and ellipsoids. Our method is applicable to cylinders through rejection sampling [Gamito, 2016] and a follow-up work considering the limit case of linear lights is already under review. Ellipsoids are more challenging but we hope that a similar iterative method will work.

More fundamentally, we show the value of iterative methods with second-order derivatives for sampling problems in graphics. Newton's method always struggles with densities approaching zero and is prone to divergent execution on GPUs. Bisection converges slowly. Our methods offer a robust solution with a low, fixed iteration count. Thus, we obtain an iterative importance sampling technique that circumvents the usual drawbacks of such algorithms.

ACKNOWLEDGMENTS

We would like to thank Tobias Zirr, Alisa Jung, Vincent Schüßler, Carsten Dachsbacher and our reviewers for their constructive input. The attic in Figs. 1, 6 and 16a is obtained from Blend Swap and created by Shuprobho Das, Fran Calvente, Asterlil, YopLand, M. Ziemys, Eva11dragon, Rob Tuytel, Greg Zaal and me. Fig. 7 uses the Bistro exterior contributed to ORCA by Amazon Lumberyard.

REFERENCES

Abdalla G. M. Ahmed and Peter Wonka. 2020. Screen-Space Blue-Noise Diffusion of Monte Carlo Sampling Error via Hierarchical Ordering of Pixels. *ACM Trans. Graph. (proc. SIGGRAPH Asia)* 39, 6 (2020). https://doi.org/10.1145/3414685.3417881

James Arvo. 1995. Stratified Sampling of Spherical Triangles. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. ACM, 437–438. https://doi.org/10.1145/218380.218500

James Arvo. 2001. Stratified sampling of 2-manifolds. In *State of the Art in Monte Carlo Ray Tracing for Realistic Image Synthesis (SIGGRAPH 2001 Course Notes)*. ACM.

Laurent Belcour, Guofu Xie, Christophe Hery, Mark Meyer, Wojciech Jarosz, and Derek Nowrouzezahrai. 2018. Integrating Clipped Spherical Harmonics Expansions. *ACM Trans. Graph.* 37, 2, Article 19 (2018). https://doi.org/10.1145/3015459

Benedikt Bitterli, Jan Novák, and Wojciech Jarosz. 2015. Portal-Masked Environment Map Sampling. *Computer Graphics Forum (proc. EGSR)* 34, 4 (2015). https://doi.org/10.1111/cgf.12674

Benedikt Bitterli, Chris Wyman, Matt Pharr, Peter Shirley, Aaron Lefohn, and Wojciech Jarosz. 2020. Spatiotemporal Reservoir Resampling for Real-Time Ray Tracing with Dynamic Direct Lighting. *ACM Trans. Graph. (proc. SIGGRAPH)* 39, 4, Article 148 (2020). https://doi.org/10.1145/3386569.3392481

James F. Blinn. 2006. How to solve a quadratic equation. Part 2. *IEEE Computer Graphics and Applications* 26, 2 (2006), 82–87. https://doi.org/10.1109/MCG.2006.35

Christopher A. Burns and Warren A. Hunt. 2013. The Visibility Buffer: A Cache-Friendly Approach to Deferred Shading. *Journal of Computer Graphics Techniques (JCGT)* 2, 2 (2013), 55–69. http://jcgt.org/published/0002/02/04/

Fernando de Goes, Katherine Breeden, Victor Ostromoukhov, and Mathieu Desbrun. 2012. Blue Noise through Optimal Transport. *ACM Trans. Graph. (proc. SIGGRAPH Asia)* 31, 6, Article 171 (2012). https://doi.org/10.1145/2366145.2366190

Zhao Dong, Bruce Walter, Steve Marschner, and Donald P. Greenberg. 2015. Predicting Appearance from Measured Microgeometry of Metal Surfaces. *ACM Trans. Graph.* 35, 1 (2015). https://doi.org/10.1145/2815618

Jonathan Dupuy, Eric Heitz, and Laurent Belcour. 2017. A Spherical Cap Preserving Parameterization for Spherical Distributions. *ACM Trans. Graph. (proc. SIGGRAPH)* 36, 4, Article 139 (2017). https://doi.org/10.1145/3072959.3073694

Manuel N. Gamito. 2016. Solid Angle Sampling of Disk and Cylinder Lights. *Computer Graphics Forum (proc. EGSR)* 35, 4 (2016). https://doi.org/10.1111/cgf.12946

Pascal Grittmann, Iliyan Georgiev, Philipp Slusallek, and Jaroslav Křivánek. 2019. Variance-Aware Multiple Importance Sampling. *ACM Trans. Graph. (proc. SIGGRAPH Asia)* 38, 6, Article 152 (2019). https://doi.org/10.1145/3355089.3356515

Ibón Guillén, Carlos Ureña, Alan King, Marcos Fajardo, Iliyan Georgiev, Jorge López-Moreno, and Adrián Jarabo. 2017. Area-Preserving Parameterizations for Spherical Ellipses. *Computer Graphics Forum (proc. EGSR)* 36, 4 (2017). https://doi.org/10.1111/cgf.13234

David Hart, Matt Pharr, Thomas Müller, Ward Lopes, Morgan McGuire, and Peter Shirley. 2020. Practical Product Sampling by Fitting and Composing Warps. *Computer Graphics Forum (proc. EGSR)* 39, 4 (2020). https://doi.org/10.1111/cgf.14060

Eric Heitz. 2020. Can't Invert the CDF? The Triangle-Cut Parameterization of the Region under the Curve. *Computer Graphics Forum (proc. EGSR)* 39, 4 (2020). https://doi.org/10.1111/cgf.14058

Eric Heitz and Eugene d'Eon. 2014. Importance Sampling Microfacet-Based BSDFs using the Distribution of Visible Normals. *Computer Graphics Forum (proc. EGSR)* 33, 4 (2014), 103–112. https://doi.org/10.1111/cgf.12417

Eric Heitz, Jonathan Dupuy, Stephen Hill, and David Neubelt. 2016. Real-time Polygonal-light Shading with Linearly Transformed Cosines. *ACM Trans. Graph. (proc. SIGGRAPH)* 35, 4, Article 41 (2016). https://doi.org/10.1145/2897824.2925895

Eric Heitz and Stephen Hill. 2017a. Linear-light shading with linearly transformed cosines. In *GPU Zen: Advanced Rendering Techniques*. Black Cat Publishing Inc., 137–162. https://hal.archives-ouvertes.fr/hal-02155101

Eric Heitz and Stephen Hill. 2017b. Physically Based Shading in Theory and Practice: Real-Time Line and Disk Light Shading. In *ACM SIGGRAPH 2017 Courses*. ACM, Article 7. https://doi.org/10.1145/3084873.3084893

Eric Heitz, Stephen Hill, and Morgan McGuire. 2018. Combining Analytic Direct Illumination and Stochastic Shadows. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (short paper) (I3D '18)*. ACM, Article 2. https://doi.org/10.1145/3190834.3190852

Claude-Pierre Jeannerod, Nicolas Louvet, and Jean-Michel Muller. 2013. Further analysis of Kahan's algorithm for the accurate computation of 2x2 determinants. *Math. Comp.* 82, 284 (2013). https://doi.org/10.1090/S0025-5718-2013-02679-8

Ondřej Karlík, Martin Šik, Petr Vévoda, Tomáš Skřivan, and Jaroslav Křivánek. 2019. MIS Compensation: Optimizing Sampling Techniques in Multiple Importance Sampling. *ACM Trans. Graph. (proc. SIGGRAPH Asia)* 38, 6, Article 151 (2019). https://doi.org/10.1145/3355089.3356565

Ivo Kondapaneni, Petr Vevoda, Pascal Grittmann, Tomáš Skřivan, Philipp Slusallek, and Jaroslav Křivánek. 2019. Optimal Multiple Importance Sampling. *ACM Trans. Graph. (proc. SIGGRAPH)* 38, 4, Article 37 (2019). https://doi.org/10.1145/3306346.3323009

Sébastien Lagarde and Charles de Rousiers. 2014. Physically Based Shading in Theory and Practice: Moving Frostbite to PBR. In *ACM SIGGRAPH 2014 Courses*. ACM, Article 23. https://doi.org/10.1145/2614028.2615431

Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. 2018. Differentiable Monte Carlo Ray Tracing through Edge Sampling. *ACM Trans. Graph. (proc. SIGGRAPH Asia)* 37, 6, Article 222 (2018). https://doi.org/10.1145/3272127.3275109

Guillaume Loubet, Tizian Zeltner, Nicolas Holzschuch, and Wenzel Jakob. 2020. Slope-Space Integrals for Specular Next Event Estimation. *ACM Trans. Graph. (proc. SIGGRAPH Asia)* 39, 6, Article 239 (2020). https://doi.org/0.1145/3414685.3417811

Christian Luksch, Lukas Prost, and Michael Wimmer. 2020. Real-Time Approximation of Photometric Polygonal Lights. *Proc. ACM Comput. Graph. Interact. Tech. (proc. i3D)* 3, 1, Article 4 (2020). https://doi.org/10.1145/3384537

Pierre Moreau, Matt Pharr, and Petrik Clarberg. 2019. Dynamic Many-Light Sampling for Real-Time Ray Tracing. In *High-Performance Graphics - Short Papers*. Eurographics. https://doi.org/10.2312/hpg.20191191

J. A. Nelder and R. Mead. 1965. A Simplex Method for Function Minimization. *Comput. J.* 7, 4 (1965), 308–313. https://doi.org/10.1093/comjnl/7.4.308

Christoph Peters and Carsten Dachsbacher. 2019. Sampling Projected Spherical Caps in Real Time. *Proc. ACM Comput. Graph. Interact. Tech. (proc. i3D)* 2, 1, Article 1 (2019). https://doi.org/10.1145/3320282

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2016. *Physically Based Rendering, 3rd Edition*. Morgan Kaufmann. http://www.pbr-book.org

Justin Talbot, David Cline, and Parris Egbert. 2005. Importance resampling for global illumination. In *EGSR 2005*. https://doi.org/10.2312/EGWR/EGSR05/139-146

Greg Turk. 1992. Generating random points in triangles. In *Graphics Gems*, Andrew S. Glassner (Ed.). Academic Press, 24–28.

Carlos Ureña. 2000. Computation of Irradiance from Triangles by Adaptive Sampling. *Computer Graphics Forum* 19, 2 (2000). https://doi.org/10.1111/1467-8659.00452

Carlos Ureña, Marcos Fajardo, and Alan King. 2013. An Area-Preserving Parametrization for Spherical Rectangles. *Computer Graphics Forum (proc. EGSR)* 36, 4 (2013). https://doi.org/10.1111/cgf.12151

Carlos Ureña and Iliyan Georgiev. 2018. Stratified Sampling of Projected Spherical Caps. *Computer Graphics Forum (proc. EGSR)* 37, 4 (2018). https://doi.org/10.1111/cgf.13471

Adriaan van Oosterom and Jan Strackee. 1983. The Solid Angle of a Plane Triangle. *IEEE Transactions on Biomedical Engineering* 30, 2 (1983), 125–126. https://doi.org/10.1109/TBME.1983.325207

Eric Veach and Leonidas J. Guibas. 1995. Optimally Combining Sampling Techniques for Monte Carlo Rendering. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. ACM, 419–428. https://doi.org/10.1145/218380.218498

Changyaw Wang. 1992. Physically correct direct lighting for distribution ray tracing. In *Graphics Gems III*. Academic Press Professional, 307–313.

Jingwen Wang and Ravi Ramamoorthi. 2018. Analytic Spherical Harmonic Coefficients for Polygonal Area Lights. *ACM Trans. Graph. (proc. SIGGRAPH)* 37, 4, Article 54 (2018). https://doi.org/10.1145/3197517.3201291

Rex West, Iliyan Georgiev, Adrien Gruson, and Toshiya Hachisuka. 2020. Continuous Multiple Importance Sampling. *ACM Trans. Graph. (proc. SIGGRAPH)* 39, 4, Article 136 (2020). https://doi.org/10.1145/3386569.3392436