(0227) Adaptive Importance Sampling for Multi-Ray Gathering

Ivan Neulander
Rhythm & Hues Studios

We present an adaptive noise reduction technique for integrating incident radiance at a fixed position. We use a Russian-roulette-based importance sampler to reshape the directional probability density of future rays in a batch, based on an *affinity map* that incorporates ratings of evaluated rays, provided by the rendering engine. Our method is unbiased, has low overhead, requires no precomputation, and works in concert with other importance sampling schemes.

Figure 1: Single-bounce diffuse reflection of mapped environment, with traditional importance sampling (left) and adaptive importance sampling (right). Ray counts differ but render times match.

Motivation

Our original motivation was to reduce shadow noise on surfaces diffusely lit by infinite area lights. We use the "PH" importance sampler [Pharr and Humphreys 2010], which selects bright pixels from a latitude-longitude environment map in logarithmic time (based on texture width) using a pair of explicit cdfs. While very fast, this sampler is unaware of occluding geometry, so it may repeatedly send rays into dark objects or outside the BSDF lobe, producing noise in occluded regions.

Blending the PH sampler with a BSDF-based sampler using multiple importance sampling (MIS)[Veach 1998] only tends to increase noise except in very dark shadows. As an alternative, we considered adjusting the PH cdfs adaptively based on evaluated rays. But the cost of these updates becomes prohibitive beyond low texture resolutions. We ultimately opted for a rejection-based approach, drawing extra samples and discarding some of them to achieve the desired ray count with a modified directional density. This method is general enough to work with any existing importance samplers, including MIS, allowing the BSDF, lighting, and occlusion to all guide ray selection.

Implementation

Our technique involves the stochastic rejection of rays based on an *affinity map*, described below. In addition, we may unconditionally reject rays that fall outside the BSDF support. To avoid introducing bias, we reject samples using Russian roulette, as described by [Veach 1998]. This can cause extra zero-contribution rays to be generated, exceeding the user-requested ray count. The sampler reports this zero count to the renderer, allowing it to increment the effective ray total by which the summed ray color is ultimately divided.

Stochastic rejection causes the total samples drawn per gather batch to fluctuate, precluding the use of low-discrepancy sequences that require a fixed sample count for proper stratification. We have found the multidimensional Halton sequence to work reasonably well for this and other forms of adaptive sampling, despite its higher dispersion compared to fixed-size sequences.

The Affinity Map

For each batch of gather rays we construct and query a multi-resolution latitude-longitude mapping to represent directional affinity for future rays. We define affinity(θ,ϕ) as the probability of accepting a ray in the direction (θ,ϕ).

The affinity map is populated based on previously traced rays, each of which is rated by the renderer. This rating compares the evaluated incident radiance of the ray with the radiance estimated by the sampler. A mip-map pyramid of 3-channel textures comprise the affinity map. The first two channels store total weighted affinity and total weight, while the third stores a counter that facilitates rapidly clearing the affinity map for the next ray batch.

At each batch, pixels are cleared to an average weighted affinity of 1, requiring low subsequent ratings to allow future rejections. This bias toward 1 prevents low-confidence affinity values derived from low sample counts from causing improper sample rejections. Each time a ray is rated with a value r, all textures in the affinity map are updated at that pixel as follows. Even though r may exceed 1 (bringing up the average accordingly) affinity is by definition clamped to 1.

$$\begin{aligned} w &= 1/\text{affinity}(\theta,\phi) \\ \text{map}(\theta,\phi)[0] &+= rw \\ \text{map}(\theta,\phi)[1] &+= w \end{aligned} \tag{1}$$

To compute affinity(θ,ϕ) for a given ray, we point-sample all resolutions of the affinity map at (θ,ϕ), starting with the highest. The first nonempty pixel determines the tentative result: $a_t = map(\theta,\phi)[0]/map(\theta,\phi)[1]$. However, we impose a penalty for large pixels by rescaling the result to $a_{min} + a_t(1-a_{min})$, where $a_{min} = 1/(tol \cdot height^2_{map})$ for some user tolerance tol. This prevents large pixels from causing significant rejection, despite their potentially low average affinity. (Our formula for a_{min} is not based on a pixel's solid angle because polar pixels subtend large latitude angles despite their low solid angles, so they are no "safer" to use than equatorial pixels.)

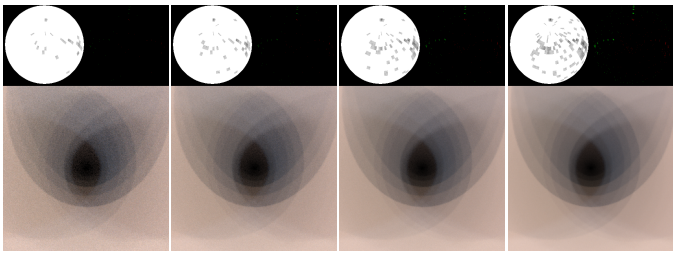

Figure 2: A shadow cast from an environment map by a sphere, rendered with 64, 128, 256, 512 rays/pixel. The upper subimages show the affinity map (left) and ray samples (right) for a single chosen pixel. Red dots mark occluded ray directions, green dots unoccluded.

Future Work

Our method works best when a large number of gather rays are sampled from the same point. Reusing an existing affinity map at subsequent nearby shading positions (with some distance-based penalty) could permit more aggressive rejection early on. Also, the use of pixel filtering and less distorting texture parametrizations for the affinity map should allow for better rejection decisions.

References

PHARR, M., AND HUMPHREYS, G. 2010. *Physically Based Rendering, Second Edition: From Theory To Implementation*, 2nd ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

VEACH, E. 1998. *Robust monte carlo methods for light transport simulation*. PhD thesis, Stanford, CA, USA. AAI9837162.