Pacific Graphics 2013
B. Levy, X. Tong, and K. Yin
(Guest Editors)

Volume 32 (2013), Number 7

Adaptive Ray-bundle Tracing with Memory Usage Prediction: Efficient Global Illumination in Large Scenes (Implementation Details)

Yusuke Tokuyoshi[1], Takashi Sekine[1], Tiago da Silva[1,2], and Takashi Kanai[2]

[1]Square Enix Co., Ltd., Japan
[2]University of Tokyo, Japan

1. Preprocessing for Importance Analysis

In this paper, since importance (i.e., light map texel density) is constant in a triangle primitive, it is precomputed and stored in each triangle for acceleration. In theory, such importance $I(t)$ for a triangle t can be computed as follows:

$$I(t) = \frac{A_l(t)}{A_o(t)}, \tag{1}$$

where $A_o(t)$ is the area of the triangle in object space and $A_l(t)$ is the area of the triangle in light map space. However, it can lead to large errors for small triangles in practice, as shown in Figure 2 (a). UV unwrapping tools can produce noise caused by lack of computation precision which is generally undetectable to artists. Even if the area of a triangle in light map space is imperceptible, it can be much larger than the area in object space for practical scenes.

To avoid noise, an image space technique is employed in this paper. Since the lowest required bandwidth is the same as the light map texel size, the texel density is estimated using the four nearest neighbor texels as shown in Figure 1 left. Let $\mathbf{x}_i$ be a neighbor texel of the target texel $\mathbf{x}$, then the density can be given by:

$$I(\mathbf{x}) = \frac{\sum_{i=0}^{3} w(\mathbf{x}_i)w(\mathbf{x}_j)}{\sum_{i=0}^{3} w(\mathbf{x}_i)w(\mathbf{x}_j)\|\mathbf{e}_i \times \mathbf{e}_j\|}, \tag{2}$$

where $j = i+1 \pmod 4$, and $w(\mathbf{x}_i) = 1$ if the neighbor texel $\mathbf{x}_i$ is mapped to an object, otherwise $w(\mathbf{x}_i) = 0$. The edge vector $\mathbf{e}_i$ is given by $\mathbf{e}_i = \mathbf{p}(\mathbf{x}_i) - \mathbf{p}(\mathbf{x})$, where $\mathbf{p}(\mathbf{x})$ is the position in object space. To obtain $\mathbf{p}(\mathbf{x})$, a position buffer is first rendered using the same projection as light maps. However, Equation (2) can produce zero divide, as shown in Figure 1 right. Therefore, instead of Equation (2), we use the following approximation:

$$I(\mathbf{x}) = \frac{\sum_{i=0}^{3} w(\mathbf{x}_i)}{\sum_{i=0}^{3} w(\mathbf{x}_i)\|\mathbf{e}_i\|^2}. \tag{3}$$

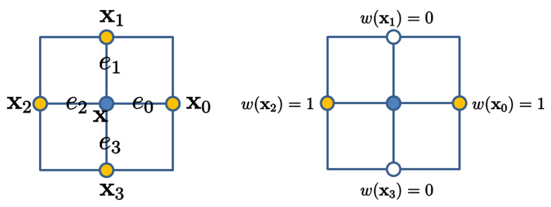


Figure 1: *The texel density is estimated using the four neighbors of the target texel in the position buffer. However, as shown in the right image, singularities can be produced. This paper uses an approximated estimation in order to reduce singularities.*

Finally, Equation (1) is clamped to the maximum value of Equation (3) $I_{max}(t) = \max_{\mathbf{x}\in t}(I(\mathbf{x}))$ as:

$$I(t) = \min\left(\frac{A_l(t)}{A_o(t)}, I_{max}(t)\right), \tag{4}$$

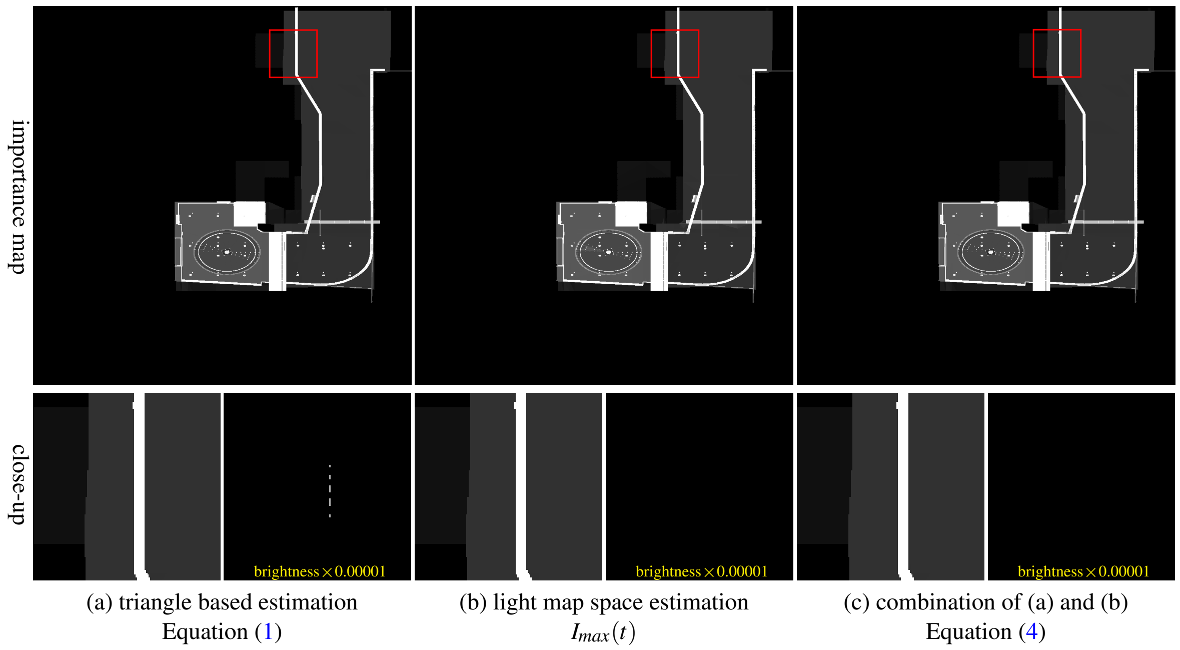
and stored in the triangle t as shown in Figure 2 (b)(c). If the triangle t is smaller than the pixel size and there are no intersected sampling points of pixels, $I_{max}(t) = 0$.

The maximum importance $I_{max}(t)$ is calculated by rendering all triangles using the same projection as the light maps. The pixel shader estimates $I(\mathbf{x})$ for each texel and computes the maximum value $I_{max}(t)$ for each triangle using an atomic instruction. Next, Equation (4) is computed for each triangle using a compute shader. Since there are only these two rendering passes and the compute pass, the preprocessing time is negligibly small compared to the total computation time.

2. Light Transport Computation

This section describes our light transport algorithm. In the light transport computation phase, once a local ray-bundle is generated, the light transport problem is solved by using the

© 2013 The Author(s)
Computer Graphics Forum © 2013 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd.



importance map
close-up
brightness × 0.00001
brightness × 0.00001
brightness × 0.00001
(a) triangle based estimation Equation (1)
(b) light map space estimation $I_{max}(t)$
(c) combination of (a) and (b) Equation (4)

Figure 2: *The left importance map estimated by Equation (1), presenting errors which are over 500 texels / mm^2. Light map space estimation (middle) avoids the errors for small triangles, while it can produce additional errors for large triangles. Their combination produces better results.*

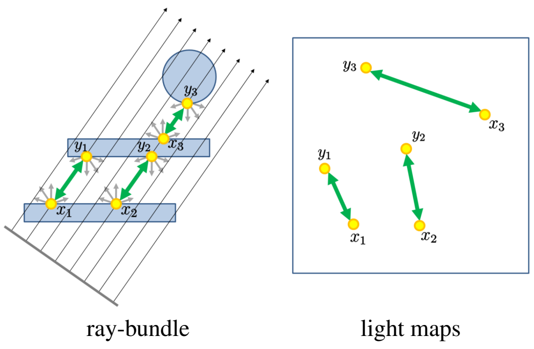


Figure 3: *Ray-bundle tracing and radiance exchange. The sample direction is randomly generated and light maps are updated in an iterative fashion. Tiled ray-bundles are created in each iteration. Light transport is computed by looking up the radiance of visible fragments from the light maps computed in the previous iteration.*

radiance exchange method described in [HHGM10]. Hermes et al. used texture atlases as intermediate data structures, whereas we use light maps directly. The light maps are updated in an iterative fashion. The light transfer is computed between all pairs of successive points. These pairs are found using the ray-bundle (see Figure 3). Taking x_1 and y_1 to be a pair of successive points, the pixel corresponding to x_1 in the light maps is updated using the radiance at y_1. The approximated radiance at y_1 is obtained by light maps computed in the previous iteration. Similarly, the pixel corresponding to y_1 in the light maps is updated using the previous radiance at x_1. This updating scheme allows multiple interreflections without additional ray generation. Although this method is biased, it converges to the truth by increasing the light map resolution and ray-bundle resolution.

3. Optimizations

Frustum Culling A local ray-bundle is often very narrow compared to an entire scene. Only light maps intersected with local ray-bundle should be updated. To reduce redundant computations, frustum culling is employed for ray-bundle creation and light map update. Frustum culling ensures scene size scalability for our tiled ray-bundle tracing.

Data Compression With our method, 16 bits float type and RGBE type [War91] are employed for light maps and ray-bundles respectively to reduce memory usage. Since our light maps have RGB and weight channels, 8 bytes per texel are used. A node data size of the linked-list is 12 bytes because it has radiance (RGBE type), depth (float type), and next node pointer (integer type). However, conversion from 32 bits float type to these lower-precision data type is a biased operation and it diverges in only a few thousand direction samplings. To avoid this problem, stochastic rounding is used as it enables unbiased conversion by rounding a value randomly. Furthermore, the resulting images are implicitly half-toned.

References

[HHGM10] HERMES J., HENRICH N., GROSCH T., MUELLER S.: Global illumination using parallel global ray-bundles. In *Proc. of VMV 2010* (2010), pp. 65–72. 2

[War91] WARD G.: *Real Pixels*. Academic Press, 1991, pp. 80–83. 2

© 2013 The Author(s)
Computer Graphics Forum © 2013 The Eurographics Association and John Wiley & Sons Ltd