Runtime Implementation of Modular Radiance Transfer

Brad Loos[1] Lakulish Antani[2] Kenny Mitchell[3] Derek Nowrouzezahrai[4] Wojciech Jarosz[4] Peter-Pike Sloan[3]

[1]University of Utah [2]UNC Chapel Hill [3]Disney Interactive Studios [4]Disney Research Zürich

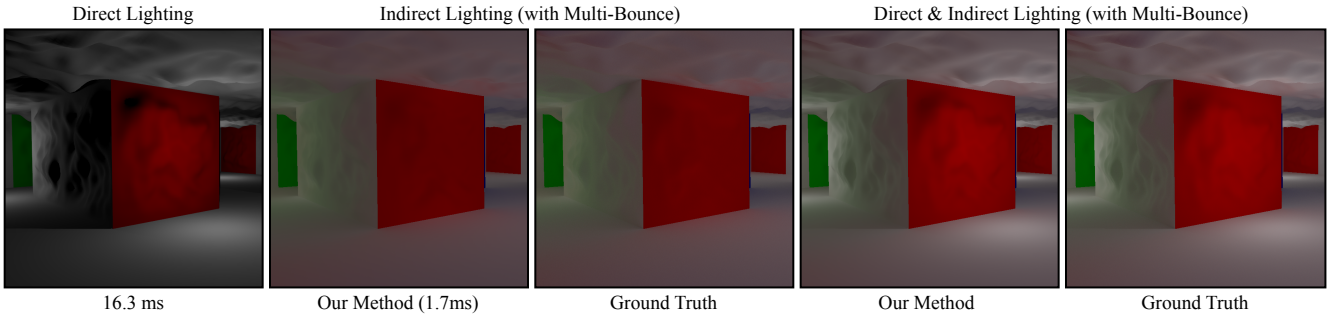

Figure 1: *Indirect light computed for a* 19 *block maze-like cave scene constructed completely at run-time. Our algorithm can model color bleeding and indirect light from surfaces with detailed normal variation and real-time performance. Timings are on an NVidia 480 GTX.*

1 Introduction

Real-time rendering of indirect lighting significantly enhances the sense of realism in video games. Unfortunately, previously including such effects often required time consuming scene dependent precomputation and heavy runtime computations unsuitable for low-end devices, such as mobile phones or game consoles. Modular Radiance Transfer (MRT) [Loos et al. 2011], is a recent technique that computes approximate direct-to-indirect transfer [Hašan et al. 2006; Kontkanen et al. 2006; Lehtinen et al. 2008] by warping and combining light transport from a small library of simple shapes. This talk complements a technical paper and focuses on implementation issues, including how our run time is designed to scale across many different platforms, from iPhones to modern GPUs.

2 Modular Radiance Transfer Overview

MRT is a novel technique for approximate direct-to-indirect transfer. The key contributions are the use of a lighting prior to reduce the dimension/entropy of the one-bounce light transport operator, approximating a scene with simpler shapes, and efficiently computing indirect lighting on these simpler shapes by expressing both self transfer and propagation between shapes using carefully constructed reduced-dimensional spaces. We achieve high performance on a wide range of platforms (see [Loos et al. 2011] for more details).

Mathematically indirect lighting is computed as

$$\mathbf{l}_{\text{ind}} = \mathbf{U_b}\ \mathbf{b} + \mathbf{U_r}\ \mathbf{r}\ ,$$

where $\mathbf{b} = \mathbf{T_{d\to b}}\ \mathbf{l}_{\text{d}}$ are the spectral indirect lighting coefficients, $\mathbf{r} = \mathbf{T}_{\text{prop}}\ \mathbf{b}$ are reduced lightfield response coefficients.

The $\mathbf{T_{d\to b}}$ operator maps direct light ($\mathbf{l}_{\text{d}}$) to reduced indirect light coefficients and $\mathbf{T}_{\text{prop}}$ maps indirect light at all the blocks scene shapes to reduced dimensional lightfields between these blocks. $\mathbf{U_b}$ is a compact basis for indirect light *inside* a shape, and $\mathbf{U_r}$ is a compact basis for mapping light flowing through a lightfield to response from a shape *to its neighbors*.

3 Implementation Details

The core run time steps for MRT are as follows:

1. Compute direct light at a reduced resolution in each block,
2. Compute per-block direct lighting spectral coefficients,
3. Compute indirect light *within* a block (into a dynamic lightmap)
4. Compute response coefficients at all the blocks' interfaces,
5. Blend response from "external" blocks into a dynamic lightmap,
6. Pad the lightmap texture to avoid texture mapping artifacts,
7. Render static elements using dynamic lightmap of indirect lighting,
8. **[optional]** Compute indirect light volume (from $\mathbf{b}$'s and $\mathbf{r}$'s),
9. **[optional]** Render dynamic objects the volume lighting.

Optional steps are currently only computed in the GPU version, but could also be computed on low-end platforms. We attempted to use GPUs on low-end devices for as many components of our algorithm as possible, however current texture format and performance limitations made this infeasible (e.g., evaluating a single unshadowed point-light takes $25ms$ on the iPad).

We instead focused on CPU optimizations: using memory layouts for the coefficients that favor code vectorization, exploiting the vector architecture itself, and balancing the CPU and GPU register counts. We transformed the GPU data layout used on higher-end platforms to a more CPU friendly one, optimizing for better cache usage, as well as investigating a few alternative CPU implementations before settling on what turned out to be most effective.

4 Conclusion

MRT is a compelling, real-time approximate global illumination technique. This talk discusses implementation details that allow the algorithm to scale across different hardware platforms with wildly varying capabilities. We believe some of the techniques employed by MRT may be useful to developers facing similar scaling challenges.

References

HAŠAN, M., PELLACINI, F., AND BALA, K. 2006. Direct-to-indirect transfer for cinematic relighting. *ACM Trans. Graph.*.

KONTKANEN, J., TURQUIN, E., HOLZSCHUCH, N., AND SILLION, F. 2006. Wavelet radiance transport for interactive indirect lighting. In *EGSR*.

LEHTINEN, J., ZWICKER, M., TURQUIN, E., KONTKANEN, J., DURAND, F., SILLION, F. X., AND AILA, T. 2008. A meshless hierarchical representation for light transport. *TOG*.

LOOS, B., ANTANI, L., MITCHELL, K., NOWROUZEZAHRAI, D., JAROSZ, W., AND SLOAN, P.-P. 2011. Modular radiance transfer (under review). In *EGSR*.