From the ACM SIGGRAPH Asia 2011 conference proceedings.

Modular Radiance Transfer

Bradford J. Loos[1,2] Lakulish Antani[1,3] Kenny Mitchell[1] Derek Nowrouzezahrai[4] Wojciech Jarosz[4] Peter-Pike Sloan[1]
[1]Disney Interactive Studios [2]University of Utah [3]UNC Chapel Hill [4]Disney Research Zurich

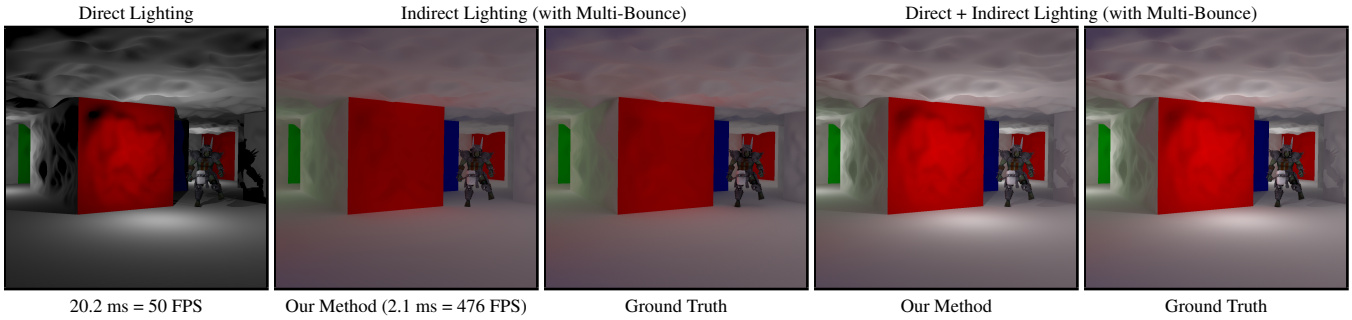


Figure 1: *Indirect light computed in reduced subspaces for a cave with 19 blocks and 4 lights. We derive low-dimensional transport operators, on simple proxy shapes, that are warped and combined at run-time, at $>$ 475 FPS on high-end GPUs and $>$ 45 FPS on mobile platforms, and can model indirect light at surfaces (with detailed normal variation) and within volumes of large-scale scene geometry.*

Abstract

Many rendering algorithms willingly sacrifice accuracy, favoring plausible shading with high-performance. Modular Radiance Transfer (MRT) models coarse-scale, distant indirect lighting effects in scene geometry that scales from high-end GPUs to low-end mobile platforms. MRT eliminates scene-dependent precomputation by storing compact transport on simple shapes, akin to bounce cards used in film production. These shapes' modular transport can be instanced, warped and connected on-the-fly to yield approximate light transport in large scenes. We introduce a prior on incident lighting distributions and perform all computations in low-dimensional subspaces. An *implicit lighting environment* induced from the low-rank approximations is in turn used to model secondary effects, such as volumetric transport variation, higher-order irradiance, and transport through lightfields. MRT is a new approach to precomputed lighting that uses a novel low-dimensional subspace simulation of light transport to uniquely balance the need for high-performance and portable solutions, low memory usage, and fast authoring iteration.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture;

Keywords: Global Illumination, GPU, Interactive

Links: DL PDF

1 Introduction

Indirect illumination increases the realism of computer generated images. The ambient term is a simple inexpensive approximation that does not respond to dynamic lighting. Accurate real-time techniques [Keller 1997] have difficulty scaling to complex scenes and often have significant performance requirements, particularly on modern console and mobile platforms. Techniques that approximate different elements of indirect lighting have been extremely successful in interactive graphics applications. Precomputation techniques used in video games [Chen 2008; Larsson and Halen 2009] tend to assume static scenes and lighting, but suffer from long authoring iteration times and memory requirements. Ambient Occlusion (AO) [Zhukov et al. 1998] captures only salient shading effects. Variants of Precomputed Radiance Transfer (PRT) [Sloan et al. 2002] generate soft lighting results. These techniques are favorable compared to more accurate techniques due to their lower storage and computation costs and the pleasing nature of their approximation. Modular Radiance Transfer (MRT) targets **coarse-scale, distant** indirect lighting in scene geometry, responds plausibly and smoothly to dynamic lighting, has extremely high-performance, and allows fast author iteration.

Our shapes are motivated by bounce cards used in live-action films. These planes only approximate indirect light from geometry in the real-world but offer a high level of control to produce the desired lighting. In digital film production, non-shadow casting lights are commonly used to allow artists to quickly iterate and achieve a desired look. The ease-of-use and controllability of these approximations outweighs their physically incorrect nature.

Our approach is very efficient, uses very little data and a quick, one-time, *scene-independent* precomputation step. It also allows real-time computation of approximate indirect light, and is designed with rapid iteration of light design in mind. We precompute light transport operators (LTOs) for a handful of simple canonical "shapes", then interactively warp and combine these shapes, along with their LTOs, to more complex geometry. These shape proxies are used to model direct-to-indirect transport which is then applied as a light map to the actual scene geometry. The flow of indirect light between proxies is modeled with lightfields, and all computations are performed on very low-dimensional subspaces. MRT results in plausible, dynamic global-illumination effects, rendered at high frame rates with low memory overhead. We design special LTOs for secondary transport effects such as light volumes for dynamic characters and higher-order irradiance for normal mapping. We illustrate our solutions ability to scale from high-end to mobile platforms and, like PRT, to provide smooth results which respond to light change.

doi.acm.org/10.1145/2024156.2024212

Keeping the design goals mentioned above in mind, MRT makes the following contributions:

- **Compact transport operators using a novel lighting prior:** we represent direct and indirect light in a specialized *light prior* basis which enables us to build compact and efficient LTOs to propagate multiple bounces of indirect light.
- **Modular, scene-independent transport computation:** LTOs from different canonical shapes are warped and combined, resulting in on-the-fly mapping of complex scene geometry (and its light transport) to simpler shapes (and their LTOs).
- **Flexible and efficient implementation:** we show that our approach runs on low-end mobile platforms as well as high-end GPUs, all while maintaining high performance.

2 Related Work

Interactive global illumination. Several approaches approximate global illumination without precomputation [Wang et al. 2009]. Instant radiosity techniques [Keller 1997] dynamically inject virtual point lights (VPL) to simulate diffuse bounced light [Dachsbacher and Stamminger 2005; Dachsbacher and Stamminger 2006; Ritschel et al. 2008]. Examples of other approaches are screen-space radiosity [Nichols and Wyman 2009; Nichols et al. 2009], multi-resolution height field sampling [Nowrouzezahrai and Snyder 2009], precomputed rigid reflectance fields [Iwasaki et al. 2007], and real-time ray-tracing [Parker et al. 1999]. Unfortunately, these techniques do not scale to large scenes, do not exploit coherence in light transport operators, or do not have performance characteristics suitable for the strict rendering budgets of modern game engines. For example, evaluating an unshadowed point light takes 25 ms on the iPad, eliminating the possibility of VPL techniques.

Light Propagation Volumes (LPVs) [Kaplanyan and Dachsbacher 2010] augment VPL approaches with a discrete, volumetric propagation phase for approximate global illumination. By storing and propagating a linear spherical harmonics (SH) radiance distribution in a volume grid encompassing the entire scene, LPVs avoid precomputation, capture indirect shadows, and attain high-performance. However, the heavyweight nature of LPVs precludes implementation on low-end platforms and radiance propagation causes energy loss which precludes distant light propagation: e.g., our large maze examples would be challenging to render with LPVs.

Precomputed light transport. PRT [Sloan et al. 2002; Lehtinen 2007; Ramamoorthi 2009] computes shading response to basis illumination, capturing soft shadows, indirect light and caustics for static geometry. Extensions to local lighting [Kristensen et al. 2005] require lengthy preprocessing and large amounts of data, as well as being scene dependent. We build on direct-to-indirect transfer [Hašan et al. 2006; Kontkanen et al. 2006; Wang et al. 2007; Lehtinen et al. 2008] although, unlike previous approaches, we represent transport using bases tailored to plausible direct and indirect lighting. We perform fast precomputation on small shape dictionaries so that, at run-time, scene-specific transport can be approximated by warping and combining transport from the dictionary elements. In this manner, we decouple scene dependence from precomputation.

A recent interactive approach [Martin and Einarsson 2010] has similar goals, but with significant differences. Both techniques use simplified geometry, though we use even coarser geometry and reuse transfer data. In addition, we employ a novel lighting prior on plausible radiance distributions to enable high performance precomputation and relighting, scaling from low- to high-end platforms.

Approximate techniques. SSAO [Mittring 2007; Bavoil et al. 2008] approximates AO using depth buffer sampling. While physically-incorrect, SSAO is easily implementable across many platforms, has high-performance, and simple artist control, leading to broad adoption in modern games. One limitation of screen space techniques is they can not model lighting out of the frame. In contrast we target coarse, soft and often distant indirect lighting effects. Our approach, while capable, is not meant for physically-accurate simulation. We purposefully trade accuracy for: extremely high-performance, soft and plausible shading that *responds* to dynamic lighting, and the ability to manipulate large scenes.

Other techniques use proxies to compute approximate visibility [Ren et al. 2006] or indirect light [Sloan et al. 2007; Guerrero et al. 2008] from dynamic objects in an efficient manner. In contrast, we target extremely high performance indirect lighting of "world geometry" (and bouncing this light to dynamic objects), with scalability to large dynamic scenes and across many platforms.

Surface normal and volume light variation. Adding high-frequency surface variation to smooth shading is a popular approach used in games [McTaggart 2004; Chen 2008]. As in meshless direct-to-indirect transfer [Lehtinen et al. 2008], we seamlessly support this "transport normal mapping". Irradiance volumes [Greger et al. 1998] store radiance distributions in a volumetric grid so that, at run-time, dynamic objects can be lit by their environment. While games have used irradiance volumes to give dynamic objects a sense of immersion, we dynamically generate indirect irradiance volumes based on dynamic light transport. We share the same mathematical framework across surface and volumetric lighting response.

Modeling surface light variation. Meyer and Anderson [2006] perform PCA on noisy indirect light, leveraging the low-frequency nature of color bleeding to quickly filter out noise. Motivated by their work, we precompute bases for physically-realizable direct light and indirect transport, coupling these spaces for efficient computation. As with our work, Ashdown [2001] also performs spectral analysis on transport operators (in a radiosity context), however we use a prior on the direct light distribution, construct low-dimensional bases for direct and indirect light, and perform transport completely in these reduced spaces. Similarly, we augment lightfields [Levoy and Hanrahan 1996; Gortler et al. 1996] with response bases to propagate and couple light flow to distant geometry.

Scene subdivision and coupling. We compute lightfields at interfaces similarly to Lewis and Fournier [1996]. However, we avoid instantiating a signal on the lightfields at run-time by using precomputed operators that couple between reduced spaces, aggregating distant light while bounding the cost of local indirect lighting evaluation. Similar concepts have been used in fluid simulation [Wicke et al. 2009] and off-line rendering [Xu et al. 1990].

3 Preliminaries

We adopt the following notation: italics for scalars and 3D points/vectors (e.g., ω), boldface lowercase for column vectors (e.g., $\mathbf{l}$), and boldface uppercase for matrices (e.g., $\mathbf{T}$).

3.1 Standard Direct-to-Indirect Transfer

Suppose we choose n surface locations on the scene to sample direct light; direct-to-indirect transfer maps direct illumination at these points to indirect lighting at these points:

$$\mathbf{l}_{\mathrm{ind}} = \mathbf{F}\,\mathbf{l}_{\mathrm{d}}\,, \tag{1}$$

Computed during precomputation then discarded:	
$\mathbf{L}_\mathrm{d}$	Matrix of all possible direct lighting signals.
$\mathbf{P}$	Light prior basis retaining k_d left singular vectors of $\mathbf{L}_\mathrm{d}$.
$\mathbf{L}_\mathrm{imp}$	Implicit lighting environment.
$\mathbf{H}_\mathrm{lf}$	Raw lightfield matrix.
$\mathbf{H}_\mathrm{rlf}$	Reduced lightfield basis retaining k_rlf modes of $\mathbf{H}_\mathrm{lf}$.
$\mathbf{M}$	Light space indirect LTO.
Computed during precomputation, used during level initialization:	
$\mathbf{R}_{\mathbf{b}\to\mathrm{rlf}}$	Projects $\mathbf{b}$'s to rlf-space (at each dictionary items' interface).
$\mathbf{R}_{\mathbf{b}\to\mathbf{b}'}$	Propagates interface lightfield (input/output are both in rlf-space).
$\mathbf{T}_{\mathrm{rlf}\to\mathbf{r}}$	Maps reduced lightfield values to $k_\mathbf{r}$ surface response modes.
Computed during precomputation, used during run time:	
$\mathbf{T}_{\mathbf{d}\to\mathbf{b}}$	Transforms direct light ($\mathbf{l}_\mathrm{d}$) to indirect light coefficients ($\mathbf{b}$).
$\mathbf{U}_\mathbf{b}$	Indirect light basis after retaining $k_\mathbf{b}$ modes. Maps $\mathbf{b}$'s to $\mathbf{l}_\mathrm{ind}$.
$\mathbf{U}_{\vec{\mathbf{b}}}$	OHLSH version of $\mathbf{U}_\mathbf{b}$ (maps $\mathbf{b}$'s to vector irradiance).
$\mathbf{U}_{\mathbf{b}\mathrm{vol}}$	Maps $\mathbf{b}$'s to indirect volumetric light represented in SH.
$\mathbf{U}_\mathbf{r}$	Indirect lightfield response basis with $k_\mathbf{r}$ modes. Maps $\mathbf{r}$'s to $\mathbf{l}_\mathrm{ind}$.
$\mathbf{T}_{\mathbf{b}\to\mathbf{r}}$	Maps $\mathbf{b}$'s to $\mathbf{r}$'s via lightfields of the level's block connectivity.
Computed during run-time:	
$\mathbf{l}_\mathrm{d}$	Direct lighting.
$\mathbf{b}$	Spectral lighting coefficients due to self-transfer ($\mathbf{T}_{\mathbf{d}\to\mathbf{b}}\ \mathbf{l}_\mathrm{d}$).
$\mathbf{r}$	Lightfield response coefficients on surfaces ($\mathbf{T}_{\mathbf{b}\to\mathbf{r}}\ \mathbf{b}$).
$\mathbf{l}_\mathrm{ind}$	Indirect lighting ($\mathbf{U}_\mathbf{b}\ \mathbf{b} + \mathbf{U}_\mathbf{r}\ \mathbf{r}$).

Table 1: *Notation used in this paper.*

where $\mathbf{l}_\mathrm{d}$ and $\mathbf{l}_\mathrm{ind}$ are n-dimensional vectors of direct and indirect irradiance, and $\mathbf{F}$ is the one bounce transport operator.

Evaluating Equation 1 quickly limits run-time direct-to-indirect performance since $\mathbf{F}$ grows as $\mathcal{O}(n^2)$. Thus, we approximate $\mathbf{F}$ using singular value decomposition (SVD), $\mathbf{F} = \mathbf{U}_\mathrm{f}\ \mathbf{\Sigma}_\mathrm{f}\ \mathbf{V}_\mathrm{f}^\mathrm{T}$, where $\mathbf{U}_\mathrm{f}$ and $\mathbf{V}_\mathrm{f}^\mathrm{T}$ are high-dimensional rotation matrices, and $\mathbf{\Sigma}_\mathrm{f}$ is a diagonal matrix of singular values σ_i. Approximate indirect light can be computed by retaining the k largest singular values, $\mathbf{l}_\mathrm{ind} \approx \tilde{\mathbf{U}}_\mathrm{f}\ \tilde{\mathbf{\Sigma}}_\mathrm{f}\ \tilde{\mathbf{V}}_\mathrm{f}^\mathrm{T}\ \mathbf{l}_\mathrm{d}$, where $\mathbf{U}_\mathrm{f}$/$\mathbf{V}_\mathrm{f}^\mathrm{T}$/$\mathbf{\Sigma}_\mathrm{f}$ are replaced with truncated matrices (retaining only the top k columns/rows) $\tilde{\mathbf{U}}_\mathrm{f}$/$\tilde{\mathbf{V}}_\mathrm{f}^\mathrm{T}$/$\tilde{\mathbf{\Sigma}}_\mathrm{f}$.

3.2 Overview

The truncated SVD of $\mathbf{F}$ requires large k for accurate computation of $\mathbf{l}_\mathrm{ind}$ (see Table 2), motivating a different approach. We define a novel **lighting prior** to compute reduced-dimensional transfer within a block (Section 4) and then couple the transfer between blocks with lightfields at their mutual interface (Section 5). Lastly, we show how to warp lighting from simple shapes to complex scene geometry (Section 6). These steps involve several intermediate spaces for derivation, but we always perform runtime computation in the low-dimensional spaces. The different spaces and quantities used in our derivations are summarized in Table 1.

Matrix	80%	90%	95%
$\mathbf{F}$ / $\mathbf{L}_\mathrm{d}$ / $\mathbf{M}$	164 / 22 / 1	240 / 41 / 5	313 / 62 / 8

Table 2: *Number of singular values to capture percentage of the energy for different matrices (for a cube with $n = 6 \times 16^2$ samples).*

4 Reduced Direct-to-Indirect Transfer

Taking the SVD of $\mathbf{F}$ assumes a uniform distribution of arbitrary n-dimensional direct light patterns, explaining the slow decay of σ_i. In real scenes, direct light at a given point obeys a (simplified) rendering equation and is not drawn from an arbitrary distribution. We define a *light prior* over the distribution of direct light to construct a space spanned by physically-plausible $\mathbf{l}_\mathrm{d}$. Our analysis will show that this space has dimensionality **significantly smaller** than n.

We first sample direct illumination patterns, $\{\mathbf{l}_{\mathrm{d}0}, \ldots, \mathbf{l}_{\mathrm{d}m}\}$ in order to construct a low-dimensional basis for plausible direct lighting. These patterns can be computed using any approach, however it is best to use the same direct illumination at run-time. We generate each sample by placing a sphere light at uniform volumetric locations in our (canonical) block shape and computing the direct lighting.

Next, we place the samples $\{\mathbf{l}_{\mathrm{d}0}, \ldots, \mathbf{l}_{\mathrm{d}m}\}$ into columns of a matrix $\mathbf{L}_\mathrm{d}$ and compute its SVD: $\mathbf{L}_\mathrm{d} = \mathbf{U}_\mathrm{d}\ \mathbf{\Sigma}_\mathrm{d}\ \mathbf{V}_\mathrm{d}^\mathrm{T}$. The left singular vectors yield our *light prior* basis: $\mathbf{P} = \tilde{\mathbf{U}}_\mathrm{d}$. Table 2 summarizes the singular value fall-off, justifying our earlier observation that physically-realizable direct light lies in a low-dimensional linear subspace: $\dim(\mathbf{L}_\mathrm{d}) \ll n$. We do not subtract the mean and compute PCA, since we wish to represent lighting with arbitrary intensities, which means all scales of input patterns should be well represented.

4.1 Light Transport in Indirect Light Space

Any plausible indirect lighting condition can be approximated as a linear combination of indirect light due to the direct light prior basis vectors: $\mathbf{l}_\mathrm{ind} = \mathbf{F}\ \mathbf{P}\ \left[\mathbf{P}^\mathrm{T}\ \mathbf{l}_\mathrm{d}\right]$, where $\mathbf{P}^\mathrm{T}$ projects $\mathbf{l}_\mathrm{d}$ onto our light prior, resulting in scaling coefficients for indirect light induced by the light prior basis vectors. In other words, each direct lighting (basis) pattern has a corresponding indirect lighting (basis) pattern.

Unfortunately, this formulation does not exploit correlations in the indirect light (the columns of $\mathbf{F}\ \mathbf{P}$ are not independent). We aim to *directly* obtain an orthogonal basis for indirect light which accounts for our direct light prior, instead of simply reconstructing direct light and applying the one-bounce operator to it (as described above).

We start by post-multiplying $\mathbf{F}\ \mathbf{P}$ by the scaling matrix $\mathbf{S} = \tilde{\mathbf{\Sigma}}_\mathrm{d}$, which leads to an equivalent problem:

$$\mathbf{l}_\mathrm{ind} = \mathbf{F}\ \mathbf{P}\ \mathbf{S}\ \mathbf{S}^{-1}\ \mathbf{P}^\mathrm{T}\ \mathbf{l}_\mathrm{d}\ , \tag{2}$$

where we define the *light space indirect LTO* $\mathbf{M} = \mathbf{F}\ \mathbf{P}\ \mathbf{S}$, which maps (scaled) direct lighting prior coefficients ($\mathbf{l}_\mathrm{d}$) to indirect light, and take its SVD: $\mathbf{M} = \mathbf{U}_\mathrm{m}\ \mathbf{\Sigma}_\mathrm{m}\ \mathbf{V}_\mathrm{m}^\mathrm{T}$. Table 2 shows that the SVD of $\mathbf{M}$ falls off much more rapidly than either $\mathbf{F}$ or $\mathbf{L}_\mathrm{d}$. We retain $k_\mathbf{b} \ll n$ singular values, yielding the approximation:

$$\mathbf{l}_\mathrm{ind} \approx \mathbf{U}_\mathbf{b}\ \mathbf{T}_{\mathbf{d}\to\mathbf{b}}\ \mathbf{l}_\mathrm{d} = \mathbf{U}_\mathbf{b}\ \mathbf{b}\ , \tag{3}$$

where $\mathbf{T}_{\mathbf{d}\to\mathbf{b}} = \tilde{\mathbf{V}}_\mathrm{m}^\mathrm{T}\ \mathbf{S}^{-1}\ \mathbf{P}^\mathrm{T}$ maps $\mathbf{l}_\mathrm{d}$ to spectral coefficients $\mathbf{b}$, used to scale columns of $\mathbf{U}_\mathbf{b} = \tilde{\mathbf{U}}_\mathrm{m}\ \tilde{\mathbf{\Sigma}}_\mathrm{m}$. These *orthogonal* columns form an *indirect light space* basis and scaling by $\tilde{\mathbf{\Sigma}}_\mathrm{m}$ makes the lengths proportional to the statistics from $\mathbf{M}$. Figure 2 illustrates several columns of $\mathbf{U}_\mathbf{b}$ and rows of $\mathbf{T}_{\mathbf{d}\to\mathbf{b}}$.

Implicit Lighting Environment. An interesting question to consider is whether there exists a direct lighting pattern that, after application of the one bounce transport operator, generates the columns of $\mathbf{U}_\mathbf{b}$ as an output indirect lighting pattern. In other words, we wish to find $\mathbf{L}_\mathrm{imp}$ such that $\mathbf{U}_\mathbf{b} = \mathbf{F}\ \mathbf{L}_\mathrm{imp}$. We derive this direct lighting pattern, which we call the *implicit lighting environment*, as follows:

$\mathbf{M} = \mathbf{U}_\mathrm{m}\ \mathbf{\Sigma}_\mathrm{m}\ \mathbf{V}_\mathrm{m}^\mathrm{T} = \mathbf{F}\ \mathbf{P}\ \mathbf{S}$ by definition, and after post-multiplying both sides by $\mathbf{V}_\mathrm{m}$, we obtain $\mathbf{U}_\mathrm{m}\ \mathbf{\Sigma}_\mathrm{m}\ \mathbf{V}_\mathrm{m}^\mathrm{T}\ \mathbf{V}_\mathrm{m} = \mathbf{F}\ \mathbf{P}\ \mathbf{S}\ \mathbf{V}_\mathrm{m}$, with the left hand side simplifying to $\mathbf{U}_\mathrm{m}\ \mathbf{\Sigma}_\mathrm{m} = \mathbf{U}_\mathrm{b}$.

And so, $\mathbf{L}_\mathrm{imp} = \mathbf{P}\ \mathbf{S}\ \mathbf{V}_\mathbf{m}$ and it will prove useful in several instances, e.g. when generating higher-order lighting variation on surfaces (Section 6.4), volume samples within the scene (Section 6.5), and interface lightfields between connected shapes (Section 5.1).

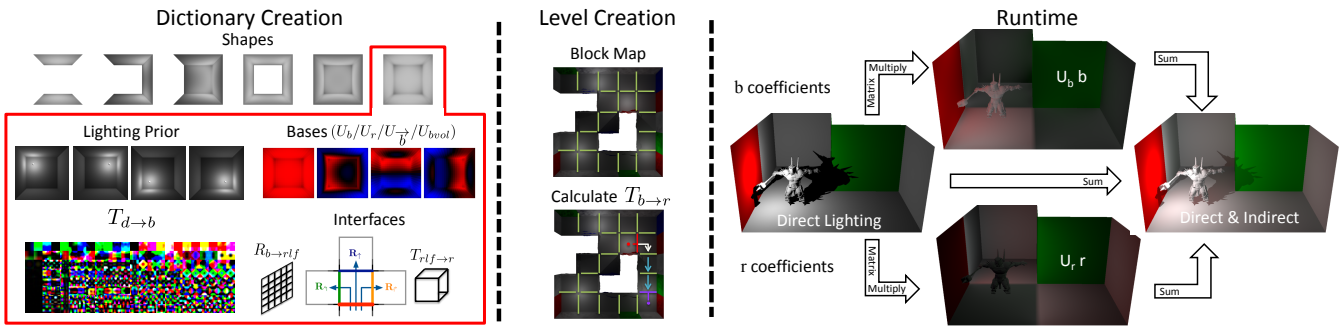


Figure 2: ***Overview:*** *dictionary creation precomputes light priors and bases for scene independent shapes. Level creation maps the shapes and generates the lightfield propagation matrix* $\mathbf{T}_{b\to r}$*. At runtime, we generate* $\mathbf{b}$ *and* $\mathbf{r}$ *vectors to compute indirect light with* $\mathbf{U}_b$ *and* $\mathbf{U}_r$ *bases.*

5 Direct-to-Indirect Transfer Between Shapes

We first couple transport between blocks in order to compute direct-to-indirect transfer on large, interconnected sets of dictionary shapes.

Our high performance relies on computing coupled transport in reduced basis spaces. Given $\mathbf{b}$ coefficients for a simple shape (e.g. a cube with missing faces), our goal is to compute operators that act directly on this vector and scatter light into neighboring shapes.

5.1 Interfaces: Far-Field Light Transport Coupling

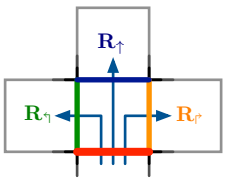
Our approach is independent of the dictionary's contents and we illustrate results with cube-based and cylinder-based dictionaries. For cubic shapes, we create five operators: $\mathbf{R}_{b\to rlf}$ describes how light leaves each shape via its missing faces or interfaces, $\mathbf{R}_{\leftarrow,\uparrow,\rightarrow}$ describes transfer between interfaces, and $\mathbf{T}_{rlf\to r}$ describes transfer from an interface onto the surface of a block. These five operators are concatenated, based on block layout, at level creation time to generate a sparse block matrix $\mathbf{T}_{b\to r}$ describing how light leaving each block illuminates all other connected blocks.

We compose a raw lightfield matrix $\mathbf{H}_{\mathrm{lf}}$ at the N_i dictionary interfaces (see Section 6.1). Columns of $\mathbf{H}_{\mathrm{lf}}$ are a resampling of implicit lights (columns of $\mathbf{L}_{\mathrm{imp}}$), for each dictionary element, at each position and direction of the interface's lightfield. $\mathbf{H}_{\mathrm{lf}}$ is an $s \times N_i\ k_{\mathrm{self}}$ matrix, where s is the spatio-directional lightfield resolution.

Given the SVD of $\mathbf{H}_{\mathrm{lf}} \approx \tilde{\mathbf{U}}_{\mathrm{rlf}}\ \tilde{\mathbf{\Sigma}}_{\mathrm{rlf}}\ \tilde{\mathbf{V}}^{\mathrm{T}}_{\mathrm{rlf}}$, the k_{rlf} left singular vectors scaled by the corresponding singular values ($\mathbf{H}_{\mathrm{rlf}} = \tilde{\mathbf{U}}_{\mathrm{rlf}}\ \tilde{\mathbf{\Sigma}}_{\mathrm{rlf}}$) form a low-rank *reduced lightfield basis*. This represents the response at the shape's interfaces to $\mathbf{L}_{\mathrm{imp}}$. We use *basis enrichment* to handle more complex propagation operations (see Appendix A).

We define a $\mathbf{R}_{\mathrm{b}\to\mathrm{rlf}}$ operator for each interface of each dictionary element to map the element's $\mathbf{b}$ coefficients to coefficients in the reduced lightfield basis. We construct this operator by lighting each element with its implicit lighting environments, resampling the implicit lighting from the surfaces to the interface(s), and projecting the resampled lightfields into the reduced lightfield basis.

We construct three additional operators to capture near-field interface-to-interfacelightfield propagation and resampling. Given an interface (red line in small figure), its lightfield can be propagated to the interface straight ahead (dark blue line) with $\mathbf{R}_{\uparrow}$, or to the interface on the left adjacent face (green line) with $\mathbf{R}_{\leftarrow}$, or to the interface on the right adjacent face (orange line) with $\mathbf{R}_{\rightarrow}$.



These square matrices (with dimensions k^2_{rlf}) resample the lightfield at one interface to either the straight-ahead, left-adjacent, or right-adjacent interface, and project the resulting lightfield back into the reduced lightfield basis.

Lastly, we define an operator to map reduced lightfield coefficients to lighting response on geometry near a lightfield: $\mathbf{G} = \mathbf{F}_{\mathrm{rlf}}\ \mathbf{H}_{\mathrm{rlf}}$, where $\mathbf{F}_{\mathrm{rlf}}$ is a transport matrix that computes the surface response to the reduced lightfield basis lighting (columns of $\mathbf{H}_{\mathrm{rlf}}$). This only needs to be done for a single canonical interface, due to symmetry.

Using a similar motivation as in Section 4.1, we compute the SVD of $\mathbf{G} = \mathbf{U}_{\mathrm{g}}\ \mathbf{\Sigma}_{\mathrm{g}}\ \mathbf{V}^{\mathrm{T}}_{\mathrm{g}}$, leveraging coherence to $\mathbf{H}_{\mathrm{rlf}}$'s response, and retain the left singular vectors (columns of $\mathbf{U}_{\mathrm{r}}$, where $\mathbf{U}_{\mathrm{r}} = \tilde{\mathbf{U}}_{\mathrm{g}}$) to form an *indirect lightfield basis*. $\mathbf{T}_{\mathrm{rlf}\to\mathrm{r}} = \tilde{\mathbf{\Sigma}}_{\mathrm{g}}\ \tilde{\mathbf{V}}^{\mathrm{T}}_{\mathrm{g}}$ maps reduced lightfield coefficients to reduced lightfield responses $\mathbf{r}$.

We only retain operators, $\mathbf{R}_{\mathrm{b}\to\mathrm{rlf}}$ and $\mathbf{T}_{\mathrm{rlf}\to\mathrm{r}}$, that act in reduced spaces. Lightfields never need to be reconstructed, resulting in significant memory and performances savings.

6 Direct-to-Indirect Transfer on Real Scenes

We now combine compact direct-to-indirect transport operators within (Section 4) and between (Section 5) simple shapes to quickly approximate direct-to-indirect transfer on *arbitrary* scenes.

Smooth, plausible and dynamic light transport is computed every frame using a mapping between dictionary shapes and scene geometry during scene creation. We derive special transport operators for vector-valued irradiance (to model high-frequency surface detail), and volumetric direct-to-indirect light probes (to relight dynamic objects), all of which will be computed entirely in reduced spaces.

6.1 2D Shape Dictionary

To simplify our exposition, we first consider 2D mazes with cubes connected to each other along missing faces (we also show cylinder-based results). For these scenes, we populate a dictionary with all possible cubes with 0 to 4 faces missing[1]. Exploiting symmetry, this dictionary has 6 entries with a total of 12 missing faces.

6.2 Authoring the Scene Proxy

We map complex geometry to *only a handful* of connected blocks from our shape dictionary. To do so, we begin by generating a set of connected blocks and associating portions of the complex scene to each block. We are motivated to model scenes with extremely coarse proxy geometry in order to capture large-scale indirect lighting effects as efficiently and compactly as possible; unlike discrete

[1]For 2D mazes, all cubes have a "top" and "bottom" face; 3D mazes do not have this constraint and result in a 9 element dictionary.

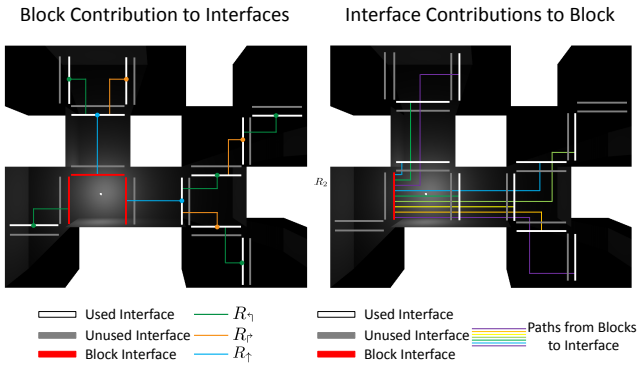


Figure 3: ***Left****: light originating from the block containing the red interfaces propagates to all other visible outgoing interfaces in the maze.* ***Right****: all the blocks that contribute light to the red interface. These are rows and columns of the matrix* $\mathbf{T}_{\mathbf{b}\to\mathbf{r}}$ *that maps* $\mathbf{b}$*s to* $\mathbf{r}$*s.*

ordinate methods that use a large number of blocks. This is analogous to the use of bounce cards when lighting for film.

The real scene must be mapped to shapes in our dictionary before lighting can be computed. Each shape in this proxy is an instanced transformation of an element in our dictionary, and also contains a region in texture space for a light map atlas that will be constructed to light the real scene. We also compute a geometry image that will be used to compute $\mathbf{l}_{\mathrm{d}}$. A key point is that we can re-use the precomputed transfer data from the dictionary. Dictionary shapes can be warped and attached by artists, or automatically for simple mappings (see Section 7.2). When applying severe warping to shapes, we can reduce artifacts by dynamically warping the prior and $\mathbf{U}$s (see Section 8 and Figure 12).

While each shape's texture is self-contained, the final mapping to real scene geometry requires continuous reconstruction between connected shapes. Moreover, clamping is required over the shape's internal creases. We rectify these continuity issues by creating a padded light map atlas, and a set of records that copy edge/corner values from neighboring un-padded regions to the padded light maps (faces translate to the center of padded faces) [Loos et al. 2011].

6.3 Interface Propagation for Distant Light Transport

Once the set of proxy shapes is generated, we compute a block sparse operator $\mathbf{T}_{\mathbf{b}\to\mathbf{r}}$ to map $\mathbf{b}$ coefficients at all the shapes to $\mathbf{r}$ coefficients at each interface. The interface matrices $\mathbf{R}_{\mathrm{b}\to\mathrm{rlf}}, \mathbf{R}_{\curvearrowleft}, \mathbf{R}_{\uparrow}, \mathbf{R}_{\curvearrowright}$, and $\mathbf{T}_{\mathrm{rlf}\to\mathrm{r}}$ are combined to propagate transport through a complex scene in our reduced spaces. It is important to note that these steps only depend on the block connectivity and the input is parameterized by the $\mathbf{b}$ coefficients, which results in an efficient runtime.

To propagate indirect light from block x, breadth-first traversal of the block connectivity (with a maximum traversal depth $d_{\max}$) builds columns of $\mathbf{T}_{\mathbf{b}\to\mathbf{r}}$ to map indirect light coefficients $\mathbf{b}$ from x to reduced lightfield coefficients $\mathbf{r}$ at the traversal's current interface.

The relevant block of $\mathbf{T}_{\mathbf{b}\to\mathbf{r}}$ is initialized to x's $\mathbf{R}_{\mathrm{b}\to\mathrm{rlf}}$ and, as each interface is traversed, is pre-multiplied with one of $\{\mathbf{R}_{\curvearrowleft}, \mathbf{R}_{\uparrow}, \mathbf{R}_{\curvearrowright}\}$, depending on the propagation direction [Loos et al. 2011]. This traversal concatenates transport operators to "drive" indirect light from source shapes, through interfaces, to the rest of the scene.

At each interface, the accumulated portion of $\mathbf{T}_{\mathbf{b}\to\mathbf{r}}$ is pre-multiplied by $\mathbf{T}_{\mathrm{rlf}\to\mathrm{r}}$ and stored. If a path from the same source block has already been computed, the matrices are summed, otherwise a new record is computed. These block matrices form a sparse representation of the full matrix that maps the block's $\mathbf{b}$ coefficients to reduced interface response $\mathbf{r}$ at all other lightfields visible from x. When this matrix is multiplied by the $\mathbf{b}$'s, we are aggregating the energy at every scene interface, avoiding more expensive reconstructions into a light map. Figure 3 is a visual representation of an column and row of the matrix. This process is only done when a scene is created, and takes less than a second for all our example scenes.

6.4 Higher-Order Irradiance in Indirect Light Space

We replace indirect irradiance (columns of $\mathbf{U}_{\mathbf{b}}$ in Equation 3) with vector-valued irradiance capable of approximating indirect light due to high-frequency, normal-mapped surface details (see Figure 4).

We derive an Optimal Hemispherical Linear SH (OHLSH) basis for vector-valued irradiance stored at the n sample locations in each canonical block. To compute the vector-valued irradiance response basis, $\mathbf{U}^{\Rightarrow}_{\mathbf{b}}$, we simply light a block with its $\mathbf{L}_{\mathbf{imp}}$ and project the irradiance distribution into quadratic SH instead. This 9D vector is analytically mapped to a 4D OHLSH vector (see Appendix B).

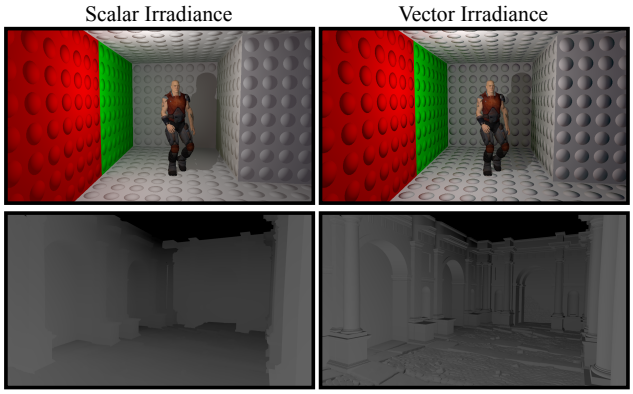


Figure 4: *Vector irradiance for surface details in a normal-mapped maze (****top****) and the Great Hall (****bottom****).* © *2011 The Authors.*

6.5 Parameterized Irradiance Volumes

We light dynamic geometry, such as animating characters and "ornamental clutter" like pillars and statues that do not map naturally to block faces (see Figure 5), by computing parameterized lightprobes in the volume of the scene. These lightprobes are represented as order-3 SH vectors sampled uniformly in space and allow us to shade animated (or "clutter") geometry using traditional SH techniques.

We compute an operator $\mathbf{U}_{\mathbf{b}\mathrm{vol}}$ (with size $9 \times k_{\mathrm{self}}$) that gathers radiance from implicit lights ($\mathbf{L}_{\mathrm{imp}}$) in a uniform volumetric grid in each block, and projects these distributions into SH. $\mathbf{U}_{\mathbf{b}\mathrm{vol}}$ maps indirect illumination in the indirect light space to SH coefficients. As with surface and interface response, we use $\mathbf{L}_{\mathrm{imp}}$ to drive the generation of these probes entirely in the indirect light space, and need only precompute this operator once for the shapes in our dictionary.

7 Implementation and Results

7.1 Dictionary Construction and Precomputation

We generate a 2D block dictionary with six basic cube shapes and a cylinder dictionary with three shapes (straight, right turn, left turn).

For these dictionaries, we construct the prior $\mathbf{P}$ by lighting the basic shapes with 6^3 sphere lights placed uniformly in the block's volume and through interfaces. We use $k_{\mathrm{d}} = 64$ and $k_{\mathrm{self}} = 32$, enriching the basis with functions that are constant on only one face at a time as well as $\mathbf{U}_{\mathbf{b}}$ so that multiple light bounces are well represented.

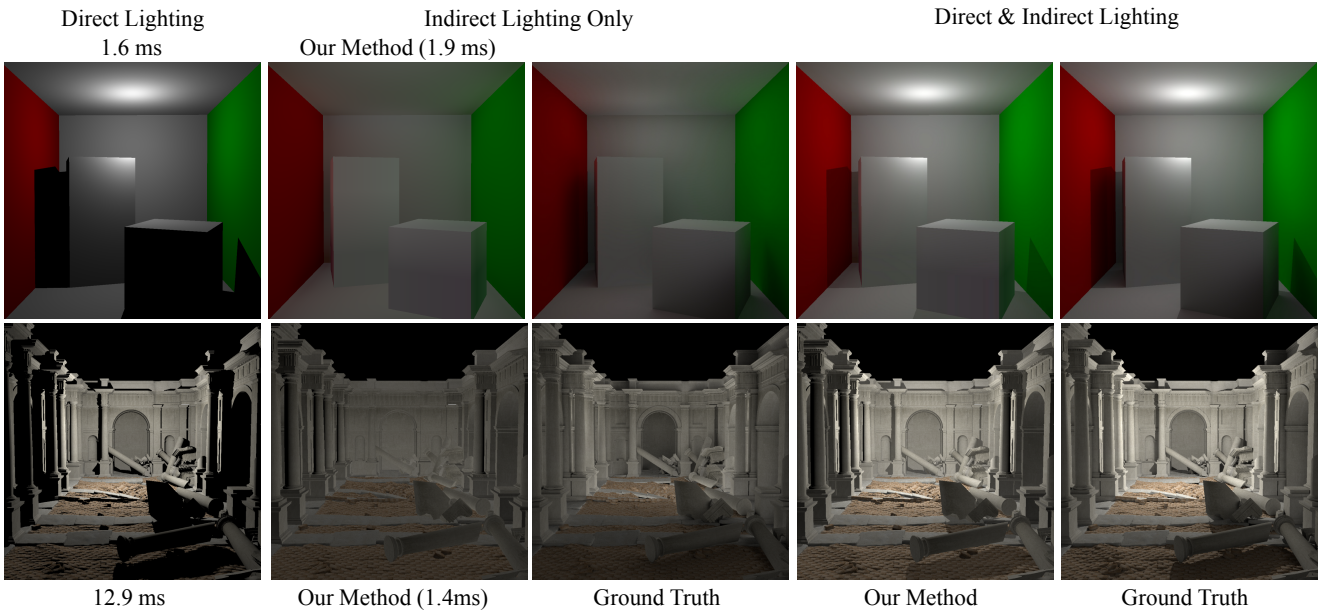


Figure 6: *Dynamic indirect light at >**500 FPS**. We ignore indirect shadows and focus on adding smooth, approximate dynamic indirect light with low computation cost. In the extreme case of the Great Hall modeled with 1 cube (and volume samples for "clutter" geometry), our results respond to direct light at a cost similar to static ambient terms but with soft shading quality common in e.g. PRT.* © *2011 The Authors.*

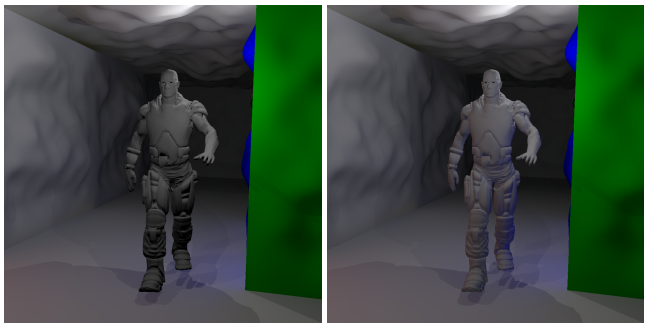


Figure 5: *Indirect transport on dynamic objects (character mesh) without (**left**) and with (**right**) volume samples.* © *2011 The Authors.*

Most of our example blocks have 16^2 samples per face. We evaluate $\mathbf{M}$ by ray tracing the scaled prior vectors (columns of $\mathbf{P}$) using 16K gather rays with importance sampling. We perform this at each of the 16^2 points on the face, from which $\mathbf{U_b}$, $\mathbf{L}_{\mathrm{imp}}$ and $\mathbf{T_{d\to b}}$ are computed. The cylinder dictionary uses 64^2 samples per shape. We also compute several lower resolution dictionaries using only scalar surface response for the iPad and x86 CPU implementations. Using fewer than 6^2 samples results in objectionable artifacts.

The $\mathbf{U}_{\overrightarrow{\mathbf{b}}}$, $\mathbf{U}_{\mathrm{r}}$, and $\mathbf{U}_{\mathbf{b}\mathrm{vol}}$ are computed by ray tracing implicit lights (columns of $\mathbf{L}_{\mathrm{imp}}$) and computing their respective output data. $\mathbf{U}_{\overrightarrow{\mathbf{b}}}$, and $\mathbf{U}_{\mathbf{b}\mathrm{vol}}$ are computed once per dictionary shape, and the interface data is computed for only a single canonical interface.

To compute $\mathbf{H}_{\mathrm{lf}}$ from $\mathbf{L}_{\mathrm{imp}}$ we use 5^2 directional super-sampling of interfaces. Three iterations of enrichment are applied to $\mathbf{H}_{\mathrm{rlf}}$, retaining $k_{\mathrm{rlf}} = 128$ lightfield modes and $k_{\mathrm{r}} = 32$ response modes.

We use raw interface resolutions of $s = 12^2$ spatial $\times$ 24^2 directional $= 82944$ total samples, mapping the hemisphere to a square [Shirley and Chiu 1997] for continuous directional interpolation.

Precomputation. The dictionary (without interfaces) requires 4.7MB, takes 59s to build, and 60s to enrich. Interfaces increase precomputation by 260s, adding 1.2MB of memory. Timings are on a dual 2.93GHz 6-core Intel CPU. Interface computation does not scale well with the 12 cores, but the other stages scale linearly.

7.2 Mapping Shapes to Complex Scenes

We use two approaches to compute texture coordinates (and position/normal geometry images) in complex scenes. For the game scene (Figure 7), artists map blocks with our level editor: a cube is warped to roughly align with part of the scene, then additional cubes are extruded/warped from existing ones (see video for a modeling session). The artist can preview indirect light directly in the tool. $\mathbf{T_{b\to r}}$ is also computed at this phase and takes a second at most.

As an extreme stress test, we model the Great Hall scene (Figure 6) using only a single block. The vertices are mapped to block faces using rasterization: a cube map camera at the center of the block renders a downsampled "geometry texture" which allows us to categorize the vertices belonging to each face. The texture coordinates for a given vertex are computed by orthographically projecting the vertex onto the chosen face, and these coordinates are used to lookup indirect light in the padded light-map texture atlas.

We create records, for each block's face, to render indirect light ($\mathbf{l}_{\mathrm{ind}} \approx \mathbf{U_b}\ \mathbf{b}$) into an (un-padded) light map. Each record is

Figure 7: *An 18 block scene from a video game: direct light (**left**, 1.2ms), direct and indirect (**right**, 2.2ms).*

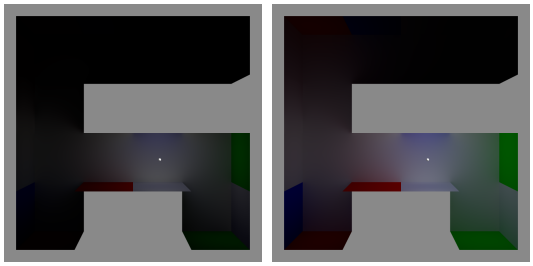


Figure 8: *Feeding back the previous iteration's indirect light buffer into the direct light buffer generates multiple bounces.*

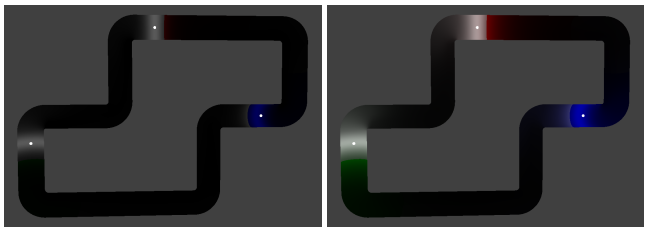


Figure 9: *Tunnel created with 10 cylinders. Direct lighting (**left**), direct and multi-bounce indirect lighting (**right**, 5.2 ms = 192 FPS).*

positioned appropriately in the light map and contains $\mathbf{U_b}$ texture coordinates and a block index for the face to look up $\mathbf{b}$ coefficients.

7.3 Simple Run-time Implementation

Our run-time is quite simple. An indirect light texture, using direct light evaluated at n surface points, is computed as follows:

1. Compute direct light at a reduced resolution in each block ($\mathbf{l}_\mathrm{d}$),
2. Compute per-block spectral coefficients ($\mathbf{b} = \mathbf{T_{d\to b}}\ \mathbf{l}_\mathrm{d}$),
3. Compute indirect light *within* a block into a lightmap ($\mathbf{U_b}\ \mathbf{b}$),
4. Compute response coefficients at interfaces ($\mathbf{r} = \mathbf{T_{b\to r}}\ \mathbf{b}$),
5. Blend response from "external" blocks into a lightmap ($\mathbf{U_r}\ \mathbf{r}$),
6. Create padded lightmap texture to eliminate texture seams,
7. Render scene using the dynamic lightmaps of indirect lighting,
8. **[optional]** Compute indirect light volume (from $\mathbf{b}$'s and $\mathbf{r}$'s),
9. **[optional]** Render dynamic objects with volume lighting.

As in Enlighten [Martin and Einarsson 2010] it is easy to feedback indirect light (scaled by albedo) to get multiple bounces with little overhead (Figures 8 and 9). Figure 1 compares our approximate multi-bounce indirect light (476 FPS) to ground truth (many hours).

7.4 Results

GPU performance was recorded on an NVIDIA 480 GTX with a DX11 runtime. Rendering of omnidirectional shadow maps for direct lighting is by far the largest bottleneck of our run-time renderer for a moderate number of blocks. Our indirect lighting performance scales linearly with the number of blocks (see Figure 10) with the most expensive stage being computing $\mathbf{b}$'s. This could potentially be optimized by using a single pass reduction or compute shader.

We also have two software implementations, one for x86 CPU's using SSE and one for iPad/iPhone using NEON instructions. The CPU versions support everything but volume samples and vector

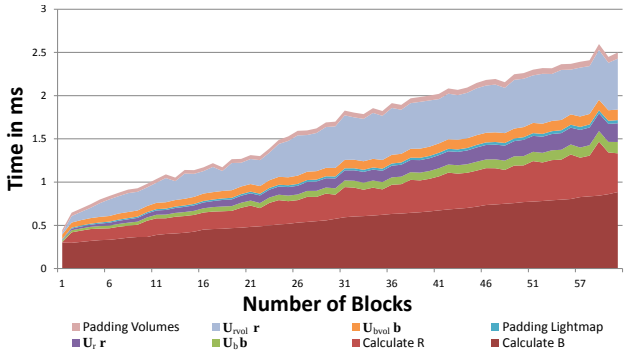


Figure 10: *Average performance taken from many random mazes.*

irradiance. We measure performance for two scenes on the CPU code paths in Table 3. For comparison one unshadowed point light takes 25 ms on the iPad, making VPL techniques difficult.

	Small Maze	Large Maze
# blocks	11	41
iPad ms	2.0	12.9
x86 ms	1.4	6.2
GPU ms	0.4	1.0

Table 3: *Performance results. The iPad uses 6^2 textures and 12 modes. The x86 and GPU use 16^2 textures and 32 modes.*

8 Discussion

Instead of computing ground truth indirect effects, we target large-scale indirect lighting that responds to dynamic direct illumination, can be properly applied to fine scale surface detail, and supports animated character meshes. Our very high-performance allows our run-time to be easily integrated across many hardware platforms, as well as significantly reducing iteration time (see accompanying video); lighting artists get **instant** indirect lighting feedback that is guaranteed to **match** the in-game rendering.

Limitations. When mapping scenes to a single box, using only the first mode is akin to an "intelligent ambient term" that is computed by averaging the direct light in the scene. Higher order terms add spatial variation. The bottom row of Figure 6 illustrates that even in the extreme case of a single cube mapped to complex geometry, our approach "fails gracefully", with smooth and plausible results. Of course, using a more accurate proxy shape would produce better results. With a single cube, light propagates too far along the longest axis of the scene's bounding box; we could alleviate this with a prior that uses the aspect ratio of the scene's bounding box, but almost equivalent results can be created by only re-raytracing the $\mathbf{U}$ vectors with an appropriately stretched prior (Figure 12), taking only seconds to "re-precompute". Our algorithm is designed for scenarios where a small number of coarse proxies are mapped to complex geometry (like bounce cards in film production); using many proxy shapes to model this scene would defeat our purpose.

Another limitation stems from performing all computation in reduced spaces: lighting conditions outside our prior cannot be captured. Figure 11 highlights this issue, where a light is positioned too close to a wall. Shadows, strong albedo changes, and direct lighting used in video game scenes are outside of our prior, but plausible results are generated in these cases using enrichment (Appendix A).

We choose a simple prior, trading off accuracy for flexibility and performance. A more complex prior can be constructed using e.g.

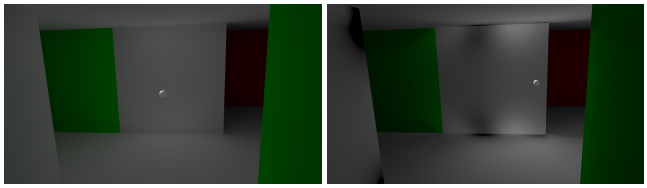


Figure 11: *Indirect light from a well-represented (**left**) and poorly-represented (**right**) direct-illumination pattern in the light prior.*

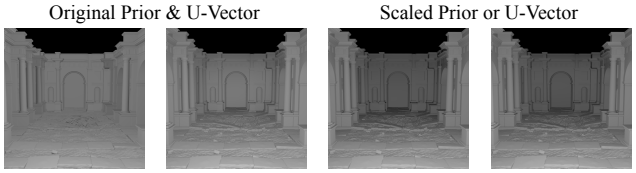


Figure 12: *Great Hall mapped to 1 cube is better than using an ambient term, but causes unrealistic lighting on the longest axis (**1 Mode / 32 Modes**). Re-raytracing* $\mathbf{U}$ *(**U-Vector Scaled**) with the cube prior stretched to the bounding box achieves results similar to recomputing the entire prior (**Both Scaled**).*

permutable blockers in a scene and different light sources, but this would require a higher-dimensional representation. As an extreme example, we could use the scene from Figure 12 as a shape library element, yielding a lighting prior that would capture higher-fidelity results for that particular scene (but for no other); however, especially when combined with direct illumination, it is clear that the additional fidelity is not worth the decrease in performance and the elimination of modularity/generality.

Our light prior bases are small, dense matrices derived using the SVD, in contrast to multiresolution bases such as wavelets which are typically represented as large sparse matrices. Our dense matrices map more effectively to texture mapping hardware, have more coherent access patterns, and require less overall memory than wavelets, which is especially important on low-power handheld devices. This is particularly true with the 12 to 32 coefficients we use in practice.

Non-diffuse Transport. Scalar (diffuse) response can be substituted with a directional radiance representation (e.g. using a basis representation such as SH or wavelets) our transport operators can also be augmented to support non-diffuse light transport. We note that low-frequency glossy reflectance models can still be applied to e.g. our SH representation of volumetric light. We have instead targeted flexible and rapid scene development with physically-plausible diffuse indirect illumination, as illustrated in our examples. Extending the simplicity of our approach to *high-frequency* glossy transport, while keeping the benefits (compactness, high-performance, plausibility) is a challenging problem left to future work. Such directional radiance transport modeling may also prove useful for handling fine-scale indirect shadows, as we discuss below.

Indirect Shadows. Our approach models coarse-scale indirect shadows (e.g. around maze corners) while ignoring large-scale indirect shadowing effects within blocks. While this is sometimes problematic (e.g. the Cornell box in Figure 6), when direct lighting is included the lack of exact indirect shadows is often difficult to notice (see Figure 1). In fact, in film production, shadow computation for the fill lights used to model indirect light is often disabled.

In the future, we plan on investigating techniques to include fine-scale indirect shadows from objects and clutter geometry *within* blocks, by applying light subtraction ideas from antiradiance [Dachsbacher et al. 2007], as well as dynamic blocker accumulation and reflection from [Sloan et al. 2007].

Dictionary Shapes. The restriction to basic shapes allows us to eliminate scene-dependent transport precomputation. To support modularity, we must restrict the generality of the shape library. This constraint is softened during authoring by combining light transport coupling, warping, volume samples, and support for normal variation. It is possible to use more complicated shapes and extend our lighting prior to support e.g. internal occluders (see above).

Alternative Real-time Indirect Approaches. We achieve much higher performance than other approximate indirect lighting solutions, and our approach can readily be used in high-end and mobile gaming applications. Moreover, MRT scales favorably with respect to both the number of direct lights and the number of indirect bounces. However, these performance gains are only a by-product of the more substantial, novel contributions of our work: the lighting prior, transport computation in reduced spaces, and modularity (which also eliminates expensive scene-dependent transport precomputation). We feel that these ideas can be applied more generally to other areas in offline and real-time rendering, including other approximate indirect illumination techniques (e.g. instant radiosity).

9 Conclusions and Future Work

We target plausible, large-scale, and soft indirect lighting with rapid content generation, and sit between a simple ambient term and more accurate and costly simulations. We leverage a novel approach to direct-to-indirect transport, using *scene-independent* transport warping and coupling. Reduced-dimensional operators act on simple shapes which are, in turn, warped and coupled to approximate transport in more complex scenes with extremely high-performance. Volumetric and vector-valued extensions model transport with high-frequency surface details and onto dynamic objects.

Blocks in our example dictionaries are cubes with one or more faces removed or cylindrical shapes. In order to support scenes such as office hallways with doors and/or windows, we would need to model blocks with "holes" cut into their faces. One option would be to subtract the response of the door/window from the implicit light coefficients of the room.

We incorporated our mapping into an internal level editing tool, producing promising results (see video). Investigating semi-automated and art-directed mapping control are interesting areas of future work.

Acknowledgements

The authors acknowledge the late Marek Romanowski for very fruitful discussions in the developing stages of the project. Ladislav Kavan and Peter Shirley also provided valuable feedback. We also thank Stephen Duck and the MIT Computer Graphics Group for allowing us to use the Great Hall scene.

References

ASHDOWN, I. 2001. *Eigenvector Radiosity*. Master's thesis, Department of Computer Science, University of British Columbia.

BAVOIL, L., SAINZ, M., AND DIMITROV, R. 2008. Image-space horizon-based ambient occlusion. In *SIGGRAPH talks*, ACM, New York.

CHEN, H. 2008. Lighting and Materials of Halo 3. In *Game Developers Conference*.

DACHSBACHER, C., AND STAMMINGER, M. 2005. Reflective shadow maps. In *ACM Symposium on Interactive 3D Graphics and Games*.

DACHSBACHER, C., AND STAMMINGER, M. 2006. Splatting indirect illumination. In *ACM Symposium on Intearactive 3D Graphics and Games*.

DACHSBACHER, C., STAMMINGER, M., DRETTAKIS, G., AND DURAND, F. 2007. Implicit visibility and antiradiance for interactive global illumination. *ACM Trans. Graph. 26*, 3.

GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. F. 1996. The lumigraph. In *SIGGRAPH*.

GREGER, G., SHIRLEY, P., HUBBARD, P. M., AND GREENBERG, D. P. 1998. The irradiance volume. *IEEE Computer Graphics & Applications*.

GUERRERO, P., JESCHKE, S., AND WIMMER, M. 2008. Real-time indirect illumination and soft shadows in dynamic scenes using spherical lights. In *Computer Graphics Forum*, vol. 27, 2154–2168.

HABEL, R., AND WIMMER, M. 2010. Efficient irradiance normal mapping. In *ACM Symposium on Interactive 3D Graphics and Games*.

HAŠAN, M., PELLACINI, F., AND BALA, K. 2006. Direct-to-indirect transfer for cinematic relighting. *ACM Trans. Graph. 25*, 3.

IWASAKI, K., DOBASHI, Y., YOSHIMOTO, F., AND NISHITA, T. 2007. Precomputed Radiance Transfer for Dynamic Scenes Taking into Account Light Interreflection . *Computer Graphics Forum*, 35–44.

KAPLANYAN, A., AND DACHSBACHER, C. 2010. Cascaded light propagation volumes for real-time indirect illumination. In *ACM Symposium on Interactive 3D Graphics and Games*.

KELLER, A. 1997. Instant radiosity. In *SIGGRAPH*.

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

KRISTENSEN, A. W., AKENINE-MÖLLER, T., AND JENSEN, H. W. 2005. Precomputed local radiance transfer for real-time lighting design. *ACM Trans. Graph. 24*, 3.

LARSSON, D., AND HALEN, H. 2009. The unique lighting of Mirror's Edge. In *Game Developers Conference*.

LEHTINEN, J., ZWICKER, M., TURQUIN, E., KONTKANEN, J., DURAND, F., SILLION, F. X., AND AILA, T. 2008. A meshless hierarchical representation for light transport. *ACM Trans. Graph. 27*, 3.

LEHTINEN, J. 2007. A framework for precomputed and captured light transport. *ACM Trans. Graph. 26*, 4.

LEVOY, M., AND HANRAHAN, P. 1996. Light field rendering. In *SIGGRAPH*.

LEWIS, R. R., AND FOURNIER, A. 1996. Light-driven global illumination with a wavelet representation of light transport. In *Rendering Techniques*.

LOOS, B., ANTANI, L., MITCHELL, K., NOWROUZEZAHRAI, D., JAROSZ, W., AND SLOAN, P.-P. 2011. Run-time implementation of modular radiance transfer. In *SIGGRAPH talks*, ACM, NY.

MARTIN, S., AND EINARSSON, P., 2010. A real-time radiosity architecture for video games. SIGGRAPH 2010 Course: Advances in Real-Time Rendering in 3D Graphics and Games.

MCTAGGART, G. 2004. Half-Life 2 source shading. In *Game Developers Conference*.

MEYER, M., AND ANDERSON, J. 2006. Statistical acceleration for animated global illumination. *ACM Trans. Graph. 25*, 3.

MITTRING, M. 2007. Finding next gen: Cryengine 2. In *SIGGRAPH courses*, ACM, New York, 97–121.

NICHOLS, G., AND WYMAN, C. 2009. Multiresolution splatting for indirect illumination. In *ACM Symposium on Interactive 3D Graphics and Games*.

NICHOLS, G., SHOPF, J., AND WYMAN, C. 2009. Hierarchical image-space radiosity for interactive global illumination. *Computer Graphics Forum 28*, 4.

NOWROUZEZAHRAI, D., AND SNYDER, J. 2009. Fast global illumination of dynamic height fields. *Computer Graphics Forum 28*, 4.

PARKER, S., MARTIN, W., SLOAN, P.-P. J., SHIRLEY, P., SMITS, B., AND HANSEN, C. 1999. Interactive ray tracing. In *ACM Symposium on Interactive 3D Graphics*.

RAMAMOORTHI, R. 2009. Precomputation-based rendering. *Foundations and Trends in Computer Graphics and Vision 3*, 4.

REN, Z., WANG, R., SNYDER, J., ZHOU, K., LIU, X., SUN, B., SLOAN, P.-P., BAO, H., PENG, Q., AND GUO, B. 2006. Real-time soft shadows in dynamic scenes using spherical harmonic exponentiation. *ACM Trans. Graph. 25*, 3 (July), 977–986.

RITSCHEL, T., GROSCH, T., KIM, M. H., SEIDEL, H.-P., DACHSBACHER, C., AND KAUTZ, J. 2008. Imperfect shadow maps for efficient computation of indirect illumination. *ACM Trans. Graph.*.

SHIRLEY, P., AND CHIU, K. 1997. A low distortion map between disk and square. *Journal of Graphics Tools*.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. 21*, 3.

SLOAN, P.-P., GOVINDARAJU, N. K., NOWROUZEZAHRAI, D., AND SNYDER, J. 2007. Image-based proxy accumulation for real-time soft global illumination. In *Pacific Graphics*, IEEE.

WANG, R., ZHU, J., AND HUMPHREYS, G. 2007. Precomputed Radiance Transfer for Real-time Indirect Lighting using a Spectral Mesh Basis. *Computer Graphics Forum*, 13–21.

WANG, R., WANG, R., ZHOU, K., PAN, M., AND BAO, H. 2009. An efficient gpu-based approach for interactive global illumination. *ACM Trans. Graph. 28*, 3.

WICKE, M., STANTON, M., AND TREUILLE, A. 2009. Modular bases for fluid dynamics. *ACM Trans. Graph. 28*, 3.

XU, H., PENG, Q.-S., AND LIANG, Y.-D. 1990. Accelerated radiosity method for complex environments. *Computers and Graphics*, 65 – 71.

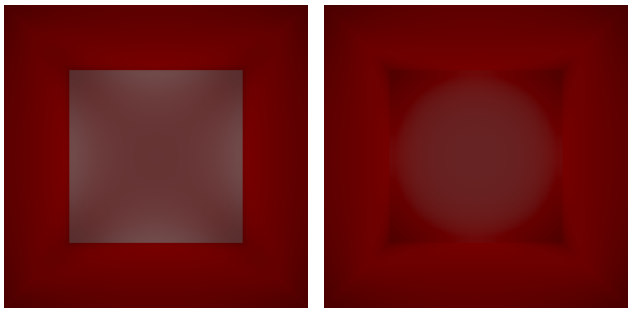
ZHUKOV, S., INOES, A., AND KRONIN, G. 1998. An ambient light illumination model. In *Rendering Techniques*, Springer-Verlag.

A Basis Enrichment

When constructing the direct light ($\mathbf{P}$) or interface ($\mathbf{H}_{\mathrm{rlf}}$) priors, some functions are not well represented in the reduced space: e.g., direct light with shadows cast by objects not present during the computation of $\{\mathbf{l}_{\mathrm{d}0}, \ldots, \mathbf{l}_{\mathrm{d}m}\}$, or indirect light $\mathbf{U}_{\mathfrak{b}}$ vectors for multiple bounces. To model targeted effects, we can *enrich* our basis.

Given bases stored in columns of matrices, we first re-scale the matrices to have a prescribed Frobenius norm, then concatenate the matrices into a larger matrix with the same number of rows and compute the SVD, retaining the left singular vectors and singular values (optionally). This new prior better spans all sub-spaces.

To generate a new prior for blocks we equally weight the scaled prior $\mathbf{P}\,\mathbf{S}$ from Section 4, fall back basis functions that are constant on one face and zero on others to approximate changes from shadows/albedo, and indirect lighting represented by $\mathbf{U_b}$.



Without Basis Enrichment | With Basis Enrichment

Figure 13: *A box has 5 red and one white face. Without enrichment, red faces bounce all colors onto the white face, which is incorrect.*

For interfaces, lightfield-to-lightfield operators $\{\mathbf{R}_{\nwarrow}, \mathbf{R}_{\uparrow}, \mathbf{R}_{\nearrow}\}$ can generate output outside of $\mathbf{H}_{\mathrm{rlf}}$'s space. In this case, we compute the reduced basis response through these operators, yielding a temporary basis $\mathbf{L}_{\mathrm{temp}}$ and balance that with $\mathbf{H}_{\mathrm{rlf}}$ generating a new basis $\bar{\mathbf{H}}_{\mathrm{rlf}}$. The new basis better captures functions from both spaces; repeated enrichment is used to handle multiple propagation steps (or bounces.)

B Quadratic SH to Hemispherical Linear SH

We present an alternative to vector irradiance: a closed-form solution for the optimal hemispherical projection of *quadratic* irradiance back into a linear (vector) model. This effectively models the higher-frequency energy that "bleeds" into the linear band after clamping reflectance response to the upper-hemisphere. We solve for linear-SH coefficients that minimize the difference, over the hemisphere, between quadratic (L_2) and linear (L_1) expansions of radiance,

$$\begin{aligned} E &= \int_{\mathcal{H}} \left[L_2(\omega) - L_1(\omega)\right]^2 d\omega \\ &= \int_{\mathcal{H}} \left[\sum_{j=1}^{9} b_j\, y_j(\omega) - \sum_{i=1}^{4} a_i\, y_i(\omega)\right]^2 d\omega, \end{aligned}$$

where $\mathcal{H}$ is the hemispherical domain, a_i (unknowns) and b_j (knowns) are order-2 and order-3 SH coefficients, and we single index the SH basis functions, $y_k(\omega)$. Setting the derivative of error with respect to the unknowns to zero and solving yields

$$\begin{aligned} \frac{dE}{da_k} &= 2\int_{H^2} \left[L_2(\omega) - L_1(\omega)\right] \frac{dL_1(\omega)}{da_k}\, d\omega = 0 \\ 0 &= 2\sum_{j=1}^{9} b_j \int_{\mathcal{H}} y_j(\omega) y_k(\omega) d\omega - 2\sum_{i=1}^{4} a_i \int_{\mathcal{H}} y_i(\omega) y_k(\omega) d\omega\,, \end{aligned}$$

which we express in matrix form: $\mathbf{H_1}\,\mathbf{a} = \mathbf{H_2}\,\mathbf{b}$, where the 4×4 matrix $[\mathbf{H_1}]_{ik} = \int_{\mathcal{H}} y_i(\omega) y_k(\omega) d\omega$ and the 4×9 matrix $[\mathbf{H_2}]_{jk} = \int_{\mathcal{H}} y_j(\omega) y_k(\omega) d\omega$. The top 4×4 block of $\mathbf{H_2}$ is $\mathbf{H_1}$, and an analytic expression for $\mathbf{a}$ is:

$$\begin{aligned} &\mathbf{a} = {\mathbf{H_1}}^{-1}\mathbf{H_2}\mathbf{b} = \{b_1 - c_1 b_7, b_2 + c_2 b_6, b_3 + c_3 b_7, b_4 + c_2 b_8\}\,, \\ &\text{where } c_1 = 3\sqrt{5}/4, c_2 = 3\sqrt{5}/8, c_3 = \sqrt{15}/2. \end{aligned}$$

The H4 basis [Habel and Wimmer 2010] and OHLSH span the same space.