Direct-to-Indirect Acoustic Radiance Transfer

Lakulish Antani, Anish Chandak, Micah Taylor and Dinesh Manocha

Abstract—We present an efficient algorithm for simulating diffuse reflections of sound in a static scene. Our approach is built on recent advances in precomputed light transport techniques for visual rendering and uses them to develop an improved acoustic radiance transfer technique. We precompute a direct-to-indirect acoustic transfer operator for a scene, and use it to map direct sound incident on the surfaces of the scene to multi-bounce diffuse indirect sound, which is gathered at the listener to compute the final impulse response. Our algorithm decouples the transfer operator from the source position so we can efficiently update the acoustic response at the listener when the source moves. We highlight its performance on various benchmarks and observe significant speedups over prior methods based on acoustic radiance transfer.

Index Terms—sound propagation, radiosity, virtual reality, precomputed transport

1 Introduction

Sound rendering can augment visual rendering and provide an enhanced spatial sense of presence. Some of the driving applications of sound rendering include video games, architectural acoustics and VR simulations.

The modeling of sound propagation effects needs to account for different wave propagation phenomena such as specular reflections, diffuse reflections, edge diffraction and interference. In this paper, we focus on modeling diffuse reflections. Many objective [1], [2] and perceptual [3] studies have demonstrated the importance of diffuse reflections in sound propagation. Further, it is computationally challenging to model high orders of diffuse reflection. Hence, modeling diffuse reflections for sound propagation is an active area of interest in many interactive sound rendering applications.

Sound propagation algorithms can be broadly classified into wave-based and geometric methods. Wave-based methods numerically solve the acoustic wave equation. However, their complexity is proportional to the volume of the scene and the fourth power of the maximum frequency of sound simulated, therefore they can be very slow for large acoustic spaces or high frequency sound sources. Geometric methods approximate sound waves by rays. Two standard methods used to simulate diffuse sound reflections are based on ray (or volume) tracing and radiance transfer. Our approach is motivated by recent developments in global illumination based on precomputed light transport algorithms [4], [5], [6]. Specifically, our work is based on direct-to-indirect transfer algorithms for visual rendering, which map direct light incident on the surfaces of a scene to indirect light on the surfaces of the scene after multiple bounces.

Main Results We present a new algorithm for modeling diffuse reflections of sound based on the direct-to-indirect transfer approach. The algorithm computes an acoustic transfer operator in matrix form which is decoupled from both the source and the listener positions, and can efficiently update the acoustic response at the listener whenever the source moves.

The algorithm approximates the transfer matrix using the singular value decomposition (SVD) to perform higher-order diffuse reflections. We show that this approximation reduces the memory requirements and increases the performance of our algorithm.

We highlight the performance of our algorithm on various models. In practice, it is much faster than prior methods based on ra-

• *L. Antani, A. Chandak, M. Taylor and D. Manocha are with the Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599. E-mail: {lakulish,achandak,taylormt,dm}@cs.unc.edu*

diance transfer. To the best of our knowledge, it is the first approach that can perform high orders of diffuse reflections in static scenes with moving sources and listeners at almost interactive rates.

The rest of this paper is organized as follows. We briefly survey related work in Section 2, and discuss mathematical prerequisites in Section 3. We present our algorithm in Section 4 and discuss implementation details in Section 5. We present experimental results in Section 6 and conclude in Section 7.

2 Related Work

Wave-based Acoustics The propagation of sound in a medium is described by the acoustic wave equation, a second-order partial differential equation [7]. Several techniques are known for numerically solving the wave equation [8], [9] and accurately modeling sound propagation in a scene. Diffuse reflections can be modeled by performing the simulation on a grid fine enough to capture the detailed "roughness" of the surfaces that results in acoustic wave scattering [10]. However, despite recent advances [9], these methods can be rather slow and are mainly limited to scenes with static sources. Precomputation-based methods have recently been developed [11] that use a numerical wave equation solver to compute the acoustic response of a scene from several sampled source positions; at run-time these responses are interpolated given the actual position of a moving source. These methods are fast, but require large amounts of precomputed data.

Geometric Acoustics Most interactive systems model sound waves in terms of rays or 3D volumes. These geometric acoustics techniques cannot accurately solve the wave equation, and cannot easily model all kinds of propagation effects, but allow efficient simulation of early reflections. However, geometric techniques have trouble handling some acoustic phenomena such as finite-size diffracting edges, or absorbers with complex boundary conditions.

Methods based on ray tracing [12], [13] can model both diffuse and specular reflections of sound. Since early specular reflections provide the listener with important directional cues, specialized techniques have been developed for modeling specular reflections, such as volume tracing [14], [15] and the image source method [16], [17]. For static scenes, which frequently arise in architectural acoustics and virtual environments, algorithms based on acoustic radiosity [18], [19], [20] or radiance transfer methods can be used to model reflections from surfaces with arbitrary bidirectional reflectance distribution functions (BRDFs) [21], [22]. Many techniques have also been designed to model edge diffraction [23], [24], [25].

Precomputed Light Transport Radiosity [26] is the classic precomputed light transport algorithm. However, it computes a full solution that has to be recomputed whenever the light source moves. In contrast, *precomputed radiance transfer* (PRT) algorithms decouple light transport effects from the light source configuration by computing a linear operator that defines how a variable light source configuration affects the radiances at surface sample points. PRT techniques can support both distant [4], [27] and local [28] source configurations.

Direct-to-indirect transfer algorithms [5], [6] are one class of precomputed light transport algorithms. These algorithms compute linear operators which map direct light incident on the surface samples to multi-bounce indirect light at the samples. They are designed to handle diffuse reflections, and some of them can also support limited glossy reflections. Our approach is based on applying these ideas to sound propagation.

3 Preliminaries

This section briefly describes the mathematical background on which our algorithm is based.

3.1 Sound Rendering vs. Visual Rendering

Light transport simulation is concerned with the *steady-state* values of radiance over the surface of the scene, since light travels fast enough (3×10^8 m/s) that transient radiance values are not observed and can be ignored. However, the speed of sound in air is much slower (340 m/s), and hence it is important to compute *time-varying* radiances over the surface.

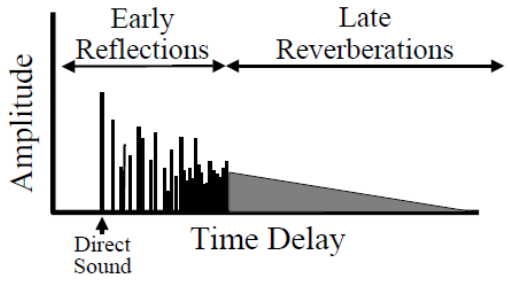


Fig. 1. Parts of a typical impulse response [30] (absolute values shown). The early response is dominated by specular reflections and diffraction; the late reverberation is dominated by diffuse reflections [29].

Furthermore, sound waves have much larger wavelengths than light waves, and are comparable in size to obstacles in typical architectural and game scenes. Therefore, diffraction plays an important role in sound propagation, and it must be modeled in order to generate plausible sounds.

The computation of sound propagation paths takes into account the knowledge of sound sources, listener locations, the 3D model of the environment, material absorption and scattering properties, and air absorption. Given the signal emitted by a sound source (i.e., a time-varying pressure wave), the signal heard by a listener (after modeling reflections, diffractions and interference) is typically computed using *impulse responses* (IRs). An IR is the signal received at the listener if the source emits a unit impulse at $t = 0$. Since room acoustics are modeled as a linear time-invariant system [29], the response at the listener to an arbitrary source sound can be computed by convolving the source sound with the IR.

IRs can be divided into three parts [30]: (a) *direct*, sound arriving directly from the source; (b) *early response*, sound reaching the listener soon after the direct sound, via a small number of reflections or diffractions; and (c) *late reverberation*, the gradual decay of amplitude after the early response (see Figure 1). Late reverberation gives an impression of the size of an acoustic space, and is mostly composed of diffuse reflections [29], therefore in this paper we focus on computing higher-order diffuse reflections; our approach can be used in tandem with existing approaches for modeling specular reflections and edge diffraction.

3.2 Impulse Response Representation

The IR at a point is a function of time. Attenuation and delay can be applied using a unified formulation [22] by representing our IRs in Fourier space. For a continuous function $f : [0, T] \rightarrow \mathbb{R}$, the Fourier transform projects f into a sinusoidal basis (with basis vectors of the form $e^{\iota\omega_i t}$). For discrete signals, we compute the Discrete Fourier Transform (DFT) using the Fast Fourier Transform (FFT) algorithm. The signal is discretized into N time-domain samples, where the value of N is chosen based on the desired audio sampling frequency and the length of the IR modeled (which could be tuned based on the expected reverberation time of a room).

Since the Fourier transform is linear, attenuations and accumulation of IRs can be performed easily (n denotes a discrete sample index):

$$\mathcal{F}(af_1(n) + bf_2(n)) = a\mathcal{F}(f_1(n)) + b\mathcal{F}(f_2(n)). \tag{1}$$

Unlike in the time domain, in the frequency domain delays can also be applied using a scale factor, since the Fourier basis vectors are eigenvectors of linear time-invariant operators:

$$\mathcal{F}(f(n - \Delta n)) = e^{-\iota\omega\Delta n}\mathcal{F}(f(n)). \tag{2}$$

Note that care must be taken to ensure that the delays align on time-domain sample boundaries, otherwise the inverse Fourier transform will contain non-zero imaginary parts. An alternative solution to this issue would be to use fractional delays [31]; we choose to address this in future work in the interests of simplicity of implementation.

A unit impulse emitted by the source at time $t = 0$ has all Fourier coefficients set to 1. Computing the IR using the above expressions for delay and attenuation results in a frequency-domain signal. Computing the inverse Fourier transform of this signal using the frequency-domain replication method described by Siltanen et al. [22] yields a periodic function which is approximately equal to the time-domain IR

at the listener. Note that this method does not compute the *steady-state* acoustic response, but the *time-varying* impulse response. The key to this is the frequency-domain delay equations described above.

3.3 Acoustic Rendering Equation

The propagation of sound in a scene can be modeled using an extension of the standard graphics rendering equation [32], called the *acoustic rendering equation* [21]:

$$L(x,\Omega) = L_0(x,\Omega) + \int_S R(x,x',\Omega) L\left(x', \frac{x-x'}{|x-x'|}\right) dx' \quad (3)$$

where L is the total outgoing radiance, L_0 is the emitted radiance and R is the *reflection kernel*, which describes how radiance at point x' influences radiance at point x. Ω is the exitant radiance direction at x; the incident radiance direction at x is implicit in the specification of x':

$$R(x,x',\Omega) = \rho(x,x',\Omega) G(x,x') V(x,x') P(x,x'). \quad (4)$$

Here, ρ is the BRDF of the surface at x, G is the form factor between x and x', V is the point-to-point visibility function, and P is a *propagation term* [21] that accounts for propagation delays (as per Equation 2).

The radiances in Equation 3 are IRs; the time variable t is hidden for the sake of brevity. This added dimension of time complicates the storage and processing requirements of algorithms based on the acoustic rendering equation.

4 Algorithm

Our algorithm provides two main improvements over the state-of-the-art acoustic radiance transfer algorithms: (a) we decouple the source position from the precomputed data by computing an *acoustic transfer operator* as opposed to simply precomputing the IRs at surface samples due to a sound source as per the method of Siltanen et al. [22]; and (b) we use the SVD to compress the transfer operator and quickly compute higher-order reflections. The rest of this section details how our algorithm achieves these improvements over the state-of-the-art.

Our overall approach is as follows (see Figure 2 and Algorithms 1 and 2):

- **Preprocessing.** We sample the surface of the scene and compute a transfer operator which models one or more orders of diffuse reflections of sound among the surface samples.
- **Run-time.** First, we shoot rays from the source to determine the direct IR at each surface sample. Next, we apply the transfer operator to the direct response to obtain the indirect response. Finally, we shoot rays from the listener and gather the direct and indirect responses from each surface sample hit by a ray. These are added to obtain the final IR at the listener.

Algorithm 1 Preprocessing

```
P ← set of samples on scene surface
p_i denotes the i^th element of P
T ← 0
for all i ∈ [0, |P| − 1] do
  for all l ∈ [0, N_rays] do
    r ← random path traced from p_i
    p_j ← final sample hit by r
    T_ij += IR contribution along r
  end for
end for
```

Algorithm 2 Run-time

```
l_0 ← direct IR from source at each sample
l_n ← T · l_0
IR ← gather from (l_0 + l_n)
```

4.1 Acoustic Transfer Operator

The acoustic transfer operator is expressed over a set of p samples chosen over the surface of the scene. The transfer operator is computed in terms of the responses at all surface samples to impulses emitted from every other surface sample. We use Fourier coefficients to represent the sample-to-sample IRs. Let there be f Fourier coefficients per surface sample. All subsequent computations are performed on each Fourier coefficient independently.

For each frequency ω_m, we define the acoustic radiance vector $\mathbf{l}(\omega_m)$, which contains p elements that represent the m^{th} Fourier coefficients of the IRs at each surface sample.

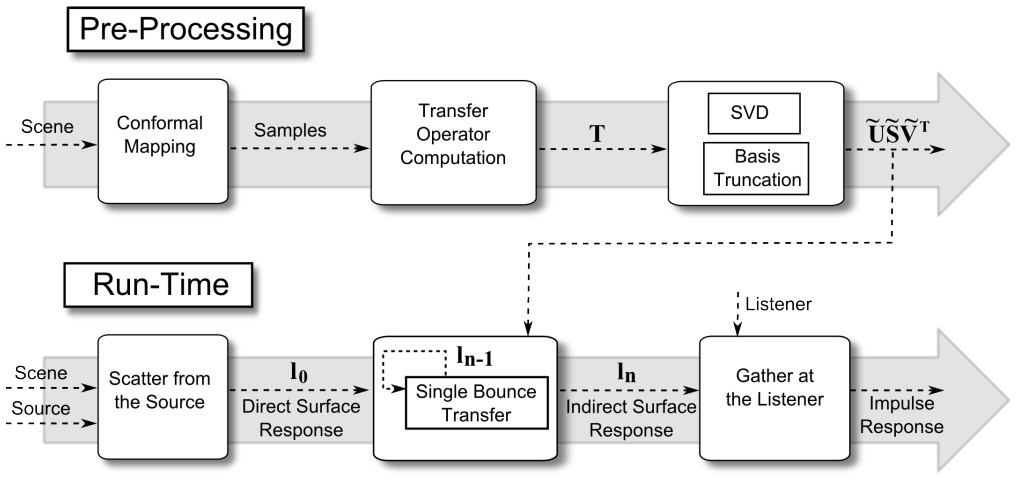


Fig. 2. Overview of our algorithm. In a precomputation step, we sample the surfaces on the scene, and compute a one-bounce transfer operator for these samples ($\mathbf{T}$). We then use the SVD to compute the modes of the transfer operator. At runtime, we shoot rays from the source (which may move freely) and compute direct IRs at the surface samples. We then apply the transfer operator (with a user-specified number of modes retained) repeatedly to quickly obtain the multi-bounce indirect IRs at the surface samples. We then compute the final IR at the listener position in a final gathering step.

For the sake of brevity, we shall omit the parameter ω_m from the equations in the rest of the paper as it may be obvious from the context.

The Neumann series expansion of Equation 3 expressed in matrix form is:

$$\mathbf{l}_{n+1}(\omega_m) = \mathbf{T}(\omega_m)\mathbf{l}_n(\omega_m), \tag{5}$$

where $\mathbf{l}_n(\omega_m)$ contains the m^{th} Fourier coefficients of the IRs at each surface sample after n reflections. The *transfer matrix* $\mathbf{T}(\omega_m)$ models the effect of one diffuse reflection. The $(i,j)^{th}$ element of $\mathbf{T}(\omega_m)$ describes how the m^{th} Fourier coefficient at surface sample j affects the m^{th} Fourier coefficient at surface sample i after one diffuse reflection. The entries in row i of $\mathbf{T}$ are computed by tracing paths sampled over the hemisphere at surface sample i; the delays and attenuations along each path terminating at any other surface sample j are added to the entry $\mathbf{T}_{ij}$ [22]. We can compute a *multi-bounce transfer operator* with n orders of reflection as the matrix sum $\mathbf{T}_n = \mathbf{T} + \mathbf{T}^2 + \cdots + \mathbf{T}^n$.

Existing acoustic radiance transfer algorithms [22] implicitly apply the transfer operator by performing path tracing from the source and precomputing the IR at each surface sample after several orders of reflection. This approach has the disadvantage of having to repeat the entire process if the source moves. We eliminate this disadvantage by precomputing $\mathbf{T}$, and multiplying it with the direct response of the source at run-time. This decoupling of the source position from the precomputed data allows rapid updates of the IR at the listener whenever the source moves.

4.2 Transfer Operator Compression

To apply the transfer operator once, the matrix-vector multiplication in Equation 5 needs to be performed once per Fourier coefficient at run-time. However, even for scenes of moderate complexity, the number of surface samples, p, can be very large. Since $\mathbf{T}$ is a $p \times p$ matrix and $\mathbf{l}_n$ is a $p \times 1$ vector, this step takes $O(p^2)$ time per Fourier coefficient per order of reflection, which can quickly become quite expensive. We use the SVD to compute a rank k approximation of $\mathbf{T}$. This allows us to reduce the complexity to $O(pk)$.

Intuitively, truncating $\mathbf{T}$ to k modes using the SVD removes some of the high spatial frequencies in the transfer operator. A lower-

order mode of $\mathbf{T}$ might model reflections from an entire wall, while higher-order modes might model details added to the acoustic response due to local variations in the wall's geometry (such as a painting on the wall). In effect, the parameter k can be used to control the level-of-detail of the acoustic response.

As we shall discuss in Section 7, there are use cases where we wish to precompute a one-bounce transfer operator and apply it repeatedly to obtain higher-order reflections. In such cases, the cost of computing transfer matrices that represent additional bounces can be further reduced to $O(k^2)$ by precomputing appropriate matrices as follows. The direct IRs at each surface sample are stored in the vector $\mathbf{l}_0$. Suppose we have a rank k approximation of $\mathbf{T}$, given by $\widetilde{\mathbf{T}} = \widetilde{\mathbf{U}}\widetilde{\mathbf{S}}\widetilde{\mathbf{V}}^T$, where $\widetilde{\mathbf{U}}$ is a $p \times k$ matrix, $\widetilde{\mathbf{S}}$ is a $k \times k$ diagonal matrix and $\widetilde{\mathbf{V}}^T$ is a $k \times p$ matrix. Then the first-order IR at each surface sample is given by:

$$\begin{aligned} \widetilde{\mathbf{T}}\mathbf{l}_0 &= \widetilde{\mathbf{U}}\widetilde{\mathbf{S}}\widetilde{\mathbf{V}}^T\mathbf{l}_0 \\ &= \widetilde{\mathbf{U}}\mathbf{b} \end{aligned}$$

where $\mathbf{b} = \widetilde{\mathbf{S}}\widetilde{\mathbf{V}}^T\mathbf{l}_0$ is $\mathbf{l}_0$ projected into the span of the first k right singular vectors of $\mathbf{T}$. The second-order response is:

$$\begin{aligned} \widetilde{\mathbf{T}}\widetilde{\mathbf{T}}\mathbf{l}_0 &= \widetilde{\mathbf{U}}(\widetilde{\mathbf{S}}\widetilde{\mathbf{V}}^T\widetilde{\mathbf{U}})\widetilde{\mathbf{S}}\widetilde{\mathbf{V}}^T\mathbf{l}_0 \\ &= \widetilde{\mathbf{U}}\mathbf{D}\mathbf{b} \end{aligned}$$

where $\mathbf{D} = \widetilde{\mathbf{S}}\widetilde{\mathbf{V}}^T\widetilde{\mathbf{U}}$ is essentially the one-bounce operator in the k-dimensional subspace spanned by the singular vectors corresponding to the top k singular values of $\mathbf{T}$. The cost of multiplying $\mathbf{b}$ by $\mathbf{D}$ is simply $O(k^2)$. Notice that the third-order response can be written as $\widetilde{\mathbf{U}}\mathbf{D}^\mathbf{2}\mathbf{b}$, and so on. This allows us to compute higher-order responses using a $k \times k$ matrix instead of a $p \times p$ matrix.

5 Implementation

Our implementation is CPU-based, and uses Microsoft DirectX 9 for visualization, and Intel Math Kernel Library (MKL) for the matrix operations.

5.1 Approximations

Our algorithm allows for the following user-controlled approximations:

Surface Samples We parameterize the scene surface by mapping the primitives to the unit square (a *uv* texture mapping) using Least Squares Conformal Mapping (LSCM) [33]. The user specifies the texture dimensions; each texel of the resulting texture is mapped to a single surface sample using an inverse mapping process. The number of texels mapped to a given primitive is weighted by the area of the primitive, to ensure a roughly even distribution of samples. We chose the LSCM algorithm for this purpose since our modeling tools (Blender [1]) have an implementation built-in; it can be replaced with any other technique for sampling the surfaces as long as the number of samples generated on a primitive is proportional to its area.

Frequency Samples We allow the user to vary the number of Fourier coefficients used to represent the IRs. We use 1K Fourier coefficients in all our experiments, since it has been shown [22] that this provides an acceptable compromise between performance and quality.

Transfer Operator Modes The SVD approximation error of the transfer operator is measured using the Frobenius norm. Figure 3 plots the error against the number of modes retained in the transfer operator. The figure suggests that we could potentially use a very small number of modes to compute IRs with diffuse reflections at runtime. Figure 4 plots the SVD approximation error (at 50 modes) with increasing orders of reflection. The figure clearly shows that the error introduced by the SVD approximation for higher orders of reflection quickly converges. In other words, the IR energy due to higher-order reflections can be modeled using very few SVD modes of the transfer operator. This matches the intuition that higher-order reflections have low spatial frequency. As a result, when computing very high orders of reflection (say 50), we can use very few SVD modes beyond the first 2-3 orders while still capturing the higher order energy (which must be captured to model the late reverberation tail of the IR) accurately.

1. http://www.blender.org

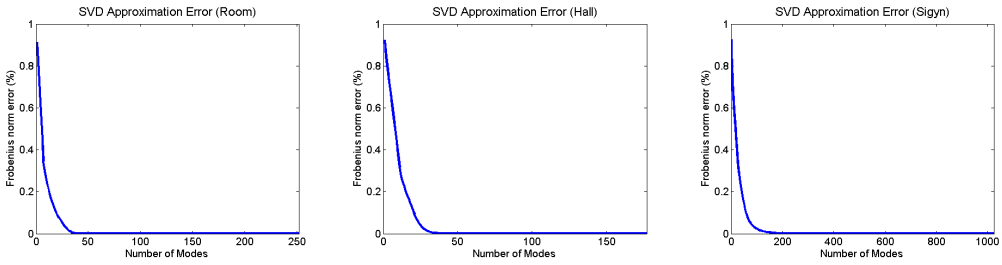


Fig. 3. SVD approximation error for transfer operators. For each benchmark scene, the plots show the relative Frobenius norm error of rank-k approximations of $\mathbf{T}$ (for one value of ω) for all possible values of k. From left to right: (a) Room (252 samples), (b) Hall (177 samples), (c) Sigyn (1024 samples).

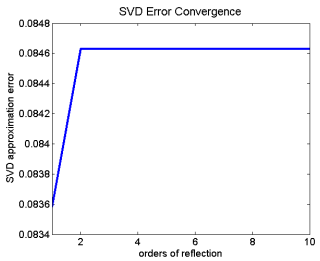


Fig. 4. SVD approximation error for each higher order of reflection, for the Sigyn scene (see Figure 5).

5.2 Audio Processing

The algorithm presented in Section 4 computes a frequency domain energy IR with 1K Fourier coefficients. The pressure IR is computed from the energy IR [34] and upsampled to encode the desired propagation delay in the IR [22].

Moving Sources and Listeners: In typical virtual environment applications, the source and listener tend to move and the audio is streamed from the source in chunks of audio samples (called audio frames). The frame size is determined by the allowed latency for the application. We choose audio frames of 4800 samples at a sampling rate of 48KHz, leading to a 100ms latency in our audio output. For a static source and listener, computing the final audio is trivial and amounts to convolving each audio frame with the IR to compute output audio frames. For moving sources and listeners, IRs evolve over time which could lead to discontinuities in the final audio when using different IRs for two adjacent audio frames. In order to minimize such discontinuity artifacts, windowing [35] is applied at the source frame and the listener frame when the source and listener are moving respectively. We use a windowing method similar to Siltanen et al. [22].

Note that the audio used in the accompanying video is generated by convolving the dry input audio with the listener IR. Ideally, one would apply the listener's Head-Related Transfer Function (HRTF) in order to model the shape of the listener's head and inter-reflections due to the head and shoulders. However, this step has been skipped in the video for simplicity, especially since HRTFs are not related to our main contributions.

6 Experiments

We now present some experimental results. All tests were performed on an Intel Xeon X5560 workstation with 4 cores (each operating at 2.80 GHz) and 4GB of RAM running Windows Vista. We report timings for all 4 cores since MKL automatically parallelizes our matrix operations over all cores of the test machine. We have benchmarked our implementation on three scenes whose complexity is typical of scenes encountered in acoustics applications. Figure 5 shows these scenes along with some details.

For comparison, we chose the state-of-the-art frequency-domain acoustic radiance transfer algorithm [22]. To the best of our knowledge, the only other algorithms that model diffuse sound reflections are time-domain radiosity and path tracing. Since time-domain radiosity requires a prohibitive amount of memory, we chose not to compare against it. Path

Fig. 5. Benchmark scenes. From left to right: (a) Room (252 samples), (b) Hall (177 samples), (c) Sigyn (1024 samples).

Scene	Surface Samples	Precomputation Time $\mathbf{T}$	Precomputation Time SVD	Modes	Runtime Initial Scatter	Runtime Transfer Operator	Runtime Final Gather
Room	252	14.2 s	94.5 s	10	43.2 ms	24.0 ms	33.7 ms
				25	45.8 ms	43.8 ms	35.0 ms
				50	42.4 ms	84.3 ms	36.4 ms
Hall	177	13.1 s	93.1 s	10	37.8 ms	26.8 ms	31.5 ms
				25	37.1 ms	45.5 ms	30.2 ms
				50	36.6 ms	79.7 ms	31.2 ms
Sigyn	1024	6.31 min	50.9 min	50	164.1 ms	218.1 ms	109.9 ms

TABLE 1

Performance characteristics of our algorithm. For each scene, we present the precomputation time required by our algorithm for 1K Fourier coefficients. Under precomputation time, we show the time required to compute the transfer operator, $\mathbf{T}$, and the time required to compute its SVD approximation. We also compare running times for varying numbers of modes from the SVD. The table shows the time spent at runtime in initial shooting from the source, applying the transfer operator, and gathering the final IR at the listener position.

tracing, while well-suited for dynamic scenes, requires even static scenes to be traversed millions of times per frame for higher-order reflections. Part of the reduction in complexity (and the memory usage) for the frequency-domain approach, is due to the restriction to a relatively small number of Fourier coefficients.

Frequency-domain acoustic radiance transfer (ART) [22] computes the transfer operator (without any SVD approximation) and iteratively applies it to the direct acoustic response until the solution converges. In order to perform a fair comparison, we restrict ART to computing as many orders of reflection as our algorithm.

Table 1 summarizes the performance of the precomputation and run-time stages of our algorithm. The run-time complexity depends on the number of modes retained during the SVD approximation; the table clearly highlights this dependency. As shown by the table, our algorithm very efficiently updates IRs when the source position changes at run-time. Note that we precompute a one-bounce transfer operator, and use the approach described in Section 4.2 to compute higher-order reflections at run-time. Depending on the application, we could also precompute a multi-bounce operator and apply it directly at run-time, further improving our performance. Our implementation uses the more flexible approach of varying the orders of reflection at runtime. As a result, it is possible to further improve the performance of our implementation.

Table 2 shows the benefit of the SVD in compressing the transfer operator. The table shows that without using SVD, the transfer operators may be too large to be used on commodity hardware. For the uncompressed ("reference") case, the transfer operator size is $p \times p$, for each Fourier coefficient (1K in our case). For the compressed ("50 Modes") case, the transfer operator size is $p \times k$ for $\tilde{\mathbf{U}}$, $k \times k$ for $\mathbf{D}$ and $k \times p$ for $\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$, where k is the number of modes retained. In the table, $k = 50$, and p is the number of surface samples in the scene.

Table 3 compares the run-time performance of our method and ART. The table shows

Scene	Samples	Reference	50 Modes
Hall	177	250.6	161.6
Room	252	508.0	221.6
Sigyn	1024	8388.6	839.2

TABLE 2
Memory requirements of the transfer operators computed by our algorithm with (column 4) and without (column 3) SVD compression. Note that since the entries of each matrix are complex numbers, each entry requires 8 bytes of storage. All sizes in the table are in MB.

the time required to update IRs at the listener when the source is moved. The table clearly shows the advantage of our approach. Since our precomputed transfer operator is decoupled from the source position, moving the source does not require recomputing the transfer operator, allowing the source position to be updated much faster than would be possible with ART.

Table 3 can also be used to derive the performance of our algorithm for the case when a multi-bounce transfer operator is precomputed. For example, suppose we precompute a transfer operator with 10 orders of reflection for the Sigyn scene. Then the run-time cost would be the same as that of the one-bounce operator, i.e., 468.5 ms. The difference, i.e. $(512.8ms - 468.5ms) \times 1024 = 45.4s$ would be the additional time spent during preprocessing to derive the multi-bounce operator from the one-bounce operator (the factor of 1024 arises due to the fact that the timings in Table 3 are for matrix-vector multiplication, whereas precomputing the multi-bounce operator from the one-bounce operator requires matrix-matrix multiplications).

Figure 6 compares the output of our algorithm and ART. The figure shows squared IRs, smoothed using a moving-average low-pass filter, for different numbers of modes. As the figure shows, reducing the number of modes significantly (down to 50 modes) has very little effect; however, if far fewer modes are used, significant errors appear in the energy decays, as expected. Coupled with the memory savings demonstrated in Table 2 and performance advantage demonstrated in Table 3, we see that using the SVD allows us to significantly reduce memory requirements and increase performance without significant degradation of the computed IRs. Along with the plots, Figure 6 shows RT_{60} (reverberation time) values estimated from the decay curves. The data demonstrates that SVD approximation upto 50 modes does not lead to significant change in reverberation time. Of course, the best way to demonstrate the benefit of our approach is by comparing audio clips; for this we refer the reader to the accompanying video.

7 Conclusion

We have described a precomputed direct-to-indirect transfer approach to solving the acoustic rendering equation in the frequency domain for diffuse reflections. We have demonstrated that our approach is able to efficiently simulate diffuse reflections for a moving source and listener in static scenes. In comparison with existing methods, our approach offers a significant performance advantage when handling moving sources.

7.1 Analysis

Our algorithm is designed to model purely diffuse reflections of sound. However, it is also important to model specular reflections and diffraction of sound in order to render plausible audio. Our algorithm is intended to be used in tandem with existing algorithms for modeling specular reflections and diffraction.

Another important design decision is how we compute the transfer operator. There are several possible use-cases:

1) Precompute a multi-bounce transfer operator, using multi-bounce path tracing from each surface sample. This is the most accurate option, but requires high precomputation time. The run-time cost is equal to that of applying a one-bounce operator.
2) Precompute a multi-bounce transfer operator by multiplying a one-bounce operator with itself repeatedly (before computing an SVD).
3) Precompute the SVD approximation of the one-bounce transfer operator. Use the method described in Section 4.2 to quickly precompute an approximate multi-bounce operator.

Scene	Orders	Radiance Transfer	Direct-to-Indirect Transfer (50 modes)
Room	2	11.7 s	131.8 ms
	5	11.8 s	154.4 ms
	10	12.0 s	179.3 ms
Hall	2	10.6 s	116.5 ms
	5	10.7 s	147.3 ms
	10	10.9 s	170.5 ms
Sigyn	2	185.3 s	468.5 ms
	5	186.7 s	491.2 ms
	10	188.7 s	512.8 ms

TABLE 3
Comparison of our approach with ART. We compare the time required by our algorithm to update the source position and recompute the IR at the listener position after a varying number of diffuse reflections. Since ART does not decouple the transfer operator from the source position, it needs to perform a costly recomputation of the transfer operator each time the source moves. On the other hand, our algorithm quickly updates the direct IR at all surface samples, then applies the precomputed transfer operator. This allows our approach to handle moving sources far more efficiently than ART.

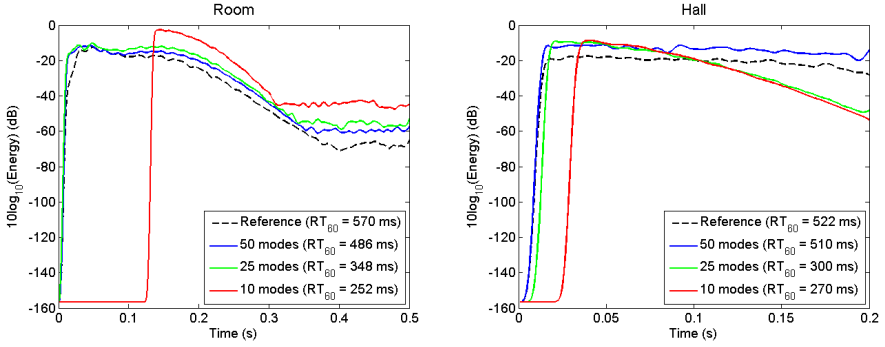


Fig. 6. Comparison of diffuse IRs (30 orders of reflection, absorption coefficient 0.2) computed by our system with and without SVD compression, for some of our benchmark scenes. The plots show squared IRs, smoothed using a moving-average low-pass filter.

4) Precompute the SVD approximation of the one-bounce transfer operator. At run-time, the orders of reflection can be easily adjusted, perhaps based on compute budget or sound quality.

Use-case (3) allows sound designers to rapidly adjust the orders of reflection baked into the precomputed transfer operator. For example, one could first compute a multi-bounce operator with 3 orders of reflection. If the resulting audio at run-time sounds unsatisfactory, the precomputed data can quickly be updated with additional orders of reflection without any further ray tracing or SVD computation.

In all other use-cases besides (3), the SVD allows the IR accuracy to be traded off for performance, providing adjustable level-of-detail for sound rendering.

7.2 Limitations

Since ours is a precomputation-based algorithm, it cannot be used for scenes with dynamic objects. In such situations, ray-tracing-based algorithms are the best available choice. However, in many applications, including games and virtual environments, scenes are mostly static, with relatively few moving parts, hence our algorithm can be used to model reflections within the static portions of the scene.

Our algorithm performs matrix-vector multiplications on large matrices at runtime. The matrix size depends on the size and complexity of the scene. Therefore, our method is useful mainly for scenes of low to medium complexity.

Another limitation arises from the approach we use [22] to reconstruct the energy response from the subsampled Fourier coefficients. The replication of Fourier coefficients leads to comb-filter artifacts in the final audio, and is an inherent limitation of the reconstruction approach. An alternative would be to treat the Fourier coefficients as defining the envelope of a noise process [36]. Both these approaches are prone to errors; further study is needed to determine the suitability of one over the other based on empirical and perceptual error.

Finally, the transfer matrix is computed using the acoustic rendering equation [21], which has its own limitations, in that it is an energy-based approach (and hence cannot easily model interference) and is based on geometric approximations to the acoustic wave equation (and hence cannot easily model low-frequency wave effects such as diffraction).

7.3 Future Work

It is crucial to develop an approach to reduce the storage requirements of IRs in order to make it feasible to implement our algorithm on hardware such as video game consoles. Moreoever, the Fourier domain representation, and the replication-based inverse transform we use to reconstruct IRs [22] lead to errors in the high-frequency component of the final audio. Specifically, the high frequencies do not decay as quickly as they should relative to the low frequencies. It would be very useful to determine alternative IR representations that are both memory-efficient and permit a more accurate reconstruction. The reconstruction process can also lead to some noise and other artifacts. While these can be cleaned up in a post-processing step, preliminary research on using alternative IR representations such as wavelets has given us results without these artifacts. However, the use of wavelets to represent IRs requires some further research.

Our direct-to-indirect transfer approach is intended to be used in tandem with existing algorithms (such as ray tracing, image source method, or numerical methods) for modeling specular reflections and edge diffractions. We are currently researching into ways of modeling paths involving both diffuse and specular reflections by modifying our initial shooting and final gathering steps to include specular reflections. We also hope to model paths including diffractions using a similar approach, since we believe that a tight integration between existing algorithms for specular reflections and edge diffraction, and our direct-to-indirect transfer approach will enable plausible sound propagation for interactive applications with moving sources and listeners at near-interactive rates.

In most complex scenes, each surface sample may influence only a few other samples, due to occlusions. We could subdivide the scene into cells separated by portals, compute transfer operators for each cell independently, and model the interchange of sound energy at the portal boundaries. Cells and portals have been previously used to model late reverberation [36], and would be a promising research direction for acoustic radiance transfer.

The acoustic response is typically a smooth function over the surfaces of the scene. Therefore, it would be beneficial to exploit spatial coherence by projecting the transfer operator into basis functions defined over the surfaces of the scene.

Some radiance transfer algorithms in graphics [4], [5] can model glossy reflections by using a directional basis such as spherical harmonics (SH) at each surface sample. Such a strategy can also be applied to model glossy reflections and diffraction of sound, however, the memory requirements for such an approach might be prohibitive.

Note that we provide a framework which allows the user to decide how many orders of reflection to simulate; how to choose this parameter appropriately is an interesting avenue for future work.

REFERENCES

[1] B.-I. Dalenbäck, M. Kleiner, and P. Svensson, "A macroscopic view of diffuse reflection," *J. Audio Eng. Soc.*, vol. 42, pp. 793–807, 1994.

[2] B.-I. Dalenbäck, "The Importance of Diffuse Reflection in Computerized Room Acoustic Prediction and Auralization," in *Proceedings of the Institute of Acoustics (IOA)*, vol. 17, no. 1, 1995, pp. 27–33.

[3] R. R. Torres, M. Kleiner, and B.-I. Dalenbäck, "Audibility of "diffusion" in room acoustics auralization: An initial investigation," *Acta Acustica united with Acustica*, vol. 86, pp. 919–927(9), November/December 2000.

[4] P.-P. Sloan, J. Kautz, and J. Snyder, "Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments," in *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 2002, pp. 527–536.

[5] M. Hašan, F. Pellacini, and K. Bala, "Direct-to-indirect transfer for cinematic relighting," in *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*. New York, NY, USA: ACM, 2006, pp. 1089–1097.

[6] J. Lehtinen, M. Zwicker, E. Turquin, J. Kontkanen, F. Durand, F. X. Sillion, and T. Aila, "A meshless hierarchical representation for light transport," in *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*. New York, NY, USA: ACM, 2008, pp. 1–9.

[7] P. Svensson and R. Kristiansen, "Computational modelling and simulation of acoustic spaces," in *22nd International Conference: Virtual, Synthetic, and Entertainment Audio*, June 2002.

[8] R. Ciskowski and C. Brebbia, *Boundary Element methods in acoustics*. Computational Mechanics Publications and Elsevier Applied Science, 1991.

[9] N. Raghuvanshi, N. Galoppo, and M. C. Lin, "Accelerated wave-based acoustics simulation," in *SPM '08: Proceedings of the 2008 ACM symposium on Solid and physical modeling*. New York, NY, USA: ACM, 2008, pp. 91–102.

[10] F. Ihlenburg, *Finite Element Analysis of Acoustic Scattering*. Springer Verlag AG, 1998.

[11] N. Raghuvanshi, J. Snyder, R. Mehra, M. Lin, and N. Govindaraju, "Precomputed wave simulation for real-time sound propagation of dynamic sources in complex scenes," in *Proc. SIGGRAPH 2010 (To appear)*, 2010.

[12] M. Vorlander, "Simulation of the transient and steady-state sound propagation in rooms using a new combined ray-tracing/image-source algorithm," *The Journal of the Acoustical Society of America*, vol. 86, no. 1, pp. 172–178, 1989. [Online]. Available: http://link.aip.org/link/?JAS/86/172/1

[13] B. Kapralos, M. Jenkin, and E. Milios, "Sonel mapping: acoustic modeling utilizing an acoustic version of photon mapping," in *In IEEE International Workshop on Haptics Audio Visual Environments and their Applications (HAVE 2004)*, 2004, pp. 2–3.

[14] A. Chandak, C. Lauterbach, M. Taylor, Z. Ren, and D. Manocha, "Ad-frustum: Adaptive frustum tracing for interactive sound propagation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1707–1722, 2008.

[15] M. T. Taylor, A. Chandak, L. Antani, and D. Manocha, "Resound: interactive sound rendering for dynamic virtual environments," in *MM '09: Proceedings of the seventeen ACM international conference on Multimedia*. New York, NY, USA: ACM, 2009, pp. 271–280.

[16] T. Funkhouser, I. Carlbom, G. Elko, G. Pingali, M. Sondhi, and J. West, "A beam tracing approach to acoustic modeling for interactive virtual environments," in *Proc. of ACM SIGGRAPH*, 1998, pp. 21–32.

[17] A. Chandak, L. Antani, M. Taylor, and D. Manocha, "Fastv: From-point visibility culling on complex models," *Computer Graphics Forum*, vol. 28, pp. 1237–1246(10), 2009. [Online]. Available: http://www.ingentaconnect.com/content/bpl/cgf/2009/00000028/00000004/art00020

[18] N. Tsingos and J.-D. Gascuel, "A general model for the simulation of room acoustics based on hierachical radiosity," in *ACM SIGGRAPH 97 Visual Proceedings: The art and interdisciplinary programs of SIGGRAPH '97*, ser. SIGGRAPH '97. New York, NY, USA: ACM, 1997, pp. 149–. [Online]. Available: http://doi.acm.org/10.1145/259081.259236

[19] A. L. Bot and A. Bocquillet, "Comparison of an integral equation on energy and the ray-tracing technique in room acoustics," *The Journal of the Acoustical Society of America*, vol. 108, no. 4, pp. 1732–1740, 2000. [Online]. Available: http://link.aip.org/link/?JAS/108/1732/1

[20] E.-M. Nosal, M. Hodgson, and I. Ashdown, "Improved algorithms and methods for room sound-field prediction by acoustical radiosity in arbitrary polyhedral rooms," *The Journal of the Acoustical Society of America*, vol. 116, no. 2, pp. 970–980, 2004. [Online]. Available: http://link.aip.org/link/?JAS/116/970/1

[21] S. Siltanen, T. Lokki, S. Kiminki, and L. Savioja, "The room acoustic rendering equation." *The Journal of the Acoustical Society of America*, vol. 122, no. 3, pp. 1624–1635, September 2007.

[22] S. Siltanen, T. Lokki, and L. Savioja, "Frequency domain acoustic radiance transfer for real-time auralization," *Acta Acustica united with Acustica*, vol. 95, pp. 106–117(12), 2009. [Online]. Available: http://www.ingentaconnect.com/content/dav/aaua/2009/00000095/00000001/art00010

[23] U. P. Svensson, R. I. Fred, and J. Vanderkooy, "An analytic secondary source model of edge diffraction impulse responses ," *Acoustical Society of America Journal*, vol. 106, pp. 2331–2344, Nov. 1999.

[24] M. Taylor, A. Chandak, Z. Ren, C. Lauterbach, and D. Manocha, "Fast edge-diffraction for sound propagation in complex virtual environments," in *EAA Auralization Symposium*, 2009.

[25] D. Schröder and A. Pohl, "Real-time hybrid simulation method including edge diffraction," in *EAA Auralization Symposium*, 2009.

[26] C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Battaile, "Modeling the interaction of light between diffuse surfaces," *SIGGRAPH Comput. Graph.*, vol. 18, no. 3, pp. 213–222, 1984.

[27] Y.-T. Tsai and Z.-C. Shih, "All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 967–976, 2006.

[28] A. W. Kristensen, T. Akenine-Möller, and H. W. Jensen, "Precomputed local radiance transfer for real-time lighting design," in *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*. New York, NY, USA: ACM, 2005, pp. 1208–1215.

[29] H. Kuttruff, *Room Acoustics*. Elsevier Science Publishing Ltd., 1991.

[30] T. Funkhouser, N. Tsingos, and J.-M. Jot, "Survey of methods for modeling sound propagation in interactive virtual environment systems," *Presence*, 2003.

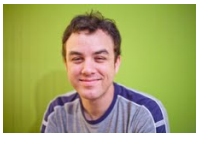
[31] T. I. Laakso, V. Välimäki, M. Karjalainen, and U. K. Laine, "Splitting the unit delay - tools for fractional delay filter design," *IEEE Signal Processing Magazine*, vol. 13, 1996.

[32] J. T. Kajiya, "The rendering equation," in *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1986, pp. 143–150.

[33] B. Lévy, S. Petitjean, N. Ray, and J. Maillot, "Least squares conformal maps for automatic texture atlas generation," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 362–371, 2002.
[34] H. K. Kuttruff, "Auralization of Impulse Responses Modeled on the Basis of Ray-Tracing Results," *Journal of the Audio Engineering Society*, vol. 41, no. 11, pp. 876–880, November 1993.
[35] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, 2nd ed. Prentice Hall, January 1999.
[36] E. Stavrakis, N. Tsingos, and P. Calamia, "Topological sound propagation with reverberation graphs," *Acta Acustica/Acustica - the Journal of the European Acoustics Association (EAA)*, 2008. [Online]. Available: http://www-sop.inria.fr/reves/Basilic/2008/STC08

Lakulish Antani Lakulish Antani received Bachelor's and Master's degrees in Computer Science from the Indian Institute of Technology (IIT) Bombay in 2008. He is currently a Ph.D. student in the Department of Computer Science at the University of North Carolina at Chapel Hill. He has previously been a research intern with INRIA Sophia-Antipolis and Disney Interactive Studios. His research interests include real-time rendering, ray tracing and sound rendering for interactive applications.

Anish Chandak Anish Chandak received the Bachelor's degree in Computer Science and Engineering (2006) from the Indian Institute of Technology (IIT) Bombay, India and the MS degree in Computer Science (2010) from the University of North Carolina at Chapel Hill. He is a Ph.D. student in the Computer Science Department at the University of North Carolina at Chapel Hill. He has worked as a research intern at Microsoft Research and Microsoft Technology Incubation. His research interests include acoustics (sound synthesis, sound propagation, and 3D audio rendering) and computer graphics (ray tracing, visibility, and GPGPU).



Micah Taylor Micah Taylor received a BS in computer science from Rose-Hulman Institute of Technology in 2004. After working in industry for several years, he returned to academia to seek his Ph.D. He is currently a research associate at the University of North Carolina at Chapel Hill working in the GAMMA research group. His research interests are interactive sound propagation and real-time ray tracing.

Dinesh Manocha Dinesh Manocha is currently the Phi Delta Theta/Mason Distinguished Professor of Computer Science at the University of North Carolina at Chapel Hill. He received his Ph.D. in Computer Science at the University of California at Berkeley 1992. He has received Junior Faculty Award, Alfred P. Sloan Fellowship, NSF Career Award, Office of Naval Research Young Investigator Award, Honda Research Initiation Award, Hettleman Prize for Scholarly Achievement. Along with his students, Manocha has also received 12 best paper and panel awards at the leading conferences on graphics, geometric modeling, visualization, multimedia and high-performance computing. He is an ACM and AAAS Fellow. Manocha has published more than 300 papers in the leading conferences and journals on computer graphics, geometric computing, robotics, and scientific computing and supervised 18 Ph.D. dissertations.