Interactive Sound Propagation using Compact Acoustic Transfer Operators

LAKULISH ANTANI and ANISH CHANDAK
University of North Carolina at Chapel Hill
and
LAURI SAVIOJA
Aalto University
and
DINESH MANOCHA
University of North Carolina at Chapel Hill

We present an interactive sound propagation algorithm that can compute high orders of specular and diffuse reflections as well as edge diffractions in response to moving sound sources and a moving listener. Our formulation is based on a precomputed acoustic transfer operator, which we compactly represent using the Karhunen-Loeve transform. At runtime, we use a two-pass approach that combines acoustic radiance transfer with interactive ray tracing to compute early reflections as well as higher-order reflections and late reverberation. The overall approach allows accuracy to be traded off for improved performance at run-time, and has a low memory overhead. We demonstrate the performance of our algorithm on different scenarios, including an integration of our algorithm with Valve's Source game engine.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Radiosity*; I.3.8 [**Computer Graphics**]: Applications—*Sound rendering*; H.5.5 [**Information Systems**]: Information Interfaces and Presentation—*Sound and music computing*

General Terms: Real-time Rendering, Geometric Acoustics

Additional Key Words and Phrases: sound propagation, acoustic radiance transfer, higher-order reflections, interactive applications

1. INTRODUCTION

Realistic sound propagation and rendering can add a whole new layer of realism to interactive applications such as video games and virtual environments. Sound propagation in a scene refers to the modeling of the sound heard by the listener after the sound emitted from each source undergoes reflections, diffraction and absorption through the scene, taking into account source and listener positions within the scene along with surface material properties. At a broad level, sound propagation effects in an acoustic environment are composed of two parts: early reflections (ER) and late reverberation (LR) ([Kuttruff 1991], pp. 97). For the listener, ER helps in localization and conveys spatial information about an environment, while LR enhances immersion, giving an impression of the size of the environment, level of furnishing and absorptivity ([Kuttruff 1991], pp. 98, 194). In order to generate plausible sound rendering, it is important to model both of these parts.

Accurate offline algorithms for sound propagation are limited to static sources and/or listeners. Several algorithms have been proposed for interactive sound propagation in dynamic scenes, but they are limited to modeling early reflections only. As a result, there has been recent interest in developing precomputation-based algorithms to model higher-order reflections. However, these methods have a very high memory overhead, or require the scene to be composed of cells and portals (see Section 2.1 for more details).

Main Results We present a novel geometric sound propagation algorithm that computes diffuse and specular reflections as well as edge diffractions at near-interactive rates. In order to model higher-order reflections and diffraction, our algorithm precomputes an *acoustic transfer operator* that models how sound energy propagates between surfaces. We use a scene-dependent Karhunen-Loeve transform (KLT) for compactly representing the transfer operators. At runtime, we use a two-pass method that uses the transfer operator to compute higher-order reflections and diffraction, along with interactive ray tracing to model early reflections and diffraction.

Some of the main benefits of our approach include:

—**Compact representation:** Our compression technique, based on KLT, results in low memory overhead for the acoustic transfer operators, resulting in a compression factor of up to two orders of magnitude over time-domain or frequency-domain representations.

—**Runtime control between accuracy and performance:** Our choice of basis for representing the acoustic transfer operator has the additional advantage of allowing control over approximation errors, and thereby trading off accuracy for performance in interactive applications.

DOI 10.1145/XXXXXXX.YYYYYYY
http://doi.acm.org/10.1145/XXXXXXX.YYYYYYY

—**Moving sources and listeners:** Our precomputed acoustic transfer operator is defined in terms of samples distributed over the surfaces of a static scene. As a result, we can efficiently handle moving sources and listeners.

—**Occlusion of sound by dynamic objects:** Our algorithm can handle (to a limited extent) the effect of introducing a dynamic object on the sound field at the listener – due to occlusion of sound emitted from the source by moving obstacles and the subsequent effect of the occlusion on propagated sound, or due to occlusion of propagated sound by moving obstacles before it reaches the listener.

In practice, our algorithm can handle indoor models at nearly interactive rates, i.e. $5-10$ frames per second on a modern desktop computer. The memory overhead of storing the precomputed transfer operators is typically a few dozen megabytes. We have also integrated our algorithm with Valve's Source engine, and use it to provide plausible sound propagation effects during real-time gameplay.

Outline The rest of the paper is organized as follows. Section 2 provides a brief overview of prior art. Section 3 presents our formulation and compression method for acoustic transfer operators. Section 4 describes our run-time algorithm. We describe our implementation in Section 5 and highlight the performance on different benchmarks, including the Source engine. Finally, we analyze the approach in Section 6 and compare it with prior approaches.

2. BACKGROUND

In this section, we give a brief overview of related work and introduce some background material used in the rest of the paper.

2.1 Related Work

We now present some key approaches used for sound propagation.

Wave-based Acoustics The propagation of sound in a medium is described by the acoustic wave equation, a second-order partial differential equation. Several numerical methods are known for solving the wave equation, such as finite-element [Ihlenburg 1998] and boundary-element methods [Ciskowski and Brebbia 1991], finite-difference time-domain methods [Savioja et al. 1994; Botteldooren 1995], etc. These techniques are the most accurate in terms of modeling sound propagation in a scene. In spite of recent advances [Raghuvanshi et al. 2009], these methods are limited to static scenes and can be very slow for high-frequency sound sources.

Geometric Acoustics Most sound propagation techniques used in interactive applications are based on geometric methods. These techniques tend to be accurate for high-frequency sources. Some of the commonly used methods are based on ray tracing [Savioja et al. 1999; Kapralos et al. 2004; Bertram et al. 2005; Lentz et al. 2007], which can be used to model both diffuse and specular reflections. Recent video games such as *Crackdown* use ray tracing to compute low-order sound propagation effects. In order to overcome the sampling issues inherent in ray tracing, techniques based on volume tracing [Funkhouser et al. 1998; Taylor et al. 2009] and image sources [Allen and Berkley 1979; Laine et al. 2009; Chandak et al. 2009] have been developed. For static scenes, radiosity [Tsingos and Gascuel 1997; Nosal et al. 2004; Alarcao et al. 2009] or radiance transfer [Siltanen et al. 2007; Siltanen et al. 2009] methods can be used to model reflections from surfaces with arbitrary BRDFs. Many techniques have also been designed to model edge diffraction [Svensson et al. 1999; Tsingos et al. 2001; Stephenson 2010].

Statistical Acoustics Most current video games use acoustic parameters such as *reverberation time* to model the acoustics of a scene. These can be derived from statistical models ([Kuttruff 1991], pp. 119) and/or controlled by an artist. There has been recent work on statistical estimation of diffuse reverberation properties [Taylor et al. 2009]. In practice, statistical models are typically valid only for simple single or coupled rooms [Summers et al. 2004], and are not considered as accurate as other methods [Stavrakis et al. 2008].

Precomputed Sound Propagation The problem of precomputing sound propagation effects for interactive applications has received significant attention in recent years. Cell-and-portal-based scene subdivision has been exploited to model the propagation of sound between cells via portals [Foale and Vamplew 2007; Stavrakis et al. 2008]. Other methods include precomputing image source gradients to quickly estimate specular reflections at run-time [Tsingos 2009], precomputing acoustic radiance transfer from static sources [Siltanen et al. 2009], using a numerical wave equation solver to precompute the acoustic response of a scene from several sampled source positions [Raghuvanshi et al. 2010], and precomputing linear operators to model diffuse reflections of sound in the frequency domain [Antani et al. 2011].

Interactive Acoustics For interactive applications, efficient scalable algorithms based on hierarchical sound source clustering have been developed [Moeck et al. 2007] in order to render sound from large numbers of sources. Many scalable sound synthesis algorithms have also been developed, which can render high-quality, physically-based sound due to collisions [James et al. 2006; Raghuvanshi and Lin 2006; Bonneel et al. 2008; Chadwick et al. 2009]. These algorithms manage the complexity of sound simulation by controlling the detail in the *generation* of sound (either by culling sources, or by simplifying the physical model used to generate the sound). In contrast, our algorithm controls the detail in the *propagation* of sound, by simplifying the representation of an acoustic transfer operator.

Precomputed Light Transport Radiosity [Goral et al. 1984] is the classic precomputed light transport algorithm. However, it computes a global illumination solution that needs to be recomputed whenever the light source moves. In contrast, *precomputed radiance transfer* (PRT) algorithms decouple light transport effects from the light source positions by computing a linear operator that defines how a variable light source configuration affects the radiance at surface sample points. PRT techniques can support both distant [Sloan et al. 2002] and local [Kristensen et al. 2005; Lehtinen et al. 2008] source configurations. We use many of these ideas to formulate our acoustic transfer operators.

2.2 Impulse Responses and Echograms

Sound travels much slower (340 m/s in air) than light (3×10^8 m/s). Hence, while light transport algorithms can compute steady-state values of lighting in the scene and neglect transient effects, it is vital to capture the time variation of the sound energy distribution in a scene. It is this time variation which gives rise to echoes and reverberation in large rooms. The additional dimension of time makes it difficult to apply light transport algorithms directly to model sound propagation.

Physically, the sound emitted by a source is a pressure wave, denoted by $p_s(t)$. The sound waves travel through the scene, undergoing reflection, diffraction and absorption. The listener then receives a *propagated* pressure wave, $p_l(t)$. Since room acoustics can be modeled as a linear time-invariant system ([Kuttruff 1991], pp. 19), the sound heard at the listener can be obtained as $p_l(t) = h_{s,l}(t) \otimes p_s(t)$, where $h_{s,l}(t)$ is the *impulse response* (IR) between the source and listener and $\otimes$ is the time-domain convolution operator. Intuitively, an IR is the signal received at the listener if the source emits a unit impulse $\delta(t)$. An *echogram* is defined as the energy contained in an IR as a function of time, and is proportional to the square of the IR. In the rest of this work, we focus on computing echograms.

2.3 Acoustic Rendering Equation

Analogous to the definition of radiance in visual rendering, *acoustic radiance* at a point x, denoted by $L(x, \Omega, t)$, is defined as the outgoing sound energy flux at point x per unit area per unit solid angle along direction Ω, as a function of time t. Similarly, we can define *acoustic irradiance* at a point x, denoted by $E(x, t)$, as the incident sound energy flux at point x per unit area as a function of time. For a point listener, the acoustic irradiance is proportional to the echogram, and hence to the square of the IR.

The propagation of sound in a scene can be modeled using an extension of the standard graphics rendering equation [Kajiya 1986], called the *acoustic rendering equation* [Siltanen et al. 2007]:

$$L(x,\Omega,t) = L_0(x,\Omega,t) + \int_S R(x,x',\Omega,t) L\left(x', \frac{x-x'}{|x-x'|}, t\right) dx' \quad (1)$$

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

$$R(x,x',\Omega,t) = \rho(x,x',\Omega,t) G(x,x') V(x,x') P(x,x',t). \quad (2)$$

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* [Siltanen et al. 2007] that accounts for propagation delays. We use this equation to define the acoustic transfer operators.

The above equations represent the acoustic rendering equation in the time domain. Hence, the BRDF (and other terms) need to be *convolved* with the acoustic radiance; the convolution operators have been hidden in the equations for brevity. Alternatively, one could formulate the acoustic rendering equation in the frequency domain (replacing the time variable t with the angular frequency ω); in the frequency domain the BRDF (and other terms) indeed are multiplied with the acoustic radiance. Thus one can model arbitrary frequency-dependent material properties using the acoustic rendering equation.

3. ACOUSTIC TRANSFER OPERATOR

The goal of our algorithm is to efficiently compute the *late response* (LR), i.e., higher-order (i.e., beyond 2-3 orders) reflections and edge diffraction of sound, which changes in response to a moving source and a moving listener in a scene. We accomplish this by precomputing an acoustic transfer operator which models the propagation of acoustic radiance between samples distributed over the surface of the scene. In this sense, our algorithm is analogous to PRT algorithms for light transport problems. The acoustic transfer operator is defined by writing the incident form of the acoustic rendering equation in terms of a linear operator:

$$\begin{aligned} L(x,\Omega,t) &= L_0(x,\Omega,t) + \mathcal{R}L(x,\Omega,t) \\ &= \mathcal{T}L_0(x,\Omega,t), \end{aligned} \quad (3)$$

where $L_0(x,\Omega,t)$ represents the *direct* acoustic radiance incident at surface point x, $L(x,\Omega,t)$ represents the *indirect* acoustic radiance incident at x after multiple reflections and diffraction through the scene, $\mathcal{R}$ is a linear operator corresponding to the reflection kernel in Equation 2 and $\mathcal{T} = (\mathcal{I} - \mathcal{R})^{-1}$ is the acoustic transfer operator.

We assume that acoustic radiance at a surface sample does not vary with direction (i.e., the surface samples are diffuse emitters and receivers). In other words, the transfer operator models sound energy which is emitted uniformly in all directions from a given surface sample, and propagates through the scene (undergoing several diffuse and specular reflections as well as diffraction) until the propagation is finally terminated upon incidence at some other surface sample. The propagated, incident sound field is averaged over all incidence directions, resulting in a directionally-invariant indirect acoustic radiance at each surface sample. This simplifying assumption is motivated by the fact that after a few orders of reflection, most of the sound energy in a scene would have typically undergone diffuse reflections [Kuttruff 1995]. This may result in some higher-order echoes being replaced with reverberation, but can be corrected when computing the early response. We now describe our approach for computing a compact representation of the acoustic transfer operator.

3.1 Transfer Operator Precomputation

In order to define the acoustic transfer operator for the scene, we first sample n points on the surface of the scene using area-weighted sampling [Hašan et al. 2006] (Figure 1 (b)). We then construct a compact, scene-dependent KLT basis for representing echograms (Figure 1 (c)), which we then use to compress echograms computed between each surface sample (Figure 1 (d)).

We use energy-based path tracing (i.e., Monte Carlo integration of the acoustic rendering equation) to compute the sample-to-sample echograms. When each path encounters a geometric primitive, it can be diffusely reflected, specularly reflected or diffracted, depending on material properties. Attenuations are applied according to standard geometric acoustics models as discussed below.

Diffuse Reflections Rays are diffusely reflected as per the Lambertian model by randomly sampling a direction on the hemisphere at the point of incidence, and sending a reflected ray along the sampled direction. The ray's energy is attenuated by the frequency-dependent *diffuse coefficient* $d(\nu) = (1 - \alpha(\nu))\sigma(\nu)$, where $\alpha(\nu)$ is the frequency-dependent absorption coefficient and $\sigma(\nu)$ is the frequency-dependent scattering coefficient of the surface material.

Specular Reflections Specular reflection of rays is performed by reflecting incident rays as per the laws of reflection. The ray's energy is attenuated by the frequency-dependent *specular coefficient* $s(\nu) = (1 - \alpha(\nu))(1 - \sigma(\nu))$.

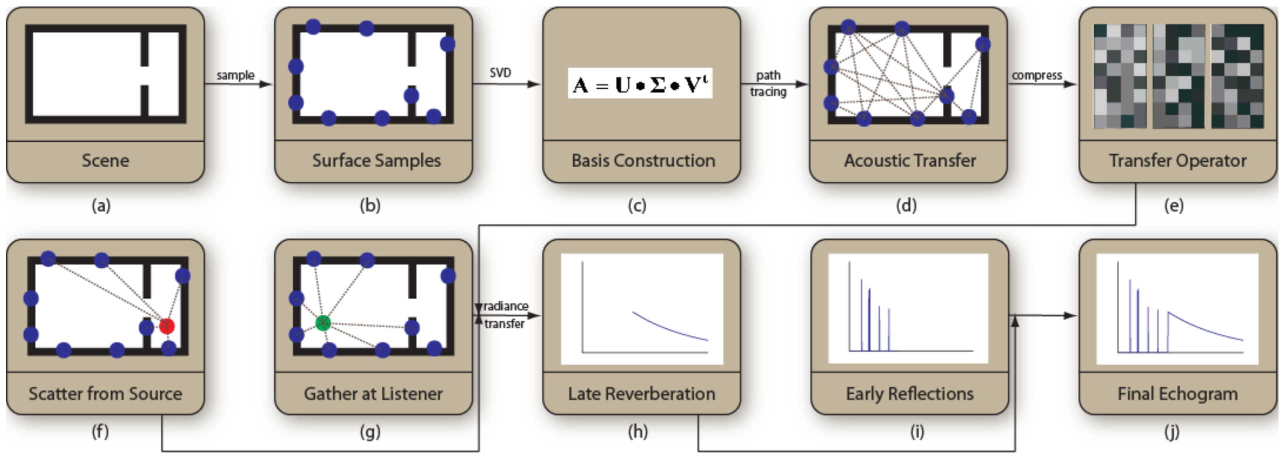


Fig. 1: Overview of our algorithm. **Top row**: Precomputation. **Bottom row**: Run-time interactive sound propagation.

Edge Diffraction Diffraction is modeled using an energy-based ray tracing model derived from Heisenberg's uncertainty principle [Stephenson and Svensson 2007; Stephenson 2010]. Rays which pass sufficiently close to a diffracting edge [Stephenson and Svensson 2007] are diffracted by deviating them in the plane normal to the diffracting edge. The angle of deviation is randomly sampled from a frequency-dependent probability distribution.

3.2 Echogram Representation

In order to capture closely-spaced echoes, which may arise in 2^{nd} or 3^{rd} order reflections captured in the transfer operator, we sample echograms at the audio sampling rate of 48 kHz. As a result, it is impractical to store precomputed sample-to-sample echograms in the time domain, since this would require 192 kB per second per echogram. For $n \approx 256$ surface samples, this would result in the transfer operator requiring 12 GB of storage per second.

Frequency-domain representations have been used in prior precomputation-based sound propagation algorithms, but require a very large number of coefficients ($m \approx 1024$) to represent either the echograms themselves [Siltanen et al. 2009], or the decay envelopes of the echograms [Stavrakis et al. 2008] (which cannot be used to model sharp echoes arising from 2^{nd} or 3^{rd} order reflections).

Commonly used signal compression techniques are based on representing the signals using transforms such as the Fourier transform, the discrete cosine transform (DCT) and the related modified discrete cosine transform (MDCT) [Wang 2003], and wavelet representations. Fourier and DCT representations require a few thousand coefficients [Stavrakis et al. 2008; Siltanen et al. 2009] in order to represent the wide range of audible sound frequencies. While the MDCT and wavelet transforms are typically sparse, they too require hundreds of coefficients in order to represent middle-to-high-frequency reverberation in large spaces. Ideally, we would prefer a basis in which echograms can be represented using relatively few coefficients.

For this, we use a scene-dependent Karhunen-Loeve basis, derived using the Karhunen-Loeve Transform (KLT) [Loève 1978]. The KLT is defined as follows. In order to derive an orthogonal basis for a d-dimensional vector space $\mathbf{S}$, we first randomly sample some number (say p) of vectors in the space. These vectors are written as column vectors and placed side-by-side to form the *data matrix* $\mathbf{A}_{d\times p}$ (subscripts denote matrix dimensions). We can then use the singular value decomposition (SVD) to decompose the data matrix: $\mathbf{A}_{d\times p} = \mathbf{U}_{d\times p}\mathbf{\Sigma}_{p\times p}\mathbf{V}^t_{p\times p}$. The columns of the orthogonal matrix $\mathbf{U}$ are then used as a basis set for $\mathbf{S}$.

To generate an orthogonal basis for sample-to-sample echograms in a given scene, we first randomly choose p pairs of surface samples, and compute echograms between them (using path tracing). The dimension of the vector space in which all echograms lie is equal to the number of samples used to represent the echograms in the time domain. These echograms are used to form the data matrix, and then the SVD is used to compute the KLT basis matrix $\mathbf{U}$. Since the basis vectors are sorted in decreasing order of singular values, we can truncate $\mathbf{U}$ and retain only the first m columns. As demonstrated in the accompanying video, the approximation error can be barely perceptible (in our benchmarks), even with very few basis vectors ($m \approx 32 - 64$).

In essence, this formulation "learns" a good basis for representing echograms in a given scene by using several *example echograms* computed in the scene. Assuming the surface sample pairs used to generate the example echograms are distributed throughout the scene, the Karhunen-Loeve transform can be used to estimate a basis of echograms that requires the fewest number of coefficients to represent an echogram in the scene for a given approximation error. Furthermore, since the storage and time complexity of this algorithm scales linearly with m, we choose the Karhunen-Loeve basis to represent the acoustic transfer operators compactly.

4. RUN-TIME SOUND PROPAGATION

At run-time, we use an approach similar to prior visual rendering algorithms [Wallace et al. 1987] and compute sound propagation effects using a two-pass algorithm (see Figure 1). The two passes work as follows:

(1) **Early Response using Ray Tracing.** Since low-order specular reflections and diffraction are important for sound localization, low-order reflections (diffuse and specular) and edge diffractions are computed using path tracing [Bertram et al. 2005].

(2) **Late Response using Radiance Transfer.** We analytically compute the direct echogram at each surface sample due to the (potentially moving) source(s) (Figure 1 (f)). The acoustic

transfer operator is then applied to the direct echograms; this yields echograms at each surface sample which model higher-order reflections and diffraction. The resulting echograms are gathered from the surface samples at the listener (Figure 1 (g)) to quickly compute the higher-order echogram from a moving source to a moving listener (Figure 1 (h)).

We now detail each pass of our algorithm.

4.1 Acoustic Radiance Transfer

The direct echogram due to a single source at surface sample j can be completely characterized by a delayed impulse with (distance) attenuation a^s_j and a delay d^s_j. Similarly, the response at a listener due to direct sound along each gather ray i can be completely characterized by a delayed impulse with (distance) attenuation a^l_i and a delay d^l_i.

For simplicity, the BRDFs at the first and last reflections are multiplied into the acoustic transfer operator. Furthermore, for simplicity of exposition, we assume that the number of gather rays traced from the listener is also n; in practice, we trace $O(n)$ gather rays, with the constant factor chosen based on run-time performance. As each gather ray hits a point on the surface of the scene, the point is mapped to a surface sample using nearest-neighbor interpolation. We denote the surface sample corresponding to gather ray i by $S(i)$.

These attenuations and delays are then combined with the compressed acoustic transfer operator to compute the final echogram as follows. We denote the precomputed echogram from sample j to sample $S(i)$ by $L_{i,j}(t)$. Then the energy received at the listener via propagation paths whose first reflection occurs at sample j and last reflection occurs at sample $S(i)$ is given by:

$$E_{i,j}(t) = a^s_j a^l_i L_{i,j}(t - d^s_j - d^l_i), \tag{4}$$

and the final echogram at the listener is obtained by adding together energy received from all possible propagation paths:

$$E(t) = \sum_{i=1}^{n} \sum_{j=1}^{n} a^s_j a^l_i L_{i,j}(t - d^s_j - d^l_i). \tag{5}$$

Since the sample-to-sample echograms in the transfer operator are stored in a basis with m coefficients, we use the basis expansion to obtain:

$$L_{i,j}(t) = \sum_{k=1}^{m} \alpha^k_{i,j} b^k(t) \tag{6}$$

$$E(t) = \sum_{k=1}^{m} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} a^s_j a^l_i \alpha^k_{i,j} \right) b^k(t - d^s_j - d^l_i) \tag{7}$$

where b^k denotes the k^{th} basis function and the α's are coefficients of echograms in the basis space. The above expression can be reformulated as a sum of convolutions:

$$E(t) = \sum_{k=1}^{m} H^k(t) \otimes b^k(t) \tag{8}$$

$$H^k(t) = \sum_{i=1}^{n} \sum_{j=1}^{n} a^s_j a^l_i \alpha^k_{i,j} \delta(t - d^s_j - d^l_i) \tag{9}$$

Therefore, at run-time, we use the source position to quickly update a^s_j and d^s_j; and the listener position to quickly update a^l_i and d^l_i. These are used along with the compressed transfer operator to construct the convolution filters $H^k(t)$; convolving the echogram basis functions with these filters and accumulating the results yields an echogram representing higher-order reflections and diffraction from the source to the listener.

4.2 Low-Order Effects

Since we assume that surface samples are diffuse emitters and receivers, the radiance transfer pass cannot model all kinds of propagation paths. Consider a variant of Shirley's regular expression notation for propagation paths, with D denoting a diffuse reflection, S denoting a specular reflection, and E denoting an edge diffraction. Then the radiance transfer pass is restricted to computing $D(D|S|E)^*D$ paths.

However, low-order specular reflections and diffraction provide important directional cues to the listener ([Kuttruff 1991], pp. 194). Therefore, in the first pass of our algorithm, low-order path tracing is performed to compute 1-3 orders of specular reflections and edge diffraction as well as first-order diffuse reflections. At each specular reflection or edge diffraction, energy can be converted to diffuse energy and transferred to the precomputed transfer operator, allowing paths of the form $(S|E)^q(D|S|E)^*(S|E)^q$ (for low values of q) to be modeled, thus allowing low-order purely specular and purely diffraction paths to be modeled. As can be seen from the corresponding regular expressions, the paths modeled in the first and second passes are disjoint (i.e., no path is traced in both passes), hence the echograms from each pass can be directly added, along with the direct source-to-listener contribution, to determine the final echogram at the listener.

4.3 Dynamic Scenes

The acoustic transfer operator is inherently decoupled from both source and listener positions. As a consequence of this formulation, our algorithm can compute higher-order sound propagation in scenes with moving sources and listeners, as mentioned above. Moreover, the computation of early reflections is performed using ray tracing, and hence we can handle fully dynamic scenes in the ER pass.

Dynamic objects may also affect the late response. We use interactive ray tracing [Taylor et al. 2009] to compute direct echograms at each surface sample (Figure 2 (a)). As a result, these rays can intersect and be blocked by dynamic objects (Figure 2 (b)). This allows dynamic objects to induce a "shadow" region and reduce the energy in the direct echograms at the surface samples in the shadow region (see Figure 2 (b)). Since these (modified) direct echograms are used as input to the precomputed acoustic transfer operator in the first pass, our formulation allows dynamic objects to affect (to a limited extent) the propagated sound field heard at the listener in the LR pass. Similarly, since interactive ray tracing is used in the final gather step, reflected and/or diffracted sound can be occluded by a dynamic object before it reaches the listener.

However, since the transfer operator is pre-computed for surface samples defined over static geometry only, we cannot model reflections or inter-reflections off dynamic objects in the radiance transfer pass. For example, we can only model ER (and not LR) due to sound reflecting off a moving car. Furthermore, since the transfer operator is computed over static surfaces only, we cannot model "indirect shadow" regions – i.e., occlusion of reflected days by dynamic objects (Figure 2 (c)). For example, we cannot accurately model the case where two static rooms are separated by a dynamic

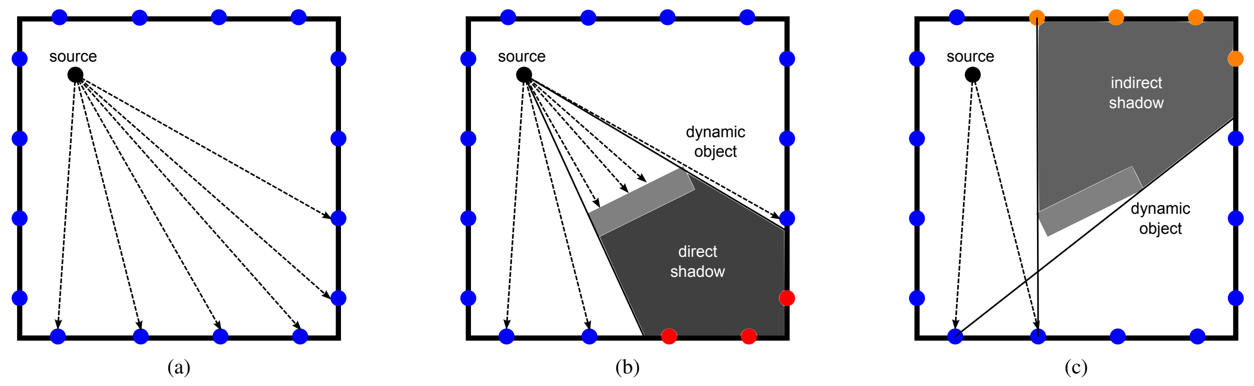


Fig. 2: Dynamic source shadowing. (a) A sound source in a static room. Blue dots indicate surface samples. (b) Adding a dynamic object (in this case, a rectangle). Some rays traced from the source are blocked by the dynamic object. This induces a "shadow" region and changes the direct response at the red dots. This in turn would affect the indirect response everywhere. This effect is modeled by our algorithm. (c) Direct energy reflected from surface samples may also be occluded by the dynamic object, inducing an "indirect shadow" region. However, since we precompute sample-to-sample transfer, indirect shadows are not modeled by our algorithm.

door, since the precomputed transfer operator cannot take into account the changes in visibility between surface samples on the walls of the two rooms caused by opening or closing the door.

4.4 Run-time Error Control

One of the advantages of our choice of echogram basis is that it allows run-time control over the accuracy of the sound propagation. Using fewer basis coefficients at run-time allows accuracy to be adapted to a limited compute budget without sacrificing the frequency content of the propagated sound. The SVD used to compute the Karhunen-Loeve basis for a scene implicitly sorts the basis functions such that most of the energy is distributed into the first few basis functions (see Figure 5). The remaining basis functions can be ignored at run-time by truncating the SVD, with only a minor impact on the accuracy of the echograms computed. Since the run-time performance scales linearly with the number of basis coefficients used (see Section 6.1), we can increase performance by using fewer basis coefficients, at the cost of a slight reduction in sound quality (as shown in the accompanying video).

5. IMPLEMENTATION AND PERFORMANCE

We now present details of our implementation and experimental results demonstrating its performance.

5.1 Implementation Details

Our implementation is written in C++, compiled using Microsoft Visual Studio 2010. We use Intel Math Kernel Library (MKL) for parallelization of linear algebra operations, and NVIDIA OptiX for interactive ray tracing. All precomputation and run-time tests were performed on an Intel Core 2 Quad 2.83 GHz CPU with 4GB RAM and an NVIDIA GeForce GTX 480 GPU, running Windows 7. Figure 3 shows the benchmark scenes used in our experiments. Figure 4 shows the surface samples generated for two of them.

Audio Processing Pipeline All of our echogram processing (during precomputation as well as at run-time) is performed in multiple frequency bands. For efficiency, we perform processing in 3 frequency bands (20Hz–250Hz, 250Hz–2kHz and 2kHz-20kHz). A similar approach has been adopted by Tsingos et al. [2004]. More accuracy can be obtained by using more frequency bands; however, the processing costs scale linearly with the number of frequency bands.

Once the final echogram is computed for the listener, it is used to estimate a pressure impulse response using standard techniques [Kuttruff 1993]. The estimated IR is then convolved with the dry audio signal to compute the final sound heard at the listener.

For simplicity, we do not model the *head-related transfer function* (HRTF) at the listener in our implementation; however, our algorithm can very easily incorporate HRTFs. This can be performed during the final gathering stage of the radiance transfer pass, and when tracing rays from the listener in the ray tracing pass. In both cases, the rays can be amplitude-panned into a distribution of directions, and the corresponding HRTF coefficients can be included in the delays and attenuations applied for each ray. This would allow more accurate spatial reproduction of the sound field at the listener. Similarly, while our algorithm can take into account source directivity, we assume isotropic point sources in all our experiments for simplicity.

Game Engine Integration We have integrated our sound propagation system with Valve's Source game engine, which has been used by many popular video games, including *Half-Life 2* and *Left 4 Dead*. From within the game engine, we spawn a parallel thread which runs our *sound propagation manager*. The game engine generates *sound events* in response to gameplay or physics events such as footsteps, collisions, gunshots or speech. Whenever a sound event is triggered, we pass all relevant information (such as the source position, orientation, and dry sound clip) to the sound propagation manager.

The sound propagation manager computes the IR from each source to the listener (which is at the player's position) and convolves it with the corresponding sound signal before playback. Propagation and convolution occur asynchronously, in order to ensure smooth, artifact-free audio. Note that propagation occurs at 5-10 FPS, while audio needs to be output in real-time (44100 samples per second). Therefore, IRs are updated independently of the streaming audio processing pipeline. As each frame of audio is pro-

Fig. 3: Benchmark scenes. From left to right: Sibenik (80K triangles), Movie Theater (120K triangles), Basement (548 triangles), Attic (1128 triangles). Basement and Attic are scenes used with our game engine integration, from Valve's Source engine SDK.

cessed, the latest available IR is used for convolution. This results in artifact-free audio at the expense of a slight lag in updating environmental audio effects.

We use the game engine's built-in ray tracer for computing ER, computing the direct response at each surface sample, and for final gathering. The transfer operators for each game scene are computed offline by exporting the scenes from the game engine and using our stand-alone preprocessor.

5.2 Performance

Table I shows the performance of the precomputation phase of our algorithm, as well as the storage requirements of the precomputed acoustic transfer operators. For each scene, we show the time spent in computing example echograms (column 4) and using them to construct a basis for echograms (column 5). We also show the time required to precompute the compressed acoustic transfer operator for each scene (columns 7 and 8). m refers to the number of example echograms used for basis construction, and n refers to the number of surface samples chosen over the surface of the scene. Finally, we show the storage required for the echogram basis in column 6, and for the transfer operators in column 9. As the table shows, our algorithm can compute compact acoustic transfer operators which require only a few tens of megabytes of storage within a few tens of minutes.

Table II demonstrates the performance of our two-pass run-time algorithm. For each scene, we show the time spent in each stage of the run-time algorithm. Column 5 shows the time taken to compute direct echograms from the source at each surface sample. Column 6 shows the time required to apply the transfer operator. Column 7 shows the time required to gather the higher-order echograms from each surface sample. Column 8 shows the time required to compute the early response using ray tracing. The table shows that our algorithm can efficiently compute higher-order reflections of sound, even for complex models consisting of tens or hundreds of thousands of triangles. Note that two of the scenes (Basement and Attic) are not shown in Table II. This is because these scenes are rendered within the game engine, so the corresponding performance numbers are not representative of our stand-alone OptiX-based implementation. In particular, the game engine's ray tracer is not optimized for ray-traced rendering workloads. As the accompanying video demonstrates, we still obtain sound propagation update rates of 5-10 FPS within the game engine.

5.3 Choice of Parameters

There are several parameters that need to be appropriately chosen when using our algorithm to compute and use acoustic transfer operators: s, the number of samples in an echogram; n, the number of surface samples; p, the number of example echograms used for basis construction; and m, the number of basis functions retained at run-time. We now discuss our choice of values for these parameters as used in our experiments.

Echogram Length The echogram length can be determined using the expected reverberation time of the scene. In our experiments, we used echograms that are 1s long, thus requiring $s = 48000$ samples to store each echogram, since our echograms are sampled at 48 kHz. As a result, our basis functions are also $s = 48000$ samples in size.

Surface Samples The number of surface samples to generate for each scene can be determined experimentally, guided by the fact that it is not possible to distinguish directions of incidence of sound with as much resolution as it is possible to distinguish directions of incidence of light [Tsingos 2007]. Audio clips generated with varying numbers of surface samples can be found in the accompanying video.

Basis Generation The values of p, i.e., the number of example echograms to use for basis construction, were arrived at through experiment. We used values of $p \in [64, 512]$ to generate the KLT basis. We then used plots of Frobenius norm error computed for the data matrix $\mathbf{A}$ to determine a sufficient value of p. Some resulting plots of Frobenius norm error are shown in Figure 5. Note that we randomly chose the surface sample pairs to generate the example echograms. For more complex environments with multiple connected rooms with a large amount of occlusion, it would be necessary to ensure (at least) that example echograms are computed between each pair of adjacent rooms.

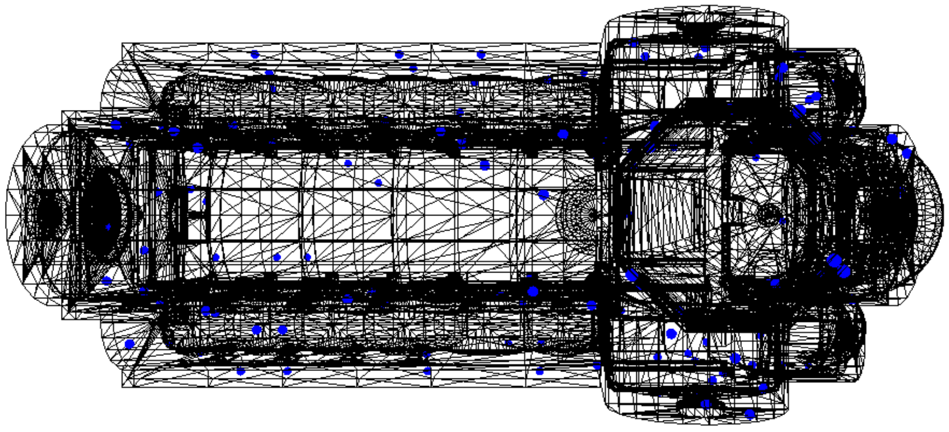
These plots were also used to determine sufficient values for m, i.e., the number of basis functions used at run-time. As the plots show, low values of m ($\approx 32 - 64$) can be used without significant Frobenius norm error. Figure 6 shows energy decay curves computed for varying values of m, compared with energy decay curves for reference path tracing solutions. The plots show that m provides a straightforward way to increase accuracy (at the cost of performance). Audio clips generated with varying numbers of KLT basis functions can be found in the accompanying video. These clips also show that low values of m can be used at run-time without significant degradation of audio quality.

6. ANALYSIS

In this section, we analyze our algorithm and compare it with prior precomputation-based methods.

6.1 Time and Storage Complexity

During precomputation, path tracing is performed from each of the n surface samples to determine echograms between each pair of surface samples. These n^2 echograms are then compressed into the Karhunen-Loeve basis with m coefficients. Hence, storing the compressed acoustic transfer operator requires $O(mn^2)$ memory. Pro-



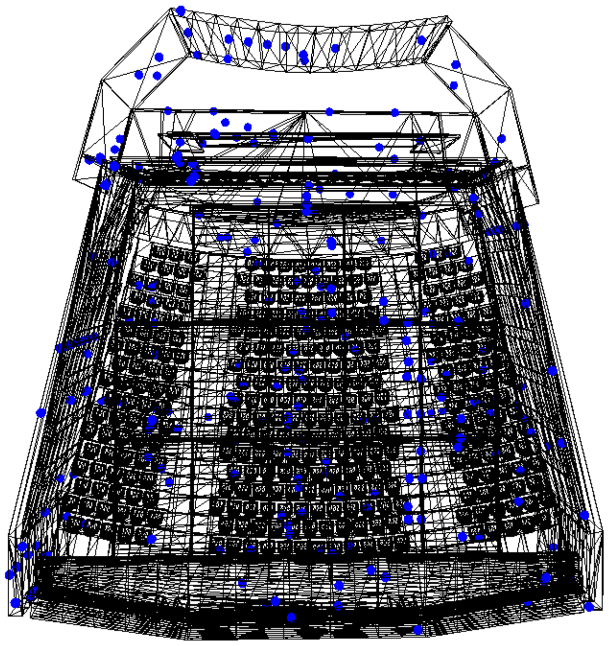


Fig. 4: Surface samples generated by our algorithm using area-weighted sampling. Sample points are shown in blue. Left: Sibenik, with 128 samples. Right: Movie Theater, with 256 samples.

jecting each echogram into the basis using a matrix-vector product requires $O(ms)$ time, where s is the number of time-domain samples used to represent the uncompressed echogram. Therefore, the total time required to compress the acoustic transfer operator is $O(m^2n^2s)$.

At run-time, the scatter and gather steps involve $O(n)$ work each; computing each convolution filter $H^k(t)$ requires $O(n^2)$ time to evaluate the double summation in Equation 9. The total time required to compute the convolution filters is, therefore, $O(mn^2)$. The basis functions $b^k(t)$ are stored in the frequency domain, hence we can use the Fourier theorem to quickly compute the convolutions in Equation 9 in $O(ms \lg s)$ time. This results in an overall run-time complexity of $O(mn^2 + ms \lg s)$ for each source.

6.2 Comparisons

There is extensive work on interactive sound propagation, including precomputation-based algorithms. We compare the main features of our algorithm with other methods in Figure 7.

Real-time Ray Tracing Recent developments in interactive ray tracing can be used to compute diffuse and specular reflections, as well as edge diffraction, for general dynamic scenes [Lentz et al.

Table I. : Performance and memory overhead of our precomputation algorithm.

Scene	Triangles	Echogram Basis				Transfer Operator		
		m	Propagation (s)	Construction (s)	Size (MB)	n	Computation (s)	Size (MB)
Sibenik	80K	256	130.29	3.34	46.9	128	481.87	16.0
Movie Theater	120K	256	186.45	0.75	46.9	256	2243.17	64.0
Attic	1128	64	27.81	0.09	11.7	64	105.66	1.0
Basement	548	64	23.52	0.07	11.7	64	93.66	1.0

Table II. : Performance of our run-time implementation.

Scene	Triangles	m	n	Scatter (ms)	Transfer (ms)	Gather (ms)	ER (ms)	**Total** (ms)	FPS
Sibenik	80K	32	128	0.4	149	42.5	1.4	193.3	5.2
Movie Theater	120K	16	256	0.6	77	82.8	2.1	162.5	6.1

2007; Taylor et al. 2009]. The former method is designed to run on powerful clusters which drive CAVE-like virtual environments; the latter can handle complex scenes containing hundreds of thousands of triangles on a modern desktop computer. However, these techniques are limited to computing the early response (2–6 orders) only. Most interactive techniques either use statistical estimation or other precomputation-based methods to model higher-order reflections.

Beam Tracing Methods based on beam tracing [Funkhouser et al. 1998] precompute a "beam tree", which represents potential propagation paths from a static source to a moving listener. While beam tracing can model higher-order specular reflections and edge diffraction, it is limited to static scenes and stationary sources.

Cells and Portals Many games and interactive applications use cell-and-portal scene decompositions. This can be used to precompute higher-order reflections of sound between moving sources and listeners using ray tracing [Foale and Vamplew 2007; Stavrakis et al. 2008] or image sources [Tsingos 2009], with relatively low memory overhead. These approaches typically store IRs sampled at a single position for each cell and/or portal encountered along the paths between the source and the listener in the scene's cell-and-portal graph. While image source gradients [Tsingos 2009] can be used for fine-grained interpolation of first-order reflections, higher-order reflections are typically modeled using reverberation decay profiles sampled coarsely throughout the scene. In contrast, our algorithm can sample echograms containing higher-order reflections at several positions on the surfaces of a general scene, and doesn't require any cell-and-portal decomposition. Moreover, our formulation allows higher-order edge diffraction to be included in the late response, unlike prior cell-and-portal-based methods.

Numerical Methods Precomputed Wave Simulation [Raghuvanshi et al. 2010] uses a numerical wave equation solver based on adaptive rectangular decomposition to compute IRs throughout a scene from several source positions sampled throughout the volume of the scene. At run-time, the IR for a moving source is obtained by interpolating between precomputed IRs of neighboring source position samples. Like any wave-based method, this approach works well for low-frequency sound sources, and can accurately model diffraction effects in such cases. The overall approach can take tens of minutes for precomputation, and generates datasets requiring hundreds of megabytes of storage. In contrast, our approach only samples the surfaces of the scene, and hence the memory requirements are about an order of magnitude lower than numerical methods which sample the volume of the scene. Furthermore, we can easily handle moving sources with dynamic direct shadowing effects.

Acoustic Radiance Transfer Frequency-domain acoustic radiance transfer [Siltanen et al. 2009] is the first precomputation-based algorithm for modeling sound propagation using the acoustic rendering equation. However, it is limited to static sources. There has been recent work on precomputing acoustic transfer operators based on the acoustic rendering equation [Antani et al. 2011]; however, it is limited to modeling only diffuse reflections. This approach also requires hundreds of megabytes to store transfer operators, since it uses Fourier coefficients to represent echograms. In contrast, our algorithm can represent acoustic transfer operators much more compactly due to our use of the Karhunen-Loeve basis. In fact, our transfer operators require $50 - 100$ times less storage than Fourier-domain acoustic transfer operators. This allows us to handle much more complicated scenes which require denser surface sampling than would be feasible with a Fourier-domain transfer operator approach.

Finally, since the Karhunen-Loeve transform sorts the basis in decreasing order of singular values, it is possible to trade off accuracy for performance by simply truncating the SVD and using fewer coefficients in these representations. This allows for a simple level-of-detail control over the acoustic simulation at run-time.

6.3 Limitations

Our approach is based on geometric sound propagation using path tracing. As a result, all the limitations of geometric acoustics apply to our method. In particular, our approach cannot accurately model low-frequency reflections and edge diffraction. Since the acoustic rendering equation is an energy-based formulation of sound propagation [Siltanen et al. 2007], phase-related effects, such as some cases of interference, cannot be modeled.

The run-time ray-tracing pass only computes early reflections (2–3 orders). Coupled with the fact that the radiance transfer pass assumes all surface samples are diffuse emitters, this implies that we cannot model higher-order purely specular reflections (such as flutter echoes) or higher-order purely diffracted paths (such as diffraction around complex curved surfaces).

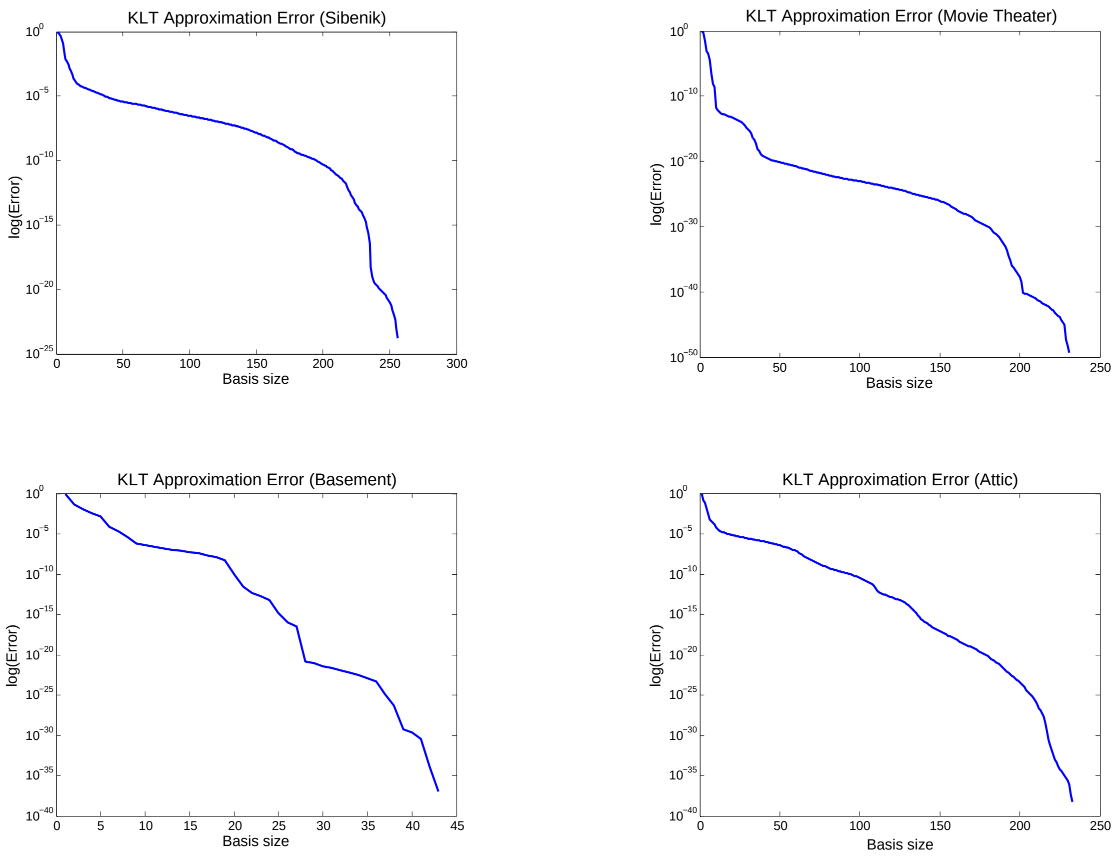


Fig. 5: Relative Frobenius norm error due to SVD truncation during KLT basis construction, for different scenes.

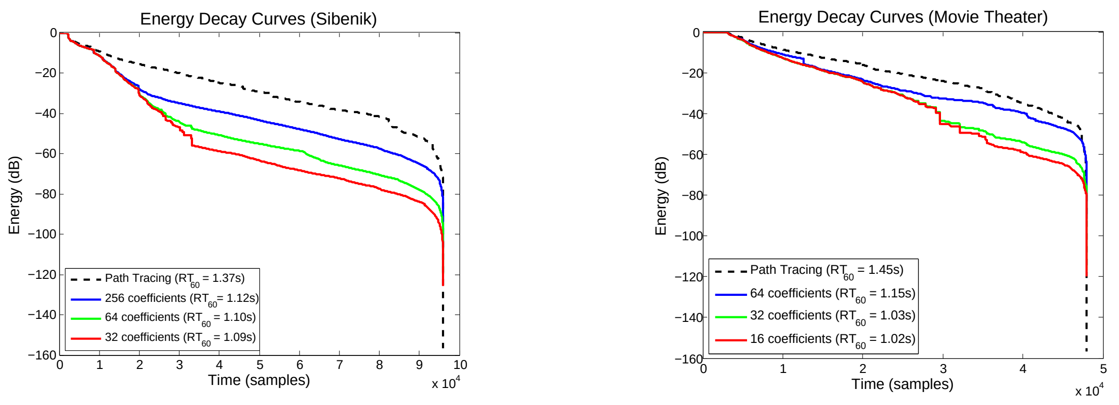
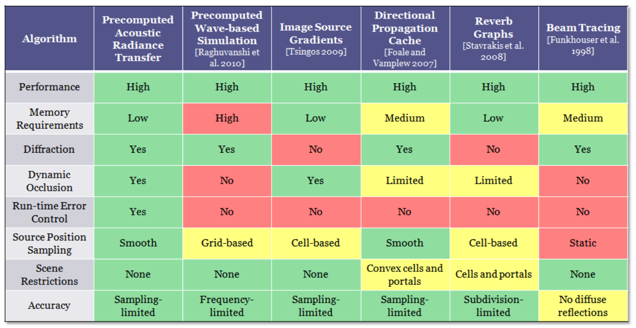


Fig. 6: Energy Decay Curves for different scenes, with varying numbers of KLT coefficients.

Our handling of dynamic objects is restricted to modeling direct "shadows" cast by a moving source due to moving objects. Since the transfer operators are computed over static surfaces only, we cannot model "indirect shadows", i.e., occlusion of reflected sound by moving objects. As a result, we cannot completely handle situations such as dynamic doors between static rooms.



Algorithm	Precomputed Acoustic Radiance Transfer	Precomputed Wave-based Simulation [Raghuvanshi et al. 2010]	Image Source Gradients [Tsingos 2009]	Directional Propagation Cache [Foale and Vamplew 2007]	Reverb Graphs [Stavrakis et al. 2008]	Beam Tracing [Funkhouser et al. 1998]
Performance	High	High	High	High	High	High
Memory Requirements	Low	High	Low	Medium	Low	Medium
Diffraction	Yes	Yes	No	Yes	No	Yes
Dynamic Occlusion	Yes	No	Yes	Limited	Limited	No
Run-time Error Control	Yes	No	No	No	No	No
Source Position Sampling	Smooth	Grid-based	Cell-based	Smooth	Cell-based	Static
Scene Restrictions	None	None	None	Convex cells and portals	Cells and portals	None
Accuracy	Sampling-limited	Frequency-limited	Sampling-limited	Sampling-limited	Subdivision-limited	No diffuse reflections

Fig. 7: Comparison of our approach with existing precomputation-based sound propagation algorithms.

Like other precomputation algorithms, our approach performs significant compression of the precomputed data in order to run on commodity hardware. As a result, our algorithm is not practical for detailed room acoustical analysis. It is designed for games and other interactive applications where approximate directional cues, echoes and reverberation can be dynamically updated to generate a plausible, immersive audio experience.

7. CONCLUSIONS AND FUTURE WORK

We have presented an efficient algorithm for computing sound propagation for interactive applications. The algorithm is a two-pass hybrid of ray tracing and radiance transfer algorithms. We have demonstrated that the algorithm can model high orders of reflection (specular as well as diffuse) and edge diffraction at near-interactive rates with low memory overhead.

Our algorithm can serve as the basis for much future research geared towards providing immersive audio in games and interactive applications. Firstly, it would be useful to investigate the possibility of using spherical harmonics to model the directional variation of acoustic radiance, while keeping memory overheads low. Another strategy for reducing memory requirements might be based on the structure of the acoustic transfer operator: in most game environments, occlusion would lead to clustering within the transfer matrix. These clusters would roughly correspond to cells in a cells-and-portals subdivision of the scene. Therefore, it might be useful to consider computing a per-cell acoustic transfer operator and modeling sound propagation between cells via the portals. The clusters may also be useful in optimizing the distribution of surface samples. In general, developing techniques for automatically choosing surface sample pairs for computing example echograms would be an interesting avenue for future research.

Since nearest-neighbor mapping is used for the hit-points of gather rays, there may be temporal error in interpolating echograms. This may lead to errors in the final echogram. It would be useful to develop delay-aware interpolation schemes to address these issues.

It would be very interesting to extend our basic precomputation framework to a pressure-based formulation by computing the sample-to-sample transfer operators using a numerical solver for the acoustic wave equation. This would essentially amount to a precomputation-based Monte Carlo solution of the Kirchoff integral formulation of the wave equation, and would result in increased accuracy in modeling wave phenomena such as diffraction.

Finally, it would be very useful to integrate our approach with a precomputation-based sound synthesis algorithm such as Precomputed Acoustic Transfer (PAT) [James et al. 2006] to perform efficient propagation of synthesized sound in interactive applications.

ACKNOWLEDGMENTS

The authors would like to thank Tapio Lokki, Ravish Mehra, Samuel Siltanen and Micah Taylor for their feedback. We would also like to thank Intel Corporation for their support, and Valve Corporation for permission to use the Source SDK and Half-Life 2 artwork for the Basement and Attic scenes. This work was supported in part by ARO Contract W911NF-10-1-0506, NSF awards 0917040, 0904990 and 1000579, and RDECOM Contract WR91CRB-08-C-0137.

REFERENCES

ALARCAO, D., SANTOS, D., AND COELHO, J. L. B. 2009. An auralization system for real time room acoustics simulation. In *Tecniacustica 2009*.

ALLEN, J. B. AND BERKLEY, D. A. 1979. Image method for efficiently simulating small-room acoustics. *Journal of the Acoustical Society of America 65,* 4, 943–950.

ANTANI, L., CHANDAK, A., TAYLOR, M., AND MANOCHA, D. 2011. Direct-to-indirect acoustic radiance transfer. *IEEE Transactions on Visualization and Computer Graphics (to appear)*.

BERTRAM, M., DEINES, E., MOHRING, J., JEGOROVS, J., AND HAGEN, H. 2005. Phonon tracing for auralization and visualization of sound. In *IEEE Visualization 2005*. 151–158.

BONNEEL, N., DRETTAKIS, G., TSINGOS, N., VIAUD-DELMON, I., AND JAMES, D. L. 2008. Fast modal sounds with scalable frequency-domain synthesis. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2008) 27,* 3.

BOTTELDOOREN, D. 1995. Finite difference time domain simulation of low frequency room acoustic problems. *Journal of the Acoustical Society of America 98,* 8, 3302–3308.

CHADWICK, J., AN, S., AND JAMES, D. L. 2009. Harmonic shells: A practical nonlinear sound model for near-rigid thin shells. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2009) 28,* 5.

CHANDAK, A., ANTANI, L., TAYLOR, M., AND MANOCHA, D. 2009. Fastv: From-point visibility culling on complex models. *Computer Graphics Forum (Proceedings of the Eurographics Symposium on Rendering) 28*, 1237–1246.

CISKOWSKI, R. AND BREBBIA, C. 1991. *Boundary Element methods in acoustics*. Computational Mechanics Publications and Elsevier Applied Science.

FOALE, C. AND VAMPLEW, P. 2007. Portal-based sound propagation for first-person computer games. In *Australasian Conference on Interactive Entertainment 2007*. 9:1–9:8.

FUNKHOUSER, T., CARLBOM, I., ELKO, G., PINGALI, G., SONDHI, M., AND WEST, J. 1998. A beam tracing approach to acoustic modeling for interactive virtual environments. In *SIGGRAPH 1998*. 21–32.

GORAL, C. M., TORRANCE, K. E., GREENBERG, D. P., AND BATTAILE, B. 1984. Modeling the interaction of light between diffuse surfaces. *Computer Graphics (Proceedings of SIGGRAPH 1984) 18,* 3, 213–222.

HAŠAN, M., PELLACINI, F., AND BALA, K. 2006. Direct-to-indirect transfer for cinematic relighting. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2006) 25,* 3, 1089–1097.

IHLENBURG, F. 1998. *Finite Element Analysis of Acoustic Scattering*. Springer Verlag AG.

JAMES, D. L., BARBIČ, J., AND PAI, D. K. 2006. Precomputed acoustic transfer: output-sensitive, accurate sound generation for geometrically complex vibration sources. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2006) 25,* 3, 987–995.

KAJIYA, J. T. 1986. The rendering equation. *Computer Graphics (Proceedings of SIGGRAPH 1986) 20,* 4, 143–150.

KAPRALOS, B., JENKIN, M., AND MILIOS, E. 2004. Sonel mapping: acoustic modeling utilizing an acoustic version of photon mapping. In *IEEE International Workshop on Haptics Audio Visual Environments and their Applications 2004*.

KRISTENSEN, A. W., AKENINE-MÖLLER, T., AND JENSEN, H. W. 2005. Precomputed local radiance transfer for real-time lighting design. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005) 24,* 3, 1208–1215.

KUTTRUFF, H. 1991. *Room Acoustics*. Elsevier Science Publishing Ltd.

KUTTRUFF, H. 1995. A simple iteration scheme for the computation of decay constants in enclosures with diffusely reflecting boundaries. *The Journal of the Acoustical Society of America 98,* 1, 288–293.

KUTTRUFF, H. K. 1993. Auralization of Impulse Responses Modeled on the Basis of Ray-Tracing Results. *Journal of the Audio Engineering Society 41,* 11 (November), 876–880.

LAINE, S., SILTANEN, S., LOKKI, T., AND SAVIOJA, L. 2009. Accelerated beam tracing algorithm. *Applied Acoustics 70,* 1, 172–181.

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 Transactions on Graphics (Proceedings of SIGGRAPH 2008) 27,* 3, 1–9.

LENTZ, T., SCHROEDER, D., VORLANDER, M., AND ASSENMACHER, I. 2007. Virtual reality system with integrated sound field simulation and reproduction. *EURASIP Journal of Applied Signal Processing 2007,* 1.

LOÈVE, M. 1978. *Probability Theory Vol. II.* Springer-Verlag.

MOECK, T., BONNEEL, N., TSINGOS, N., DRETTAKIS, G., VIAUD-DELMON, I., AND ALOZA, D. 2007. Progressive perceptual audio rendering of complex scenes. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*.

NOSAL, E.-M., HODGSON, M., AND ASHDOWN, I. 2004. Improved algorithms and methods for room sound-field prediction by acoustical radiosity in arbitrary polyhedral rooms. *Journal of the Acoustical Society of America 116,* 2, 970–980.

RAGHUVANSHI, N. AND LIN, M. C. 2006. Interactive sound synthesis for large scale environments. In *Symposium on Interactive 3D Graphics and Games*.

RAGHUVANSHI, N., NARAIN, R., AND LIN, M. C. 2009. Efficient and accurate sound propagation using adaptive rectangular decomposition. *IEEE Transactions on Visualization and Computer Graphics 15*, 789–801.

RAGHUVANSHI, N., SNYDER, J., MEHRA, R., LIN, M., AND GOVINDARAJU, N. 2010. Precomputed wave simulation for real-time sound propagation of dynamic sources in complex scenes. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2010) 29,* 4, 68:1 – 68:11.

SAVIOJA, L., HUOPANIEMI, J., LOKKI, T., AND VÄÄNÄNEN, R. 1999. Creating interactive virtual acoustic environments. *Journal of the Audio Engineering Society 47,* 9, 675–705.

SAVIOJA, L., RINNE, T., AND TAKALA, T. 1994. Simulation of room acoustics with a 3-D finite difference mesh. In *International Computer Music Conference*. 463–466.

SILTANEN, S., LOKKI, T., KIMINKI, S., AND SAVIOJA, L. 2007. The room acoustic rendering equation. *Journal of the Acoustical Society of America 122,* 3, 1624–1635.

SILTANEN, S., LOKKI, T., AND SAVIOJA, L. 2009. Frequency domain acoustic radiance transfer for real-time auralization. *Acta Acustica united with Acustica 95*, 106–117.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2002) 21,* 3, 527–536.

STAVRAKIS, E., TSINGOS, N., AND CALAMIA, P. 2008. Topological sound propagation with reverberation graphs. *Acta Acustica united with Acustica*.

STEPHENSON, U. M. 2010. An analytically derived sound particle diffraction model. *Acta Acustica united with Acustica 96*, 1051–1068.

STEPHENSON, U. M. AND SVENSSON, U. P. 2007. An improved energetic approach to diffraction based on the unvertainty principle. *19th International Congress on Acoustics (ICA)*.

SUMMERS, J. E., TORRES, R. R., AND SHIMIZU, Y. 2004. Statistical-acoustics models of energy decay in systems of coupled rooms and their relation to geometrical acoustics. *Journal of the Acoustical Society of America 116,* 2, 958–969.

SVENSSON, U. P., FRED, R. I., AND VANDERKOOY, J. 1999. An analytic secondary source model of edge diffraction impulse responses. *Journal of the Acoustical Society of America 106*, 2331–2344.

TAYLOR, M. T., CHANDAK, A., ANTANI, L., AND MANOCHA, D. 2009. Resound: interactive sound rendering for dynamic virtual environments. In *ACM Multimedia 2009*. 271–280.

TSINGOS, N. 2007. Perceptually-based auralization. In *International Congress on Acoustics*.

TSINGOS, N. 2009. Precomputing geometry-based reverberation effects for games. In *Audio Engineering Society Conference: Audio for Games*.

TSINGOS, N., FUNKHOUSER, T., NGAN, A., AND CARLBOM, I. 2001. Modeling acoustics in virtual environments using the uniform theory of diffraction. In *SIGGRAPH 2001*. 545–552.

TSINGOS, N., GALLO, E., AND DRETTAKIS, G. 2004. Perceptual audio rendering of complex virtual environments. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2004) 23,* 3.

TSINGOS, N. AND GASCUEL, J.-D. 1997. A general model for the simulation of room acoustics based on hierachical radiosity. In *ACM SIGGRAPH 97 Visual Proceedings*.

WALLACE, J. R., COHEN, M. F., AND GREENBERG, D. P. 1987. A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods. *Computer Graphics (Proceedings of SIGGRAPH 1987) 21,* 4, 311–320.

WANG, YE; VILERMO, M. 2003. Modified discrete cosine transform: Its implications for audio coding and error concealment. *J. Audio Eng. Soc 51,* 1/2, 52–61.