Precomputed Wave Simulation for Real-Time Sound Propagation of Dynamic Sources in Complex Scenes

Nikunj Raghuvanshi*† John Snyder* Ravish Mehra† Ming Lin† Naga Govindaraju*

* Microsoft Research † University of North Carolina at Chapel Hill

Abstract

We present a method for real-time sound propagation that captures all wave effects, including diffraction and reverberation, for multiple moving sources and a moving listener in a complex, static 3D scene. It performs an offline numerical simulation over the scene and then applies a novel technique to extract and compactly encode the perceptually salient information in the resulting acoustic responses. Each response is automatically broken into two phases: early reflections (ER) and late reverberation (LR), via a threshold on the temporal density of arriving wavefronts. The LR is simulated and stored in the frequency domain, once per room in the scene. The ER accounts for more detailed spatial variation, by recording a set of peak delays/amplitudes in the time domain and a residual frequency response sampled in octave frequency bands, at each source/receiver point pair in a 5D grid. An efficient run-time uses this precomputed representation to perform binaural sound rendering based on frequency-domain convolution. Our system demonstrates realistic, wave-based acoustic effects in real time, including diffraction low-passing behind obstructions, sound focusing, hollow reverberation in empty rooms, sound diffusion in fully-furnished rooms, and realistic late reverberation.

Figure 1: Train station scene from Valve's Source™ game engine SDK (`http://source.valvesoftware.com`). Our method performs real-time auralization of sounds from dynamic agents, objects and the player interacting in the scene, while accounting for perceptually important effects such as diffraction low-pass filtering and reverberation.

1 Introduction

Acoustic propagation gives a visceral and immersive sense of a virtual environment. It provides cues that cannot be obtained visually, alerting the listener to activity behind walls or his head with information about the distance, direction, and indirectness of path to this activity's location. Realistic sound propagation must simulate two interrelated wave effects: diffraction and scattering. Many gross acoustic effects arise from diffraction. Smooth reduction in volume as one walks through a doorway or behind a building is due to high-order diffraction. Smooth loudness variation in the sound field of a furnished room results from diffracted scattering off its complex geometry. Late reverberation arises from very high order scattering. Neglecting diffraction leads to clicking artifacts and incoherent loudness fluctuations, as well as unnatural termination of sound before it reaches occluded regions.

Capturing these effects in real-time within a complex 3D environment presents a challenging problem for current approaches. By "complex", we mean "containing acoustically relevant scene features at length scales down to centimeters" (see Figure 5(a)). Sound frequencies up to 5 kHz scatter diffusely from "rough" surface features at centimeter scales and are mostly unresponsive to finer detail while higher frequencies are more strongly absorbed as they travel through air or scatter off surfaces [Kuttruff 2000, p. 27]. Our method is limited to *static* scene geometry which largely determines how sound propagates in many common architectural/virtual spaces, thus covering a useful set of applications.

Existing methods to solve this problem are *geometric*, and trace rays or beams from the source into the scene to collect their contributions near the listener. These methods have a number of limitations, although some can handle dynamic scenes unlike ours. Methods based on conservative beam tracing split the beam when it encounters geometric discontinuities, leading to slow performance in complex scenes and exponential scaling in the reflection order [Funkhouser et al. 2004; Chandak et al. 2008]. A related problem arises for ray-tracing methods, which must sample a huge number to capture multiple-bounce propagation and avoid missed contributions. In practice, meshes must be simplified and detailed interior objects replaced by simpler proxies to achieve interactive performance. Automatic methods to do this are a nascent area of research [Siltanen 2005]. Handling diffraction with geometric approaches is challenging, especially for complex scenes [Calamia 2009]. At long sound wavelengths, current geometric edge-diffraction models either ignore global effects from complex geometry or require too much computation to run interactively.

Wave-based simulation offers a solution to many of these problems, but its computational burden is at least 3 orders of magnitude too slow for on-the-fly evaluation. Our solution is to precompute an off-line, wave-based simulation for a given static environment in terms of its 7D spatially-varying acoustic *impulse response* (IR), $S(t, p_s, p_r)$, where t is time, p_s is source position, and p_r is receiver position. This can be reduced to a 6D problem by restricting the listener to a 2D plane in the scene and leveraging acoustic reciprocity. Even so, the computational and storage requirements are huge. With a recent, fast technique [Raghuvanshi et al. 2009], sim-

ulating a one-second response at a single source position in a scene of volume $12^3 m^3$ bandlimited to frequencies up to 1kHz requires 30 minutes of computation and generates 24GB of data. Simulating over many different source locations becomes intractable.

To make our approach practical, decrease offline computation, and reduce run-time computation and memory, we exploit human auditory perception using the well-known *ER/LR perceptual model* [Kuttruff 2000, p. 98]. Following standard practice, the IR is divided into two time intervals. The *early reflections* (ER) phase comprises sparse, high-energy wavefronts which are detected and processed individually in human perception, followed smoothly by the *late reverberation* (LR) phase, comprising dense arrival of many low-amplitude wavefronts which are fused perceptually to infer aggregate properties such as the decay envelope. Perceptually, the ER conveys a sense of location, while the LR gives a global sense of the scene – its size, level of furnishing and overall absorptivity.

We extract the LR by performing a 1-2 second simulation from a source placed at the room's centroid and analyzing its IR at a receiver in the same place. This result determines the time length of the ER, as well as the per-room LR filter called the *late reverberation impulse response* (LRIR). ER length is 50-200ms for rooms of typical size. We then run simulations with the source placed at all sample points on the 2D grid described above. For each source position, the resulting *early reflections impulse responses* (ERIRs) are recorded at sample points over the scene's 3D volume. This two-step approach reduces the time duration of expensive ER simulations from 1-2s to 50-200ms, saving at least $10\times$ in precomputation time and runtime storage.

We then extract and compactly encode ERIRs recorded at each source-listener pair using a novel, perceptually-based technique. ERIRs contain distinct pressure peaks in time, corresponding to wavefronts arriving at the listener. We extract the highest-energy peaks (yielding roughly 30-40 in our examples) and store their delay and attenuation in the time domain. Peak data is wide-band information that captures reverberation and interference but ignores diffraction low-pass filtering, so we add a residual *frequency trend* representation which restores these effects and allows them to be plausibly extrapolated to frequencies beyond the ones simulated. Compared to direct storage of simulated ERIRs, this reduces memory use tenfold while encoding all-frequency information.

A novel run-time system propagates source sounds based on this precomputed data. Spatial interpolation of IRs is required as sources and listener move; these positions are subsampled in our system to about 1m. Straightforward waveform interpolation yields erroneous timbral coloration and "gurgling" artifacts; we interpolate in the encoded space of peak times and amplitudes to avoid such artifacts. The ERIR is then *spatialized* to the listener's left and right ears, the LRIR added, and the source sound convolved in the frequency domain with the final IRs for each ear.

Our work is the first to achieve real-time, wave-based sound propagation, including high-order, frequency-dependent sound scattering and diffraction, for moving sources and listener in complex environments (see Figure 1). We propose a novel decomposition of the computation into three parts: an off-line wave-based simulation in the static scene, an off-line perceptually-based encoding of the resulting IRs, and a run-time engine. We automatically compute from simulation both the ER and LR and separate them based on echo density. Our new IR encoding extracts peaks and a residual frequency trend. Computing this information from the bandlimited results of numerical simulation, extrapolating it in a perceptually plausible manner, employing a grid-based sampling for moving sources and listener, and using one which exploits reciprocity are all novel contributions. Our run-time system then efficiently decodes, interpolates, and convolves source sounds with these encoded IRs in the frequency domain, supporting tens of moving sources and a moving listener in real time.

2 Related Work

An introduction to acoustics can be found in the classic texts [Kuttruff 2000; Kinsler et al. 2000] and auralization in [Lokki 2002]. Techniques for simulating sound propagation may be classified into *geometric acoustics* (GA) and *numerical acoustics* (NA).

Geometric acoustics GA assumes rectilinear propagation of sound waves, a valid approximation only if the wavelength is much smaller than the local feature size. Extensive research and commercial software apply geometric methods to room acoustic prediction; here we focus on interactive approaches.

Beam tracing [Funkhouser et al. 2004; Antonacci et al. 2004] bundles rays to compute their interaction with scene geometry. For fixed sources, a beam-tree is built for the scene in an offline processing step, and then used at run-time to render acoustic propagation paths to a moving listener. The method handles low-order specular reflections. Diffraction can also be included, via the *geometric theory of diffraction* [Tsingos et al. 2001], but is applicable only for edges much larger than the wavelength.

More recent work [Chandak et al. 2008] supports moving sources without guaranteeing exact visibility. Minimal preprocessing is required and dynamic geometry can be handled; diffraction is disregarded. [Taylor et al. 2009] includes diffraction effects by using the Biot-Tolstoy-Medwin (BTM) theory of diffraction [Calamia 2009]. The method discretizes diffraction edges and integrates contributions from each element. Second-order diffraction must consider all pairs of elements and quickly becomes intractable at higher orders. Even finding all potential diffraction edges is computationally challenging in complex scenes.

Numerical acoustics NA directly solves the *wave equation* governing physical propagation of sound in a scene. With enough computation, all wave phenomena can be captured, including diffraction and scattering in complex scenes. Numerical approaches are insensitive to the complexity and polygon count of a scene and instead scale with its physical dimensions.

Expressing the problem in the frequency domain yields the Helmholtz equation. Many interactive sound applications rely on this formulation, to capture the frequency-dependent directional distribution of sound emitted by an object [James et al. 2006], or the frequency-dependent local scattering of sound off complex surfaces [Tsingos et al. 2007]. The latter considers only first-order scattering and neglects diffraction and time-domain effects. In general, Helmholtz solvers are efficient for simulating at one or a few frequencies, but become much less efficient than time-domain solvers for capturing transient information such as propagation delays.

Finite difference time domain (FDTD) methods have been used in room acoustics for more than a decade [Botteldooren 1995]. A scene's 3D air volume is uniformly discretized and the pressure field in each cell is solved with a time-marching scheme. Simulation from a single source location yields the acoustic response at all listener cells in the scene. FDTD is computationally intensive, scaling linearly in the volume and in the fourth power of maximum simulated frequency. Advances in computational power have prompted recent investigation into FDTD for medium-sized scenes [Sakamoto et al. 2006].

A recent technique [Raghuvanshi et al. 2009] achieves speedups

of nearly 100 times over FDTD by partitioning the scene's interior volume into rectangular regions and exploiting an analytic solution within each. The technique simulates large, static 3D scenes in minutes, and forms the basis for our precomputed simulation.

Precomputed methods for interactive auralization A division of computation into offline and run-time parts similar to ours is used in [James et al. 2006], but for rendering sounds at select frequencies from an impulsively struck, precomputed object rather than propagating arbitrary, broadband sounds between moving sources and a receiver inside a precomputed 3D scene. We assume sources are monopole (point sources), but our system could easily be extended to handle multiple monopole and dipole sources.

Previous work has also dealt with precomputed IRs. [Pope et al. 1999] proposes sampling IRs on a dense grid but does not describe how moving sources might be handled, nor how to represent or interpolate IRs for handling the moving listener.

[Tsingos 2009] uses manually-specified source/listener location pairs (roughly one location per room) to sparsely sample spatially-varying IRs. The ER component is approximated using globally computed (virtual) image sources from these locations. The overall approach is well-suited for scenes with large, specular reflectors but insufficient for handling spatially detailed acoustic effects that we target such as diffracted shadowing behind walls or focusing/diffraction/scattering effects off complex geometry.

ER/LR separation LR effects are typically specified through *artificial reverberation*, based on IIR (Infinite Impulse Response) filters whose parameters (e.g. exponential decay time and peak density) are hand-tuned to match scene geometry. Based on a well-known idea [Gardner 1998], [Min and Funkhouser 2000] separate the ER and LR using echo density as we do, but handle the LR with artificial reverberation. Recent geometric techniques use ray-tracing to explicitly compute the LR from geometry [Stavrakis et al. 2008; Tsingos 2009]. However, the former doesn't account for the ER at all while the latter segments based on a user-specified number of initial wavefronts arriving at the receiver. Our approach automatically computes and segments both the ER and LR using a wave-based simulation on the scene geometry – the LR is sampled sparsely, while the ER is sampled densely in space.

Peak detection Peak detection is proposed in [Emerit et al. 2007] to encode a single acoustic response with the goal of accelerating run-time convolution of input signals during playback of MPEG streams. Neither spatial interpolation nor compactness is a concern. Our method for IR encoding differs substantially: we resolve peaks possibly merged by the bandlimited simulation to recover individual peaks; [Emerit et al. 2007] segments the input IR into a sparse (~10) set of coalesced "peaks" and computes detailed spectral properties for each in 64 frequency sub-bands. This requires at least $10\times$ more memory than our approach.

IR interpolation Dynamic Time Warping (DTW) [Masterson et al. 2009] finds correlations between two signals of lengths N and M in $O(NM)$ time and could be used to interpolate simulated acoustic responses. Our representation is based on sparse peaks which can be correlated and interpolated much more efficiently.

3 Perception in Acoustic Spaces

Sound propagation in an arbitrary acoustic space from a source location, p_s, to a receiver location, p_r, is completely characterized by the corresponding acoustic *impulse response* (IR), defined as the sound received at p_r due to an ideal Dirac-delta impulse emitted at p_s. The propagated result of an *arbitrary* source sound is then found by convolving it with this IR. Simulating and storing IRs at all possible sample pairs, p_s and p_r, is prohibitively expensive. We attack this problem by exploiting human perception, based on established ideas in room acoustics [Kuttruff 2000]. This analysis breaks IRs down into two distinct time phases: early reflections (ER) followed by late reverberation (LR). We briefly discuss each below.

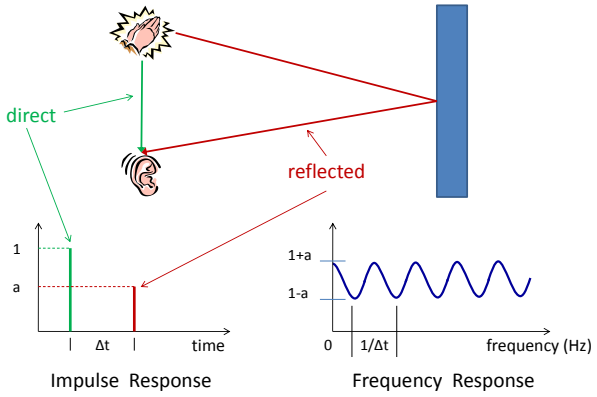

Figure 2: Impulse response (IR) and corresponding frequency response (FR) for a simple scene with one reflection.

Note that the propagated sound received by a human listener depends both on the scene's IR and on the original sound emitted at the source. A listener does not perceive the acoustic IR directly. Nevertheless, a specific IR usually leads to perceptible characteristics when applied to a wide variety of source sounds, prompting the shorthand "The IR sounds like this." The more impulsive the source sound, the more it generally reveals about the scene's acoustic response, but the perceptual effect of even a continuous source sound like music is substantially changed by propagation.

3.1 Early Reflections (ER)

The ER phase is characterized by a sparse set of high-amplitude pressure peaks, and captures position-dependent variation in loudness, spectral coloration and echoes. Consider the scene shown in Figure 2. The scene's propagated response can be examined in two ways: in the time domain, as the impulse response shown on the bottom-left in the figure, and equivalently, in the frequency domain, as the complex *frequency response* (FR) whose magnitude is shown on the bottom-right. The IR is a set of two impulse peaks, separated in time by Δt. Assuming the amplitudes of the peaks to be 1 and a, the FR is a comb-filter that oscillates between $1+a$ and $1-a$ with "between-teeth" bandwidth of $1/\Delta t$.

When Δt is above 60-70ms, known as the "echo threshold" [Litovsky et al. 1999], we perceive a distinct echo, especially for impulsive sources. If the delay is much smaller, the peaks fuse in our perception. However, the corresponding FR's between-teeth bandwidth increases (as $1/\Delta t$), and our auditory system is able to extract the selective attenuation of certain frequencies. The result is perceived as a comb-filtered, "colored" sound, typical of small spaces such as a narrow corridor or shower. Following [Kuttruff 2000, p. 203], if the delay is larger than 20-25ms (FR between-teeth bandwidth less than 50Hz), coloration becomes inaudible due to our ears' limited frequency discrimination [Halmrast 2007]. We thus assume that errors in peak delays and amplitudes are tolerable as long as they preserve FR features up to a resolution of 50Hz. Between these thresholds (25ms-60ms), our perception transitions

from coloration to echoes and sounds "rough".

ERs in real scenes are determined by numerous interactions with the boundary. In an empty room, the ER has low temporal density and audible coloration. In a furnished room, sound is scattered more highly, yielding an ER with dense, smaller peaks and leading to a warmer sound. The ER also depends on occlusion, as the sound field diffracts around obstacles leading to spatial variation in its loudness and frequency response.

3.2 Late Reverberation (LR)

After undergoing many boundary interactions, the sound field enters the LR phase, becoming dense and chaotic in time and space. The LR between any source and receiver consists of numerous small-amplitude peaks. These peaks are not individually perceived, only their overall decay envelope which causes reverberations to fade out over time. The decay curve stays largely constant within a room [Kuttruff 2000, p. 209] and depends on its global geometry.

An important feature of our approach is that it directly simulates the LR, and so captures its decay curve automatically from scene geometry. This is possible because numerical simulation scales linearly in simulated time duration and is insensitive to the order of boundary interactions. Computing the LR from scene geometry also ensures that the ER and LR match, preserving realism. The LR is characterized by peak density: 1000 peaks per second indicate a fully-developed LR phase [Gardner 1998]. We use a value of 500 peaks per second as the LR's onset.

4 Acoustic Precomputation

The precomputation begins by running an *LR probe* simulation, which places a source at the centroid of each room in the scene as described in Section 4.2. These LR simulations determine the time at which the ER phase transitions to the LR phase, denoted T_{ER}. T_{ER} depends on the room size. Larger spaces require more time for the sound field to become stochastic; i.e., to reach the required peak density. This is detected automatically by processing on the LR probe data and taking the maximum over all rooms in the scene. We obtain values of 50-200ms for our experimental scenes.

In many applications, the listener's position is more constrained than the sources'. This can be exploited to reduce memory and preprocessing time. Uniform samples are placed over a region representing possible locations of the listener at runtime, but are considered as sources in the precomputed simulation. The principle of acoustic reciprocity means that we can reverse the simulated sense of source and listener at run-time. Simulations are then run over multiple source positions, p_s, for time length T_{ER}, to record the spatially-varying ER impulse response (ERIR). For each source, the resulting time-varying field is recorded at samples, p_r, uniformly distributed over the entire 3D scene volume. Each source and receiver sample generates a time-varying sound signal, $s(t)$, which is then processed as described in Section 4.3 to extract its relevant content and yield a compact result.

4.1 Numerical Simulation

We use the simulator described in [Raghuvanshi et al. 2009]. It is fast yet still able to propagate an input signal over long distances and many reflections without distorting the waveform due to numerical dispersion errors that arise in finite difference approaches. This is crucial in our subsequent processing – dispersion introduces artificial ringing and leads to the detection of spurious peaks and artifacts in the final auralization.

Input to the simulator is the room geometry, absorption parameters (potentially varying per triangle), grid resolution, source reference signal, and source location. The simulator then generates the resulting sound at each time-step and at all grid cell centers in 3D. A single run of the simulator thus captures the whole room's acoustic response at every possible listener location, from a fixed source location. Currently, our simulator models spatially-varying but not frequency-dependent sound absorption. Including frequency-dependent absorption requires a more sophisticated simulator but otherwise does not affect our approach.

The reference input signal propagated from the source location is a Gaussian pulse of unit amplitude, given by:

$$G(t) = \exp\left(-\frac{(t-5\sigma)^2}{\sigma^2}\right), \tag{1}$$

where $\sigma = (\pi\eta_f)^{-1}$ and η_f is the simulation's frequency limit, typically 1000Hz. This function's Fourier transform is a broadband Gaussian spanning all frequencies from 0 to η_f.

Bandlimited simulation To keep preprocessing time and storage manageable, we limit our simulation to frequencies up to η_f. A practical limit used in most of our examples is η_f = 1000Hz, with a corresponding grid resolution of about 10cm. Information at higher frequencies must be extrapolated. This is a limitation of the precomputation only and becomes less of a problem as computational power increases; the run-time supports the results of higher-frequency simulation without change or additional cost. Frequencies in the range 100Hz to 5000Hz are most important perceptually for room acoustics [Kuttruff 2000, p. 27]. Though humans can hear higher frequencies, they quickly attenuate in air and are highly absorbed by most surfaces.

Our simulation's frequency gap between 1kHz and 5kHz yields two major types of error. First, it limits the time resolution at which peaks can be separated to 500μs. Merging a high-amplitude peak with another which follows closer than this limit eliminates coloration effects. Our auralizations thus neglect some high-frequency coloration. Second, diffraction information is unavailable beyond 1kHz. Our method for extrapolating the frequency trend in Section 4.3.2 produces a plausible, natural-sounding result in this gap.

Spatial sampling The spatial accuracy at which we perceive sound fields is limited. A human head acts as a diffractive occluder which destroys interference patterns at smaller spatial resolution. We thus believe that its size (~10cm) is a guide to the smallest sampling resolution required for convincing auralization in typical scenes. To our knowledge the question is an open research problem.

Our simulator generates spatial samples for the receiver near this resolution, but we further downsample them by a factor of 8× in each dimension to reduce run-time memory requirements. Typical receiver grid resolution is about 1m. Using the interpolation method proposed in the next section, subsampled IRs still provide artifact-free results without clicks or gurgling, as demonstrated in the accompanying video. The sound field changes convincingly as the sources and listener move around obstructions. Increasing spatial resolution would use more memory but would not significantly affect runtime performance, since the expense is dominated by FFTs rather than spatial interpolation.

4.2 LR Processing

The purpose of the LR probe simulation is to compute the duration of the ER phase, T_{ER}, and the LRIR. We run a peak detector on the

simulated probe response to generate a set of peak delay/amplitude data, $\{t_i, a_i\}$, sorted on delays, t_i. This is shown in the upper right of Figure 3. Simulations bandlimited to 1kHz imply a quantization of peak delays at roughly 500μs. Such quantization can introduce artificial periodicities and lead to audible but false coloration. We perturb peak times by a random value in the range -500 to +500 μs. Any periodicities above a few milliseconds, including "fluttering" effects in the LR, are still preserved.

Peak detection We use a straightforward algorithm to detect peaks in a signal. It scans the time-dependent signal for maxima and minima by comparing each value with both its neighbors. Detected signal extrema are recorded in terms of their time and amplitude. Such a simple detector is sensitive to noise and would be inappropriate for physically-measured IRs. It works well however on the results of noise-free bandlimited simulations.

LR duration After peak detection, we conservatively truncate the LR when it has decayed by 90dB. The resulting duration, T_{LR}, is calculated as follows. We first construct the signal's decay envelope, $d(t)$, from the peaks detected previously via

$$d(t) = 10 \times \log({a_i}^2), \;\; t_i \leq t < t_{i+1} \tag{2}$$

This operation converts pressure amplitudes to intensities and fills out spaces between peaks. We then smooth in time by applying a moving average filter on $d(t)$ of width 40ms. T_{LR} is given by the smallest t such that $d(0) - d(t) > 90$. Peaks with delays beyond T_{LR} are then removed.

ER duration As explained in Section 3, we calculate T_{ER} as the time when peak density reaches 500 peaks per second. Starting from time t = 0, we consider all peaks within the range $[t, t + 20ms]$ and count those within 20 dB of the highest intensity peak in the interval. If the number of such peaks exceeds 500 per second $\times$ 0.02 seconds = 10 peaks, we record this t as the ER duration. Otherwise, we increase t by 20ms and continue until the criterion is satisfied. Since peak density increases with time, this procedure terminates.

Increasing LR peak density Our frequency-bandlimited simulator can detect no more than 1000 peaks per second. While this is sufficient for music and voice, it is not for impulsive sources such as gunshots or footsteps, which require roughly 10,000 peaks per second [Gardner 1998]. Following practice in artificial reverberators, we add peaks stochastically to increase the density by a factor of 10. We do this while preserving the decay envelope, as follows.

We first compute a *densification factor*, $F(t)$, $T_{ER} \leq t < T_{LR}$ as

$$F(t) = 10 \left(\frac{t - T_{ER}}{T_{LR} - T_{ER}} \right)^2. \tag{3}$$

F(t) builds quadratically in time from 0 to a terminal value of 10, based on the expected quadratic growth in peak density in physical scenes [Kuttruff 2000, p. 98]. Next, for each peak $\{t_i, a_i\}$ in the LR after time T_{ER}, $\lfloor F(t_i) \rfloor$ additional peaks are added with amplitudes $a_i \times \text{rand}(-1, 1)$. This preserves the simulated decay envelope and yields the final LRIR for use at runtime.

Handling multiple rooms For scenes comprising multiple rooms, we perform an LR probe simulation and store the LRIR separately for each room. The user manually marks volumetric scene regions as rooms with enclosing, closed surfaces. An LR probe simulation is run for each of these rooms, as described earlier. ER processing, described in the next section, does not rely on room partitioning but instead operates on the entire scene at once.

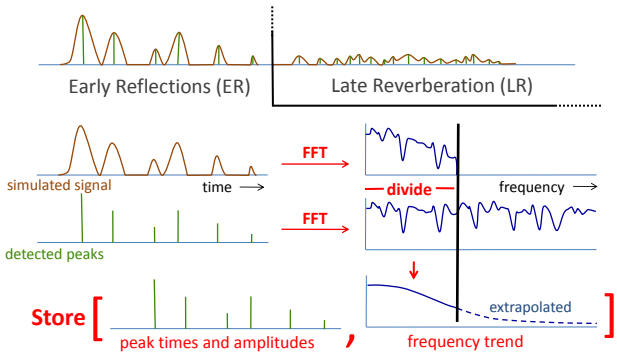

Figure 3: IR encoding. The late reverberation (LR) filter is stored once per room. The early reflections (ER) filter is represented as a set of peaks and a frequency trend, and stored at each source/listener grid point.

4.3 ER Processing

At each source and listener location, the simulator produces a propagated signal, $s(t)$, of length T_{ER}. The ERIR can be computed from this signal using straightforward deconvolution. Convolving input sounds with it then yields a simple method for run-time auralization. Such a direct approach has three problems: it ignores frequencies above η_f and so muffles sounds, it is difficult to spatially interpolate without producing artifacts, and it uses too much memory. We solve these problems by converting the ERIR to a compact representation having two parts as shown in Figure 3: a set of peaks with associated delays and amplitudes, and a residual frequency response magnitude, called the *frequency trend*. ER peaks capture the main time-domain information such as high-amplitude reflections, as well as coloration effects as described in Section 3.1, while the frequency trend accounts for residual frequency-dependent attenuation including low-pass filtering due to diffraction.

Peak information is naturally all-frequency, and so applies to arbitrary inputs without muffling. Only the frequency trend needs to be extrapolated to higher-frequency input signals.

4.3.1 Peak Extraction

A simple method for detecting peaks in the recorded response, $s(t)$, is to run the peak detector introduced in Section 4.2. Let's denote this set of peaks as P. We first remove low-amplitude peaks from P using a threshold of 40dB below the highest-amplitude peak present. Such peaks are masked by the higher energy peaks.

Unfortunately, this method merges peaks separated by less than 1ms, since sums of closely-separated Gaussians have only one extremal point. We can do better: using results from a simulation bandlimited to 1kHz, it is possible in theory to resolve peaks separated by as little as 0.5ms. The following describes an approach that extracts more information from the simulation in order to preserve coloration effects and simplify later processing to extract the frequency trend. It doesn't guarantee all theoretically-resolvable peaks are detected but provides good results in practice.

The ideal impulse response, $I(t)$, is computed by performing a deconvolution on $s(t)$ with the input signal $G(t)$ given in (1). Using $\otimes$ to denote convolution, $\odot$ to denote element-wise complex multiplication, and $\widehat{x}$ to denote the Fourier transform of x, the convolution theorem states that

$$s(t) = G(t) \otimes I(t) \Leftrightarrow \widehat{s}(f) = \widehat{G}(f) \odot \widehat{I}(f). \tag{4}$$

To solve for the IR I given s and G, we deconvolve using a frequency coefficient complex division to obtain

$$\widehat{I}(f) = \frac{\widehat{s}(f)}{\widehat{G}(f)}. \tag{5}$$

An inverse FFT on $\widehat{I}(f)$ then yields $I(t)$. Before performing it though, we must low-pass filter $\widehat{I}$ to eliminate frequencies above η_f, since these are outside the simulator's range and contain large numerical errors.[1] This can be done by zero-padding the FR vector above η_f up to the target rate of 44.1kHz. It is well-known that such an abrupt zeroing or "brick-wall filtering" leads to ringing artifacts in the time domain and so to fictitious peaks. At the same time, it provides the best possible time resolution for separating peaks.

The problem then is to generate a set of super-resolved peaks P' from $I(t)$ that are guaranteed to be real and not "rings" of earlier peaks, in order to separate high-energy peaks to the fullest possible extent and preserve their audible coloration.

Finding super-resolved peaks P' A bound on ringing error in $I(t)$ can be computed by observing that brick-wall filtering turns an ideal peak into a Sinc function, having a $1/t$ time-decaying amplitude. We can therefore build an error envelope $e(t)$ against which later peaks in $I(t)$ can be culled to account for "rings" of earlier ones. P' is initialized to contain all peaks detected from the ideal IR, $I(t)$, including rings. We retain only peaks within 20dB of the highest amplitude one, since we are interested in closely-spaced, high-amplitude peaks that create strong oscillations in the FR.

A ringing envelope, $S(t)$, is then defined as the low-pass filtering result of a unit ideal impulse at $t = 0$ to a frequency limit of η_f:

$$S(t) = \begin{cases} 0, & |t| < t_0 \\ |sin(\omega t)|/\omega t, & t_0 \le |t| < t_1 \\ 1/\omega t, & |t| \ge t_1 \end{cases} \tag{6}$$

where $t_0 = 0.5/\eta_f$, $t_1 = 0.75/\eta_f$, and $\omega = 2\pi\eta_f$. The time t_0 represents the first zero of the Sinc while t_1 represents the following minimum of its negative side lobe. The envelope thus bounds ringing that occurs after a peak's main lobe. If an ideal peak is band-passed, peak detection run on it, and all peaks with absolute amplitudes less than this envelope culled, only the original peak remains.

To build the complete error envelope, $e(t)$, we accumulate these over each peak in $P' = \{a_i, t_i\}$ via

$$e(t) = \sum_i |a_i| S(t - t_i) \tag{7}$$

The fact that P' itself contains ringing peaks serves to make the envelope conservative in the sense that in the worst case we may remove a real peak but never fail to remove a ringing peak. Culling is straightforward: a peak (a_i, t_i) is removed if $|a_i| < e(t_i)$.

Supplementing P with P' We use P' to supplement P since multiple peaks in P' may have merged to a single peak in P. We scan through all peaks in P' and assign each to the peak in P closest to it in time. We then replace each peak in P with the set of peaks in P' that were assigned to it, yielding the final set of peaks. This automatic procedure generates about N=20-50 peaks in our tests.

[1] The reader might wonder why such low-pass filtering was not also applied to the signal $s(t)$ before performing peak detection on it. The reason is that the input source signal $G(t)$ and its response $s(t)$ contain little energy above η_f. It is only when frequency-domain values are divided by each other during deconvolution that a non-negligible result is obtained at higher frequencies, requiring cleanup.

4.3.2 Frequency Trend Extraction

Peak extraction ignores diffraction and implicitly assumes every peak acts on all frequencies uniformly. Diffraction introduces frequency-dependent filtering not captured by this set of peaks. This residual information is captured by our method of frequency trend extraction, as illustrated in the middle of Figure 3. It compares the simulated FR, $\widehat{I}(f)$, to the FR of the idealized IR composed of the extracted peaks, P, in order to extract any residual low-pass trend due to diffraction. The following description assumes η_f=1kHz, our typical frequency limit.

We construct the impulse response corresponding to the extracted ER peaks, I', by summing over its peaks via

$$I' = \sum_{i=1}^{N} a_i \, \delta(t - t_i) \tag{8}$$

where $\delta(t)$ is the discrete analog of the Dirac-delta function – a unit-amplitude pulse of one sample width. Its corresponding FR is denoted $\widehat{I'}$. We also construct the FR of the ideal IR of the original simulation, $\widehat{I}$, which contains complete information up to frequencies of 1kHz. The overall frequency-dependent diffraction trend for $f \le$ 1kHz is obtained via

$$T(f) = \frac{\left|\widehat{I}(f)\right|}{\left|\widehat{I'}(f)\right|}. \tag{9}$$

Before performing this division, we smooth both the numerator and denominator with a Gaussian of width 50Hz. The unsmoothed $\widehat{I'}(f)$ often has near-zero values; smoothing takes care of this problem and makes the above operation numerically stable. As explained in Section 3.1, this has little perceptual impact because we are insensitive to finer details in the magnitude frequency response. $T(f)$ is then smoothed again with the same Gaussian to yield the final frequency trend. Average values of $T(f)$ in each octave band 0-62.5Hz, 62.5-125Hz, …, 500-1000Hz are then stored.

This trend contains information only up to 1kHz. Fortunately, much of the perceivable diffraction-related occlusion effect in common acoustic spaces manifests itself in frequencies below 1kHz [Kuttruff 2000]. Sound wavelength for 1kHz is about 34cm, which is comparable to large features such as pillars, doors and windows.

We can also plausibly extrapolate this trend to frequencies higher than were simulated. To do this, we express $T(f)$ on a log-log scale, which corresponds to plotting the power at each frequency, in dB, against the frequencies in octaves. These scales better match our loudness and pitch perception. Moreover, the low-pass diffraction trend for physical edges is roughly linear on such a log-log scale for mid-to-high frequencies [Svensson et al. 2009]. We then fit a line to the log-log trend in the frequency range 125-1000Hz. If the slope is negative, indicating a low-pass trend, the line is extrapolated and stored at octave bands higher than 1000Hz up to 22,050Hz. If the slope is positive, we do not extrapolate and instead just copy the value at the 500-1000Hz octave to higher ones.

5 Interactive Auralization Runtime

The precomputed, perceptually-encoded information from numerical simulation supports interactive sound propagation from moving sources to a moving listener. Our approach performs perceptually-smooth spatial interpolation of the IRs, and then propagates source sounds by convolving them with these IRs. Because we generate realistically dense IRs, our run-time works in the frequency domain

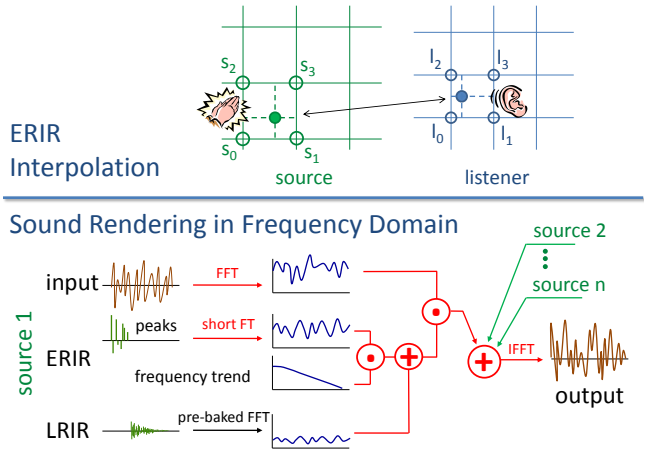

Figure 4: Run-time processing. Operations computed at run-time are colored red. Processing for only one channel (ear) is shown at figure bottom.

to perform the convolution efficiently. A schematic diagram of our real-time auralization system is shown in Figure 4.

5.1 Load-time Computation

At load-time, per-room LR filters are loaded and processed. The initial T_{ER} part of the LRIR is zeroed out, to be replaced at run-time with the spatially-varying ERIR. The LRIR's Fourier Transform $\widehat{I}_{LR}$ is then computed and stored, along with its time-domain peaks.

Next, ERIR filters for the whole scene are loaded, yielding a table, $I_{ER}(p_s, p_r)$, where points p_s lie on a 2.5D region of potential listener positions and p_r sample sources over the entire 3D volume of the scene. Note the reversal of sense of source/listener from the original simulation, which is justified by acoustic reciprocity. Each sample in the $I_{ER}(p_s, p_r)$ contains a set of peaks with associated delays and amplitudes, and an octave-band frequency trend.

5.2 ERIR Interpolation

Spatial interpolation Given the current source and listener locations, p_s and p_r, the ERIR table is indexed and interpolated to reconstruct the ERIR as shown in the top of Figure 4. This interpolation is bilinear over p_r and tri-linear over p_s, and so involves 32 point pairs (8 over p_s and 4 over p_r). The result is denoted I_{SL}, and is based on the temporal interpolation described next.

Temporal Interpolation High-quality interpolation of IRs is a challenging problem. Direct linear interpolation (cross-fading) leads to unrealistic oscillations in the sound amplitude and audible "gurgling" artifacts. Each peak represents a wavefront from the source arriving at the listener. As the listener moves, this peak smoothly translates in time; cross-fading instead generates twin "ghosts" which only modify amplitudes at fixed peak times evaluated at the two spatial samples. Frequency-domain interpolation fails to help because the time and frequency domains are linearly related. Interpolating over the peak delays and amplitudes we extract better matches the physical situation and produces no artifacts in our experiments.

Interpolating between delays and amplitudes of two peaks assumes that they correspond; i.e., belong to the same propagating wavefront. The finite speed of sound dictates that the peaks from the same wavefront at two points in space separated by a distance Δd can be separated in time by no more than

$$\Delta t \leq \frac{\Delta d}{c}, \tag{10}$$

where Δd is the spatial sampling distance (1m) and c is the speed of sound in air, 343m/s at 20°C. The following procedure computes correspondences and achieves convincing interpolation.

Denote the peak sets of the two IRs as P_1 and P_2, and assume both are sorted over peak times t_i. We construct a graph whose edges represent potential correspondence between a peak in P_1 and a peak in P_2; in other words, difference between peak times satisfies (10). Edge weight is assigned the absolute value of the peaks' amplitude difference. The algorithm iterates greedily by selecting the edge of smallest weight currently in the set, finalizing it as a correspondence, and removing all other edges sharing either of the two peaks selected. The operation is commutative in P_1 and P_2 and interpolates associated peak delays and amplitudes, yielding a perceptually smooth auralization for moving sources and listener.

In addition to the peaks, the ERIR's frequency trend must also be interpolated. In this case, straightforward linear interpolation of amplitudes in each octave band works well.

5.3 LRIR Scaling

The overall propagation IR combines the interpolated ERIR between the source and listener, I_{SL}, with the room's LRIR, I_{LR}. We currently choose the LRIR of the room in which the source lies. This approach yields good results in practice but could be extended in future work [Stavrakis et al. 2008]. Since convolutions are already performed in the frequency domain, chains of IRs from multiple rooms can be computed by element-wise complex multiplication, so such an extension would not incur much run-time cost. To make a natural-sounding transition between ERIR and LRIR (at time T_{ER}), we must scale the LRIR.

The LRIR's scaling factor is calculated by first computing the RMS peak amplitude of the ERIR during the time interval $[t_0 + 5ms, T_{ER}]$, where t_0 is the time of the first peak in the ERIR. The result, denoted A_{ER}, discards the first (direct) peak and any that closely follow it. We also calculate the LRIR's maximum absolute peak amplitude in $[T_{ER}, 2T_{ER}]$, yielding A_{LR}. Finally, we account for the LRIR's attenuation from the frequency trend; we do this by computing the mean of amplitudes over all the ERIR's frequency bins, yielding F_{ER}. The final scaling factor is given by

$$\beta_{LR} = \frac{A_{ER}\, F_{ER}}{A_{LR}}. \tag{11}$$

This scaling is then applied to the LRIR's frequency response, $\widehat{I}_{LR}$ computed during load-time, before adding it to the ERIR.

5.4 Binaural Processing

Human auditory perception is binaural; that is, based on two ears. This allows us to estimate direction and distance to sound sources, a capability known as *localization* [Blauert 1997]. Ideally, this requires augmenting each peak in the ERIR with information about the corresponding wavefront gradient; i.e., the direction in which the sound is propagating. It may be possible to extract such information from our simulation, but its calculation is challenging. This is especially true because we exploit reciprocity, which would require tracking simulated waves all the way back to their sources.

Fortunately, a well-known property of localization is the "precedence effect", also known as the "law of the first wavefront", which

(a) living room scene

(b) outdoor walkway scene

(c) "Citadel" scene from Valve's Source™SDK

Figure 5: Test scenes used for auralization.

states that our perception of the direction to a source is determined almost exclusively by the first arriving sound. We therefore assign to the first peak the direction from source to listener, and the remaining peaks to random directions. Each peak in I_{SL} is processed depending on its assigned direction and two ERIRs generated for the left and right ear respectively, I_{SL}^{left} and I_{SL}^{right}.

Binaural perception is sensitive to the exact geometry of the individual listener's ears, head and shoulders, which can be encapsulated as his *head-related transfer function* (HRTF). Non-individualized HRTFs can lead to large errors in localization [Hartmann and Wittenberg 1996]. Our system is easily extensible to customized HRTFs and supports them with little additional run-time cost. To avoid the complexity and present results to a general audience, we currently use a simple model [Hartmann and Wittenberg 1996], based on a spherical head and cardioid directivity function.

5.5 ERIR Short Fourier Transform

To perform convolutions, we transform the left/right ERIRs, I_{SL}^{left} and I_{SL}^{right}, to the frequency domain. This processing is identical for both; we simplify notation by referring to the ERIR as I_{SL}. Denote the number of audio time samples in the ER and LR phases as N_{ER} and N_{LR}, respectively. $T_{ER} << T_{LR}$ and so $N_{ER} << N_{LR}$. Because the ERIR and LRIR are later added in the frequency domain before convolving with the source signal, a straightforward approach is to perform an FFT of length N_{LR} on I_{SL}. However, it contains only zeros beyond sample N_{ER}. Zero-padding a signal in the time-domain corresponds to interpolating in the frequency domain, a fact we exploit to reduce computation. We perform a short FFT on I_{SL} of length $4N_{ER}$ and then upsample the resulting frequency coefficients by a factor of $N_{LR}/4N_{ER}$. The interpolation filter used is a Sinc truncated at the fifth zero on either side, multiplied by the Lanczos window. These choices of intermediate buffer size ($4N_{ER}$) and windowing function reduce FFT wrap-around effects enough to avoid ghost echoes.

The same idea can be applied to compute the Fourier transform on audio buffers representing each input sound source. Overall, we reduce all required per-source FFTs from length N_{LR} to $4N_{ER}$, a speedup of 2-4$\times$ compared to the straightforward approach.

5.6 Auralization

Audio processing is done in fixed-sized buffers at a constant sampling rate. The size of FFTs is clamped to the longest LR filter over all rooms. For each source, a sample queue is maintained in which buffers are pushed from the front at each audio time step. The input sound signal for the i-th source is denoted $u_i(t)$. Refer to Figure 4 for the overall organization of the auralization pipeline.

Processing begins by performing an FFT on the current buffer for the source sound, u_i, yielding the transformed signal $\widehat{u}_i$. Next, the interpolated ERIR, I_{SL}, is computed based on the current source and listener locations as discussed in Section 5.2. The LRIR is accessed depending on the room containing the source, and its scaling factor β_{LR} computed as described in Section 5.3. Binaural processing from Section 5.4 is performed to yield ERIRs for the two ears, I_{SL}^{left} and I_{SL}^{right}. These are transformed to the frequency domain as described in Section 5.5, and the scaled LRIRs added to yield

$$\begin{aligned} \widehat{I}^{\text{left}} &= \widehat{I}_{SL}^{\text{left}} + \beta_{LR}\,\widehat{I}_{LR}, \\ \widehat{I}^{\text{right}} &= \widehat{I}_{SL}^{\text{right}} + \beta_{LR}\,\widehat{I}_{LR}. \end{aligned} \tag{12}$$

The source signal is then efficiently convolved in the frequency domain, yielding the propagated versions of this sound for each ear:

$$\begin{aligned} \widehat{v}_i^{\text{left}} &= \widehat{I}^{\text{left}} \odot \widehat{u}_i, \\ \widehat{v}_i^{\text{right}} &= \widehat{I}^{\text{right}} \odot \widehat{u}_i. \end{aligned} \tag{13}$$

In this way, we accumulate contributions from all sources in the frequency domain, at each ear. The final result is transformed back to the time domain using two inverse FFTs:

$$\begin{aligned} v^{\text{left}} &= \text{FFT}^{-1}\left(\textstyle\sum_i \widehat{v}_i^{\text{left}}\right), \\ v^{\text{right}} &= \text{FFT}^{-1}\left(\textstyle\sum_i \widehat{v}_i^{\text{right}}\right). \end{aligned} \tag{14}$$

The first audio buffers for v^{left} and v^{right} are then sent to the sound system for playback. Between consecutive buffers in time, we perform linear interpolation within a small (5%) window of overlap.

6 Implementation and Results

Our system is implemented in C++ and uses the Intel MKL library for computing FFTs. Vector operations are optimized using SSE3. Performance was measured on a 2.8 GHz Quad-core Intel Xeon processor, with 2.5 GB RAM. We intentionally ensure that the entire sound engine runs inside one thread on a single core, mirroring the practice in interactive applications such as games. Precomputation and run-time statistics are summarized in Table 1. In all examples, the preprocessing time is dominated by the numerical simulation; the perceptually-based encoding is comparatively negligible. The accompanying video shows real-time results collected from our system, and a demonstration of integration with Valve's Source™ game engine. Geometry input to the acoustic simulator is exactly what is shown rendered in each video segment. No manual simplification is performed.

Audio buffer length in our system is 4096 samples, representing about 100ms at a sample rate of 44.1kHz. Our system takes 1.7-3.5ms per source for every audio buffer, which allows about 30 moving sources and a moving listener in real time. Our sound engine utilizes XAudio2 for buffer-level access to the sound-card.

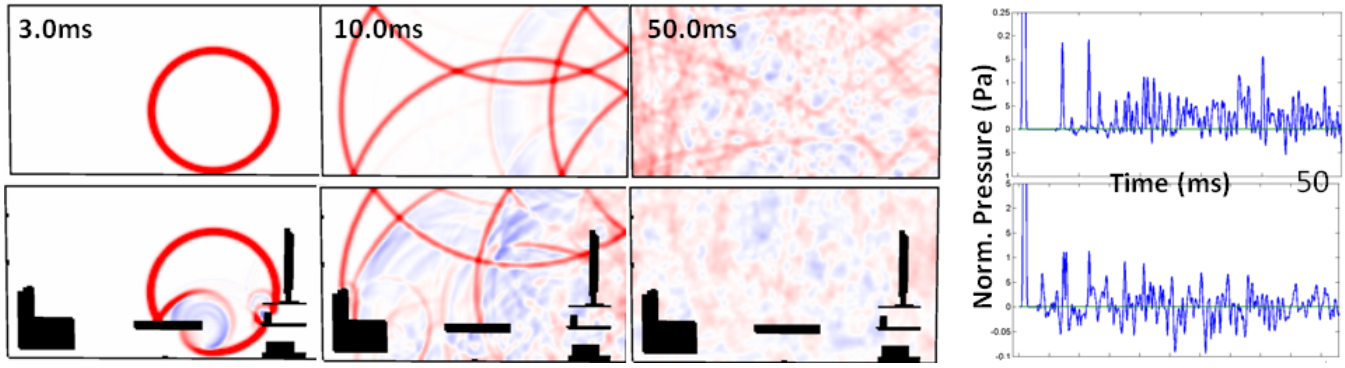

Figure 6: Sound scattering and diffusion in the living room scene. The top row shows an empty room while the bottom row is fully-furnished. The left three columns show a 2D slice of the sound field generated by a Gaussian impulse emitted near the room's center, while the right column shows the entire IR at a single receiver point placed at the source location. Red/blue represents positive/negative pressure in the sound field. Black areas represent solid geometry in the scene. Note the large difference in wave propagation in the two scenes because of scattering and diffraction. Refer to the video for comparative auralizations.

Scene	Dim.	η_f	#ERL	#ERS	#R	ST	Mem.	T_{ER}	T_{LR}	#S	ABT
walkway	19m×19m×8m	1000Hz	1.5M	100	1	120min	600MB	70ms	1100ms	1	1.7ms
living room	6m×8m×3m	2000Hz	4.9M	129	1	75min	1000MB	45ms	250ms	2	1.8ms
Citadel	28m×60m×32m	1000Hz	1.7M	155	6	350min	620MB	80ms	1900ms	8	27.2ms
train station	36m×83m×32m	500Hz	1.1M	200	3	310min	390MB	200ms	2600ms	~15	60ms

Table 1: Performance statistics. The leftmost columns show precomputation statistics while the rightmost two are for the run-time. Abbreviated column headings are as follows. "Dim." is the scene dimensions "#ERL" is the simulated number of ER listener probes (before downsampling). "#ERS" is the simulated number of ER source probes. "#R" is the number of rooms in the scene. "ST" is total simulation time, including LR and ER probes. "Mem." is the total memory used at runtime to store ERIRs for all source and receiver positions, including extracted peaks and frequency trend. T_{ER} is the length of the ER phase, and T_{LR} the LR phase, maximized over all rooms in the scene. "#S" is the number of different sources simulated at run-time. "ABT" is the total time needed to process each audio buffer, summed over all run-time sources.

6.1 Living room

Our technique handles complex scenes like the furnished living room shown in Figure 5(a). Scattering off furnishings has a noticeable effect on a room's acoustics. Figure 6 compares visualizations of the sound field in a 2D slice of this scene, between the furnished room and an empty version with all interior objects and carpet removed. In the empty room, specular wavefronts dominate the sound field and substantial energy still remains after 50ms. In the furnished room, the greater scattering and larger absorbing surface area more quickly reduce the sound field's coherence and energy. Refer to the accompanying video to hear the difference. The right column of the figure plots pressure as a function of time at a single receiver location. The empty room's IR is dominated by high, positive-amplitude peaks, while the furnished room's contains negative pressure peaks due to diffraction at geometric discontinuities and more closely resembles a real room's response qualitatively.

6.2 Walkway

Figure 5(b) shows an outdoor scene designed to demonstrate various acoustic effects. Scene surfaces are all highly reflective with a pressure reflection coefficient of 0.95. Specular reflections from the walls and lack of scattering yield a fairly long T_{LR}=1.1s, with a characteristic hollow reverberation due to the parallel walls and lack of intervening geometry. The walkway's concave ceiling (blue) focuses sounds, so that they become louder when the source and receiver both move below it. Occlusion effects are also important in this scene because diffraction is the only major source of energy transport behind the walls. The sound loudness changes realistically and smoothly as the listener walks behind a wall separating him from the source, and demonstrates a convincing diffracted shadowing effect. Refer to the video for the auralization.

6.3 Citadel

Figure 5(c) shows a larger scene taken from the "Citadel" scene of Half Life 2. Sound sources include a walking narrator, his footsteps, as well as other sources both fixed and moving within the environment. Realistic, spatially-varying acoustic propagation is captured automatically from scene geometry, including a varying LR, and is especially dramatic as the narrator moves from a large room into a narrow corridor. Interesting reverberation is produced in the largest room because of its high ceiling, which causes it to "flutter" especially audible for impulsive sounds.

6.4 Train station

Figure 1 shows a frame from the game Half-Life 2™, with which we have integrated our sound engine. We have written a game "mod" that broadcasts all in-game sound events over a pipe to our sound engine running in a separate process. Sound events include information about the WAV file played as well as the source/listener locations. Our engine then uses this information to propagate sounds in the scene, based on precomputed results over the scene, without requiring any access to the scene geometry at runtime. We can handle up to 15 sources in real time on this scene, including the game's ambient sounds as well as main source sounds, such as gunshots, footsteps and voices. Refer to the accompanying video for acoustic results and a comparison of our engine's sounds with the original game's with its "sound quality" set to "high".

6.5 Error Analysis

To validate our approach, we compared it against a reference numerical simulation bandlimited to a higher frequency of 4kHz. This tests the three sources of error in our approach: our simulation's frequency limit, our perceptually-based parametrization, and spatial interpolation of IRs. The comparison was done on the fully furnished living room. The IR from the numerical simulator was convolved directly with the input sound for producing the reference output. Our system's sound closely matches the reference; refer to the accompanying video for the audio comparison.

Figure 7 quantitatively analyzes error in the same scene. Errors are calculated in third-octave bands with respect to a 4kHz reference simulation. Frequency responses for our decoded result based on a bandlimited working simulation (1kHz) are compared to this reference. The top graph shows errors due to compression alone, by placing the listener on a simulation grid point and avoiding interpolation. Below the simulated frequency limit of 1kHz, our error stays within 2 dB of the reference and increases only moderately to 4 dB in the extrapolated range. Including spatial interpolation (bottom graph) increases error to a maximum of 5 dB, with an average of 2-3 dB. These errors may be compared to the human loudness discrimination threshold at roughly 1 dB over all audible frequencies [Jesteadt et al. 1977]. Our method preserves high frequencies better than linear interpolation and so controls the "gurgling" artifacts heard when linearly interpolating IRs as the listener moves between sample points. For results at other listener locations, refer to the supplementary material.

7 Conclusion, Limitations and Future Work

Ours is the first real-time method for wave-based acoustic propagation from multiple moving sources to a moving listener. It exploits human auditory perception to express the precomputed, spatially-varying impulse response of a complex but static scene in a compact form. Our run-time technique convolves in the frequency-domain, allowing arbitrarily dense impulse responses. Overall, our system captures realistic acoustic effects including late reverberation, diffuse reflections, reverberation coloration, sound focusing, and diffraction low-pass filtering around obstructions.

Some limitations of our approach are due to the high computational cost of wave simulators on today's desktops – the simulation's frequency limit and restricted volume. Others arise from precomputation – our restriction to static scenes and high runtime memory use (hundreds of MBs) even with fairly low spatial sampling. Memory use could be reduced by extracting an even more compact set of ERIR perceptual parameters such as loudness, clarity, etc. Finding a "perceptually complete" set is an open research problem, as is determining spatial sampling requirements for perceptually accurate auralization. Our technique might be practically extended to dynamic scenes by simulating low-dimensional parameterized scenes, such as an opera house at various degrees of seating occupation. It could also benefit from approximations to handle dynamic objects, perhaps by separately precomputing frequency-dependent occlusion factors and then applying them on the fly.

8 Acknowledgements

We thank Micah Taylor and Anish Chandak at UNC Chapel Hill for their help. Our thanks to the Apps Incubation group at Microsoft Research for their support. Also, thanks to the Valve™Corporation for permission to use the Source™SDK and Half Life 2™artwork for the Citadel, train station, and walkway demos.

References

ANTONACCI, F., FOCO, M., SARTI, A., AND TUBARO, S. 2004. Real time modeling of acoustic propagation in complex environments. *Proceedings of 7th International Conference on Digital Audio Effects*, 274–279.

BLAUERT, J. 1997. An introduction to binaural technology. In *Binaural and Spatial Hearing in Real and Virtual Environments*, R. Gilkey and T. R. Anderson, Eds. Lawrence Erlbaum, USA.

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

CALAMIA, P. 2009. *Advances in Edge-Diffraction Modeling for Virtual-Acoustic Simulations*. PhD thesis, Princeton University.

CHANDAK, A., LAUTERBACH, C., TAYLOR, M., REN, Z., AND MANOCHA, D. 2008. Ad-frustum: Adaptive frustum tracing for interactive sound propagation. *IEEE Transactions on Visualization and Computer Graphics 14*, 6, 1707–1722.

EMERIT, M., FAURE, J., GUERIN, A., NICOL, R., PALLONE, G., PHILIPPE, P., AND VIRETTE, D. 2007. Efficient binaural filtering in QMF domain for BRIR. In *AES 122th Convention*.

FUNKHOUSER, T., TSINGOS, N., CARLBOM, I., ELKO, G., SONDHI, M., WEST, J. E., PINGALI, G., MIN, P., AND NGAN, A. 2004. A beam tracing method for interactive architectural acoustics. *The Journal of the Acoustical Society of America 115*, 2, 739–756.

GARDNER, W. G. 1998. Reverberation algorithms. In *Applications of Digital Signal Processing to Audio and Acoustics*, M. Kahrs and K. Brandenburg, Eds., 1 ed. Springer, 85–131.

HALMRAST, T. 2007. Coloration due to reflections. further investigations. In *International Congress on Acoustics*.

HARTMANN, W. M., AND WITTENBERG, A. 1996. On the externalization of sound images. *The Journal of the Acoustical Society of America 99*, 6 (June), 3678–3688.

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

JESTEADT, W., WIER, C. C., AND GREEN, D. M. 1977. Intensity discrimination as a function of frequency and sensation level. *The Journal of the Acoustical Society of America 61*, 1, 169–177.

KINSLER, L. E., FREY, A. R., COPPENS, A. B., AND SANDERS, J. V. 2000. *Fundamentals of acoustics*, 4 ed. Wiley, December.

KUTTRUFF, H. 2000. *Room Acoustics*. Taylor & Francis, October.

LITOVSKY, R. Y., COLBURN, S. H., YOST, W. A., AND GUZMAN, S. J. 1999. The precedence effect. *The Journal of the Acoustical Society of America 106*, 4, 1633–1654.

LOKKI, T. 2002. *Physically-based Auralization*. PhD thesis, Helsinki University of Technology.

MASTERSON, C., KEARNEY, G., AND BOLAND, F. 2009. Acoustic impulse response interpolation for multichannel systems using dynamic time warping. In *35th AES Conference on Audio for Games*.

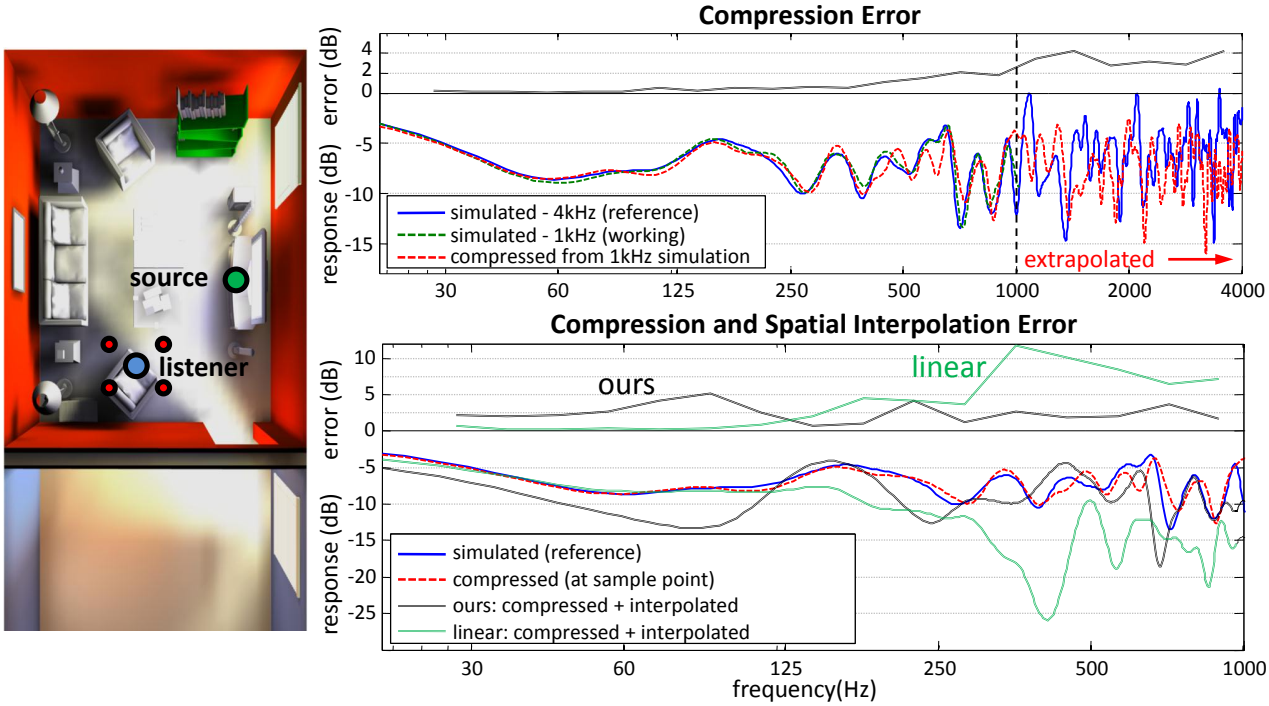

Figure 7: Error analysis of our technique in the living room scene, at the source/listener points shown on the left. The top graph isolates errors from IR parametrization (peaks and frequency trend) alone, while the bottom graph accounts for interpolation as well, at a point midway between listener samples (red dots on left). The error between our approximated response and the reference appears in the top part of each plot. The bottom graph also compares linear interpolation (green curve) against our method (black curve). Linear interpolation produces more error overall and incorrectly attenuates higher frequencies.

MIN, P., AND FUNKHOUSER, T. 2000. Priority-driven acoustic modeling for virtual environments. *Computer Graphics Forum* (September), 179–188.

POPE, J., CREASEY, D., AND CHALMERS, A. 1999. Realtime room acoustics using ambisonics. In *The Proceedings of the AES 16th International Conference on Spatial Sound Reproduction*, Audio Engineering Society, 427–435.

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*, 5, 789–801.

SAKAMOTO, S., USHIYAMA, A., AND NAGATOMO, H. 2006. Numerical analysis of sound propagation in rooms using the finite difference time domain method. *The Journal of the Acoustical Society of America 120*, 5, 3008.

SILTANEN, S. 2005. *Geometry Reduction in Room Acoustics Modeling*. Master's thesis, Helsinki University of Technology.

STAVRAKIS, E., TSINGOS, N., AND CALAMIA, P. 2008. Topological sound propagation with reverberation graphs. *Acta Acustica/Acustica - the Journal of the European Acoustics Association (EAA)*.

SVENSSON, U. P., CALAMIA, P., AND NAKANISHI, S. 2009. Frequency-domain edge diffraction for finite and infinite edges. In *Acta Acustica/Acustica 95*, 568–572.

TAYLOR, M. T., CHANDAK, A., ANTANI, L., AND MANOCHA, D. 2009. Resound: interactive sound rendering for dynamic virtual environments. In *MM '09: Proceedings of the seventeen ACM international conference on Multimedia*, ACM, New York, NY, USA, 271–280.

TSINGOS, N., FUNKHOUSER, T., NGAN, A., AND CARLBOM, I. 2001. Modeling acoustics in virtual environments using the uniform theory of diffraction. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 545–552.

TSINGOS, N., DACHSBACHER, C., LEFEBVRE, S., AND DELLEPIANE, M. 2007. Instant sound scattering. In *Rendering Techniques (Proceedings of the Eurographics Symposium on Rendering)*.

TSINGOS, N. 2009. Pre-computing geometry-based reverberation effects for games. In *35th AES Conference on Audio for Games*.