Efficient Auralization for Moving Sources and Receiver

Anish Chandak
University of North Carolina at Chapel Hill
Chapel Hill, NC, USA
achandak@cs.unc.edu

Lakulish Antani
University of North Carolina at Chapel Hill
Chapel Hill, NC, USA
lakulish@cs.unc.edu

Dinesh Manocha
University of North Carolina at Chapel Hill
Chapel Hill, NC, USA
dm@cs.unc.edu

ABSTRACT

We address the problem of generating smooth, spatialized sound for interactive multimedia applications, such as VoIP-enabled virtual environments and video games. Such applications have moving sound sources as well as moving receiver (MS-MR). As a receiver moves, it receives sound emitted from prior positions of a given source. We present an efficient algorithm that can correctly model auralization for MS-MR scenarios by performing sound propagation and signal processing from multiple source positions. Our formulation only needs to compute a portion of the response from various source positions using sound propagation algorithms and can be efficiently combined with signal processing techniques to generate smooth, spatialized audio. Moreover, we present an efficient signal processing pipeline, based on block convolution, which makes it easy to combine different portions of the response from different source positions. Finally, we show that our algorithm can be easily combined with well-known geometric sound propagation methods for efficient auralization in MS-MR scenarios, with a low computational overhead (less than 25%) over auralization for static sources and a static receiver (SS-SR) scenarios.

1. INTRODUCTION

Auralization is the technique of creating audible sound from computer-based simulation. It involves modeling sound propagation from a source, followed by signal processing to recreate the binaural listening experience at the receiver. Auralization is important in many interactive multimedia applications including games, telepresence, and virtual reality. For example, in massive multiplayer online (MMO) game such as World of Warcraft, spatialization of voice chat using Voice over IP (VoIP) could greatly enhance the gaming experience for multiple players [?]. In MMOs, the game players are moving and correspond to the location of sound sources (source of voice chat) as well as the receivers in the game. In tele-collaboration systems, spatial reproduction of sound is necessary to achieve realistic immersion [?]. Auralization can also provide environmental context by modeling acoustics in absence of visual cues, provide a feeling of a human condition, or set the mood [?]. For example, reverberation provides a sense of warmth or immersion in the environment. Auralization is indispensable in telepresence systems as it can provide important sound cues, like the direction of sound sources and the size of the environment, and is frequently used for training [?], therapy [?], tourism [?], exploration [?], and learning [?]. There are recent efforts to provide distributed musical performances using telepresence systems and the acoustic reproduction of the virtual environment is very important [?]. In many of these applications moving sound sources and receivers are common, e.g. in virtual reality exposure (VRE) therapy (virtual Iraq) [?] to treat post-traumatic stress disorder (PTSD), the game player (receiver) is moving as are the sound sources (e.g., helicopters, tanks). The simultaneous movement of the sources as well as the receiver results in additional challenges with respect to computational overhead and accurate modeling of the acoustic response at each receiver position.

The two key components of auralization are sound propagation and signal processing. During sound propagation, an impulse response (IR) is computed for every source-receiver pair, which encodes reflections, diffraction, and scattering of sound in the scene from the source position to the receiver position. During the signal processing stage, the IR computed during propagation is convolved with the anechoic (dry) audio signal emitted by the source, yielding the audio signal heard at the receiver. Since sound may arrive at the receiver from many previous source positions, there are three main challenges in auralization of MS-MR scenes. Firstly, IRs need to be computed between many previous source positions and the current receiver position; the number of IRs that need to be computed is proportional to the maximum delay modeled in the IR. Secondly, during signal processing, many different IRs need to be convolved with the dry audio signal, incurring a substantial computational overhead. Finally, the convolved signals due to different IRs need to be combined to generate a smooth, correct audio signal at the receiver.

Various approaches have been proposed to handle sound propagation and signal processing for auralization in dynamic scenes. However, current methods do not accurately model dynamic scenes where both the sources and receivers are moving simultaneously. We present novel auralization techniques for such dynamic scenes. Some of the new components of our work include:

- **Auralization for MS-MR Scenes**: We present an

WOODSTOCK '97 El Paso, Texas USA

algorithm to accurately model auralization for dynamic scenes with simultaneously moving sources and a moving receiver (MS-MR). We show that our technique can also be used to perform efficient auralization for moving sources and a static receiver (MS-SR) as well as static sources and a moving receiver (SS-MR).

- **Efficient Signal Processing Pipeline**: We present a signal processing pipeline, based on block convolution, that efficiently computes the final audio signal for MS-MR scenarios by convolving appropriate blocks of different IRs with blocks of the input source audio.

- **Modified Sound Propagation Algorithms**: We extend existing sound propagation algorithms, based on the image-source method and pre-computed acoustic transfer, to efficiently model propagation for MS-MR scenarios. Our modified algorithms are quite general and applicable to all MS-MR scenarios.

- **Low Computational Overhead**: We show that our sound propagation techniques and signal processing pipeline have a low computational overhead (less than 25%) over auralization for SS-SR scenarios.

The rest of the paper is organized in the following manner. We discuss prior work related to sound propagation and signal processing for auralization in Section 2. Section 3 gives a brief overview of auralization. In Section 4, we present our algorithm for auralization in MS-MR scenes. We present a signal processing pipeline, and extend two recent sound propagation algorithms for MS-MR scenes in Section 4.2 and in Section 5, respectively. Finally, we describe our results in Section 6.

2. RELATED WORK

In this section, we briefly review prior work related to sound propagation and signal processing for auralization.

2.1 Sound Propagation

The propagation of sound in a medium is described by the acoustic wave equation, a second-order partial differential equation. Various methods have been proposed to solve the wave equation, and we summarize them below.

Numerical Methods: Various classes of numerical methods have been applied to solve the wave equation [?], such as the Finite Element Method (FEM), the Boundary Element Method (BEM), the Finite-Difference Time-Domain (FDTD) method, and Digital Waveguide Meshes (DWM). FDTD methods are popularly used in room acoustics [?] due to their simplicity. However, FDTD methods are computationally intensive, and scale as the fourth power of the maximum simulated frequency and linearly with the scene volume. These methods, when applied to medium-sized scenes using a cluster of computers, can take tens of GBs of memory and tens of hours of computation time [?]. Recently, an Adaptive Rectangular Decomposition (ARD) technique [?] was proposed, which achieves two orders of magnitude speed-up over FDTD methods and other state-of-the-art numerical techniques. In practice, these numerical algorithms can compute an accurate acoustic response. However, they are quite expensive in terms of handling large acoustic spaces or dynamic scenes.

Geometric Methods: Geometric acoustics provides approximate solutions to the wave equation for high-frequency sound sources. Broadly, geometric acoustics algorithms can be divided into *pressure-based* and *energy-based methods*.

Pressure-based methods model specular reflections and edge diffraction, and are essentially variations of the image-source method [?]. Ray tracing [?], beam tracing [?], frustum tracing [?], and several other techniques [?] have been proposed to accelerate the image-source method for specular reflections. *Edge diffraction* is modeled by the Uniform Theory of Diffraction (UTD) [?] or the Biot-Tolstoy-Medwin (BTM) model of diffraction [?].

Energy-based methods are typically used to model diffuse reflections and propagation effects when interference of sound waves is not important. The *room acoustic rendering equation* [?] is an integral equation generalizing energy-based methods. Many different methods, including ray tracing [?], phonon tracing [?], and radiosity [?], have been applied to solve the room acoustic rendering equation.

Geometric methods are efficient and can handle dynamic scenes, but cannot accurately model low-frequency interactions with objects in a scene.

Precomputation-based Methods: Sound propagation is computationally challenging, and many interactive applications like video games allow a very small compute and memory budget ($<$ 10% of the total compute and memory budget) for auralization. Hence, precomputation-based auralization approaches are becoming increasingly popular. Precomputation-based methods using ARD [?] have recently been developed. They compute the acoustic response of a scene from several sampled source positions; at run-time these responses are interpolated given the actual position of a moving source. Energy-based precomputation approaches that can model high orders of reflection using precomputed surface responses [?] or pre-computed transfer operators [?] have also been proposed. These methods precompute acoustic response on points on the surface of the scene and use them at run-time to compute the impulse response based on the source and receiver positions. Pressure-based precomputation methods based on image-source gradients have also been proposed [?].

2.2 Signal Processing for Auralization

Auralization is a vital component in some VR systems, such as DIVA at the Helsinki University of Technology [?], and RAVEN at Aachen University [?]. In addition, real-time auralization systems have been developed based on beam tracing [?] and frustum tracing [?]. These systems use sound propagation engines that compute IRs at interactive rates and signal processing pipelines that use these IRs to generate smooth, spatialized audio signals for a receiver.

Artifact-Free Audio: IRs change when sources and receivers move in a dynamic scene. This could lead to artifacts in the final audio signal. Approaches based on parametric interpolation of delays and attenuations [?, ?], image-source interpolation [?, ?], and windowing-based filtering [?] have been proposed to handle discontinuities in IRs and audio signals.

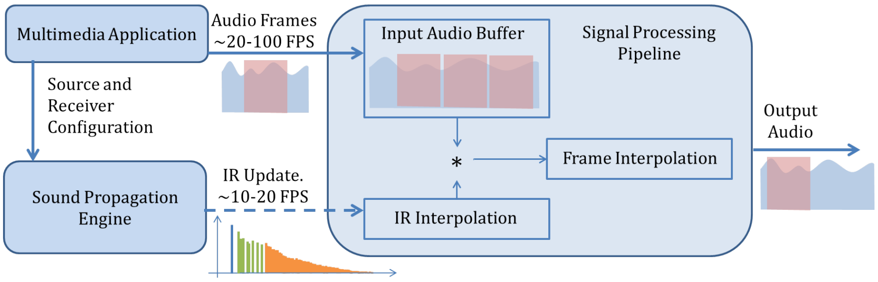


Figure 1: An overview of auralization for multimedia applications. Source and receiver configuration is sent by the application to the sound propagation engine. Sound propagation engine asynchronously updates the impulse response (IR) for the source and receiver configuration to the signal processing pipeline. The signal processing pipeline buffers the incoming audio frames, interpolate IRs, convolve IRs with the buffered input audio, and interpolate the convolve result to generate smooth final audio signal.

Efficient Signal Processing: Processing a large number of IRs from a large number of sound sources at interactive rates is computationally challenging. Efficient techniques to compute the final audio signal based on perceptual optimizations [?], Fourier-domain representations [?], and clustering of sound sources [?, ?] have been proposed.

3. BACKGROUND

In this section, we present an overview of auralization for multimedia applications (see Figure 1).

3.1 Impulse Responses

Most indoor sound propagation algorithms operate on the assumption that the propagation of sound from a source to a receiver can be modeled by a *linear, time-invariant* (LTI) system. As a result, the propagation effects are described by computing an *impulse response* (IR) between the source and the receiver. The IR encodes spatial information about the size of the scene and the positions of objects in it by storing the arrival of sound waves from the source to the receiver as a function of time. Given an arbitrary sound signal emitted by the source, the signal heard at the receiver can be obtained by convolving the source signal with the IR.

Figure 2 shows an example of an IR; it is usually divided into direct response (DR), early response (ER), and late response (LR). For example, in a church or a cathedral, an IR could be 2-3 seconds long, as sound emitted by a source will reflect and scatter and reach the receiver with a delay of upto 2-3 seconds, decaying until it is not audible. For such an IR, the DR is the direct sound from the source to the receiver, ER is typically the sound reaching the receiver within the first 80-100 ms, and LR is the sound reaching the receiver after 100 ms of being emitted from the source.

However, this assumption is only valid for SS-SR scenarios. For MS-MR scenarios, sound propagation cannot be modeled as an LTI system. Therefore, there is no well-defined IR between the source and the receiver. Fortunately, propagation for MS-MR scenarios can be modeled using *time-varying IRs*. We shall use this observation in Section 4 to develop an auralization framework for MS-MR scenarios.

3.2 Auralization Pipeline

There are two key components of an auralization pipeline: (a) the sound propagation engine, and (b) the signal processing pipeline. Figure 1 shows an overview of auralization for interactive multimedia applications. The simple convolution framework described in Section 1 is sufficient for acoustical analysis of a space, where sound sources and receiver are fixed and the user is not actively interacting with the system. But this is not sufficient for interactive multimedia applications and they pose many challenges.

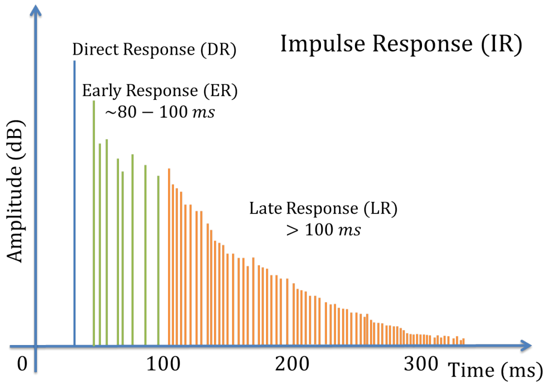
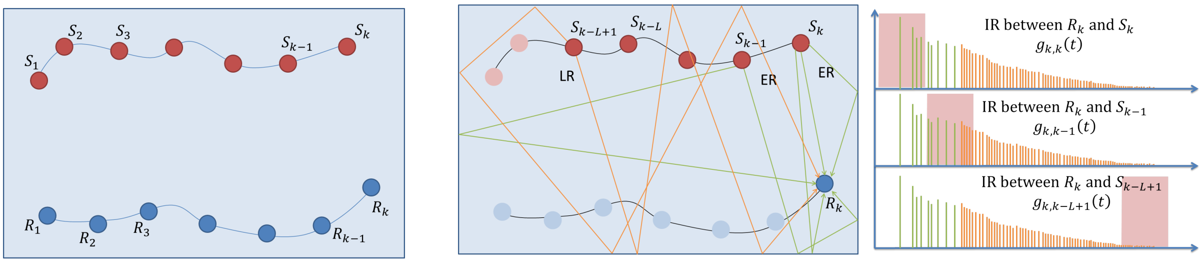


Figure 2: Example of a typical IR. It is composed of direct response (DR), early response (ER), and late response (LR). DR is the direct sound from the source to the receiver, ER is typically the sound arriving at the receiver in the first 80-100 ms, and LR is the sound arriving at the receiver after 100 ms.

Firstly, due to the interactive nature of such applications, audio must be streamed to the user in real-time. For example, voice chat among players in MMOs must be played back to the player in real-time for effective in-game communication. Therefore, the audio played at the source is sent to the signal processing pipeline in small chunks, called *audio frames*. Sound propagation and binaural audio effects are applied to each audio frame. The size of the audio frames is chosen by the application, typically based on allowable latency in the system and the time required to apply propagation effects on a given audio frame. The size of an audio frame could be anywhere from 10-50 ms depending on the application, and therefore 20-100 audio frames need to be processed per second. Thus, a signal processing pipeline


S_1
S_2
S_3
S_{k-1}
S_k
R_1
R_2
R_3
R_{k-1}
R_k
S_{k-L+1}
S_{k-L}
S_{k-1}
S_k
LR
ER
ER
R_k
IR between R_k and S_k $g_{k,k}(t)$
IR between R_k and S_{k-1} $g_{k,k-1}(t)$
IR between R_k and S_{k-L+1} $g_{k,k-L+1}(t)$

(a) (b)

Figure 3: Sound propagation for MS-MR. (a) Path for moving source and moving receiver. Source and receiver positions are updated during each frame. S_k and R_k denote the position of source and receiver during the k^{th} frame. (b) The sound received at the receiver R_k comes from the source positions $\{S_k(t), S_{k-1}, \ldots, S_{k-L+1}\}$, where corresponding audio frames $\{s_k(t), s_{k-1}(t), \ldots, s_{k-L+1}(t)\}$ are modified by direct and early response for the latest source positions $\{S_k, S_{k-1}\}$ and by late response from the earlier source positions $\{S_{k-2}, \ldots, S_{k-L+1}\}$.

for auralization in interactive applications needs to handle streaming audio frames in real-time.

Secondly, these interactive applications could involve complex scenes, and current algorithms may not be able to perform sound propagation at 20-100 frames per second. Therefore, sound propagation is performed asynchronously with respect to signal processing, as shown in Figure 1. For interactive applications, 50-100 ms is an acceptable latency to update the IRs in a scene [**?**] and therefore, the sound propagation engine should be able to asynchronously update the IRs at 10-20 FPS.

Thirdly, as these interactive applications involve dynamic MS-MR scenes, it is important to reduce any artifacts due to the dynamic scenarios, and the final audio signal must be smooth. Due to the movement of sources and receiver, the IRs used for two subsequent audio frames may be different, as these IRs are updated asynchronously. This could lead to a discontinuity in the final audio signal at the boundary between two audio frames. Hence, interpolation of IRs or smoothing of the audio at the frame boundaries is performed to ensure an artifact-free final audio signal.

Finally, as the application may have a large number of sound sources, the signal processing pipeline must be efficient and should be able to handle the convolution of IRs for a reasonable number of sound sources at 20-100 FPS.

4. AURALIZATION FOR MS-MR

In this section, we present our auralization technique for MS-MR scenarios. Moreover, our technique can also be specialized for MS-SR and SS-MR scenarios. Table 1 presents the notation used throughout this section and the rest of the paper. Note that the audio emitted in different frames could overlap in time, depending on the window function chosen. A windowing function limits the support of a signal. For clarity of exposition, we use a square window function such that $w(t) = 1$ for $0 \leq t < \Delta T$, and $w(t) = 0$ elsewhere. However, to compute a smooth audio signal, an overlapping windowing function like Hamming window is used. Next, we present a mathematical formulation to compute the audio signal heard at the receiver in a given frame, $r_k(t)$, for MS-MR scenarios (see Figure 3 and Figure 4).

In a given MS-MR scenario, sound reaching R_k arrives from many previous source positions $\{S_k, S_{k-1}, ..., S_{k-L+1}\}$, as the sound emitted by previous source positions (upto L audio frames ago) propagates through the scene before arriving at R_k. For any previous source position S_{k-i}, the sound reaching the listener can be obtained by convolving $s_{k-i}(t)$, the sound emitted from the source in frame $k - i$, with $g_{k,k-i}(t)$, the IR between S_{k-i} and R_k. Since the listener is at R_k only during audio frame k, we isolate the relevant portion of the convolved signal by applying a square window $w_k(t)$. This results in the following equation:

Notation	
$\cdot$	multiplication operator
$*$	convolution operator
t	time in seconds
ΔT	length of an audio frame in seconds
N	frame rate ($= 1/\Delta T$)
k	frame number (range from 0 to ∞)
S_k	position of source in frame k
R_k	position of receiver in frame k
$w(t)$	window function (non-zero for ΔT time period)
$w_k(t)$	window function used to select frame k $= w(t - k\Delta T)$
$s(t)$	signal emitted by source
$s_k(t)$	signal emitted by source in frame k $= s(t) \cdot w_k(t)$
$r_k(t)$	signal received by receiver in frame k
$r(t)$	signal received by receiver $= \sum_{k=0}^{\infty} r_k(t)$
$g_{k_1,k_2}(t)$	impulse response between R_{k_1} and S_{k_2}
l	length of impulse response in seconds $g_{k_1,k_2}(t) = 0$ for $t > L$
L	length of impulse response in frames $= l/\Delta T$

Table 1: Definitions of symbols and other notation used in this paper.

$$r_k(t) = \sum_{i=0}^{L-1} [g_{k,k-i}(t) * s_{k-i}(t)] \cdot w_k(t) \tag{1}$$

Equation 1 correctly models the audio signal that will

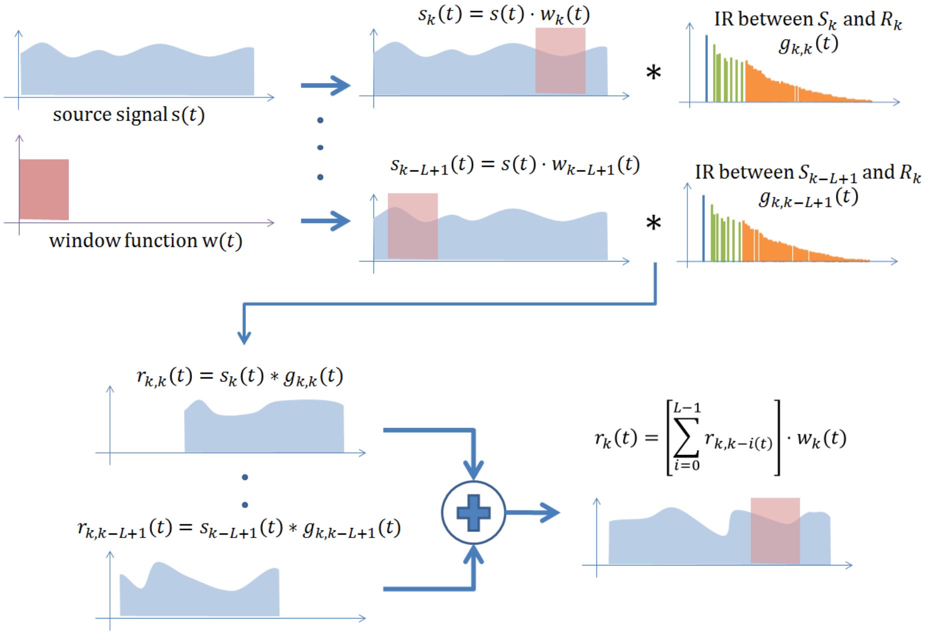

$s_k(t) = s(t) \cdot w_k(t)$
IR between S_k and R_k
$g_{k,k}(t)$
*
source signal s(t)
$s_{k-L+1}(t) = s(t) \cdot w_{k-L+1}(t)$
IR between S_{k-L+1} and R_k
$g_{k,k-L+1}(t)$
*
window function w(t)
$r_{k,k}(t) = s_k(t) * g_{k,k}(t)$
$r_k(t) = \left[\sum_{i=0}^{L-1} r_{k,k-i(t)}\right] \cdot w_k(t)$
$r_{k,k-L+1}(t) = s_{k-L+1}(t) * g_{k,k-L+1}(t)$

Figure 4: Signal processing pipeline for MS-MR: The input audio signal $s(t)$ is multiplied with window function $w(t)$ to generate the audio frames $\{s_k(t), s_{k-1}(t), \ldots, s_{k-L+1}(t)\}$ for source positions $\{S_k, S_{k-1}, \ldots, S_{k-L+1}\}$. These audio frames are convolved with the corresponding IRs $\{g_{k,k}(t), g_{k,k-1}(t), \ldots, g_{k,k-L+1}(t)\}$ and multiplied with the window function $w_k(t)$ to generate the audio frame for the receiver position R_k.

reach the receiver R_k from prior source positions. Note that none of the IRs $\{g_{k,k}(t), g_{k,k-1}(t), \ldots, g_{k,k-L+1}(t)\}$ were computed in previous frames, and therefore L new IRs need to be computed during every frame between each of the prior L source positions, $\{S_k, S_{k-1}, \ldots, S_{k-L+1}\}$, and the current receiver position R_k. A naïve auralization technique may compute L new IRs during sound propagation per frame and may perform L full convolutions of the IRs with the current audio frame. For example, for an IR of length 1 second and audio frames of size 100 ms, sound propagations for 10 different IRs would be performed per frame and 10 full convolutions between the IRs and the current audio frame would be computed per frame. This may not fit within the convolution budget of 10-50 ms per frame or the 50-100 ms budget per frame for sound propagation, and may lead to glitches in the final audio signal or high latency in updating the IRs.

Key Observations: Our formulation is based on the following property:

Lemma 1: *For a given receiver position R_k and source position S_{k-i}, only the interval of the IR $g_{k,k-i}(t)$ in $[(i-1)\Delta T, (i+1)\Delta T]$ will contribute to $r_k(t)$.*

Intuitively, the sound emitted at source position S_{k-i} can arrive at R_k only within a delay of $[(i-1)\Delta T, (i+1)\Delta T]$, assuming that a square window $w_k(t)$ is applied to compute the final audio signal at R_k. Thus, only an interval of the IR $g_{k,k-i}(t)$ needs to be computed to generate the final audio signal at R_k.

Furthermore, a block convolution framework is well-suited for signal processing in MS-MR scenarios as different blocks of IRs can be convolved with corresponding audio blocks. To compute the final audio signal at R_k, the block of the IR $g_{k,k-i}(t)$ in the interval $[(i-1)\Delta T, (i+1)\Delta T]$ is convolved with the input audio frame $s_{k-i}(t)$ to generate the audio signal at R_k. This minimizes the computation in the signal processing pipeline, and we will show in Section 6 that this leads to efficient auralization for MS-MR scenarios.

4.1 Auralization for SS-MR and MS-SR

Equation 1 can be specialized to derive similar equations for SS-MR and MS-SR scenarios. In SS-MR scenarios, the source is static, i.e., $S_1 = S_2 = S_3 \ldots = S_k$. Therefore, $g_{k,k-i}(t) = g_{k,k}(t)$, and $g_{k,k-i}(t)$ can be moved out of the summation in Equation 1, yielding the following simplified equation:

$$r_k(t) = [g_{k,k}(t) * \sum_{i=0}^{L-1} s_{k-i}(t)] \cdot w_k(t) \tag{2}$$

Hence, SS-MR scenarios are easier to model, as they require computation of only one IR, $g_{k,k}(t)$, per frame. However, the past audio frames $\{s_k(t), s_{k-1}(t), \ldots, s_{k-L+1}(t)\}$ need to be buffered in memory.

In MS-SR scenarios, the receiver is static, i.e., $R_1 = R_2 = R_3 \ldots = R_k$. Therefore, $g_{k-i,k}(t) = g_{k,k}(t)$, i.e., of the L IRs needed in every frame, $L-1$ would have been computed in the previous frames, and can be re-used. This observation results in the following simplified equation:

$$r_k(t) = [\sum_{k=0}^{L-1} r_{k,k-i}(t)] \cdot w_k(t) \tag{3}$$

$$r_{k,k-i}(t) = g_{k,k-i}(t) * s_{k-i}(t) \tag{4}$$

Thus, MS-SR scenarios are easier to model, as they require the computation of only one IR, $g_{k,k}(t)$, per frame. Moreover, the convolved output from the previous $L-1$ frames, $\{r_{k,k-1}(t), r_{k,k-1}(t), \ldots, r_{k,k-L+1}(t)\}$, can be cached in a buffer and used to compute $r_k(t)$.

4.2 Signal Processing for Auralization

In this section, we present our signal processing pipeline. The pipeline uses block convolution to compute the final audio signal for MS-MR scenarios. During block convolution for receiver R_k, the block $[(i-1)\Delta T, (i+1)\Delta T]$ of IR $g_{k,k-i}(t)$ is convolved with the source signal block $s_{k-i}(t)$. The results from these block convolutions for the past L source positions are added together and played back at the receiver after applying the window function $w_k(t)$.

In our formulation, the block convolution is implemented by computing a short-time Fourier transform (STFT) of each frame of input audio and the IRs. Let the STFT of $s_k(t)$ be $s_k(\omega)$ and the STFT of $g_{k,k-i}(t)$ for the interval $[(i-1)\Delta T, (i+1)\Delta T]$ be $g_{k,k-i}(\omega)$. Then the final audio $r_k(\omega)$ can be computed as follows:

$$r_k(\omega) = \sum_{i=0}^{L-1} g_{k,k-i}(\omega) \cdot s_{k-i}(\omega) \tag{5}$$

Finally, we compute an inverse-STFT to compute $r_k(t)$ from $r_k(\omega)$. Note that as the IRs change from one frame to another, there may be discontinuities in the final audio signal between the boundaries of consecutive frames. Therefore windowing functions, such as the Hamming window, are used instead of a square window to minimize such artifacts.

5. SOUND PROPAGATION FOR MS-MR

In this section, we present two efficient modifications of sound propagation algorithms for modeling MS-MR scenarios. First, based on Lemma 1, we can efficiently compute L IRs from the prior source positions, $\{S_k, S_{k-1}, \ldots, S_{k-L+1}\}$ using the image-source method. Second, we extend a precomputation based sound propagation algorithm [?] that stores responses from the sources at the surface samples, by observing that the response at the surface samples from past source locations $\{S_k, S_{k-1}, \ldots, S_{k-L+1}\}$ would have been computed in previous frames and can be stored to efficiently generate L IRs corresponding to the new receiver position.

5.1 MS-MR Image Source Method

Figure 5 gives a short overview of the image-source method. This method is used to compute specular reflections and can be extended to handle edge diffraction. The image-source method can be divided into two main steps: (a) *image tree* construction based on the geometric representation of the environment, and (b) *path validation* to compute valid specular paths in the image tree.

In the first step, we construct a image tree from receiver R_k up to a user-specified m orders of reflection. It is constructed by recursively reflecting the receiver or an image of the receiver over the scene primitives and storing the image-sources as nodes in the tree. Each node in the tree thus corresponds to a single specular propagation path. We denote this image tree by $T(R_k, m)$. We construct an image tree using the receiver position as the root of the tree as it leads to a more efficient implementation over constructing an image tree from the source position. Figure 5 shows an example of an image tree.

To compute the final specular paths from the image tree, the source position is required. Hence, in the second step, given a source position S_k, we traverse the tree, and for each node in $T(R_k, m)$, we determine which of the corresponding propagation paths are valid. Some of the paths may not be valid because of the following reasons: (a) a path from the source S_k to an image-source or between two image-sources might be obstructed by other objects in the scene, or (b) a path may not lie within the scene primitive which induced the image source. In such cases, a specular path does not exist for the corresponding node in the image-source tree.

In MS-MR scenarios, path validation needs to be performed from all the source positions $\{S_k, S_{k-1}, ..., S_{k-L+1}\}$. However, we require only a portion of the IR in the interval $[(i-1)\Delta T, (i+1)\Delta T]$ for a given receiver position R_k and source position S_{k-i}. Therefore, we use the time interval and compute the distance interval $[(i-1)c\Delta T, (i+1)c\Delta T]$, where c is the speed of the sound, to eliminate paths during the path validation step. A path is not considered valid if the length of the path lies outside this distance interval. We will show in Section 6 that our formulation substantially reduces the overhead of computing image-source IRs for MS-MR scenarios.

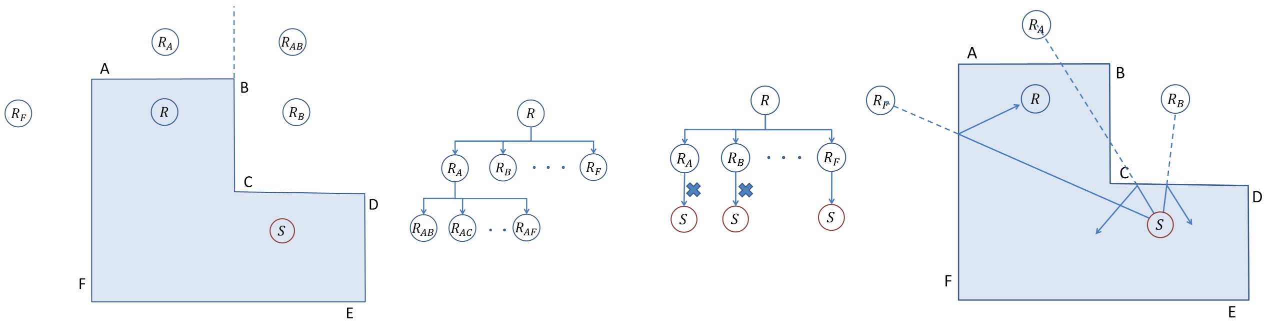
5.2 MS-MR Direct-to-Indirect Acoustic Radiance Transfer

Direct-to-indirect (D2I) acoustic radiance transfer is a recently proposed precomputation-based approach to compute high orders (up to 100 or more) of diffuse reflections interactively [?]. The key idea is to sample the scene surfaces and precompute a complex-valued *transfer matrix*, which encodes higher-order diffuse reflections between the surface samples. At run-time, the IR is computed as follows: (a) Given the source position, compute a *direct response* at each sample point. (b) Given the direct response at the sample points, *indirect responses* are computed at these sample points by performing a matrix-vector multiplication with the transfer matrix. (c) Given the direct and indirect responses, the final IR is computed at the receiver by tracing rays (or "gathering") from the receiver.

In MS-MR scenarios, we need to compute IRs for L previous source locations $\{S_k, S_{k-1}, ..., S_{k-L+1}\}$. A naïve implementation would perform all three steps above L times. This is highly inefficient, and would lead to an L-fold slow-down over SS-SR scenarios. We observe that in this case, all the indirect responses from the previous $L-1$ source locations $\{S_{k-1}, ..., S_{k-L+1}\}$ can be stored at the surface samples (as they are independent of the receiver position), and therefore only one indirect response from source position S_k needs to be computed for each surface sample per frame. In the final step, the final IR can be gathered from L different indirect responses. This is much more efficient than the naïve implementation, as can be seen by the overall performance of direct-to-indirect acoustic radiance transfer for MS-MR in Section 6.

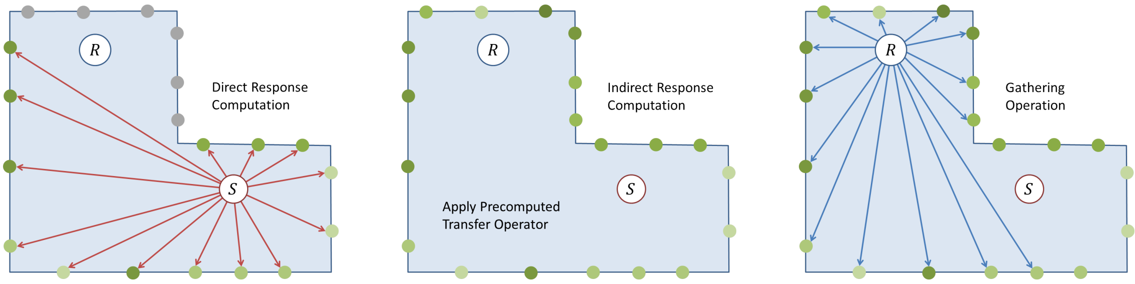
6. RESULTS

In this section, we present the results of performance evaluation and benchmarking experiments carried out on our proposed framework. All tests were carried out on an Intel Core 2 Quad desktop with 4GB RAM running Windows 7.



(a) (b)

Figure 5: Our modified image-source method. (a) An image tree is constructed by reflecting the receiver recursively across the surfaces in the scenes. Here, receiver R is reflected across the surfaces $\{A, B, C, D, E, F\}$ to construct first order images. Image R_A is reflected across surface B to construct higher order image R_{AB}. An image tree, $T(R, m)$, is constructed from these image sources. (b) A path validation is performed by attaching the source S to nodes in the image tree to compute valid specular paths from receiver R to the source S. A path is invalid if it is obstructed by objects in the scenes, e.g. image R_A, or does not lie within the scene primitive, e.g. image R_B. For MS-MR scenarios, we test that the delay of the path lie with the appropriate delay range ($[(i-1)\Delta T, (i+1)\Delta T]$) for receiver R_k and source S_{k-i}.



R
Direct Response Computation
S
R
Indirect Response Computation
S
Apply Precomputed Transfer Operator
R
Gathering Operation
S

(a) (b) (c)

Figure 6: An overview of our modified Direct-to-Indirect Acoustic Radiance Transfer method. (a) *Direct responses* are computed at the sample points from the source. (b) *Indirect responses* are computed at these sample points by performing a matrix-vector multiplication with the transfer matrix. The transfer matrix is computed in a pre-processing step. (c) The final IR is computed at the receiver by tracing rays (or "gathering") from the receiver. For MS-MR scenarios, the response from the past sources is stored at the surface samples and gathered at run time.

(a) (b) (c)

Figure 7: Benchmark scenes. (a) Room (876 triangles), (b) Hall (180 triangles), (c) Sigyn (566 triangles).

Figure 7 shows the scenes we used for benchmarking.

We first summarize the performance of our signal processing pipeline (as described in Section 4.2). All timings are reported for a single CPU core. As the table shows, a signal processing pipeline based on block convolution can efficiently handle a large number of sound sources per audio frame.

Table 6 quantifies the computational cost of modeling MS-MR scenarios in the image-source method for sound propagation (see Section 5). In our implementation, the image tree is computed using a single CPU core, whereas path validation is performed using all 4 CPU cores. Column 3 shows the time taken for computing the image tree for 2 orders of reflection. Columns 4 and 6 show the time taken for performing path validation, for SS-SR scenarios and MS-MR scenarios respectively. The computational overhead is shown in Column 8.

Table 6 quantifies the computational cost of modeling MS-MR scenarios using direct-to-indirect acoustic radiance transfer (see Section 5). Our implementation of this algorithm uses all 4 CPU cores. Column 3 shows the number of surface samples used for each scene. Column 4 shows the time taken to compute the direct response at each surface sample, and apply the transfer matrix to compute the indirect response at each surface sample. Columns 5 and 7 show the time taken for gathering the final IR at the receiver, for SS-SR scenarios and MS-MR scenarios respectively. The computational overhead is shown in Column 9.

Frame Size IR length	10 ms	20 ms	50 ms	100 ms
1 sec	50.6 ms	18.3 ms	10.4 ms	7.9 ms
2 sec	97.8 ms	36.9 ms	19.1 ms	14.8 ms
3 sec	147.0 ms	54.4 ms	27.0 ms	21.2 ms

Table 2: Timing results for signal processing pipeline based on STFT. The table shows the time needed to compute 1 second of final audio output for different frame sizes and the length of the IR. Thus, for IR of length 2 sec and audio frames of size 50 ms, our signal processing pipeline takes about 19 ms to compute 1 second of output audio, and therefore it can efficiently handle up to 50 sound sources.

Since higher-order reflections are precomputed and stored in a transfer matrix, the overhead percentages shown are constant regardless of the number of orders of reflections.

7. CONCLUSION AND FUTURE WORK

Real-time auralization for dynamic scenes is a challenging problem. We have presented a framework for performing real-time auralization in scenes where both the sound sources and the receiver may move. A key component of this framework is an equation describing how time-dependent impulse responses can be convolved with streaming audio emitted from a sound source to determine the sound signal arriving at the receiver. This equation can be used to derive simpler equations which describe auralization in situations where either the sources or the receiver or both may be static.

Another key component is the insight that for the receiver position in a given audio frame, we do not need full impulse responses from all previous source positions. This insight allows us to suitably modify sound propagation algorithms (such as the image-source method or direct-to-indirect acoustic radiance transfer) to handle moving sources and a moving receiver without incurring a significant computational cost as compared to the naïve approach of computing full impulse responses from all previous source positions.

Correctly performing auralization for scenes with dynamic geometry remains an interesting avenue for future work. Such scenes commonly occur in interactive virtual environments and video games. Another important question that remains to be addressed is that of the perceptual impact of correctly modeling auralization for dynamic scenes.

8. REFERENCES

[1] J. B. Allen and D. A. Berkley. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America*, 65(4):943–950, April 1979.

[2] L. Antani, A. Chandak, M. Taylor, and D. Manocha. Direct-to-Indirect Acoustic Radiance Transfer. *IEEE Transactions on Visualization and Computer Graphics*, to appear.

[3] C. Avendano. *Audio Signal Processing for Next-Generation Multimedia Communication Systems*, chapter 13. Kluwer Academic Publishers, Norwell, MA, USA, 2004.

[4] `https://axon.dolby.com`.

[5] D. Botteldooren. Finite-difference time-domain simulation of low-frequency room acoustic problems. *Acoustical Society of America Journal*, 98:3302–3308, December 1995.

[6] A. Chandak, L. Antani, M. Taylor, and D. Manocha. Fast and Accurate Geometric Sound Propagation Using Visibility Computations. In *International Symposium on Room Acoustics, ISRA*, Melbourne, Australia, August 2010.

[7] J. R. Cooperstock. Multimodal telepresence systems. *Signal Processing Magazine, IEEE*, 28(1):77–86, 2011.

[8] T. Funkhouser, N. Tsingos, I. Carlbom, G. Elko, M. Sondhi, J. E. West, G. Pingali, P. Min, and A. Ngan. A beam tracing method for interactive architectural acoustics. *The Journal of the Acoustical Society of America*, 115(2):739–756, 2004.

[9] T. Funkhouser, N. Tsingos, and J.-M. Jot. Survey of Methods for Modeling Sound Propagation in Interactive Virtual Environment Systems. *Presence and Teleoperation*, 2003.

[10] M. Gerardi, B. O. Rothbaum, K. Ressler, M. Heekin, and A. Rizzo. Virtual reality exposure therapy using a virtual iraq: Case report. *Journal of Traumatic Stress*, 21(2):209–213, 2008.

[11] B. Kapralos. *The Sonel Mapping Acoustical Modeling Method*. PhD thesis, York University, Toronto, Ontario, September 2006.

[12] M. Kleiner, B.-I. Dalenbäck, and P. Svensson. Auralization - an overview. *JAES*, 41:861–875, 1993.

[13] R. G. Kouyoumjian and P. H. Pathak. A uniform geometrical theory of diffraction for an edge in a perfectly conducting surface. *Proceedings of the IEEE*, 62(11):1448–1461, November 1974.

[14] A. Krokstad, S. Strom, and S. Sorsdal. Calculating the acoustical room response by the use of a ray tracing technique. *Journal of Sound and Vibration*, 8(1):118–125, July 1968.

[15] T. Lentz, D. Schröder, M. Vorländer, and I. Assenmacher. Virtual reality system with integrated sound field simulation and reproduction. *EURASIP J. Appl. Signal Process.*, 2007:187–187, January 2007.

[16] B. L. Mann. The evolution of multimedia sound. *Computers & Education*, 50(4):1157–1173, 2008.

[17] D. McGookin, S. Brewster, and P. Priego. Audio bubbles: Employing non-speech audio to support tourist wayfinding. In M. Altinsoy, U. Jekosch, and S. Brewster, editors, *Haptic and Audio Interaction Design*, volume 5763 of *Lecture Notes in Computer Science*, pages 41–50. Springer Berlin / Heidelberg, 2009.

[18] H. Medwin, E. Childs, and G. M. Jebsen. Impulse studies of double diffraction: A discrete huygens interpretation. *The Journal of the Acoustical Society of America*, 72(3):1005–1013, 1982.

[19] A. Melzer, M. Kindsmuller, and M. Herczeg. Audioworld: A spatial audio tool for acoustic and cognitive learning. In R. Nordahl, S. Serafin, F. Fontana, and S. Brewster, editors, *Haptic and Audio Interaction Design*, volume 6306 of *Lecture Notes in Computer Science*, pages 46–54. Springer Berlin / Heidelberg, 2010.

[20] T. Moeck, N. Bonneel, N. Tsingos, G. Drettakis,

Scene	Triangles	Tree Construction	SS-SR		MS-MR		Overhead
			Validation	Total	Validation	Total	
Hall	180	0.185 s	1.49 ms	0.186 s	3.91 ms	0.189 s	1.3 %
Room	876	1.001 s	12.09 ms	1.013 s	15.22 ms	1.016 s	0.3 %
Sigyn	566	1.030 s	11.09 ms	1.041 s	19.35 ms	1.049 s	0.8 %

Table 3: Computational overhead due to modeling MS-MR scenarios using the image-source method.

Scene	Triangles	Surface	Direct + Indirect	SS-SR		MS-MR		Overhead
		Samples	Response	Gather	Total	Gather	Total	
Hall	180	177	116.3 ms	31.2 ms	147.5 ms	52.8 ms	169.1 ms	14.6 %
Room	876	252	126.7 ms	36.4 ms	163.1 ms	76.9 ms	203.6 ms	24.8 %
Sigyn	566	1024	369.2 ms	122.9 ms	492.1 ms	234.5 ms	603.7 ms	22.7 %

Table 4: Computational overhead due to modeling MS-MR scenarios using direct-to-indirect acoustic radiance transfer.

I. Viaud-Delmon, and D. Alloza. Progressive perceptual audio rendering of complex scenes. In *I3D '07: Proceedings of the 2007 symposium on Interactive 3D graphics and games*, pages 189–196, New York, NY, USA, 2007. ACM.

[21] E.-M. Nosal, M. Hodgson, and I. Ashdown. Improved algorithms and methods for room sound-field prediction by acoustical radiosity in arbitrary polyhedral rooms. *The Journal of the Acoustical Society of America*, 116(2):970–980, 2004.

[22] M. Pielot, N. Henze, W. Heuten, and S. Boll. Tangible user interface for the exploration of auditory city maps. In I. Oakley and S. Brewster, editors, *Haptic and Audio Interaction Design*, volume 4813 of *Lecture Notes in Computer Science*, pages 86–97. Springer Berlin / Heidelberg, 2007.

[23] N. Raghuvanshi, R. Narain, and M. C. Lin. Efficient and accurate sound propagation using adaptive rectangular decomposition. *IEEE Trans. Vis. Comput. Graph.*, 15(5):789–801, 2009.

[24] N. Raghuvanshi, J. Snyder, R. Mehra, M. C. Lin, and N. K. Govindaraju. Precomputed wave simulation for real-time sound propagation of dynamic sources in complex scenes. *ACM Trans. Graph.*, 29(4), 2010.

[25] S. Sakamoto, T. Yokota, and H. Tachibana. Numerical sound field analysis in halls using the finite difference time domain method. In *RADS 2004*, Awaji, Japan, 2004.

[26] L. Savioja, J. Huopaniemi, T. Lokki, and R. Väänänen. Creating Interactive Virtual Acoustic Environments. *J. Audio Eng. Soc.*, 47(9):675–705, 1999.

[27] S. Siltanen, T. Lokki, S. Kiminki, and L. Savioja. The room acoustic rendering equation. *The Journal of the Acoustical Society of America*, 122(3):1624–1635, September 2007.

[28] S. Siltanen, T. Lokki, and L. Savioja. Frequency domain acoustic radiance transfer for real-time auralization. *Acta Acustica united with Acustica*, 95:106–117(12), 2009.

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

[30] C. M. J. Tsilfidis, Alexandros; Papadakos. Hierarchical perceptual mixing. In *Audio Engineering Society Convention 126*, 5 2009.

[31] N. Tsingos. Artifact-free asynchronous geometry-based audio rendering. In *Proceedings of the Acoustics, Speech, and Signal Processing, 2001. on IEEE International Conference - Volume 05*, pages 3353–3356, Washington, DC, USA, 2001. IEEE Computer Society.

[32] N. Tsingos. A versatile software architecture for virtual audio simulations. In *International Conference on Auditory Display (ICAD)*, Espoo, Finland, 2001.

[33] N. Tsingos. Scalable perceptual mixing and filtering of audio signals using an augmented spectral representation. In *Proceedings of the International Conference on Digital Audio Effects*, September 2005. Madrid, Spain.

[34] N. Tsingos. Precomputing geometry-based reverberation effects for games. In *Audio Engineering Society Conference: 35th International Conference: Audio for Games*, 2 2009.

[35] M. Vorlander. Simulation of the transient and steady-state sound propagation in rooms using a new combined ray-tracing/image-source algorithm. *The Journal of the Acoustical Society of America*, 86(1):172–178, 1989.

[36] M. Wand and W. Straßer. Multi-Resolution Sound Rendering. In *SPBG'04 Symposium on Point-Based Graphics 2004*, pages 3–11, Zürich, Switzerland, 2004.

[37] E. Wenzel, J. Miller, and J. Abel. A software-based system for interactive spatial sound synthesis. In *International Conference on Auditory Display (ICAD)*, Atlanta, GA, April 2000.

[38] G. R. White, G. Fitzpatrick, and G. McAllister. Toward accessible 3d virtual environments for the blind and visually impaired. In *Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*, DIMEA '08, pages 134–141, New York, NY, USA, 2008. ACM.