EFFICIENT TECHNIQUES FOR WAVE-BASED SOUND PROPAGATION IN INTERACTIVE APPLICATIONS

Ravish Mehra

A dissertation submitted to the faculty at the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2015

Approved by:

Dinesh Manocha

Ming Lin

Anselmo Lastra

Turner Whitted

Marc Niethammer

ABSTRACT

RAVISH MEHRA: Efficient Techniques for Wave-Based Sound Propagation in Interactive Applications
(Under the direction of Dinesh Manocha)

Sound propagation techniques model the effect of the environment on sound waves and predict their behavior from point of emission at the source to the final point of arrival at the listener. Sound is a pressure wave produced by mechanical vibration of a surface that propagates through a medium such as air or water, and the problem of sound propagation can be formulated mathematically as a second-order partial differential equation called the *wave equation*. Accurate techniques based on solving the wave equation, also called the wave-based techniques, are too expensive computationally and memory-wise. Therefore, these techniques face many challenges in terms of their applicability in interactive applications including sound propagation in large environments, time-varying source and listener directivity, and high simulation cost for mid-frequencies.

In this dissertation, we propose a set of efficient wave-based sound propagation techniques that solve these three challenges and enable the use of wave-based sound propagation in interactive applications. Firstly, we propose a novel equivalent source technique for interactive wave-based sound propagation in large scenes spanning hundreds of meters. It is based on the equivalent source theory used for solving radiation and scattering problems in acoustics and electromagnetics. Instead of using a volumetric or surface-based approach, this technique takes an object-centric approach to sound propagation. The proposed equivalent source technique generates realistic acoustic effects and takes orders of magnitude less runtime memory compared to prior wave-based techniques.

Secondly, we present an efficient framework for handling time-varying source and listener directivity for interactive wave-based sound propagation. The source directivity is represented as a linear combination of elementary spherical harmonic sources. This spherical harmonic-based representation of source directivity can support analytical, data-driven, rotating or time-varying

directivity function at runtime. Unlike previous approaches, the listener directivity approach can be used to compute spatial audio (3D audio) for a moving, rotating listener at interactive rates. Lastly, we propose an efficient GPU-based time-domain solver for the wave equation that enables wave simulation up to the mid-frequency range in tens of minutes on a desktop computer. It is demonstrated that by carefully mapping all the components of the wave simulator to match the parallel processing capabilities of the graphics processors, significant improvement in performance can be achieved compared to the CPU-based simulators, while maintaining numerical accuracy.

We validate these techniques with offline numerical simulations and measured data recorded in an outdoor scene. We present results of preliminary user evaluations conducted to study the impact of these techniques on user's immersion in virtual environment. We have integrated these techniques with the Half-Life 2 game engine, Oculus Rift head-mounted display, and Xbox game controller to enable users to experience high-quality acoustics effects and spatial audio in the virtual environment.

To my loving family who supported me whole-heartedly in this endeavor and have always been a source of strength to me.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my adviser, Prof. Dinesh Manocha, who gave me the freedom to pursue my own research problems and provided me with all the necessary tools to solve those problems. His guidance and support was of tremendous help and kept me focused on my research goals.

I would also like to thank my committee member, Prof. Ming Lin, for introducing me to the fascinating field of sound propagation and guiding me through many of the initial hurdles of this field. I am grateful to my committee members, Prof. Alselmo Lastra, Prof. Marc Niethammer, and Prof. Turner Whitted, for their guidance and insightful feedback which enabled me to refine and polish this dissertation to the highest standards.

This dissertation would not have been possible without the help of my colleagues who spent countless late nights in the lab alongside me to make sure we met the many SIGGRAPH deadlines. I had the great fortune of collaborating with some of the sharpest minds and would like to personally thank the following people: Nikunj Raghuvanshi for providing valuable guidance during the GPU-ARD parallelization and ESM projects, Lakulish Antani for co-discovering the derivative-based plane wave decomposition method, Anish Chandak for helping out with initial framework for the acoustic validation work, and Atul Rungta for helping me conduct user evaluations for the equivalent source technique. I would also like to thank Zhimin Ren, Hengchin Yeh, Sujeong Kim, Carl Schissler, Nicoles Morales, Alok Meshram, and Abhinav Golas, for collaborating with me over the last five years. I am also grateful to my fellow GAMMA group members, Rahul Narain, Jason Sewall, and Micah Taylor, for the countless stimulating discussions.

I also had the the privilege to work with some of the leading minds in the field of acoustics and computer graphics. I am grateful to Dr. Keith Wilson and Dr. Donald Albert from the US Army Cold Regions Research Lab, Prof. Lauri Savioja from Aalto university, and Dr. John Snyder from

Microsoft Research. They were instrumental in various project discussions and provided me with feedback on my research ideas.

The love and support of my parents, Sushil Kumar Mehra and Geeta Mehra, and my sister Neha Mehra has been instrumental in the successful completion of this dissertation. My time in graduate school has been made easier and enjoyable due to the many amazing people that I met here and now have the privilege to call my friends: Debapriya Basu, Abhinav Golas, Shabbar Ranapurwala, Rebecca Bruening, Aleksandra Rebeka, Elizabeth Rogawski, and Sridutt Balachandra.

Last but not the least, I would like to thank the various funding agencies whose support enabled this research: Link Foundation Fellowship in Advanced Simulation and Training, ARO Contracts W911NF-12-1-0430, W911NF-13-C-0037,W911NF-14-1-0437, and the National Science Foundation (NSF awards 1320644 and 1305286).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ADM	Average Decibel Metric
ARD	Adaptive Rectangular Decomposition
BEM	Boundary Element Method
BTM	Biot-Tolstoy-Medwin
CPU	Central Processing Unit
DCT	Discrete Cosine Transform
DTW	Dynamic Time Warping
ESM	Equivalent Source Method
FDTD	Finite-Difference Time Domain
FEM	Finite Element Method
FFT	Fast Fourier Transform
FMM	Fast Multipole Method
GPU	Graphics Processing Unit
GA	Geometric Acoustics
HRTF	Head-Related Transfer Function
HMD	Head-Mounted Display
IDCT	Inverse Discrete Cosine Transform
LOS	Line-Of-Sight
NLOS	Non-Line-Of-Sight
PML	Perfectly Matched Layer
SDM	Spectrogram Difference Metric
SH	Spherical Harmonics

CHAPTER 1: INTRODUCTION

Sound is ubiquitous in the physical world surrounding us and forms the basic medium of human communication *via* language and human artistic expression *via* music. Sound is a pressure wave produced by mechanical vibration of a surface that propagates through a medium, such as air and water, and consists of frequencies within the range of human hearing (20 Hz - 20 kHz). Sound waves with frequencies lower than 20 Hz are called *infrasound* and higher than 20 kHz are called *ultrasound*.

Sound propagation or acoustic wave propagation is the process of transmission of sound waves as they are emitted by the source, interact with the environment, and reach the listener to generate the sensation of hearing. The interaction of sound waves with the environment leads to multiple propagation effects such as specular reflection, diffuse reflection, scattering, diffraction, and interference. Therefore, the sound waves that reach the listener become different from the sound waves emitted by the source. Mathematically, the phenomenon of sound propagation can be modeled as a second-order partial differential equation called the *acoustic wave equation* in time-domain or the *Helmholtz equation* in frequency domain.

1.1 Motivation

Sound propagation has applications in wide variety of areas including engineering, architecture, entertainment and music, telephony, and noise control. In the field of virtual reality (VR), realistic sound propagation is crucial to immerse the user in the virtual environment and create a *sense of presence*. In augmented reality (AR), sound propagation can be used to reinforce the physicality of virtual objects in AR environments by producing sound events that are synchronized with the interaction of virtual objects with the real-world. Sound propagation is important in the field architectural design and planning to determine the acoustics of a design before physically building it.

This is extremely critical for large scale projects, such as designing airports or concert halls, where the cost of fixing an acoustic defect post-construction is exorbitantly high. In urban planning and city design, sound propagation can be used to determine noise levels at different places in the urban environment and help inform decisions regarding the locations of noise sensitive buildings such as schools or hospitals. In computer gaming and visual effects industry, sound propagation effects can make the gaming environment more immersive and improve the overall gaming experience. In the field of data visualization, well designed visualization techniques are used to convey information about scientific, engineering, or medical data. Similarly, sound can be used as a channel for conveying information by *auralizing* the data also known as *sonification*. A simple example of sonification would be the clicking rate of a Geiger counter to convey information about the level of radiation in the vicinity. In the field of human-computer interaction, sound propagation effects, such as ultrasound interference and Doppler shift, can be used to create novel user interfaces for volumetric haptics and hands-free gestures.

Sound propagation techniques model the effect of the environment on the sound waves and predict the behaviour of sound waves from the point of emission at the source to the final point of arrival at the listener. The existing techniques for performing sound propagation can be classified into three categories: (a) heuristic, (b) geometric, and (c) wave-based techniques. Typical heuristic techniques involve parametric digital filters to describe the overall acoustics of the space such as loudness and reverberation. These filters are manually designed by artists and game-audio designers based on their intuition and experience, and assigned to different locations in the game environment. Due to the fast runtime computation and low memory requirements, these techniques are typically used in computer gaming applications. Geometric techniques assume rectilinear propagation of sound waves i.e. ray-like behavior. This assumption is valid for high frequencies of sound where the wavelength of sound waves is much smaller than the size of everyday objects. These techniques trace rays (or beams) of sound from each sound source, propagate (reflect, scatter) these rays through the environment, and accumulate contributions at the listener. Typical geometric techniques include image source method, ray tracing, and beam tracing. Geometric techniques are approximate techniques which are used for high-frequency sound propagation. Due to their low computational and memory requirements, these techniques are used for sound propagation in interactive applications in the fields of virtual reality, architectural acoustics, and gaming. Wave-based techniques, also

called numerical acoustic techniques, solve the wave equation of sound propagation numerically and compute the propagated sound pressure field in the environment. Since these techniques solve the physics of sound propagation from first principles, they can achieve high accuracy for all the frequencies of sound. Typical wave-based techniques are based on numerical solution methods such as finite element method, boundary element method, and finite-difference time-domain. These techniques have high computational and memory requirements due to which they are mainly used for high-accuracy simulations in offline engineering applications such as underwater acoustics, aircraft design, and automotive design.

1.2 Sound propagation in Interactive Applications

The state-of-the-art of sound propagation techniques for interactive applications is as follows:

Virtual Reality VR simulations have wide applications in training, therapy, and education. These simulations have been used for military training purposes in combat, flight and submarine simulators, for treating post-traumatic stress disorder (PTSD) in war veterans *via* VR exposure therapy, and for training medical personnel for emergency scenarios. Realistic sound propagation is extremely important in VR for improving the sense of presence and immersion of the user in the virtual environment and augmenting the visual sense of the user for increased situational awareness. For combat training simulations (Pellerin, 2011), sound propagation cues can provide additional information about the environment (small vs big, inside vs outside), about events happening outside the field-of-view (enemy sneaking from behind, fellow soldier giving instructions), and can help localize the direction of gunfire, etc. These cues would be available to the soldier in real-life scenario and it would be important for him/her to train with these cues in the VR simulation (Hughes et al., 2006). In VR exposure therapy, patients are asked to experience the trauma-related virtual environment and exposed to "trigger" stimuli (visual, auditory, olfactory, and tactile) in a controlled manner for therapeutic gain (Rizzo et al., 2007). In this scenario, sound propagation is necessary to generate accurate auditory cues to increase the patient's sense of immersion in the virtual environment.

Accurate propagation is crucial for creating high-fidelity VR simulations since the consequences of approximate (or low accuracy) technique could be severe, for example mis-learning and wrong skill acquisition, patients not responding to the VR exposure therapy. Current sound propagation

techniques used in most VR systems are either based on heuristic or geometric techniques. The first class of techniques has no direct correlation with the real world acoustics and the second one is an approximation that cannot accurately capture propagation effects dominant at low frequencies such as diffraction and interference. Therefore, it is important to develop efficient wave-based propagation techniques to perform accurate and realistic sound propagation for these interactive VR applications.

Gaming Computer graphics and visual effects have reached the pinnacle of visual realism in current generation games. However, game sound is still at its infancy and based on 15 year old technology (IASIG, 1999). The current state-of-the-art for sound propagation in games is based on heuristic techniques such as artist-designed reverb filter, ambient sound zones, and real-time parameter control (Damian, 2010). These techniques are not physically accurate and require a significant amount of manual-effort and precious artist time. Artists and game audio designers have developed good intuition for designing sound filters for simple indoor scenes. However, this intuition does not extend well to outdoor scenes such as forest, desert, etc. Realistic sound effects generated from wave-based techniques have the capability of making the gaming world more immersive and improve the overall gaming experience. In addition, wave-based propagation can provide a good starting point for these sound filters which the artists can then tune based on the game requirements. Therefore, there is need for interactive wave-based techniques which can fulfill the strict performance and memory constraints of these computer gaming applications.

Acoustic Auralization Auralization refers to the creation and reproduction of sound based on computer data. Acoustic auralization is the auralization of sound based on propagation of sound waves in a 3D environment. It has applications in many fields such as architectural acoustics, aircraft and automobile design, and city planning. The main purpose of acoustic auralization in these applications is to get a sense of how an environment would sound like before actually building it. Virtual walkthroughs that includes visual and aural feedback can be extremely useful in getting a sense of the space, improving the design and removing any acoustic defects. Current acoustic auralization softwares are based on geometric sound propagation techniques such as image source method, ray-tracing or a combination of both. These techniques employ the geometric approximation (wavelength $<<$ object size) and can only model specular and diffuse reflections. Wave effects, such as diffraction and interference, are hard to model using these techniques. However, all these

applications contain objects whose size is of the order of wavelength (like pillars, chairs, tables, trees, etc), due to which modeling the wave effects becomes extremely important. Therefore, there is a need to develop efficient wave-based techniques for these auralization applications.

1.3 Challenges and Goals

The state-of-the-art of wave-based sound propagation techniques include standard numerical solvers such as finite-difference time domain (FDTD), boundary element method (BEM), and finite element method (FEM). The computation complexity of these techniques scales with the volume or surface area of the scene and at least fourth power of frequency; the memory complexity scales as at least third power of frequency. Due to high computational and memory requirements, these techniques are mostly limited to offline applications. Recently, there has been some work on developing interactive wave-based sound propagation techniques (James et al., 2006; Tsingos et al., 2007; Raghuvanshi et al., 2010). These techniques typically split the computation in an offline preprocessing step and an interactive runtime step. This is similar to visual rendering approaches used for global illumination in the field of computer graphics (Sloan et al., 2002). The key is to keep the precomputation times feasible (turnaround time of few hours), and keep the runtime computation lightweight (10-100s of ms) and runtime memory low (in MBs). These techniques can perform wave-based sound propagation at interactive rates for certain cases such as sound radiation in free space (James et al., 2006), first-order scattering from surfaces (Tsingos et al., 2007), and sound propagation in small-to-medium sized scenes (Savioja, 2010; Raghuvanshi et al., 2010). However, interactive wave-based sound propagation still faces many challenges:

Large Environments Large scenes, which arise in many interactive applications ranging from games to VR simulations to auralization, present significant challenges for interactive, wave-based sound propagation techniques. The computational complexity of prior wave-based techniques scale as the fourth power of the scene dimension and the memory complexity scales as the third power of the scene dimension (Savioja, 2010; Raghuvanshi et al., 2010). These techniques take hours of precomputation and gigabytes of runtime memory for sound propagation in small to medium-sized scenes spanning tens of meters. For sound propagation in large scenes spanning hundreds of meters

typically found in interactive applications, it would not be feasible to run these prior interactive wave-based techniques.

Directional Sources and Listeners Most sound sources we encounter in real life, ranging from human voices through speaker systems, machines noises, and musical instruments, are directional sound sources with specific directivity pattern. Source directivity has a significant effect on the propagation of sound waves and the corresponding acoustics of the environment (Vigeant, 2008). Analogous to directional sources, listeners also do not receive sound equally from all directions. The directional sound at the listener combined with scattering of sound around the listener's head generates subtle differences in the sound received at the left and right ear. The human auditory system obtains significant directional cues from these subtle differences in the sound at each ear and can use it to perceive the direction of incoming sound. An audio signal containing these directional cues which when played at the listener's ears to give him/her a sense of direction of incoming sound is called *spatial audio* or *3D audio*.

Current interactive wave-based techniques can only model source directivity during the offline precomputation phase. As a result, the source directivity gets baked (precomputed and stored) into the final solution. Since current wave-based techniques have a high precomputation overhead, it is not possible to modify the directivity at runtime for interactive applications. This can be a problem for sound sources with time-varying directivity such as a person moving around or turning while talking. Additionally, integrating listener directivity into wave-based techniques requires decomposition of sound field into plane waves, also called the *plane-wave decomposition*. All the previous techniques for performing plane-wave decomposition are computationally expensive and limited to offline applications.

Mid-frequencies The frequency range of sound propagation can be divided into three categories: low-, mid- and high-frequencies (Hidaka et al., 1995; Abdou and Guy, 1996; Savioja, 2010). Low frequency range consists of frequencies less than few hundred Hz. In this range, the wavelength is of the same order as the size of the everyday objects and wave-effects, such as diffraction and interference, are dominant. High frequency range consists of frequencies greater than few kHz and less than the maximum audible frequency (20 kHz). In this frequency range, sound waves can be approximated by rays and sound propagation can be handled accurately by geometric techniques.

Mid-frequency range contains sounds with frequency greater than few hundred Hz and less than few kHz. This is a transition region where the acoustic effects transition from wave-effects to ray-based effects. Sound propagation modeling in the mid-frequency range is critical since many everyday sound sources emit sound in this range, for example human voice. For sound propagation at low-and mid-frequencies, wave-based sound propagation is necessary as wave-effects are prevalent at these frequencies. Current interactive wave-based propagation techniques have a high precomputation cost which varies as the fourth power of frequency. This results in hours of a preprocessing time for low frequencies and tens of hours (or even days) for mid-frequencies. This is too slow a turnaround time for scene design, noise control and auralization applications in games, architectural acoustics and virtual reality. Therefore, there is need to develop efficient wave-based techniques with faster precomputation time that can compute simulation results for both low and mid-frequencies in few hours.

1.4 Thesis Statement

Complex acoustic effects, such as diffraction, interference, high-order reflection, scattering, directivity, and spatial sound, can be simulated efficiently and practically for interactive applications through the use of wave-based sound propagation techniques.

In this dissertation, we propose a set of efficient wave-based sound propagation techniques that can handle large environments, source and listener directivity, and mid-frequencies of sound. The preprocessing times of these techniques is few hours, the runtime compute is in tens to hundreds of milliseconds, and the runtime memory is in tens to hundreds of megabytes. By solving the challenges faced by wave-based propagation techniques and satisfying the practical compute and memory requirements, these techniques have enabled wave-based sound propagation in interactive applications. We have integrated these techniques in a commercial game engine and demonstrated complex acoustic effects generated by wave-based sound propagation in a variety of scenarios.

1.5 Main Results

In this thesis, we present three efficient wave-based sound propagation techniques for interactive applications. Firstly, we present a novel equivalent source method for sound propagation in large

scenes which has a small runtime compute and memory footprint. Secondly, we present a general source and listener directivity framework to incorporate time-varying source directivity and interactive spatial audio in wave-based sound propagation techniques. Lastly, we present the GPU-based adaptive rectangular decomposition technique to precompute wave-based propagation results efficiently for mid-frequencies of sound. We discuss each of these contributions in detail:

1.5.1 Equivalent Source Method for Sound Propagation

Equivalent source method (ESM) is an interactive wave-based sound propagation technique based on the fundamental solutions of the wave equation known as *equivalent sources*. This is a novel technique for wave-based sound propagation in large, open scenes spanning hundred of meters with a small runtime memory footprint. In the preprocessing stage, the scene is decomposed into disjoint right objects. The free-field acoustic behavior of each object is captured by a compact per-object transfer function that relates the incoming field of each object to the outgoing scattered field. Pairwise acoustic interactions between the objects are computed analytically to yield inter-object transfer functions. The global sound field accounting for all orders of interaction (reflection, diffraction, scattering) is computed by solving a linear system consisting of the per-object and inter-object transfer functions. This field is stored compactly using the equivalent sources which form an efficient basis for representing the sound field. The runtime system uses a fast summation over the equivalent sources to auralize the sound field for a moving listener at interactive rates. Realistic acoustic effects, such as diffraction low-passing behind objects, sound focusing by concave reflectors, sound scattering by rough surfaces, high-order reflections, and echoes, are demonstrated. The key contributions of this work includes:

1. **Object-based sound field decomposition**: This technique takes an object-centric approach to wave-based sound propagation by decomposing the sound field into per-object and inter-object interactions of sound, thereby enabling interactive, wave-based sound propagation for large open scenes.
2. **Compact per-object transfer**: The per-object transfer function is encoded efficiently and compactly by the use of the equivalent source basis.

3. **Compact analytical coupling of objects**: The analytical coupling between the objects is achieved by expressing the inter-object transfer functions in the same equivalent source basis as the per-object transfer function.

4. **Fast, memory efficient runtime**: This enables interactive auralization for a moving listener while requiring tens of megabytes of runtime memory.

This technique has been tested on a variety of scenarios and integrated with the Valve's Source^{TM} game engine. This equivalent source technique generates realistic acoustic effects and takes orders of magnitude less runtime memory compared to other state-of-the-art wave-based techniques, enabling interactive performance.

1.5.2 Source and Listener Directivity Framework

This is an approach to model time-varying source and listener directivity in interactive wave-based sound propagation techniques. The source directivity is represented as a linear combination of elementary spherical harmonic sources. In the preprocessing stage, the propagated sound fields corresponding to each spherical harmonic source are precomputed, encoded, and stored for runtime use. At runtime, the spherical harmonic decomposition of the time-varying source directivity is performed at interactive rates to compute the spherical harmonic coefficients. The total sound field at the listener is computed as a linear combination of precomputed propagated sound fields of the spherical harmonic sources with their respective coefficients. A novel plane-wave decomposition technique based on derivatives of the pressure field is proposed. This technique can compute spatial audio for a moving listener with a time-varying listener directivity at interactive rates. A general framework is proposed for incorporating source and listener directivity in interactive wave-based propagation techniques. The main contributions of this work are as follows:

1. **Time-varying source directivity**: The spherical harmonic-based representation of source directivity can support analytical, data-driven, rotating or any time-varying function at runtime. Pressure field due to spherical harmonic sources are precomputed and then used at runtime to to compute the pressure field due to a time-varying directional source.

2. **Efficient plane-wave decomposition**: The plane-wave decomposition approach is based on derivatives of the pressure field at the listener position. Unlike previous approaches, this

technique can compute plane-wave decomposition at interactive rates. This enables listener motion, head rotation, and dynamic listener directivity at runtime.

3. **General framework** to integrate both the source and listener directivity approach in any offline or interactive wave-based sound propagation technique.

4. **Fast, memory efficient auralization system**: This technique has a low runtime memory footprint and computes wave-based sound propagation at interactive rates.

This approach has been incorporated in the equivalent source technique and integrated with Valve's Source^{TM} game engine. Realistic acoustics effects due to source and listener directivity are demonstrated on a variety of scenes, such as people talking on the street, loudspeakers between buildings, a rotating siren, and musical instruments being played in amphitheater.

1.5.3 Parallel Adaptive Rectangular Decomposition

In this work, an efficient GPU-based time-domain solver for the acoustic wave equation is presented. It is based on the adaptive rectangular decomposition (ARD) algorithm (Raghuvanshi et al., 2009b) which uses analytical solutions within rectangular partitions for spatial invariant speed of sound. The proposed GPU-based ARD technique is suitable for performing fast computations for wave-based sound propagation in both low- and mid-frequency range. It is demonstrated that by carefully mapping all the components of the ARD algorithm to match the parallel processing capabilities of the graphics processors, significant improvement in performance can be achieved compared to the corresponding CPU-based solver whilst maintaining numerical accuracy. Substantial performance gain over a high-order finite-difference time-domain method is observed. Using this technique, a 1 second long simulation can be performed on scenes of air volume 7500 m^3 till 1650 Hz within 18 minutes compared to the corresponding CPU-based ARD solver that takes around 5 hours and a high-order finite-difference time-domain solver that could take up to three weeks on a desktop computer. The main results of this work are as follows:

1. **Mid-frequency simulations**: A parallel technique in combination with efficient utilization of the GPU architecture to enable wave simulation in the mid frequency range on a desktop computer.

2. **Two levels of parallelism**: The technique exploits both coarse- and fine-grained parallelism exhibited by the ARD algorithm to achieve high performance.

3. **Avoiding host-device data transfer bottleneck**: The technique avoids any host-device data transfer bottleneck between the host (CPU) and the device (GPU) by parallelizing all the steps of the algorithm on the GPU.

4. **Computationally-efficient decomposition**: The approach utilizes a modified rectangular decomposition to take advantage of the computational efficiency of the Fast Fourier Transform (FFT) operations on grids of size powers of two. This results in significant speedup for the computationally heavy steps of the algorithm.

This technique demonstrates that it is possible to effectively parallelize all the steps of the ARD algorithm on current GPU architectures and to exploit the raw computational power of the high number of GPU cores. The parallel GPU-based ARD solver achieves a speedup of up to 25 times over the optimized CPU-based ARD solver, and three orders of magnitude speedup over the high-order CPU-based finite difference solver.

1.6 Thesis Organization

The rest of the dissertation is organized as follows.

- In **Chapter 2**, the mathematical details of sound propagation are provided along with a literature review of the prior sound propagation techniques, and existing work in the area of directional sources and listeners.
- **Chapter 3** introduces the equivalent source method for performing wave-based sound propagation in large environments. The chapter starts by providing background on the theory of equivalent sources, discusses the different steps of the technique, provides implementation details, and presents the results and analysis of the proposed technique.
- In **Chapter 4**, the source and listener directivity formulation based on spherical harmonics is introduced. A general framework is provided for integrating these directivities with wave-based sound propagation techniques along with implementation details and results of the proposed approach.

- **Chapter 5** starts with the background on the adaptive rectangular decomposition technique. Next, it discusses the GPU-based adaptive rectangular decomposition solver for efficiently computing simulation results in the mid-frequency range for wave-based sound propagation.
- **Chapter 6** deals with the validation of the techniques discussed in the previous chapters by comparing their results with analytical solutions, state-of-the-art numerical solvers and measured data from real world.
- **Chapter 7** presents the results of the two preliminary user evaluations conducted for the equivalent source method and the source and listener directivity approach.
- **Chapter 8** concludes the thesis and discusses avenues for future work.

CHAPTER 2: BACKGROUND AND PREVIOUS RESEARCH

This chapter provides a literature review of the field of sound propagation.

2.1 Acoustic Wave Equation

Sound is a pressure wave produce by mechanical vibration of a surface. Sound propagation algorithms predict the behavior of sound waves as they interact with the environment. The physics of sound propagation can be described by the well-known time-domain formulation of the acoustic wave equation:

$$\frac{\partial^2 p}{\partial t^2} - c^2 \nabla^2 p = f\left(\mathbf{x}, t\right). \tag{2.1}$$

The wave equation models sound waves as a time-varying pressure field $p\left(\mathbf{x}, t\right)$. While the speed of sound in air (denoted by c) exhibits fluctuations due to variations in temperature and humidity, the acoustic effects of these fluctuations are ignored for most indoor scenes and many outdoor scenes i.e. a homogenous media assumption is made. Sound sources in the scene are modeled by the forcing field denoted by $f\left(\mathbf{x}, t\right)$ on the right hand side in the Equation 2.1. The operator $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$ is the Laplacian operator in 3D. The wave equation succinctly captures wave propagation phenomena, such as interference and diffraction, observed in real world.

In the frequency-domain, sound propagation can be expressed as a boundary value problem for the Helmholtz equation:

$$\nabla^2 P + \frac{w^2}{c^2} P = 0 \ \text{ in } \Omega, \tag{2.2}$$

where $P\left(\mathbf{x}, \omega\right)$ is the (complex-valued) pressure field in the frequency domain, ω is the angular frequency, and Ω is the acoustic domain.

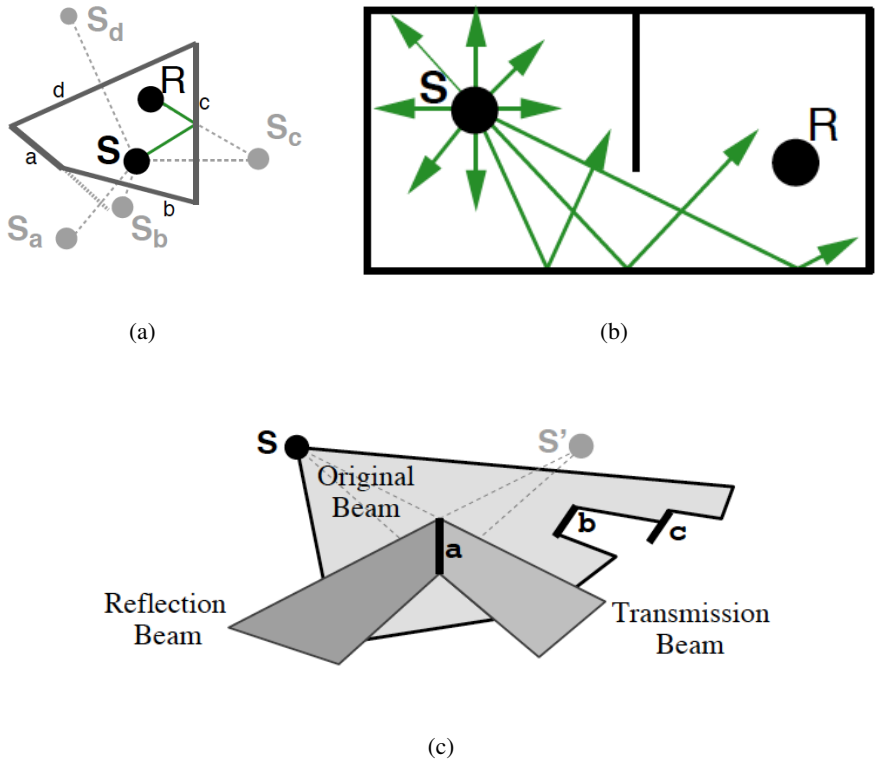
2.2 Sound Propagation Techniques

The techniques for computing the propagated sound field in an environment can be classified into three categories:

Heuristic Techniques These are propagation techniques that do not solve the physics of sound propagation but are geared more towards quick evaluation and artist-control. These are light-weight techniques that typically have very small runtime memory or compute requirement. Due to these reasons, these techniques are popular in the gaming industry to generate different types of acoustic effects. The first among these is *reverb*, short for *reverberation*, which corresponds to the decay envelope of the sound field in an environment. Sound produced by the source undergoes reflections, scattering and absorption in the environment resulting in decay of sound pressure over time. Reverb gives the player a sense of size and type of the environment (small vs large, indoor vs outdoor, cave vs desert). In case of heuristic techniques, predetermined reverb filters designed by artists are assigned to different regions in the environment also called *reverb zones*. At runtime, based on the position of the player with respect to the reverb region, an appropriate reverb filter is applied to the sound. Additionally, these reverb settings can be extended to the sources such that the sound coming from a source in a different reverb zone can use the reverb setting of its origin.

The second most popular technique is called real-time parameter control (RTPC). In this technique, the parameters of the audio content are changed based on the events or actions in the game, for example, the tempo of the footstep sound can be changed based on the walking speed of the character, "swoosh" sound of a sword can be made shriller based on its swinging speed, or loudness of the sound can be reduced based on the distance between the source and the player. Audio middleware toolsets, such as AudioKinetic's WWise or FMOD, support such capabilities. In recent years, certain advanced heuristic techniques have been developed, for example, ray-tracing and convolution technique were used in the game Crackdown. In this technique, few rays are shot in the scene to get an approximation of the size of the environment. Based on this information, the reverb filter is changed dynamically and applied to the source audio signal using the convolution operation. Dolby has recently introduced the Axon middleware toolkit that enables surround-sound live chat between the game players (Dolby, 2010). More details about these heuristic techniques can be found in (Damian, 2010).

Geometric Techniques These techniques are based on the high-frequency approximation for sound propagation which assumes that the wavelength of sound waves is smaller than the object size. Under this assumption, sound waves can be treated as rays and sound propagation can be performed using ray-based techniques (Funkhouser et al., 2003). However, this assumption is only valid for high frequencies of sound (> few kHz) thereby preventing the applicability of these techniques in the low- or mid-frequency range (< few kHz).



(a) (b)

(c)

Figure 2.1: The most common geometric techniques of sound propagation: (a) Image Source Method, (b) Ray Tracing, and (c) Beam Tracing (images © (Funkhouser et al., 2003)).

The most common geometric technique is the image source method (Allen and Berkley, 1979; Borish, 1984). This technique is used for modeling specular reflections of sound. It computes the specular reflection paths of sound by mirroring the sound source over the polygonal surfaces of the environment thereby creating *virtual sources*. First-order specular reflection paths are constructed by joining the lines between the virtual sources and the listener. This process can be repeated recursively to generate higher-order specular reflection paths. The computation complexity of this technique

grows exponentially with the order of reflections. Also, each specular path needs to be validated to make sure it is feasible since not all specular paths generated are physically realizable.

Ray tracing is another method to compute sound propagation paths from sources to listener as the sound ray gets specularly reflected, diffusely reflected, scattered, or transmitted by the environment. It is interesting to note that ray tracing methods were first developed for sound propagation (Krokstad et al., 1968) before being widely adopted for light transport in the computer graphics community (Whitted, 1980). The primary operation used in all ray tracing techniques is the ray-surface intersection test. This technique is popular due to ease of implementation and sub-linear growth in computational complexity with number of polygons. One of the main drawbacks is the aliasing artifact caused by discrete sampling of a continuous sound field. In order to minimize these artifacts, a large number of rays are required. Image source method and ray tracing are among the two most popular geometric techniques and widely used in acoustic prediction softwares, such as ODEAN, CATT, EASE, in the field of architectural and room acoustics.

Beam and frustum tracing are geometric techniques that use pyramidal beams/frusta to compute propagation paths from source to listener through the environment (Dadoun et al., 1985; Funkhouser et al., 1998; Lauterbach et al., 2007; Chandak et al., 2008). These techniques trace pyramidal beams (or frusta) through the environment and compute beam-polygon (or frustum-polygon) intersection. After the intersection, the original beam (frustum) is split into reflection and transmission parts which are recursively traced in the scene to compute high-order reflections. Similar to ray-tracing, beam and frustum tracing can handle specular and diffuse reflections. However, in contrast with ray-tracing, these techniques take advantage of the spatial coherence in the scene to reduce the aliasing artifacts. Beam and frustum tracing are more computationally-expensive techniques due to complicated geometric operations involved in beam-polygon and frustum-polygon intersection test.

Geometric techniques are applicable only when the wavelength of sound is much smaller than the object size. Therefore, modeling of wave-effects, such as diffraction, has to be done separately. Diffraction is a wave phenomenon of sound that arises when the wavelength of waves is proportional to the object size. In this case, the sound waves bend around the object resulting in gradual reduction in the loudness as one enters the occluded region. The two formulations that incorporate diffraction in geometric techniques are the Uniform Theory of Diffraction (UTD) and the Biot-Tolstoy-Medwin (BTM) method. UTD is an approximate formulation that computes the

diffraction filter of a propagation path over an edge (Kouyoumjian and Pathak, 1974; McNamara et al., 1990). In this technique, a ray incident on a diffraction edge gives rise to a cone of diffraction rays to which frequency-dependent diffraction coefficients are applied. UTD is an approximate but efficient formulation used mainly for interactive applications (Tsingos et al., 2001). BTM is a more accurate formulation for modeling the diffraction phenomenon (Svensson et al., 1999). It computes a dense sampling of diffraction edges using elementary point sources which are then used to compute the frequency-dependent diffraction filter. BTM is an offline technique and used mainly for room acoustic prediction applications (Torres et al., 2001). Recently, interactive geometric techniques based on the UTD formulation have been proposed to handle high-order diffraction effects (Taylor et al., 2009; Schissler et al., 2014).

Wave-based Techniques These techniques discretize the volume or surface area of the scene and solve the wave equation numerically on this discretization to compute the propagating pressure field in the environment. Based on the type of discretization, these techniques can be classified as either *volumetric* or *surface-based* techniques. Volumetric techniques discretize the entire volume of the scene into grid cells and perform the numerical computation on a per-grid cell basis. The number of grid cells along with the computational and memory complexity of these techniques scale as at least the third power of scene dimension. Surface-based techniques, on the other hand, discretize the surface area of the scene into discrete surface elements and perform numerical computations on each discrete element. The number of surface elements as well as the compute and memory requirements of surface-based techniques scale as at least the square of scene dimension. Volumetric techniques are well-suited for scenes with high surface area and low air volume, which makes them highly applicable to indoor spaces. Similarly, surface-based techniques are better suited for scenes with high volume and (comparatively) smaller surface area such as outdoor scenes. Wave-based techniques can also be grouped into *time-domain* or *frequency-domain* techniques based on whether they are solving the acoustic wave equation (equation 2.1) or the Helmholtz equation (equation 3.1), respectively.

The Finite Difference Time Domain (FDTD) method is a volumetric technique which was explicitly designed for solving the time-domain wave equation, although in the context of electromagnetic simulation (Yee, 1966). The FDTD method for sound propagation solves for the time-dependent pressure field on a Cartesian grid by making discrete approximations of the spatial derivative opera-

tors and using an explicit time-stepping scheme. Assuming that the space has been discretized into a uniform Cartesian grid with spatial spacing h and the time-step is Δt, the pressure field $p\left(ih, jh, kh\right)$ at time $n\Delta t$ can be denoted as $p^{(n)}_{i,j,k}$. The spatial derivative is approximated in finite-difference approaches by applying a constant linear stencil β of size d as –

$$\nabla^2 p = \sum_{l=-d}^{d} \beta_l \left(p_{i+l,j,k} + p_{i,j+l,k} + p_{i,j,k+l}\right) + O(h^{2d}) \tag{2.3}$$

The spatial differentiation error is $O(h^{2d})$. The stencil has a compact support of $2d+1$. For most FDTD implementations, $d = 1$, yielding second-order spatial accuracy, with $\beta = \frac{1}{h^2}\{1, -2, 1\}$. The same analysis can be applied for the time derivative as well. It is typical in time-domain solvers to use second-order accurate time-stepping –

$$\frac{\partial^2 p}{\partial t^2}^{(n)} = \frac{1}{\Delta t^2}\left(p^{(n+1)} - 2p^{(n)} + p^{(n-1)}\right) + O(\Delta t^2). \tag{2.4}$$

Standard von Neumann analysis can be used to show that the spatio-temporal errors in FDTD appears as frequency-dependent phase velocity, known as numerical dispersion: as waves propagate, their shape is gradually destroyed due to loss of phase-coherence. FDTD has been applied to medium-sized scenes in 3D for room acoustic computations (Sakamoto et al., 2006, 2008). The authors calculated typical room acoustic parameters and compared the calculated values with the actual measured values in the scene. The implementation can take days of computation on a small compute cluster. A recent technique proposed in (Savioja, 2010) allows for real-time auralizations till a usable frequency of roughly 500 Hz on geometries with large volume using the Interpolated Wideband (IWB) FDTD scheme running on GPUs. The IWB-FDTD scheme uses optimized compact stencils for reducing numerical dispersion while keeping the computational expenditure low (Kowalczyk and van Walstijn, 2010).

Spectral techniques achieve much higher accuracy by expanding the field in terms of global basis functions. Typically, the basis set is chosen to be the Fourier basis or Chebyshev polynomials (Boyd, 2001) as the fast fourier transform (FFT) can be employed for the basis transformation. The Pseudo-Spectral Time Domain (PSTD) method is a spectral method proposed for underwater acoustics as an alternative to FDTD to control its numerical dispersion artifacts (Liu, 1997) . The PSTD is also

a volumetric technique that solves the time-domain wave equation. The key difference in PSTD compared to FDTD is to utilize spectral approximations for the spatial derivative (Liu, 1997). The spatial error in PSTD shows *geometric convergence*, $O(h^n), \ \ \forall n > 0, n \in Z$, allowing for meshes with samples per wavelength approaching the Nyquist limit, and still allowing vanishingly small dispersion errors in the spatial derivative. However, this holds only if the pressure field is periodic which is not commonly the case. Errors in ensuring periodicity appear as wrap-around effects where waves exiting from one end of the domain enter from the opposite end. Time update is done using a second-order explicit scheme similar to the FDTD method. Therefore, although spatial errors are controlled in PSTD, the errors due to temporal derivative approximation are still present and are of a similar magnitude as FDTD.

The most common wave-based techniques in frequency-domain include the boundary element method (BEM) (Cheng and Cheng, 2005) and the finite element method (FEM) (Zienkiewicz et al., 2006; Thompson, 2006). BEM is a numerical solution technique that solves the wave equation of sound propagation in the frequency-domain, i.e. the Helmholtz equation. BEM transforms the Helmholtz equation into boundary integral equations and solves the boundary integral equations on the surface of the scene by expressing the pressure field as the sum of elementary fields radiating from monopole and dipole sources placed on a uniform, sub-wavelength sampling of the scene's surface. Next, BEM reformulates the integral equations into a linear system and solves for the strengths of these monopole/dipole sources. These strengths can then be used to evaluate pressure at any point inside the acoustic domain by evaluating the boundary integral equations. Traditional BEM scales as the square of the surface area but recent research on the fast multipole method for BEM (FMM-BEM)(Liu et al., 2009; Gumerov and Duraiswami, 2009) has improved the complexity to linear in surface area. This was made possible by creating a hierarchical clustering of BEM monopoles and dipoles using an octree, and approximating their interactions compactly using high-order multipole Green's functions.

FEM is a volumetric technique that solves Helmholtz equation on a volumetric discretization of the scene (Zienkiewicz et al., 2006; Thompson, 2006). One of the main strengths of FEM is the capability to use unstructured grids cells with complex shapes, typically tetrahedras, thus allowing the scene boundary to be represented with much higher accuracy. One of the main concerns of FEM is the generation of good quality discretization for arbitrary domains. The stability and accuracy of

FEM computations be severely impacted by "thin" tetrahedras. Both FEM and BEM are employed for solving the Helmholtz equation, with FEM applied mainly to interior and BEM to exterior scattering problems.

In recent years, there has been increasing interest in developing interactive wave-based sound propagation techniques in indoor and outdoor spaces for applications in the field of computer graphics and visual effects. The first among these techniques deals with the problem of sound radiation from vibrating objects in free space (empty space with no surrounding objects) (James et al., 2006). In the first step, modal analysis is performed on the object to compute all the different modes of vibration. Sound radiation behavior of each mode of the vibrating object is then computed by solving the exterior radiation problem of the Helmholtz equation. This radiation behavior is efficiently encoded in terms of the acoustic transfer function which approximate the radiation behavior of a complicated geometry by expressing it in terms of an efficient basis of pressure field called the *equivalent sources*. The use of equivalent sources enables quick runtime evaluation of pressure field and interactive performance. However, this technique only handles sound radiation by an object in empty space and does not model sound propagation between different objects in the scene. (Tsingos et al., 2007) handles sound scattering between objects by solving the boundary integral formulation of the Helmholtz equation subject to the Kirchhoff approximation. The approximation enables efficient computation on graphics processors to achieve interactive performance. However, by using this approximation, only the first-order effects between the objects are modeled and high-order interactions, such as multiple reflections, diffractions, are ignored. In the work of (Raghuvanshi et al., 2010), wave-based sound propagation for dynamic source and listeners is performed by precomputing a volumetric sampling of impulse responses on a spatial grid of source and listener positions. This dataset is compressed by using perceptual encoding based on the acoustic properties of indoor spaces and stored for runtime use. At runtime, the impulse response for the instantaneous position of source and listener is computed by looking up the closest match in the dataset and interpolating the resulting impulse responses.

2.3 Directional Sources

Most sound sources that we come across in real life have a characteristic directivity pattern that varies with frequency (Meyer and Hansen, 2009). Source directivity has a significant effect on the propagation of sound in 3D space and the acoustics of an environment (Vigeant, 2008). Vigeant's PhD thesis is an excellent reference for details on the effect of source directivity on room acoustics (Vigeant, 2008). Over the last decade, many researchers have tried to measure directivity of real world sound sources. Meyer et al. measure the directivity of brass, woodwind and string instruments in an anechoic chamber (Meyer and Hansen, 2009). This data contains magnitude information only, measured in terms of relative sound pressure levels as a function of directions in different octave bands. (Otondo and Rindel, 2004) measured the tone-specific directivity of a trumpet, French horn, and clarinet. (Ahnert, 2005) have investigated the use of complex frequency response data (magnitude and phase) for loudspeaker directivity. They demonstrated that incorporating the phase data reduces the acoustic prediction error by an order of magnitude as compared to the magnitude-only data. Recently, directivities of male and female singing voices have also been measured (Jers, 2005). Many of these datasets are publicly available and have been widely in various sound propagation techniques.

The development of sound propagation algorithms for direction sources is a topic of ongoing research. Physically-based sound synthesis algorithms can simulate sound radiation from directional sources directly (Chadwick et al., 2009). However, these techniques handle only free-space sound radiation and do not model propagation effects (reflections, diffraction, reverberation) from other objects in the scene. Interactive GA techniques can incorporate high-frequency source directivities at runtime (Funkhouser et al., 2003; Otondo and Rindel, 2004). These methods essentially involve enumerating sound propagation paths from the source to the listener. The directions of rays emitted from the source (received at the listener) can be used to apply attenuation to the corresponding propagation paths based on any arbitrary source (listener) directivity. (Rindel and Otondo, 2005) proposed a data-driven approach using GA techniques for multi-channel auralization to account for time-variant nature of the directivity of musical instruments. In this method, a musical instrument is placed in an anechoic chamber and sound signals emitted by it is captured by microphones placed in fixed number of directions. Each microphone recording is stored in a separate audio channel. In

the computer acoustic simulation, a virtual source with directivity corresponding to the solid angle captured by each microphone, is simulated and its impulse response convolved with the appropriate channel and then combined to created a complete auralization. However, the geometric techniques are not accurate in the low frequency range (e.g. 20-1000Hz), as sound waves can diffract (bend) around obstacles and undergo interference and other wave effects. Interactive wave-based sound propagation techniques (Raghuvanshi et al., 2010; Mehra et al., 2013) can handle elementary directional sources such as monopoles, dipoles, quadrupoles, and their linear combinations. Other techniques have been proposed to incorporate measured directivities in wave-based techniques (Hacihabiboglu et al., 2008; Southern and Murphy, 2009). But the source directivity is modeled during the offline simulation stage and its effects on the sound propagation gets baked into the simulation results. Therefore, a source which at runtime either rotates or has a time-varying directivity, cannot modeled by current interactive wave-based techniques.

2.4 Spatial Audio

The human auditory system obtains significant directional cues from the subtle differences in sound received by each ear, caused by the scattering of sound around the head (Begault, 1994). These effects are represented using a *head-related transfer function* (HRTF). Measurements to compute the HRTF are performed in controlled environments and the recorded data is available online (Algazi et al., 2001). Interactive GA techniques can incorporate high-frequency HRTF-based listener directivity at runtime by enumerating sound propagation paths from the source to the listener. The directions of rays received at the listener can be used to look up into the HRTF and apply the corresponding attenuation to the propagation paths. However, handling listener directivity effects at low frequency arising due to the wave nature of sound (e.g. diffraction, interference) remains a significant challenge with interactive GA techniques. Integrating HRTFs into wave-based techniques involves computation of propagation directions using *plane wave decomposition*. Prior plane-wave decomposition techniques either use spherical convolution (Park and Rafaely, 2005; Rafaely and Avni, 2010) or solve a linear system (Zotkin et al., 2010), and are computationally expensive. Interactive wave-based techniques resort to simpler listener directivity models based on a spherical head and a cardioid function (Raghuvanshi et al., 2010). However, these simplified models are not accurate

for sound localization and externalization, both of which are necessary for immersion in virtual environments (Begault, 1994).

2.5 Parallel Wave Solvers

State-of-the-art wave solvers, such as FDTD or BEM, can take hours of compute and gigabytes of memory to solve the wave equation in a small-to-medium sized scene up to a few hundred Hz. The computational complexity of wave solvers scales as the fourth power of simulation frequency and memory complexity scales as third power of frequency. Therefore, for large scenes and frequencies up to a few kiloHertz, these solvers can take days (or even weeks) of compute and terabytes of memory making them unfeasible for most applications in the field of virtual reality, gaming, and auralizations. Significant research effort has been spent on parallelization of these wave solvers on modern generation graphics processor units (GPUs) and large compute clusters.

The FDTD method is by far the most popular choice for parallelization due to its simplicity and ease of implementation. Over the last two decades, many parallel solvers based on the FDTD method have been proposed on both GPUs (Sypek et al., 2009; Savioja, 2010) as well as on large compute clusters (Guiffaut and Mahdjoubi, 2001; Yu et al., 2005; Komatitsch et al., 2010). Other techniques have explored parallelization of the FEM method on distributed memory systems such as CPU clusters (Diekmann et al., 1996; Bhandarkar and Kal, 2000). Besides these solvers, many specialized parallel solvers have been developed for solving the wave equation in specific applications. These include a scalable distributed-memory solver for the non-linear wave equation (Alghamdi et al., 2011), a parallel numerical solver for modeling for the elastic wave equation in heterogenous media (Bao et al., 1998), and a discontinuous Galerkin solver for solving electromagnetic and aeroacoustic wave equations (Bernacki et al., 2006).

This concludes the discussion of previous work in the field of sound propagation, directional sources, spatial audio, and parallel wave-solvers. The next chapter discusses the first contribution of this dissertation: an interactive wave-based technique for sound propagation in large environments.

CHAPTER 3: EQUIVALENT SOURCE METHOD FOR SOUND PROPAGATION

3.1 Introduction

Large, open scenes, which arise in many applications ranging from games to training simulations, present a significant challenge for interactive, wave-based sound propagation techniques. State-of-the-art wave-based propagation methods can take hours of computation and gigabytes of memory for performing sound propagation in indoor scenes such as concert halls (Sakamoto et al., 2006; Raghuvanshi et al., 2009b). For large, open scenes spanning hundreds of meters, these techniques would take days or even weeks of compute and terabytes of memory, making them unfeasible for interactive applications in virtual reality, gaming, and architectural acoustics. On the other hand, geometric acoustic techniques can provide interactive performance for such applications. However, geometric techniques are better suited for higher frequencies of sound due to the inherent assumption of rectilinear propagation (ray-like behavior) of sound waves. Accurately modeling wave effects, such as diffraction and interference, remains a significant challenge with these techniques.

In this chapter, we present a novel approach for precomputed, wave-based sound propagation in large, open scenes. It is based on the *equivalent source method*, a technique widely studied for radiation and scattering problems in acoustics and electromagnetics (Doicu et al., 2000) and more recently used for free-space sound radiation in computer graphics (James et al., 2006). we extend this technique for wave-based sound propagation in large, open spaces spanning hundreds of meters. This technique takes an object-centric approach to wave-based sound propagation. The scene is decomposed into disjoint rigid objects. The free-field acoustic behavior of each object is captured by a compact per-object transfer-function relating incoming pressure fields to outgoing pressure fields. Pairwise acoustic interactions between objects are computed analytically to yield compact inter-object transfer functions. Efficient basis function of pressure fields, known as *equivalent sources*, are used to compactly represent the incoming and outgoing pressure fields of an object. The global sound

field accounting for all orders of interaction between objects is computed using the per-object and inter-object transfer functions. The runtime system uses fast summation over the outgoing equivalent sources for all objects to auralize the sound field for a moving listener in real-time. Realistic acoustic effects such as diffraction, low-passed sound behind obstructions, focusing, scattering, high-order reflections, and echoes, are demonstrated on a variety of scenes.

3.2 Background

The equivalent source method is a frequency-domain technique for solving sound radiation or scattering problems by using free-space, radiating functions called the *equivalent sources*. In this section, we give a brief overview of the equivalent source method as used in sound radiation.

Helmholtz Equation Consider the exterior scattering problem (Thompson and Pinsky, 2004): a solid three-dimensional object A immersed in an unbounded air volume radiating sound (see Figure 3.1). By considering only time-harmonic vibrations, with angular frequency ω and a homogeneous medium with constant speed of sound c, acoustic wave propagation can be expressed as a boundary value problem for the Helmholtz equation:

$$\nabla^2 P + \frac{w^2}{c^2} P = 0 \text{ in } A^+, \tag{3.1}$$

where $P = P(\mathbf{x}, \omega)$ is the (complex-valued) pressure field varying with space and frequency, A^+ is the domain exterior to the object, and $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$ is the Laplacian operator.

Since sound radiation in three-dimensional space is considered, the behavior of pressure P at infinity must be specified by the *Sommerfeld radiation condition* (Pierce, 1989) which can also be interpreted as a boundary condition at infinity:

$$\lim_{r \to \infty} r \left[\frac{\partial P}{\partial r} - \hat{j} \frac{w}{c} P \right] = 0, \tag{3.2}$$

where $r = \|\mathbf{x}\| = \sqrt{x^2 + y^2 + z^2}$ is the distance of point $\mathbf{x} = (x, y, z)$ from the origin and $\hat{j} = \sqrt{-1}$. To complete the problem specification, the boundary condition at the domain of the boundary ∂A needs to be specified. The boundary condition can be of three types:

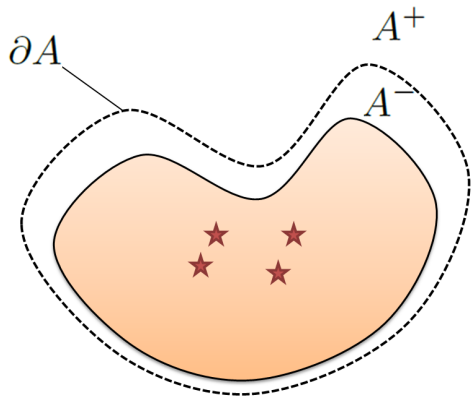


Figure 3.1: A diagram illustrating a radiating object A, its corresponding boundary ∂A, exterior region A^+, interior region A^-, and the set of equivalent sources (denoted by star shapes).

Dirichlet boundary condition: In this case, pressure on the domain of the boundary is specified:

$$P = \Upsilon(\mathbf{x}) \text{ on } \partial A. \tag{3.3}$$

Neumann boundary condition: The gradient of pressure in the normal direction or the normal velocity v is specified.

$$\frac{\partial P}{\partial n} = -\hat{j}\omega\rho v = \Upsilon \text{ on } \partial A, \tag{3.4}$$

where ρ is the fluid velocity.

Mixed boundary condition: This is also called the general impedance boundary condition where both pressure and gradient of pressure are specified:

$$Z\frac{\partial P}{\partial n} + P = \Upsilon \text{ on } \partial A, \tag{3.5}$$

where Z and Υ are functions defined on the domain boundary.

Equivalent Sources Mathematically, the equivalent source is the solution of the Helmholtz equation (3.1) subject to the Sommerfeld radiation condition (3.2). The equivalent source, also called the *Green's function*, can be represented as a pressure field $q(\mathbf{x}, \mathbf{y}_i)$ generated by a point source located at $\mathbf{y}_i$ ($\mathbf{x} \neq \mathbf{y}_i$) and can be expressed as

$$q(\mathbf{x}, \mathbf{y}_i) = \sum_{l=0}^{L-1} \sum_{m=-l}^{l} d_{ilm}\varphi_{ilm}(\mathbf{x}) = \sum_{k=1}^{L^2} d_{ik}\varphi_{ik}(\mathbf{x}), \tag{3.6}$$

where k is a generalized index for (l, m). The fundamental solution $\varphi_{ilm}(\mathbf{x})$ is the pressure field by a *multipole* source located at $\mathbf{y}_i$, defined as

$$\varphi_{ilm}(\mathbf{x}) = \Gamma_{lm}\, h_l^{(2)}(\omega r_i/c)\;\; \psi_{lm}(\theta_i, \phi_i) \tag{3.7}$$

where (r_i, θ_i, ϕ_i) is the vector $(\mathbf{x} - \mathbf{y}_i)$ expressed in spherical coordinates, $h_l^{(2)}(\omega r_i/c)$ are the (complex-valued) spherical Hankel functions of the second kind (Abramowitz and Stegun, 1964), $\psi_{lm}(\theta_i, \phi_i)$ are the (complex-valued) spherical harmonic functions (Hobson, 1955), and Γ_{lm} is the (real-valued) normalizing factor that makes the spherical harmonic functions orthonormal. The term d_{ilm} is the strength of the multipole and L is the order of the multipole. $L = 1$ is for monopole, $L = 2$ includes dipole terms as well, and so on.

Equivalent Source Method for Radiation and Scattering The equivalent source method (ESM), also called the *source simulation technique* or *method of fundamental solutions* (Ochmann, 1995, 1999; Pavic, 2006) relies on the existence of equivalent sources. The basic idea of the equivalent source method is to replace a sound radiating or scattering object by equivalent sources placed in the interior of the object. Consider the outgoing scattered field due to an object and the associated boundary value problem on ∂A. Consider a discrete set of R equivalent sources located at $\{\mathbf{y}_i\}_{i=1}^R$, all contained in the interior region A^-. The total field due to these equivalent sources at any $\mathbf{x} \in A^+$ is

$$P(\mathbf{x}) = \sum_{i=1}^{R} c_i\;\; q(\mathbf{x}, \mathbf{y}_i) = \sum_{i=1}^{R} \sum_{k=1}^{L^2} c_{ik}\;\; \varphi_{ik}(\mathbf{x}), \tag{3.8}$$

where c_i's are the strengths of the equivalent sources and $c_{ik} = c_i d_{ik}$'s are the strength of the multipoles. The main idea of the ESM is that if the equivalent source strengths and their positions are chosen to match the boundary condition on ∂A (for the sake of example, lets say the Dirichlet boundary condition),

$$P(\mathbf{x}) = \sum_{i=1}^{R} \sum_{k=1}^{L^2} c_{ik}\;\; \varphi_{ik}(\mathbf{x}) = \Upsilon(\mathbf{x}); \quad \mathbf{x} \in \partial A, \tag{3.9}$$

then the uniqueness of the boundary value problem dictates that $P(\mathbf{x})$ is the correct solution over the entire exterior region A^+ (Chapter 3 in (Colton and Kress, 2013)). The same holds for Neumann or mixed boundary conditions.

This process can also be used to represent the incident field of an object, the only difference in this case is that the equivalent sources are now placed in the exterior region A^+. Again, by matching the boundary condition (3.9), we get the correct solution $P(\mathbf{x})$ for all $\mathbf{x}$ in the interior region A^-.

In practice, the boundary conditions (3.9) can only be satisfied approximately for a finite value of R, and the degree of approximation can be controlled by changing R. Since the strengths of multipoles of all equivalent sources must be stored and its contribution evaluated at runtime, R is the main parameter for trading accuracy for runtime performance and memory requirements. This flexibility makes the equivalent source method highly suitable for interactive applications. This method can yield large gains in performance and memory-efficiency for scattering and radiation problems, and has been used widely in both acoustic and electromagnetic applications (Doicu et al., 2000).

ESM was also used in the field of computer graphics to efficiently model sound radiation by vibrating objects in free space (James et al., 2006). ESM is an attractive starting point for such precomputation-based approaches, as it allows very flexible performance-to-accuracy tradeoffs. More importantly, the compactness of the solutions reduces runtime memory and compute requirements by orders of magnitude, making them amenable to interactive evaluation.

The equivalent source method is related to the boundary element method (BEM) in many ways. For a detailed discussion of the similarities and differences between ESM and BEM, please refer to the seminal work by (Ochmann, 1995, 1999).

3.3 Equivalent Source Method for Sound Propagation

In this section, we present a novel equivalent source method for wave-based sound propagation in large, open scenes. We extend the previous equivalent source method for sound radiation to the case of sound propagation. The two key differences between sound radiation and sound propagation are as follows: Firstly, in case of sound radiation, an object radiates a field outwards generating only an outgoing field. In sound propagation, an object receives an incoming sound field which is

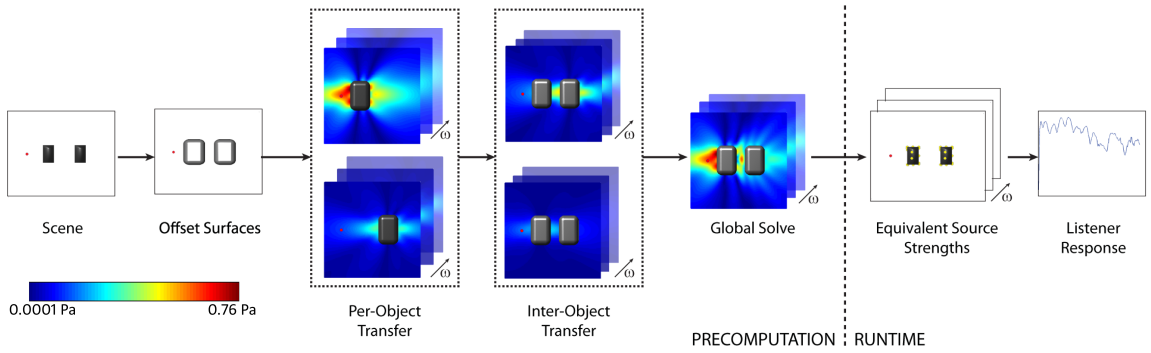


Figure 3.2: Overview of the different stages of the proposed equivalent source method for sound propagation: (a) **Offset surface calculation**: The input scene is classified into well-separated objects and the offset surface for each object is calculated. (b) **Per-object transfer function**: For each object, the *per-object transfer function* is precomputed which maps the incoming field incident on the object to the outgoing scattered field. (c) **Inter-object transfer function**: For each object pair, the *inter-object transfer function* is precomputed which encodes how the outgoing scattered field of one object becomes the incoming field for the other object. (d) **Global solve**: Based on the per-object and inter-object transfer functions and a sound source configuration, acoustic interactions between the objects in the scene is modeled and the global sound field is solved. The strengths of all the equivalent sources are computed and stored for runtime use. (e) **Run-time pressure evaluation**: The pressure is computed for all equivalent sources and added for each frequency to generate the frequency response at the listener. (The magnitudes of pressure fields are visualized using the color scheme shown.)

then reflected, scattered, diffracted, by this object resulting in an outgoing field. Therefore, there are two kinds of fields in sound propagation: an incoming field and an outgoing field. Secondly, sound radiation deals with the acoustic interactions of a single object due to its own geometry. Sound propagation involves dealing with acoustic interactions between multiple objects in the environment.

An overview of the ESM technique for wave-based sound propagation is provided in Figure 3.2. This is a frequency-domain sound propagation technique which generates a complex-valued frequency response at regularly sampled set of frequencies in the range $[0, \nu_{\max}]$, where $\nu_{\max}$ is the maximum simulated frequency. This technique assumes that all the objects in this scene are static. The overall technique can be split into two main stages: preprocessing and runtime. During preprocessing, the scene is split into disjoint, well-separated rigid objects. The acoustic behavior of each object, taken independently, is characterized by its *per-object transfer function* that maps an arbitrary incoming field incident on the object to the resulting scattered field. This transfer function is represented using the equivalent sources into a compact *scattering matrix*. Pairwise acoustic coupling between objects is then modeled by computing *inter-object transfer functions* between all pairs of objects

which maps the outgoing scattered field of one object to the incoming field of another object. These inter-object transfer functions are again represented compactly by using equivalent sources to yield *interaction matrices*. Finally, the acoustic response (frequency response) of the scene to a static source distribution is computed by solving a global linear system, consisting of scattering and interaction matrices, that accounts for all orders of intra- and inter-object wave propagation. This acoustic response is stored in terms of strengths of equivalent sources. At runtime, the acoustic response at the listener position is computed by a fast summation of pressure fields generated by all the outgoing equivalent sources (for all objects) weighted by their individual strengths. This acoustic response is used for real-time sound rendering for a moving listener.

All the individual steps of the ESM technique for sound propagation are discussed as follows.

3.3.1 Offset Surface Calculation

The first step is to split the input scene into *well-separated* objects. In order to decide if two objects are well-separated or not, the notion of an *offset surface* is used. The offset surface is defined by taking a constant offset along the normal direction at each point on the boundary surface of the object. Two objects are considered disjoint if and only if their offset surfaces do not intersect. Otherwise, the objects are combined and treated as a single object (see Figure 3.3). The offset surface of an object is computed using the distance field and the marching cubes algorithm (James et al., 2006). Typical values of voxel resolution of distance field h and offset distance δ are specified in Table 3.1. The offset surface serves as the boundary of the object's acoustic domain ∂A. After splitting the scene into well-separated objects, the scattering properties for each object are computed independently.

3.3.2 Per-object Transfer Function

In order to capture an object's scattering behavior, the notion of *per-object transfer function* f is defined as a function that maps an arbitrary incoming field incident on the object to the corresponding outgoing field after reflection, scattering and diffraction due to the object's own geometry. This function is linear owing to the linearity of the wave equation and depends only on the shape, size, and material properties of the object.

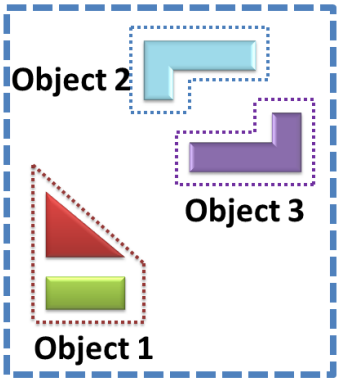


Figure 3.3: Object classification: the triangle and rectangle constitute a single object, as their offset surfaces overlap. On the other hand, L-shaped shapes are classified as separate objects.

The incoming and outgoing fields of an object A are compactly represented using the equivalent source basis. The outgoing field is represented by placing equivalent sources $\{q_1^{out}, q_2^{out}, q_3^{out}, ...\}$ in the interior region A^- of the object. Similarly, the incoming field is represented by placing equivalent sources $\{q_1^{in}, q_2^{in}, q_3^{in}, ...\}$ in the exterior region A^+. The transfer function f maps the basis of the incoming field (i.e. multipoles φ_{ik}^{in}) to the corresponding outgoing field expressed as a linear combination of its basis functions (i.e. multipoles φ_{jh}^{out}):

$$f(\varphi_{ik}^{in}) = \sum_{(j,h)=(1,1)}^{(\mathbb{P},N^2)} \alpha_{jh}^{ik} \ \varphi_{jh}^{out}; \tag{3.10}$$

$$\begin{bmatrix} f(\varphi_{11}^{in}) \\ f(\varphi_{12}^{in}) \\ . \\ . \\ f(\varphi_{\mathbb{Q}M^2}^{in}) \end{bmatrix} = \begin{bmatrix} \alpha_{11}^{11} & \alpha_{12}^{11} & ... & \alpha_{\mathbb{P}N^2}^{11} \\ \alpha_{11}^{12} & \alpha_{12}^{12} & ... & \alpha_{\mathbb{P}N^2}^{12} \\ . & . & ... & . \\ . & . & ... & . \\ \alpha_{11}^{\mathbb{Q}M^2} & \alpha_{12}^{\mathbb{Q}M^2} & ... & \alpha_{\mathbb{P}N^2}^{\mathbb{Q}M^2} \end{bmatrix} \begin{bmatrix} \varphi_{11}^{out} \\ \varphi_{12}^{out} \\ . \\ . \\ \varphi_{\mathbb{P}N^2}^{out} \end{bmatrix} \tag{3.11}$$

$$= \ T_A \ \Phi_A^{out}, \tag{3.12}$$

where $\alpha_{jh}^{ik} \equiv T_A(ik, jh)$ is the (complex-valued) strength of the outgoing multipole φ_{jh}^{out} induced by a unit-amplitude incoming multipole φ_{ik}^{in}. The per-object sound transfer function for object A is encoded in the coefficient matrix T_A, which is referred to as the *scattering matrix*. Next, the strength of the outgoing field multipoles α_{jh}^{ik} are computed as described below. The details on how to choose the number and positions of incoming and outgoing equivalent sources are described in Section 3.3.4.

Computing the scattering matrix For an object A, its scattering matrix is computed as follows: For each multipole φ_{ik}^{in} of the incoming field, a unit-amplitude multipole source is placed at its corresponding position around the object and a numerical wave solver is used to compute the total pressure field at n uniformly-sampled positions $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ on the offset surface ∂A of the object. The incident field is subtracted from the total pressure field to compute the outgoing field at these sampled locations (see Figure 3.4), denoted by $\bar{\mathrm{P}}_{ik} = \{P(\mathbf{x}_1), P(\mathbf{x}_2), ..., P(\mathbf{x}_n)\}$.

The outgoing field multipole expansion is fitted to the sampled outgoing field in the least-squares sense by solving an over-determined linear system ($n > \mathbb{P}N^2$) subject to a pre-specified error threshold σ.

$$\sum_{(j,h)=(1,1)}^{(\mathbb{P},N^2)} \varphi_{jh}^{out}(\mathbf{x}_t)\ \alpha_{jh}^{ik} = p(\mathbf{x}_t), \text{ for } t = 1, ..., n; \tag{3.13}$$

$$\mathrm{V}\boldsymbol{\alpha}^{ik} = \bar{\mathrm{P}}_{ik}. \tag{3.14}$$

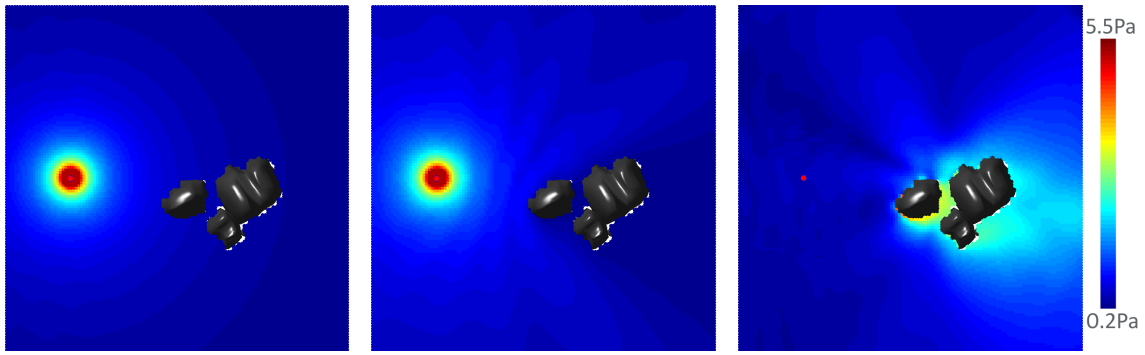
The least-squares solution yields the coefficient vector $\boldsymbol{\alpha}^{ik} = \begin{bmatrix} \alpha_{11}^{ik} & \alpha_{12}^{ik} & ... & \alpha_{PN^2}^{ik} \end{bmatrix}^T$ which corresponds to the ik^{th} row of the scattering matrix T. This process is repeated for all incoming field multipoles to compute the scattering matrix. The solution can be computed efficiently using a single combined linear system

$$\mathrm{V}\ T_A^{tr} = \begin{bmatrix} \bar{\mathrm{P}}_{11} \ldots \bar{\mathrm{P}}_{\mathbb{Q}M^2} \end{bmatrix}, \tag{3.15}$$

where T_A^{tr} is the transpose of T_A. The per-object transfer function is computed for all the objects, independently for each frequency. The error threshold σ decides the number and placement of equivalent sources such that the above linear system gives error less than σ (Section 3.3.4).

3.3.3 Inter-object Transfer Function

Scenes with multiple objects exhibit object-to-object interactions, where the outgoing field from one object serves as the incoming field for the other objects. For example, with two objects A and B,



(a) Incident field (b) Total field (c) Outgoing field ($\times 8$)

Figure 3.4: Magnitude of the pressure field (in Pa) at 170 Hz in a simple scene with a single object (rocks) and a single sound source (red dot). The difference between total and incident fields is the outgoing field (scaled 8 times for visualization). Note the high amplitude of the outgoing field between the rocks representing the large difference in incident and total field that results from diffracted occlusion.

source s and listener l, the possible acoustic interactions that can occur from s to l are: 0^{th} order (direct sound) $s \rightarrow l$; 1^{st} order $s \rightarrow A \rightarrow l, s \rightarrow B \rightarrow l$; 2^{nd} order $s \rightarrow A \rightarrow B \rightarrow l, s \rightarrow B \rightarrow A \rightarrow l$ and so on. These interactions are modeled by formulating the *inter-object transfer function*. For two objects A and B, the inter-object transfer function g_A^B expresses the outgoing field of A in terms of the basis (i.e. multipoles φ_{ik}^{in}) of the incoming field of B. Like the per-object transfer function, the inter-object transfer function is also a linear function. The inter-object transfer function g_A^B maps each basis function of the outgoing field of A (i.e. multipoles φ_{jh}^{out}) to the corresponding incoming field of B expressed as a linear combination of its basis functions (i.e. multipoles φ_{ik}^{in}):

$$g_A^B(\varphi_{jh}^{out}) = \sum_{(i,k)=(1,1)}^{(\mathbb{Q},M^2)} \beta_{ik}^{jh} \varphi_{ik}^{in}; \tag{3.16}$$

$$\begin{bmatrix} g_A^B(\varphi_{11}^{out}) \\ g_A^B(\varphi_{12}^{out}) \\ . \\ . \\ g_A^B(\varphi_{\mathbb{P}N^2}^{out}) \end{bmatrix} = \begin{bmatrix} \beta_{11}^{11} & \beta_{12}^{11} & \dots & \beta_{\mathbb{Q}M^2}^{11} \\ \beta_{11}^{12} & \beta_{12}^{12} & \dots & \beta_{\mathbb{Q}M^2}^{12} \\ . & . & \dots & . \\ . & . & \dots & . \\ \beta_{11}^{\mathbb{P}N^2} & \beta_{12}^{\mathbb{P}N^2} & \dots & \beta_{\mathbb{Q}M^2}^{\mathbb{P}N^2} \end{bmatrix} \begin{bmatrix} \varphi_{11}^{in} \\ \varphi_{12}^{in} \\ . \\ . \\ \varphi_{\mathbb{Q}M^2}^{in} \end{bmatrix}$$

$$= G_A^B \ \Phi_B^{in}, \tag{3.17}$$

where $\beta_{ik}^{jh} \equiv G_A^B(jh, ik)$ is the (complex-valued) strength of the incoming multipole φ_{ik}^{in} of B induced by a unit-amplitude outgoing multipole φ_{jh}^{out} of A. The inter-object transfer function from A to B is thus encoded as G_A^B, which we call the *interaction matrix*. Generally, the interaction matrix is *not* symmetric, i.e., $G_A^B \neq G_B^A$. Since the outgoing field of an object is not defined in its interior region, G_A^A and G_B^B are zero matrices. The method to compute the strengths β_{ik}^{jh} of the incoming field multipoles are described below.

Computing the Interaction Matrix The interaction matrix G_A^B can be computed using a least-squares formulation similar to the one used for computing scattering matrices. However, the pressure values at the offset surface samples of B, $\bar{\mathbf{P}}_{jh} = \{P(\mathbf{x}_1), P(\mathbf{x}_2), ..., P(\mathbf{x}_n)\}$ are simpler to compute in this case. In a homogenous medium (constant speed of sound), the outgoing field due to a multipole is the same as the free-space field, for which analytical expressions exist (Equation 3.7). Therefore, we simply evaluate the analytical expressions of the outgoing field multipoles φ_{jh}^{out} of A at the sample points on the offset surface of B. The resulting linear system is solved subject to a separate error threshold η :

$$\sum_{(i,k)=(1,1)}^{(\mathbb{Q},M^2)} \varphi_{ik}^{in}(\mathbf{x}_t)\ \beta_{ik}^{jh} = P(\mathbf{x}_t), \text{ for } t = 1, ..., n. \tag{3.18}$$

Again, this process is repeated for each outgoing multipole of B or solved efficiently as a single combined linear system:

$$\mathrm{U}\, G_A^{B^{tr}} = \left[\bar{\mathbf{P}}_{11} \ldots \bar{\mathbf{P}}_{\mathbb{P}N^2}\right]. \tag{3.19}$$

The inter-object transfer functions are computed for all object pairs, independently for each frequency.

3.3.4 Equivalent Source Positions

Choosing offset surface samples Solving equations (3.15) and (3.19) at frequency ν involves computing the pressure at sampled locations $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ on the offset surface of each object. The number of sampled locations n depends on the spatial variation of the pressure field, which in turn depends directly on its frequency ν or inversely on its wavelength λ (since $\nu = c/\lambda$). As per the Nyquist Theorem, representing a signal of frequency ν with a finite number of samples requires a sampling rate of 2ν. The spatially-varying pressure field defined on the 2D offset surface must be

sampled at a rate of 2ν in both dimensions. The distance between samples become $2\nu/c = 2/\lambda$. Therefore, we place $n \propto (2/\lambda)^2 \times$ *surface area* samples uniformly on the offset surface.

Choosing incoming equivalent sources Since the nature of the incoming field is not known *a priori*, it is difficult to optimize the number and position of incoming equivalent sources. This problem is resolved by generating another offset surface at distance $\Delta > \delta$ from the object, where δ is the original offset surface's distance (see Table 3.1 for value of Δ). Next, incoming equivalent sources are placed on this new surface. The number of incoming equivalent sources $\mathbb{Q}$ depends on the spatial variation of the incoming pressure field. As before, $\mathbb{Q} \propto (2/\lambda)^2 \times$ *surface area* number of equivalent sources are uniformly placed. This allows us to represent any incoming field on the inner offset surface to good accuracy.

Choosing outgoing equivalent sources The number of outgoing equivalent sources $\mathbb{P}$ and their positions are decided based on a multi-level source placement algorithm similar to (James et al., 2006). The previous algorithm was designed to satisfy a single radiating field $\bar{\mathrm{P}}$ of an object at each frequency. It placed equivalent sources in a greedy manner, where at each step a set of candidate positions χ are ranked based on their ability to reduce the pressure residual vector $\bar{\mathrm{r}} = \bar{\mathrm{P}}/\|\bar{\mathrm{P}}\|_2$ on the offset surface. The best candidate position $\mathbf{x}^*$ is chosen via the largest projection, i.e., $\mathbf{x}^* = \arg\max_{x\in\chi} u$, where projection $u = \|(U_{\mathbf{x}})^H \bar{\mathrm{r}}\|_2$. The subspace unitary matrix corresponding to the subspace spanning all the previously selected positions is updated. The residual vector is updated by removing its component in that subspace. The process is repeated until the value of the residual $||\bar{\mathrm{r}}||_2$ falls below the error tolerance. The set of best candidate positions selected in the process is the set of outgoing equivalent sources and its size gives us the value of $\mathbb{P}$. Multi-level scheme can be employed to accelerate the multipole selection process. For more details, please refer to Section 4 in (James et al., 2006).

The ESM source placement algorithm for sound propagation is designed to satisfy multiple outgoing fields simultaneously. In this case, at each frequency, there are as many outgoing fields $[\bar{\mathrm{P}}_1 \ldots \bar{\mathrm{P}}_m]$ as the number of incoming multipoles (i.e. $m = \mathbb{Q}M^2$). This results in a matrix of pressure residuals $\mathbf{r} = [\bar{\mathrm{r}}_1 \ldots \bar{\mathrm{r}}_m]$ and a corresponding vector of projections $\bar{\mathrm{u}} = [u_1 \ldots u_m]^T$ where $u_i = \|(U_{\mathbf{x}})^H \bar{\mathrm{r}}_i\|_2$. The best candidate is chosen as the one that minimizes all the pressure residuals simultaneously via a modified largest projection $\mathbf{x}^* = \arg\max_{x\in\chi} \|\bar{\mathrm{u}}\|_2$. The subspace

Algorithm 1 EQUIVALENT_SOURCE_PLACEMENT

Input: Outgoing radiating fields $[\bar{\mathrm{P}}_1 \ldots \bar{\mathrm{P}}_m]$, error tolerance σ

1: $\mathrm{r} = [\bar{\mathrm{r}}_1 \ldots \bar{\mathrm{r}}_m] \leftarrow [\bar{\mathrm{P}}_1 \ldots \bar{\mathrm{P}}_m]$
2: $\forall i,\ \bar{\mathrm{r}}_i = \bar{\mathrm{r}}_i / \|\bar{\mathrm{r}}_i\|_2$ init residual
3: $\mathrm{Q} \leftarrow 0$ init subspace
4: $\xi \leftarrow \emptyset$ init selected points
5: **while** $\|\mathrm{r}\|_{(2,2)} > \sigma$ **do**
6: $\quad \mathrm{x}^* \leftarrow$ SELECT_MULTIPOLE_POSITION(r)
7: $\quad$ EXPAND_SUBSPACE_AND_UPDATE_RESIDUAL(Q, r, x^*)
8: $\quad \xi \leftarrow \xi \bigcup \mathrm{x}^*$

Output: List of equivalent source positions ξ

SELECT_MULTIPOLE_POSITION(r)

Input: Residual matrix $\mathrm{r} = [\bar{\mathrm{r}}_1 \ldots \bar{\mathrm{r}}_m]$

1: $\chi \leftarrow$ draw T random candidate source positions
2: $\bar{\mathrm{u}} \leftarrow [u_1 \ldots u_m]^T$ where $u_i = \|(U_{\mathbf{x}})^H\, \bar{\mathrm{r}}_i\|_2$
3: $\mathbf{x}^* \leftarrow \arg\max_{x \in \chi} \|\bar{\mathrm{u}}\|_2$

Output: Best candidate position x^*

EXPAND_SUBSPACE_AND_UPDATE_RESIDUAL(Q, r, x)

Input: Subspace unitary matrix Q, residual matrix r, candidate position x

1: $\mathrm{Q} \leftarrow [\mathrm{Q} \mid \mathrm{Q_x}] \leftarrow$ MOD_GRAM_SCHMIDT($[\mathrm{Q} \mid \mathrm{V_x}]$)
2: $\forall i,\ [\mathrm{Q_x} \mid \bar{\mathrm{r}}_i^*] \leftarrow$ UPDATE_RESIDUAL($[\mathrm{Q_x} \mid \bar{\mathrm{r}}_i]$)
3: $\forall i,\ \bar{\mathrm{r}}_i \leftarrow \bar{\mathrm{r}}_i^*$

unitary matrix is updated as before and for each residual vector its component in the chosen subspace is removed. The value of the modified residual $\|\mathbf{r}\|_{(2,2)}$ is defined as $\|\mathbf{r}\|_{(2,2)} = \|\,[d_1 \ldots d_m]^T\,\|_2$ where $d_i = \|\bar{\mathrm{r}}_i\|_2$. This process is repeated until the relative value of the modified residual falls below the user-specified error tolerance (σ). This process is described in the Algorithm 1. The term $\mathrm{V_x}$ refers to the multipole matrix corresponding to a multipole placed at position x, MOD_GRAM_SCHMIDT refers to the modified Gram-Schmidt orthogonalization, and UPDATE_RESIDUAL corresponds to operation $\bar{\mathrm{r}}_i^* \leftarrow \bar{\mathrm{r}}_i - \mathrm{Q_x}(\mathrm{Q_x})^* \bar{\mathrm{r}}_i$ performed using modified Gram-Schmidt orthogonalization. Multilevel placement can be used to accelerate the selection of multipoles (James et al., 2006).

Similar to the number of incoming equivalent sources $\mathbb{Q}$, the number of outgoing equivalent sources $\mathbb{P}$ also increases with frequency. But the actual value of $\mathbb{P}$ strongly depends on the shape of the object and the complexity of the outgoing field that the object generates. As many outgoing equivalent sources are placed as required to satisfy the error threshold. As the frequency increases, more outgoing equivalent sources are needed but the accuracy of the technique is maintained. The

candidate position set χ is chosen randomly on the surface of the object in the same manner as the previous algorithm (James et al., 2006). However, a minimum distance between any two equivalent sources is enforced to improve the condition number of the system; extremely close equivalent sources dominate the eigenvalues of the resulting system, adversely affecting its condition number. A minimum distance of half the wavelength is chosen at any given frequency.

3.3.5 Global Solve

Once the scattering and interaction matrices are computed and the sound source position fixed, one can solve for the global sound field to compute the strengths of outgoing equivalent source for all the objects in the scene. An intuitive explanation is provided here for a simple two-object scene and the detailed derivation can be found in Appendix A.1. For a scene composed of multiple objects, the same equation can be derived, as described in detail in Appendix A.2.

Let the scattering matrix for the scene be $\mathbf{T}$, the interaction matrix for the scene be $\mathbf{G}$ and the sound source generates a sound field $\mathbf{S}$. Assume that the outgoing field of the scene is $\mathbf{C}$. This field when propagated through the scene, transferred via all possible object pairs using interaction matrix $\mathbf{G}$, generates an incoming field $\mathbf{GC}$. This field, in addition to the source field $\mathbf{S}$, generates the total incoming field $(\mathbf{GC} + \mathbf{S})$ on the objects. This incoming field is then scattered by the objects, via scattering matrix $\mathbf{T}$, to produce an outgoing field $\mathbf{T}(\mathbf{GC} + \mathbf{S})$. Under steady state, this outgoing field must equal the outgoing field we started with i.e. $\mathbf{C}$. Mathematically, this can be written as

$$\mathbf{C} = \mathbf{T}(\mathbf{GC} + \mathbf{S}) \tag{3.20}$$

This yields a linear system for the outgoing source strengths for all objects:

$$(\mathbf{I} - \mathbf{TG})\mathbf{C} = \mathbf{TS}. \tag{3.21}$$

This linear system is solved for $\mathbf{C}$ for each frequency. This step has to be repeated for every sound source generating a distinct source field $\mathbf{S}$. In the absence of a source, the solution is identically zero.

3.3.6 Runtime Computation

At the end of the preprocessing stage, we obtain the outgoing equivalent source strengths for all objects, at each frequency, corresponding to each sound source. During runtime, we use these strengths to compute the pressure field at any listener position $\mathbf{x}$ corresponding to each source as

$$P(\mathbf{x}) = \sum_{j=1}^{\kappa} C_{A_j}^{tr} \Phi_{A_j}^{out}(\mathbf{x}) + s(\mathbf{x}), \tag{3.22}$$

where κ is the number of objects in the scene, $C_{A_j}^{tr}$ and $\Phi_{A_j}^{out}$ are the strengths and multipoles of the outgoing equivalent sources for object A_j respectively, and $s(\mathbf{x})$ is the field generated by the sound source. This computation is performed for all the sampled frequencies and repeated for each source to compute a band-limited frequency response per source. Evaluating equation (3.22) for a new value of $\mathbf{x}$ is very efficient, allowing a moving listener to be handled naturally in realtime. Unlike previous grid-based approaches, such as FDTD or (Raghuvanshi et al., 2010), the analytical expressions for multipoles of equivalent sources allows the pressure to be evaluated at any position $\mathbf{x}$ in space and not necessarily at sampled grid positions. Therefore, no spatial interpolation is required at runtime with the ESM sound propagation technique. This equivalent source method is independent of the spatial discretization, resulting in a much smoother auralization for a moving listener.

The discussion till now has focused on how the equivalent source technique can be used to compute acoustic responses in a scene with multiple static sources and a moving listener. However, the technique can also handle the case of multiple moving sources and a static listener. This can be done by employing the principle of acoustic reciprocity which states that the sense of source and listener can be reversed without changing the acoustic response of the scene (Pierce, 1989, p. 195-199). To handle a scene with multiple moving sources and a static listener, the sense of sources and listener in the scene are reversed resulting in moving sources and a static listener. Then the ESM propagation technique computes the acoustic response for this reversed scene (with moving listeners and a static source) in the manner described earlier. The reciprocity principle dictates the the acoustic responses of the reversed scene and the original scene are the same. This gives us the acoustic response for the original scene with multiple moving sources and a static listener.

3.4 Implementation

In this section, the implementation details of the equivalent source technique for sound propagation is described. Typical parameter values used in the experiments are specified in Table 3.1.

Parameter	Value	Description
c	340 m/s	speed of sound
$\nu_{\max}$	1 kHz	highest frequency simulated
h	$c/2\nu_{\max} = 0.17$ m	voxel resolution of distance field
δ	$5h = 0.85$ m	inner offset distance
Δ	$8h = 1.36$ m	outer offset distance
σ	15%	error threshold for scattering matrix
η	1%	error threshold for interaction matrix
M, N	1, 2	order of incoming, outgoing multipoles resp.

Table 3.1: Parameters used in the implementation of the proposed equivalent source technique.

Hardware and Timing Results The offset surface generation code is written in C++. When computing per-object transfer functions, outgoing fields are computed on the offset surface using an efficient GPU-based implementation of the adaptive rectangular decomposition (ARD) wave-solver (Raghuvanshi et al., 2009b; Mehra et al., 2012). The solver treats the objects as rigid (with no transmission) and handles the material properties using perfectly matched layer interfaces. The remaining parts of the preprocessing stage i.e. solving the linear system for per-object transfer functions, for inter-object transfer functions, and for computing equivalent source strengths, are all implemented in MATLAB. The runtime code is implemented in C++ and has also been integrated with Valve's Source™ engine.

The timing results for the offset surface generation code, the ARD solver, and runtime code are measured on a single core of a 4-core 2.80 GHz Xeon X5560 desktop machine with 4 GB of RAM and NVIDIA GeForce GTX 480 GPU with 1.5 GB memory. The offset surface generation code takes $<$ 1sec for each object. The timing results for the MATLAB-based precomputation are measured on a 64-node CPU cluster (Xeon X5560 processor nodes, 8 cores, 2.80 GHz, 48 GB). Detailed statistics are provided in Table 3.2 and Table 3.3. Precomputation for each frequency is performed in parallel over all the nodes (and individual cores) of the CPU cluster. The precomputation step is computationally heavy and takes a few hours to run on a CPU cluster. However, this step is trivially parallel and could be performed easily on cheap and widely available cloud computing

Scene	#objs.	# freq.	# srcs	wave sim. (total, per obj.)	per-object (per freq)	inter-object (per freq)	source field (per freq, per src)	global solve (per freq, per src)	wall clk time
Concave	1	250	1	80 min	51 min	NA	1 min	0.1 min	132 min
Wall	1	250	1	50 min	101 min	NA	3 min	0.1 min	154 min
Rock	1	250	1	80 min	87 min	NA	1 min	0.1 min	168 min
Parallel	2*	250	1	50 min	101 min	13 min	6 min	1 min	171 min
Desert	4*+2*	500	3	180 min	196 min	98 min	9 min	26 min	509 min
Reservoir	4*+1	500	2	146 min	224 min	63 min	7 min	15 min	455 min
Christmas	2*+2*+1	500	2	297 min	301 min	71 min	7 min	18 min	694 min

Table 3.2: **Precomputation performance:** Abbreviations are as follows : "#objs." denotes the number of objects in the scene, "#freq." is the number of frequency samples in the range [0-1 kHz] and "#srcs" is the number of sound sources. For the precomputation stage, the term "wave sim." is the total simulation time of the numerical wave solver for all frequencies, "per-object" denote the compute time for the per-object transfer function for all unique objects and "inter-object" denote the compute time for inter-object transfer functions for all object pairs, "source-field" is time to express each sound source in terms of incoming multipoles for all objects, and "global-solve" is time to compute equivalent source strengths for all objects. The "wave sim." step is parallelized over all unique objects whereas the remaining precomputation steps are parallelized over all frequencies. The term "wall-clk time" is the total wall-clock time computed by uniformly distributing all the parallel processes over all the cores of the 64-node cluster with 8 cores per node (512 cores in total). For column "#objs.", notation $a^* + b^*$ denotes that first object has been instanced a times and second object instanced b times, but their per-object transfer functions are computed only once for each unique object.

resources, such as Amazon EC2. Given more nodes on the cluster, the per-object, inter-object, and source-field computations can be further parallelized over all unique objects, object-pairs, and objects, respectively. Moreover, the current implementation for the preprocessing stage is in MATLAB and a 10x improvement can be expected with an optimized C++ implementation.

For the purpose of wave simulations, the sources are treated as band-limited with frequency range bounded by maximum frequency $\nu_{\max}$ (see Table 3.1). This is done due to the computational overhead of the precomputation stage. The pressure is computed at regularly sampled set of frequencies in the range $[0, \nu_{\max}]$ with a step size of $\Delta\nu$ and extrapolated to the full audible frequency range (up to 20 kHz). The value of parameter $\Delta\nu$ is 4.08 Hz for concave, wall, rock, and parallel walls scenes and 2.04 Hz for desert, reservoir, and Christmas scenes.

Handling Ground Reflections To simplify the handling of ground reflections, the ground is assumed to be an infinite plane. We split this discussion into two stages: a) handling only the last level of ground reflection, and b) handling all orders of ground reflection.

Last ground reflection

This method deals with only those cases where the ground reflection happens a step before the sound reaches the listener i.e. last level of ground reflection. To incorporate the last level of ground reflection, the sound sources and the equivalent sources are reflected about the ground plane and their source strengths are multiplied by the (complex-valued) reflection coefficient of the ground. This is similar to the image source method proposed in (Allen and Berkley, 1979). Since sound waves traveling in air maintain their phase upon reflection from a hard surface, the strengths of sound sources or equivalent sources need not be inverted. More accurate physical models of ground reflection coefficient based on Darcy's law can also be used (Taraldsen and Jonasson, 2011). This reflection step can be performed during the runtime step resulting in no changes to the preprocessing stage. To give an example, lets start with a scene with source S, objects A and B, and listener L. Let GP be an infinite ground plane for this scene. The ground reflections can be handled at runtime by reflecting all the sound sources s_i and equivalent sources q_i about the ground plane and multiplying their source strengths by the (complex) ground reflection coefficient. The original sources along with the reflected ones give the outgoing pressure field at the listener L. As discussed, this step will only handle the last level of ground reflections:

- $S \rightarrow GP \rightarrow L$ (ground reflection of direct sound),
- $S \rightarrow (A, B) \rightarrow GP \rightarrow L$ (last level of ground reflection),

where (A, B) denotes all levels of interactions between objects A and B.

All orders of ground reflections

The above method deals with only those cases where the ground reflection happens a step before the sound reaches the listener i.e. last level of ground reflection. But this method does not model interactions of the type $S \rightarrow A \rightarrow GP \rightarrow B \rightarrow L$ or $S \rightarrow A \rightarrow GP \rightarrow A \rightarrow L$, where the ground reflection happens between objects or between the same object. Also, the above technique does not handle interactions of type $S \rightarrow GP \rightarrow A \rightarrow GP \rightarrow B \rightarrow L$ and $S \rightarrow GP \rightarrow A \rightarrow GP \rightarrow A \rightarrow L$. The goal is to handle all interactions of the type $S \rightarrow (A, B, GP) \rightarrow L$ where (A, B, GP) denotes all levels of interactions between objects A, B and ground plane GP. In order to handle all levels of ground reflections, we propose the following approach:

1. Firstly, we need to model the ground reflections that happen when the sound travels from one object to another i.e. interactions of type $A \rightarrow GP \rightarrow B$. This can be modeled by modifying the corresponding interaction matrix G_A^B between the two objects. Earlier, for computing the interaction matrix G_A^B, for each outgoing equivalent source q_i of A, we computed the incoming field on the offset surface of B by q_i and used that to compute the strengths of the corresponding incoming equivalent sources of B. This modeled interactions of type $A \rightarrow B$. Now, while computing the interaction matrix G_A^B, for each outgoing equivalent source of A, the total incoming field on the offset surface of B will be the field produced by equivalent source q_i and the corresponding equivalent source reflected about the ground plane $\hat{q}_i$. The strength of equivalent source $\hat{q}_i$ is strength of q_i multiplied by the ground reflection coefficient. Now, this modified incoming field is used to compute the strength of incoming equivalent sources of B. This will model interactions of the both types $A \rightarrow B$ as well as $A \rightarrow GP \rightarrow B$.

2. Secondly, we need to model the ground reflections that happen when the sound travels out of one object, gets reflected by the ground and reaches back to the same object. These interactions are of the type $A \rightarrow GP \rightarrow A$. This can be modeled by modifying the interaction matrix G_A^A for object A. In the previous method, since the outgoing field of an object A is not defined in its interior region, the interaction matrix G_A^A was a zero matrix. But now the outgoing field of A can get reflected from the ground plane and enter the interior region of A. Thus, G_A^A is no longer a zero matrix. Thus, to compute this interaction matrix, for each outgoing equivalent source q_i of A, we find the incoming field on the offset surface of A by the corresponding equivalent source reflected about the ground plane $\hat{q}_i$ and use it to compute the strengths of the incoming equivalent sources of A. The strength of equivalent source $\hat{q}_i$ is strength of q_i multiplied by the ground reflection coefficient. This will model all the per-object interactions as well as interaction of the type $A \rightarrow GP \rightarrow A$.

3. The sound field generated by the source in the "Global Solve" step (term $\mathbf{S}$ in equation 3.21) needs to be modified to incorporate ground reflections. The modified source field is now computed using both source S and its reflected counterpart $\hat{S}$. Now, after performing the global solve step, the interactions of the type $S \rightarrow (A, B, GP) \rightarrow A \rightarrow L$ and $S \rightarrow (A, B, GP) \rightarrow B \rightarrow L$ are modeled.

Scene	#objs.	# freq.	# srcs	# eq. srcs (total, per src)	eval. (total, per src)	storage (total, fixed + per src)
Concave	1	250	1	0.1 M	3 ms	(1 + 4) MB
Wall	1	250	1	0.1 M	4 ms	(2 + 5) MB
Rock	1	250	1	0.4 M	10 ms	(4 + 11) MB
Parallel	2*	250	1	0.2 M	8 ms	(4 + 10) MB
Desert	4*+2*	500	3	1.1 M	26 ms	(12 + 33) MB
Reservoir	4*+1	500	2	1.3 M	33 ms	(15 + 41) MB
Christmas	2*+2*+1	500	2	1.5 M	38 ms	(18 + 47) MB

Table 3.3: **Runtime performance:** Abbreviations are as follows : "#objs." denotes the number of objects in the scene, "#freq." is the number of frequency samples in the range [0-1 kHz] and "#srcs" is the number of sound sources. At runtime, the total number of equivalent sources "# eq. srcs" (in million M), performance "eval." and storage requirement "storage" (fixed and per source cost) for all objects for all frequencies are also specified. For column "#objs.", notation $a^* + b^*$ denotes that first object has been instanced a times and second object instanced b times, but their per-object transfer functions are computed only once for each unique object.

4. Lastly, we need to handle the last level of ground reflection. This is done in the same manner as described in the section before: we take all the sound sources and equivalent sources, reflect them about the ground plane and multiply their source strengths by the reflection coefficient.

Combining the interactions modeled in Step 1, 2, 3 and 4, it can be deduced that all interactions of the type $S \rightarrow (A, B, GP) \rightarrow L$ have been modeled.

The assumption of infinite flat plane works very well for cases where the size of the ground perturbations is smaller than the minimum wavelength simulated (34 cms for 1 kHz). For cases where the ground contains terrain features that are much larger, like hillocks, these should be handled as separate objects in the ESM framework. Due to the increased number of objects in this case, the precomputation time and runtime memory would increase but the accuracy of the technique would be maintained.

Spectral Extrapolation The band-limited nature of the frequency response necessitates a plausible extrapolation to higher frequencies at runtime. Prior work on interactive wave-based methods has shown that spectral extrapolation techniques can be used to produce plausible results for higher frequencies (Raghuvanshi et al., 2010). However, using this method with the equivalent source technique would incur an extra inverse FFT cost at every audio frame for time-domain processing.

Therefore, a simple and fast extrapolation technique based on the edge-diffraction spectra (Svensson et al., 1999) is used.

Based on observations, it is known that the typical edge diffraction spectra are roughly linear on a log-log scale. Hence, we first estimate a trend-line by a least-squares fit to the maxima of the log magnitude spectrum till the maximum simulation frequency. We then adjust for the trend to create a flat response by multiplying the spectrum with the inverse of the trend on a log frequency scale. This adjusted response is replicated to higher frequencies and then multiplied by the trend again for the entire frequency range, yielding the final wide-band spectrum. If the trend-line has positive slope, indicating a high-pass response, we flatten the trend-line for frequencies beyond $\nu_{\max}$. Note that this extrapolation technique does not change the spectrum up to the maximum simulation frequency.

Real-Time Auralization The sound sources used in our implementation play a pre-recorded audio clip. Audio is rendered using FMOD, an audio middleware solution, and processed in frames of 1024 audio samples at a sampling rate of 44.1 kHz. In-game ("dry") audio clips are pre-processed by computing a windowed short-time Fourier transform (STFT) on each frame (Blackman window). The STFTs are computed on audio frames after zero-padding by the maximum impulse response length to prevent aliasing artifacts. Real-time auralization is performed using overlap-add STFT convolutions using the Intel MKL library. In each rendered frame, the dry audio frame for each source is multiplied in the frequency-domain with the corresponding frequency response. The results are then mixed, and an inverse FFT performed on the mixed audio. Finally, *overlap* from previous frames is added in, and overlap from the current frame is cached in a ring buffer. Frequency responses are updated asynchronously from the actual convolution processing. Spatialization is achieved by using a simplified spherical head model with two listeners, one for each ear. Better spatialization that uses geometry information of an individual listener's ears, head, and shoulders can be modeled using *head-related transfer functions* (HRTFs), and can be easily integrated in this approach, but at the cost of added computational expense.

3.5 Results and Analysis

In this section, the results of the equivalent source technique for sound propagation is demonstrated on different scenarios.

Scenarios The auralizations corresponding to the scenes discussed below are provided in the supplementary video.

Single Object We have considered various objects having different scattering characteristics: rocks, a wall, and a concave reflector. The rocks scatter sound in all directions (see Figure 3.4). We show the magnitude of the outgoing field for the wall as generated by the proposed ESM technique and by the BEM technique in Figure 6.3. As shown in the figure, the wall strongly scatters sound in the direction perpendicular to itself. As a more challenging scene, the magnitude of the scattered sound field for a concave reflector is also included. The reflector generates significant interference effects, resulting in caustic formation in the focal region. This is clearly captured by the proposed technique showing that the equivalent source technique is able to approximate the phase of the scattered wave field to reasonable accuracy. The relative error between the total pressure fields generated by the proposed technique and by the BEM technique is less than 2% for the wall and 5% for the concave reflector.

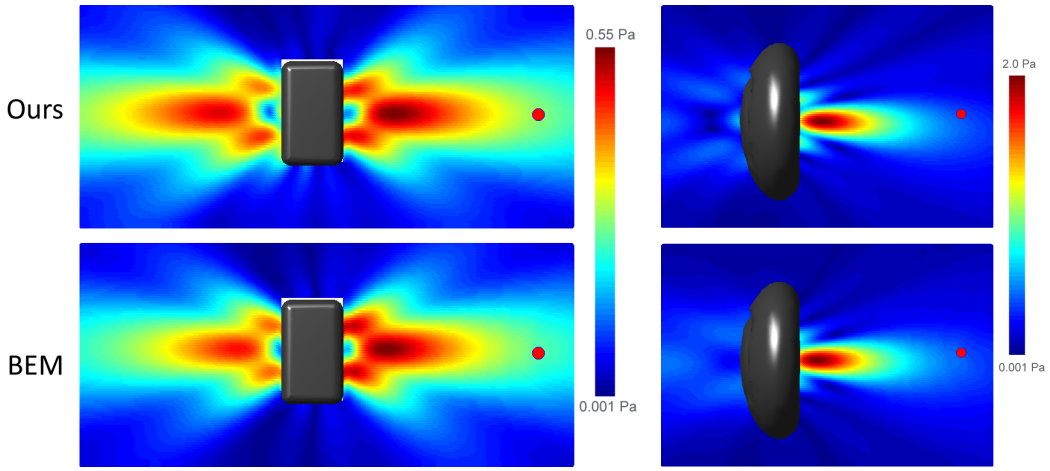


Figure 3.5: Scattering behavior of a wall ($2.3m$ x $4.5m$ x $3.7m$) and a concave reflector (diameter $8m$, thickness $1.2m$) at $160Hz$ is shown using the proposed equivalent source technique (top row) and BEM (bottom row). The sound source is shown with a red dot.

Parallel Buildings This scene consists of two buildings situated parallel to one another. Two walkthroughs of this scene are shown, one with a flying helicopter, and the other with a person speaking. As the helicopter moves behind a building, diffraction leads to a distinct low-pass occlusion effect. In the second walkthrough, the two walls trap sound producing high-order reflections such that

the loudness of someone talking between the buildings is markedly higher than someone standing to the side.

Desert This is a large scene with three sound sources spread throughout the scene: a jeep, a bird, and a radio. As the listener walks through the scene, the sound received from the various sources changes significantly depending on whether or not the listener is in the line-of-sight of the source(s). We also demonstrate the effect of second-order diffracted occlusion of the jeep sound around the two buildings.

Christmas Town This scene demonstrates sound propagation in a village with many houses, a church, a bell tower and large buildings. It shows reflection from buildings, diffraction around houses, sound propagation over large distances, and reflections between two parallel buildings.

Reservoir In this scene, we show that our technique can be integrated with an existing game engine to generate realistic wave acoustic effects in a large, outdoor game map. Our method is the first wave-based sound propagation technique that can accurately model wave phenomena, such as diffraction behind the rocks and scattering around buildings, over large distances on such a large scene in real time.

Error Analysis Figure 3.7 shows the convergence of the proposed ESM technique as the error threshold decreases. Since the number of outgoing equivalent sources is inversely proportional to the error threshold, it also shows convergence of the technique with increasing number of outgoing equivalent sources. We also plot the variation of the number of outgoing equivalent sources required to achieve given error thresholds alongside the simulation frequency (see Figure 3.8).

Computational Complexity Consider a scene with κ objects. To perform analysis for frequency ν, let the number of offset surface samples, incoming equivalent sources and outgoing equivalent sources at this frequency be n, $\mathbb{Q}$ and $\mathbb{P}$, respectively. Also assume that all objects have equal volume u.

Scattering Matrix: For each of the $\mathbb{Q}M^2$ incoming multipoles of an object, wave simulations are performed and a dense linear system of size $n \times \mathbb{P}N^2$ is solved to find the object's scattering matrix. The cost for each simulation is $u \log u$, and the cost of solving the linear system[1] is $n\mathbb{P}^2N^4$. Hence, the total cost is $O\left(\kappa\mathbb{Q}M^2\left(n\mathbb{P}^2N^4 + u\log u\right)\right)$.

[1] To solve a dense linear system of size m x n $(m > n)$, the cost is mn^2

Figure 3.6: **Benchmarks:**The proposed equivalent source technique accurately models realistic acoustic effects, such as diffraction, scattering, focusing, and echoes, in large, open scenes. The runtime memory usage of this technique is orders of magnitude smaller compared to state-of-the-art wave solvers, enabling real-time, wave-based sound propagation in scenes spanning hundreds of meters: a) reservoir scene (Half-Life 2), b) Christmas scene, c) desert scene and (d) parallel walls scene.

Interaction Matrix: For every pair of objects, $\mathbb{P}N^2$ linear systems of size $n \times \mathbb{Q}M^2$ need to be solved to find the interaction matrix. In total, we have κ^2 object pairs. The cost of evaluating analytical expressions for multipole pressure is $O(1)$ each, and is dominated by the cost of solving the linear systems. Hence the total cost is $O\left(\kappa^2\mathbb{P}N^2n\mathbb{Q}^2M^4\right)$. The size of these linear systems vary linearly with n, which in turn varies quadratically with frequency (see Section 3.3.4). Thus, ensuring a few hours precomputation time on a small computational cluster (see Table 4.1) limits the implementation of this technique to a maximum simulation frequency of 1-2 kHz on typical outdoor scenes.

Computing Strengths: The incoming field produced by each sound source is represented in terms of the incoming equivalent sources of the objects. This requires solving κ linear system of size $n \times \mathbb{Q}M^2$ resulting in cost $O\left(\kappa n\mathbb{Q}^2M^4\right)$. The size of the final linear system for solving the outgoing equivalent source strengths for all objects is $\kappa\mathbb{P}N^2 \times \kappa\mathbb{P}N^2$. Solving it takes $O\left(\kappa^3\mathbb{P}^3N^6\right)$ time.

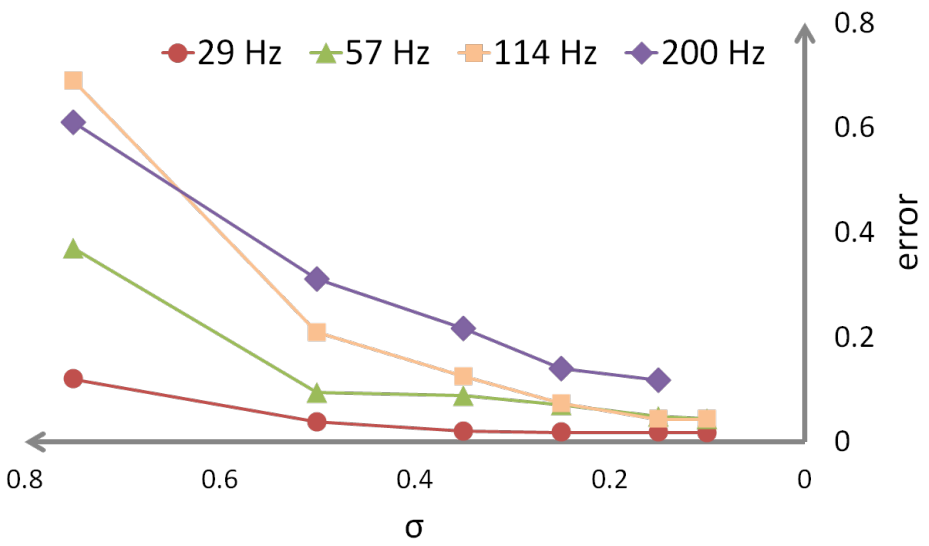


Figure 3.7: **Convergence**: We show the variation of error $\|P_{ref} - P_{ESM}\|^2/\|P_{ref}\|^2$ between the reference wave solver and our technique for varying values of scattering matrix error threshold σ for the two parallel walls scene (fixed $\eta = 1\%$). P_{ref} and P_{ESM} are vectors consisting of (complex) pressure values at all the receiver locations as given by the reference wave solver and our technique, respectively. The receivers are placed on a XY grid for this scene.

It follows that the total pre-processing cost at frequency ν thus scales as

$$O(\kappa \mathbb{Q} M^2 (n\mathbb{P}^2 N^4 + u \log u + \kappa \mathbb{P} N^2 n \mathbb{Q} M^2 + n \mathbb{Q} M^2) + \kappa^3 \mathbb{P}^3 N^6) \tag{3.23}$$

At runtime, we evaluate equation (3.22), which takes $O\left(\kappa \mathbb{P} N^2\right)$ at frequency ν. The runtime memory requirement consists of positions (3 floats) and (complex-valued) strengths (2 floats) of equivalent sources, which comes out to be $\kappa(3\mathbb{P} + 2\mathbb{P}N^2)$ at frequency ν.

The precomputation and runtime complexity and memory requirement depend on the number of equivalent sources $\mathbb{P}$, which scales quadratically with frequency, in an asymptotic sense. However, for practical objects and frequencies up to 1 kHz, linear scaling of equivalent sources with frequency is observed, as shown in Figure 3.8.

We have to compute the pressure at the listener position over a regularly sampled set of frequencies in the range $[0, \nu_{max}]$ with a step of $\Delta\nu$. The total number of frequency samples becomes $\nu_{max}/\Delta\nu$. Thus, the above expressions are summed for all frequency samples in this range to give the total computational complexity and memory requirement. Since the computational cost and runtime memory scales with the multipole order, the equivalent sources are limited to monopoles and

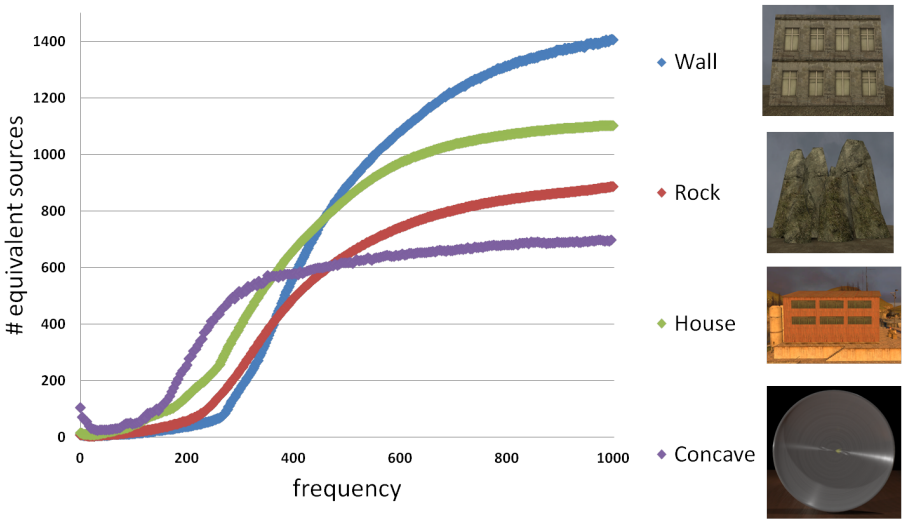


Figure 3.8: **Scaling:** Variation of the number of outgoing equivalent sources with frequency, for four different objects. As the frequency increases (wavelength decreases), surface details of the size of the wavelength increase the complexity of the sound field. This results in a larger number of equivalent sources. When all the details of the object are captured, increasing the frequency has little effect and the number of equivalent sources begin to stabilize. Error thresholds are $\sigma = 15\%$ and $\eta = 1\%$.

dipoles. Low multipole orders (N, M) result in a larger number of equivalent sources for satisfying the same error thresholds. However, low multipole order does not effect the quality of the final result since as many equivalent sources are placed as required to satisfy the error thresholds. The theoretical runtime memory requirements for other wave-based sound propagation techniques are discussed in Appendix A.3. The runtime memory requirements of these other techniques are compared with the proposed ESM technique on a variety of scenes (see Table 3.4).

Comparison with Prior Interactive Techniques Our usage of equivalent sources for sound propagation is in a similar vein to prior work (James et al., 2006), where the authors represent arbitrary outgoing radiation field from a single, geometrically complex object. Our work differs primarily in three regards: First, we model *mutual interactions* between objects in arbitrary scenes using inter-object transfer functions, accounting for high-order interactions, such as echoes and multiple diffraction. Secondly, we model acoustic scattering from objects (as opposed to radiation), which requires an approximation of both the *incoming and outgoing* pressure fields for an object. Finally, our outgoing equivalent sources are chosen to satisfy multiple outgoing fields as opposed to a single radiation field.

Scene	air. vol.	surf. area	FDTD	ARD	BEM/ FMM	Ours
Concave	$(85m)^3$	107 m^2	33 TB	0.9 TB	0.5 GB	5 MB
Wall	$(85m)^3$	71 m^2	33 TB	0.9 TB	0.3 GB	7 MB
Rock	$(85m)^3$	159 m^2	33 TB	0.9 TB	0.8 GB	15 MB
Parallel walls	$(85m)^3$	142 m^2	33 TB	0.9 TB	0.7 GB	14 MB
Desert	$(180m)^3$	1626 m^2	625 TB	17 TB	15 GB	45 MB
Reservoir	$(180m)^3$	950 m^2	625 TB	17 TB	9 GB	56 MB
Christmas	$(180m)^3$	2953 m^2	625 TB	17 TB	27 GB	65 MB

Table 3.4: Runtime memory requirements per source, for FDTD (Taflove and Hagness, 2005), ARD (Raghuvanshi et al., 2009b), BEM/FMM-BEM (Liu et al., 2009), and our ESM technique with error thresholds $\sigma = 15\%$, $\eta = 1\%$ at maximum simulation frequency $\nu_{\max} = 1018Hz$.

The problem of real-time acoustic scattering has been previously addressed using GPUs (Tsingos et al., 2007). First-order scattering effects are incorporated, but acoustic interactions between objects are not modeled. In contrast, our work can handle all orders of interactions between the objects using inter-object transfer functions.

A recent technique for interactive acoustics based on wave simulation was proposed in (Raghuvanshi et al., 2010), which relies on sampling the volume of the scene, and uses a perceptual compression specific to indoor scenes. The runtime memory requirement of their technique (per source) on our scenes after assuming a spatial sampling of $1m$ is 187 MB for the parallel walls and 1.8 GB for the reservoir scene. This technique is complimentary to our approach; it works best in indoor spaces with a lot of geometric clutter but limited volume, while our technique is better suited to large outdoor spaces with well-separated objects. In fact, it would be quite natural to integrate this method with ours, with the indoor and outdoor propagation models coupled through transport operators defined on doors and windows.

3.6 Conclusion

This chapter presents a novel wave-based sound propagation algorithm that captures acoustic effects, such as high-order diffraction and scattering, using an equivalent source formulation. As a result, this technique can perform accurate sound propagation on large, open scenes in real-time, has a small memory footprint, and allows flexible efficiency-to-accuracy tradeoffs. Compared to directly

storing and convolving wave-solver solutions for auralization, this technique reduces the memory usage by more than 2-3 orders of magnitude

Limitations and Future Work The approach is currently limited to static scenes, due to the computational cost of recomputing inter-object transfers as objects move. We would like to combine this approach with the Fast Multipole Method (FMM)(Liu et al., 2009; Gumerov and Duraiswami, 2009) to accelerate inter-object transfer evaluations using progressive far-field approximations. Moreover, real-time performance could be achieved by further using GPU-based dense linear solvers. The incoming field strength computation for a moving source is similar to inter-object transfer. Thus, the combination of FMM and GPU-based computations could enable dynamic sources along with a moving listener. Also, the current runtime system does not model the Doppler effect, which we would like to address in future work.

Currently, the sound sources emit a pre-recorded audio clip. An interesting direction of future research would be to integrate acoustic radiators based on mechanical physical models (Chadwick et al., 2009; Zheng and James, 2010) as potential sound sources, thus enabling physically-based real-time sound synthesis and propagation. The computational complexity and runtime memory requirement of the technique scale linearly with number of frequency samples which in turn scales linearly with the scene size since number of frequency samples $\propto$ length of impulse response $\propto$ scene size. Thus, the technique can easily handle scenes that are hundreds of meters wide. However, for massive outdoor scenes that span kilometers, the runtime memory requirement would become too high (GBs per source). We plan to address this in future by using FMM-based far field approximations to reduce the number of equivalent sources.

The precomputation depends heavily on the maximum simulation frequency thereby limiting it to 1-2 kHz. This behavior is consistent with other wave-based techniques like BEM and FDTD which are also computationally limited to a few kHz. Geometric approximations become quite accurate for outdoor scenes at higher frequencies because buildings and terrain have much larger dimensions than the wavelength of 17 cm at 2 kHz. Thus, hybridization of our technique with geometric methods could lead to accurate wide-band propagation techniques for open scenes. Hybridization is an active area of research in acoustics (Southern et al., 2011).

CHAPTER 4: DIRECTIONAL SOURCES AND LISTENERS

4.1 Introduction

Most sound sources we come across in real life, ranging from human voices through speaker systems, machine noises, and musical instruments, are directional sources that have a specific *directivity pattern* (Jers, 2005; Meyer and Hansen, 2009). This directivity depends on the shape, size, and material properties of the sound source, as well as a complex interaction of the processes of vibration and sound radiation, resulting in varying directivity at different frequencies. Source directivity has a significant impact on the propagation of sound and the corresponding acoustics of the environments (Vigeant, 2008) that is noticeable in everyday life: a person talking towards/away from a listener, the positioning of different types of musical instruments in an orchestra (Meyer and Hansen, 2009), and good-sounding places (sweet spots) in front of the television in the living room, aircraft, helicopter, or fire trucks in an urban environment. Analogous to directional sources, listeners also do not receive sound equally from all directions. The human auditory system obtains significant directional cues from the subtle differences in the sound received at the left and right ear, caused by the scattering of sound around the head. This listener directivity is encoded as the head-related transfer function (HRTF). Spatial sound based on listener directivity can be used to enhance a user's immersion in the virtual environment by providing binaural cues corresponding to the direction the sound is coming from, thereby enriching the experience (Begault, 1994).

Existing wave-based propagation techniques for handling source and listener directivity have many limitations. Current interactive wave-based techniques (James et al., 2006; Raghuvanshi et al., 2010; Mehra et al., 2013) have a high precomputation overhead and can only model source directivity during the offline computation stage. As a result, the source directivity gets baked (precomputed and stored) into the final solution, and it is not possible to modify the directivity pattern at runtime for interactive applications e.g. a rotating siren or a person covering his/her mouth. Additionally,

integrating listener directivity into wave-based techniques requires a plane-wave decomposition of the sound field (Zotkin et al., 2010). Previous techniques for performing plane-wave decomposition (Park and Rafaely, 2005; Rafaely and Avni, 2010; Zotkin et al., 2010) are computationally expensive and limited to offline applications. In this chapter, we address the problem of incorporating dynamic source and listener directivity in interactive wave-based sound propagation techniques.

4.2 Background

Spherical Harmonics Spherical harmonics (SH) are the angular portion of solutions to the Laplace equation when restricted to a sphere (Hobson, 1955) and form an orthonormal basis for functions defined on a sphere. These are complex-valued functions defined as:

$$Y_{lm}(\theta, \phi) = \Gamma_{lm}\, e^{\hat{j}m\phi}\, P_l^m(\cos\theta), \tag{4.1}$$

where P_l^m are the Associated Legendre polynomials, $\Gamma_{lm} = \sqrt{\frac{(2l+1)}{4\pi}\frac{(l-m)!}{(l+m)!}}$ are normalization constants and $\hat{j} = \sqrt{-1}$.

These functions have been extensively used in graphics rendering techniques, where the corresponding real-valued SH functions are used.

$$y_{lm} = \begin{cases} \sqrt{2}\,\text{Real}(Y_{l|m|}) & m > 0 \\ Y_{l0} & m = 0 \\ \sqrt{2}\,\text{Imag}(Y_{l|m|}) & m < 0 \end{cases} = \begin{cases} \sqrt{2}\,\Gamma_{l|m|}\, P_l^{|m|}(\cos\theta)\, \cos(|m|\phi) & m > 0 \\ Y_{l0} & m = 0 \\ \sqrt{2}\,\Gamma_{l|m|}\, P_l^{|m|}(\cos\theta)\, \sin(|m|\phi) & m < 0 \end{cases} \tag{4.2}$$

For detailed discussion on SH in graphics, refer to Sloan et al. (Sloan, 2008). These visual rendering techniques are not directly applicable in sound propagation due to the wave-effects such as diffraction and interference, which are typically ignored in light transport.

SH have been also been used in acoustics for beamforming with spherical microphone arrays (Meyer and Elko, 2002), spatialization in high-order ambisonic systems (Malham, 1999), plane-wave decomposition (Ward and Abhayapala, 2001) and wave-field synthesis (Ahrens and Spors, 2012). Warfusel et al. (Warusfel and Misdariis, 2004) generate source directivity from a combination of elementary SH directivities using a 3D loudspeaker array. However, the system does

not support dynamic, data-driven source directivity at runtime, and only handles morphing between elementary directivities. Also, listener directivity was handled by recording binaural responses of the room at a fixed listener head position and orientation. Recording binaural responses for all possible listener positions and orientations becomes an infeasible 5D sampling problem, requiring huge memory and manual effort. Noisternig (Noisternig and Katz, 2009) proposed a source directivity approach for GA techniques based on the discrete SH transform. However, geometric techniques cannot model directivity at low frequencies.

Head-Related Transfer function The human auditory system obtains significant directional cues from the subtle differences in sound received by each ear, caused by the scattering of sound around the head (Begault, 1994). These effects are represented using a *head-related transfer function* (HRTF). HRTF describes the effect of the listener's outer ear, head and body on the sound arriving along different directions.

Interactive GA techniques can incorporate high-frequency HRTF-based listener directivity at runtime by enumerating sound propagation paths from the source to the listener. The directions of rays received at the listener can be used to look up into the HRTF and apply the corresponding attenuation to the propagation paths. However, handling listener directivity effects at low frequency arising due to the wave nature of sound (e.g. diffraction, interference) remains a significant challenge with interactive GA techniques.

Integrating HRTFs into wave-based techniques involves computation of the propagation directions using *plane wave decomposition*. Prior plane-wave decomposition techniques either use spherical convolution (Park and Rafaely, 2005; Rafaely and Avni, 2010) or solve a linear system (Zotkin et al., 2010), and are computationally expensive. Interactive wave-based techniques resort to simpler listener directivity models based on a spherical head and a cardioid function (Raghuvanshi et al., 2010; Mehra et al., 2013). However, these simplified models are not accurate for sound localization and externalization, both of which are necessary for immersion in virtual environments (Begault, 1994).

Directivity Measurements (Meyer and Hansen, 2009) measure the source directivity of brass, woodwind and string instruments in an anechoic chamber. This data contains magnitude information only, measured in terms of relative sound pressure levels as a function of directions in different

octave bands. Otondo and Rindel (Otondo and Rindel, 2004) measured the tone-specific directivity of a trumpet, French horn, and clarinet, to understand the change of directivity with tones. Recently, directivities of male and female singing voices have also been measured (Jers, 2005). These datasets are available online and have been widely used in various sound propagation techniques. Fiestel and Ahnert (Ahnert, 2005) have investigated the use of complex frequency response data (magnitude and phase) for loudspeaker directivity; they demonstrated that incorporating the phase data reduces the acoustic prediction error by an order of magnitude as compared to the magnitude-only data. Measurements to compute HRTF-based listener directivity have been performed in controlled environments and the recorded data is available online (Algazi et al., 2001).

Plane wave decomposition In the frequency domain, the global sound field can be expressed as a superposition of pressure due to plane waves (Zotkin et al., 2010):

$$p(\mathbf{x}) = \frac{1}{4\pi} \int_S \psi_{\mathbf{s}}(\mathbf{x}) \mu(\mathbf{s}) d\mathbf{s}, \tag{4.3}$$

where $\psi_{\mathbf{s}}(\mathbf{x}) = e^{\hat{j}k\mathbf{s}\cdot(\mathbf{x}-\mathbf{x}_0)}$ is the plane wave basis (also called *Herglotz wave basis*), $\mathbf{x}_0$ is the listener position, $\mathbf{s} = (s_x, s_y, s_z)$ is the unit vector in the direction of plane wave propagation, $k = 2\pi\nu/c$ is the wave number, and $\mu(\mathbf{s})$ is the *signature function* that specifies the complex-valued amplitude of the plane wave traveling along direction $\mathbf{s}$ for a given frequency ν.

The pressure received at the left ear due to a plane wave traveling along direction $\mathbf{s}$ is given by: $H_L(\mathbf{s})\psi_{\mathbf{s}}(\mathbf{x}_0)$ where $H_L(\mathbf{s})$ is the HRTF function for the left ear. The total pressure received at the left ear is obtained by integration:

$$\begin{aligned} p_L &= \frac{1}{4\pi} \int_S H_L(\mathbf{s}) \psi_{\mathbf{s}}(\mathbf{x}_0) \mu(\mathbf{s}) d\mathbf{s} && (4.4) \\ &= \frac{1}{4\pi} \int_S H_L(\mathbf{s}) \mu(\mathbf{s}) d\mathbf{s}, \ \ (\text{since } \ \psi_{\mathbf{s}}(\mathbf{x}_0) = 1). && (4.5) \end{aligned}$$

As discussed in (Rafaely and Avni, 2010), by using the SH decomposition of the HRTF $H_L(\mathbf{s}) = \sum_{l'} \sum_{m'} \ \beta^L_{l'm'} Y_{l'm'}(\mathbf{s})$ and the signature function $\mu(\mathbf{s}) = \sum_l \sum_m \alpha_{lm} Y^*_{lm}(\mathbf{s})$ (where Y^*_{lm} is conjugate SH), the above integral gets simplified to a dot product using the orthonormality

property of SH as:

$$p_L = \frac{1}{4\pi} \sum_l \sum_m \beta_{lm}^L \alpha_{lm}. \tag{4.6}$$

An analogous equation can be derived for the right ear. Thus, spatial sound at both ears can be computed as a dot product of SH coefficients of the plane-wave decomposition and the HRTFs.

4.3 Source and Listener Directivity Formulation

An overview of our approach is given in Figure 4.1. Given a scene and a source position, a set of pressure fields due to elementary spherical harmonic (SH) sources are precomputed using a frequency-domain, wave-based sound propagation technique. Next, these pressure fields are encoded in the basis functions (e.g. multipoles) and stored for runtime use. At runtime, a SH decomposition of the dynamic source directivity is performed to compute the corresponding SH coefficients. The final pressure field is computed as a summation of the pressure fields due to SH sources (evaluated at the listener position) weighted by the appropriate SH coefficients. In order to incorporate dynamic listener directivity in wave-based techniques, an interactive plane-wave decomposition approach based on derivatives of the pressure field is proposed. Acoustic responses for both ears are computed at runtime by using this efficient plane-wave decomposition of the pressure field and the HRTF-based listener directivity. These binaural acoustic responses are convolved with the (dry) audio to compute the propagated spatial sound at the listener position.

4.3.1 Source Directivity

In this section, a far-field source representation is presented that can be used to efficiently handle dynamic, data-driven source directivity. Next, a novel directivity formulation is proposed to incorporate this source representation into a general frequency-domain wave-based propagation technique. All the variables used henceforth, except the SH basis functions, positions and speed of sound, are frequency-dependent. For the sake of brevity these dependencies are not mentioned explicitly.

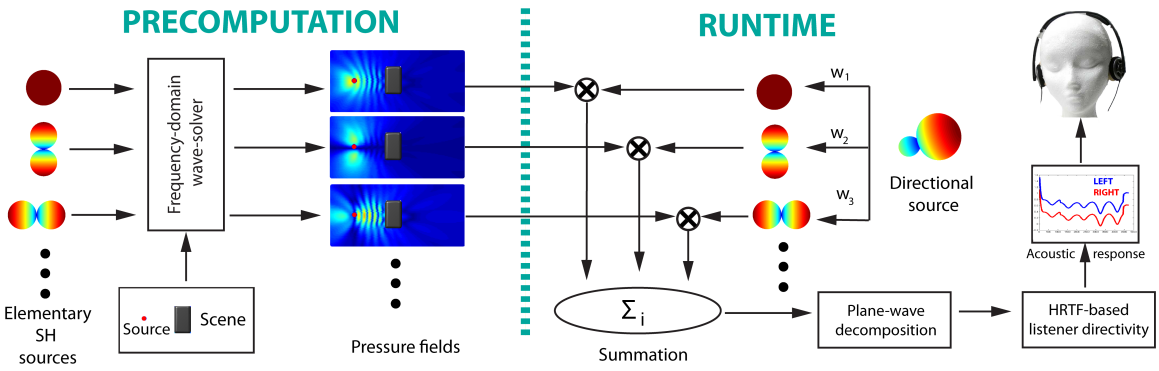


Figure 4.1: **Overview of the source and listener directivity formulation:** Given a scene and a source position, a set of pressure fields due to elementary spherical harmonic (SH) sources are pre-computed using a frequency-domain, wave-based sound propagation technique. Next, these pressure fields are encoded in the basis functions (e.g. multipoles) and stored for runtime use. At runtime, a SH decomposition of the dynamic source directivity is performed to compute the corresponding SH coefficients. The final pressure field is computed as a summation of the pressure fields due to SH sources (evaluated at the listener position) weighted by the appropriate SH coefficients. In order to incorporate dynamic listener directivity in wave-based techniques, an interactive plane-wave decomposition approach based on derivatives of the pressure field is proposed. Acoustic responses for both ears are computed at runtime by using this efficient plane-wave decomposition of the pressure field and the HRTF-based listener directivity. These binaural acoustic responses are convolved with the (dry) audio to compute the propagated spatial sound at the listener position.

Source representation The radiation pattern of a generic directional source can be expressed using the one-point multipole expansion (Ochmann, 1995) as:

$$s(\mathbf{x},\mathbf{y}) = \sum_{l=0}^{L-1} \sum_{m=-l}^{l} b_{_{lm}}\, h_l^2(2\pi\nu r/c)\, Y_{_{lm}}(\theta,\phi), \tag{4.7}$$

where $s(\mathbf{x},\mathbf{y})$ is the pressure field at point $\mathbf{x}$ of the directional source centered at point $\mathbf{y}$, $h_l^2(.)$ are the spherical Hankel functions of second kind, $Y_{_{lm}}(.)$ are complex-valued SH basis functions, (r,θ,ϕ) is the vector $(\mathbf{x}-\mathbf{y})$ expressed in spherical coordinates, $b_{_{lm}}$ are weights and L is order of the expansion, ν is the frequency, and c is the speed of sound. This source formulation is valid in both near and far-field of the source[1].

The choice of source representation for directional sources is motivated by the measured directivity data that is currently available for real-world sound sources. Most available measurements have been collected by placing sources in an anechoic chamber and recording their directivity by

[1] Far-field refers to the region of space where the distance d of any point in that region to the source is greater than the wavelength λ of the sound emitted by the source. The complementary region is the near-field (Kino, 1987, p. 165).

rotating microphones every few degrees at a fixed distance from the source. Typically, these measurements are carried out at a distance of a few meters, which corresponds to the far-field region for the frequencies emitted by these sources[2]. Keeping this in mind, we chose a source representation that corresponds to the far-field radiation pattern of a directional source. Under far-field approximation, $h_l^2(z) \approx \hat{j}^{l+1} e^{-\hat{j}z}/z$ where $\hat{j} = \sqrt{-1}$ simplifying equation (4.7) to the following source representation (Menzies, 2007):

$$s(\mathbf{x}, \mathbf{y}) = \frac{e^{-\hat{j}2\pi\nu r/c}}{r} \sum_{l=0}^{L-1} \sum_{m=-l}^{l} a_{lm} Y_{lm}(\theta, \phi), \tag{4.8}$$

$$= \sum_{l=0}^{L-1} \sum_{m=-l}^{l} a_{lm} s_{lm}(\mathbf{x}, \mathbf{y}), \tag{4.9}$$

$$= \frac{e^{-\hat{j}2\pi\nu r/c}}{r} D(\theta, \phi), \tag{4.10}$$

where $a_{lm} = b_{lm} \hat{j}^{l+1} c/(2\pi\nu)$ are the SH coefficients, $s_{lm}(\mathbf{x}, \mathbf{y}) = \frac{e^{-\hat{j}2\pi\nu r/c}}{r} Y_{lm}$ are the elementary SH sources, and $D(\theta, \phi) = \sum\sum a_{lm} Y_{lm}(\theta, \phi)$ is the source directivity function specified for each frequency either analytically or measured or simulated at discrete sample directions. The directivity function can be complex-valued (both magnitude and phase) or real-valued (magnitude-only) function. Typically, the measured data is magnitude-only and available as directivities averaged over octave-wide frequency bands (PTB, 1978).

Sound propagation for directional sources We now describe the approach for incorporating directional sources in a general frequency-domain, wave-based sound propagation technique. The steps outlined below are repeated for frequency samples in the range $[0, \nu_{\max}]$, where $\nu_{\max}$ is the maximum frequency simulated.

The linearity of the Helmholtz equation implies that the pressure field of a linear combination of sources is a linear combination of their respective pressure fields (Pierce, 1989). The proposed source representation for directional sources is a linear combination of elementary SH sources $s_{lm}(\mathbf{x}, \mathbf{y})$ with different weights a_{lm} (equation (4.9)). Therefore, for a given scene, if the pressure field $p_{lm}(\mathbf{x})$ corresponding to each of the elementary SH sources $s_{lm}(\mathbf{x}, \mathbf{y})$ is precomputed, then the pressure

[2] For a distance $d > 3.43$m, corresponds to far-field for all the frequencies $\nu > 100$ Hz (for all wavelengths $\lambda = c/\nu < 3.43$m).

field $p(\mathbf{x})$ due to any directional source $s(\mathbf{x}, \mathbf{y})$ can be expressed as the linear combination of the precomputed pressure fields $p_{lm}(\mathbf{x})$ of these elementary SH sources with the same weights a_{lm}:

$$\underbrace{\sum_l \sum_m a_{lm} s_{lm}(\mathbf{x}, \mathbf{y})}_{s(\mathbf{x},\mathbf{y})} \longrightarrow \underbrace{\sum_l \sum_m a_{lm} p_{lm}(\mathbf{x})}_{p(\mathbf{x})}.$$

The pressure fields for elementary SH sources can be computed using any wave-based sound propagation technique. In the case of interactive applications, this computation is performed during the preprocessing stage, and the resulting pressure field data is efficiently encoded and stored. This pressure field data completely defines the acoustic response to any directional source at the given position up to the SH approximation order. At runtime, the specified source directivity $D(\theta, \phi)$ is decomposed into a SH-based representation using SH projection methods, and the resulting weights a_{lm} are used to compute the final pressure field at listener position, as described above.

SH projection Given an analytical expression for the directivity function $D(\theta, \phi)$, the SH coefficients a_{lm} can be computed by SH projection as follows:

$$a_{lm} = \int_0^{2\pi} \int_0^{\pi} D(\theta, \phi)\, Y_{lm}(\theta, \phi)\, \sin\theta \, d\theta \, d\phi. \tag{4.11}$$

This expression can be evaluated either symbolically or numerically, depending on $D(\theta, \phi)$ (Green, 2003).

Given the directivity function $D(\theta, \phi)$ at sampled locations $\{(\theta_1, \phi_1), (\theta_2, \phi_2), ..., (\theta_n, \phi_n)\}$, the spherical harmonic expansion can be fitted to this function in the least-square sense, by solving an over-determined linear system ($n > L^2$) to compute the coefficients a_{lm}:

$$\sum_l \sum_m Y_{lm}(\theta_k, \phi_k)\ a_{lm} = D(\theta_k, \phi_k). \quad \text{for } k = 1, ..., n; \tag{4.12}$$

The source representation can handle both complex-valued and real-valued directivities. In case the directivity function is real-valued (as the widely available measured data is magnitude only), the real-valued SH functions (Green, 2003) are used in the aforementioned expressions.

Dynamic and Rotating Directivity For a source with dynamic directivity, the SH decomposition of the source directivity function $D(\theta, \phi)$ is computed at runtime. This is performed by solving equation (4.12) at interactive rates using fast linear solvers (such as Intel MKL). For the special case of a rotating directional source, the new spherical harmonic coefficients a_{lm}^{rot} after rotation can be computed by applying the spherical harmonic rotation matrix to the original coefficients $a_{l'm'}$ as follows:

$$a_{lm}^{rot} = \sum_{l'} \sum_{m'} Z(lm, l'm') \;\; a_{l'm'}, \tag{4.13}$$

where $Z(j, k)$ is the $(j, k)^{th}$ entry of the spherical harmonic rotation matrix Z, described in detail by Green et al. (Green, 2003). Matrices can also be derived for more general rotations about arbitrary axes (Green, 2003, p. 21-26).

Note that the linearity of the Helmholtz equation and the use of the spherical harmonic basis enables this source directivity framework to handle dynamic directivity at runtime without running the offline simulation again.

4.3.2 Spatial Audio

In this section, we discuss the method for computing spatial sound using efficient plane-wave decomposition of the pressure field and the HRTF-based listener directivity. To support a moving and rotating listener (dynamic listener directivity), one needs to update the SH coefficients of the HRTFs β_{lm} and plane-wave decomposition α_{lm} interactively. The head rotation of listener can be incorporated by applying SH rotation techniques to the HRTF's SH coefficients (equation (4.12)). However, in order to handle listener translation, the SH coefficients of the plane-wave decomposition will have to be recomputed at runtime. Previous techniques for plane-wave decomposition (Park and Rafaely, 2005; Rafaely and Avni, 2010; Zotkin et al., 2010) cannot compute these coefficients at interactive rates.

Plane-wave decomposition using derivatives A novel method to compute the SH coefficients of the plane-wave decomposition at interactive rates using the derivatives of the pressure field is proposed next.

Theorem : *The SH coefficients of the plane-wave decomposition of the pressure field at a point in space can be expressed as a linear combination of pressure field derivatives at that point.*

Proof : Given the polynomial expression of the n^{th} order and q^{th} degree SH as

$$Y_{nq}(\mathbf{s}) = A \sum_{(a,b,c)} \Gamma_{a,b,c} \quad s_x^a \, s_y^b \, s_z^c,$$

where $\Gamma_{a,b,c}$ is a constant, $\mathbf{s} = (s_x, s_y, s_z)$ is the unit vector in the direction of plane wave propagation, $a \geq 0, b \geq 0, c \geq 0$, and $a + b + c = n$. For computing the SH coefficient of the plane-wave decomposition α_{nq}, multiply both sides by $\alpha_{lm} Y_{lm}^*$ (where Y_{lm}^* is conjugate SH) and apply summation and integral operators to give

$$\sum_l \sum_m \alpha_{lm} \int Y_{nq} \, Y_{lm}^* d\mathbf{s} = A \sum_{(a,b,c)} \Gamma_{a,b,c} \quad \sum_l \sum_m \alpha_{lm} \int s_x^a \, s_y^b \, s_z^c \, Y_{lm}^* d\mathbf{s}.$$

Using the orthonormality property of SH, we get

$$\alpha_{nq} = A \sum_{(a,b,c)} \Gamma_{a,b,c} \quad \sum_l \sum_m \alpha_{lm} \int s_x^a \, s_y^b \, s_z^c \, Y_{lm}^* d\mathbf{s}. \tag{4.14}$$

We denote the partial a^{th} x-derivative, partial b^{th} y-derivative, partial c^{th} z-derivative of pressure field as $p^{(a,b,c)} = \frac{\partial^{a+b+c} p}{\partial x^a \partial y^b \partial z^c}$. The total pressure at point $\mathbf{x}$ (equation 4.3) along with the SH-expansion for signature function $\mu(s)$ gives

$$p(\mathbf{x}) = \frac{1}{4\pi} \sum_l \sum_m \alpha_{lm} \int_S \psi_{\mathbf{s}}(\mathbf{x}) Y_{lm}^*(\mathbf{s}) d\mathbf{s}.$$

On differentiating and evaluating the above expression at $\mathbf{x}_0$, we get

$$\frac{4\pi}{(\hat{\jmath}k)^{a+b+c}} \; p^{(a,b,c)}(\mathbf{x}_0) = \sum_l \sum_m \alpha_{lm} \int s_x^a \, s_y^b \, s_z^c \, Y_{lm}^* d\mathbf{s}, \;\; (\text{since } \; \psi_{\mathbf{s}}(\mathbf{x}_0) = 1).$$

Substituting above expression in equation (4.14), we get

$$\alpha_{nq} = A \sum_{(a,b,c)} \Gamma_{a,b,c} \quad \frac{4\pi}{(\hat{\jmath}k)^{a+b+c}} \; p^{(a,b,c)}(\mathbf{x}_0). \tag{4.15}$$

The above expression relates the pressure field derivatives to the SH coefficients of the plane-wave decomposition, at the listener position $\mathbf{x}_0$. This gives us an efficient method to compute the plane-wave decomposition using derivatives of the pressure field.

Sound propagation for directional listeners At runtime, the pressure field derivatives at the listener position are computed interactively by differentiating the pressure field's basis functions corresponding to the wave-based propagation technique *analytically* rather than using a finite difference stencil (Section 4.4). Therefore, higher-order derivatives do not suffer from numerical instabilities allowing the SH coefficients of the plane-wave decomposition to be computed to higher-orders (equation (4.15)). Finally, we compute the spatial sound for the left and right ears as a dot product of the SH coefficients of the plane-wave decomposition and the HRTFs (equation (4.6)).

Over the years years, several SH-based models have been proposed for representing HRTFs (Rafaely and Avni, 2010; Pollow et al., 2012). This work is orthogonal to those techniques. As discussed above, to compute spatial audio, a dot product of SH coefficients of the HRTFs and the plane wave decomposition of pressure field is required (Rafaely and Avni, 2010). These techniques focus on the first part - computing better HRTF coefficients. The main contribution of this work lies in the second part - efficiently computing SH coefficients of the plane-wave decomposition at interactive rates to support a moving and rotating listener. This allows efficient computation of spatial sound for a moving listener at runtime. The approach can compute spatial sound for a head-tracked listener at interactive rates and provides the flexibility to use personalized (user-customized) HRTFs without recomputing the simulation results.

4.4 Integration with Sound Propagation Techniques

In this section, we discuss the integration of the proposed source and listener directivity with wave-based sound propagation techniques.

4.4.1 Boundary element method (BEM)

The boundary element method (Liu, 2009) is a numerical technique used for solving the 3D Helmholtz equation. It accurately models sound propagation in indoor and outdoor spaces, and is widely used in many fields such as computational acoustics, noise predictions, room acoustics, and

underwater acoustics. BEM transforms the Helmholtz equation into the boundary integral equation, then solves for pressure and velocity on the boundary and thereby pressure at any point in the domain.

Simulation Given a scene (indoor or outdoor) with source position $\mathbf{y}$, the Helmholtz equation is solved corresponding to the elementary SH source $s_{lm}(\mathbf{x}, \mathbf{y})$ using BEM. This gives us pressure $q_{lm}(\mathbf{z})$ and velocity $v_{lm}(\mathbf{z})$ on the domain boundary where $\mathbf{z}$ is a point on the boundary. This step is repeated for all the elementary sources $s_{lm}(\mathbf{x}, \mathbf{y})$ in the SH expansion (equation (4.9)). The resulting pressure and velocity on the domain boundary is stored and then used at runtime for evaluating pressure and its high-order derivatives at the listener position.

Pressure evaluation We compute the pressure field $p_{lm}(\mathbf{x})$ at point $\mathbf{x}$ due to the elementary SH sources $s_{lm}(\mathbf{x})$ by using the boundary integral equation:

$$p_{lm}(\mathbf{x}) = \int_S \left[G(\mathbf{x}, \mathbf{z}, \omega) q_{lm}(\mathbf{z}) - F(\mathbf{x}, \mathbf{z}) v_{lm}(\mathbf{z}) \right] dS(\mathbf{z}) + s_{lm}(\mathbf{x}, \mathbf{y}),$$

where $G(\mathbf{x}, \mathbf{z}, \omega) = \frac{1}{4\pi d} e^{\hat{j}\omega d/c}$ is the Green's function ($d = \|x - z\|$), $F(\mathbf{x}, \mathbf{z}) = \frac{\partial G(\mathbf{x}, \mathbf{z}, \omega)}{\partial n(\mathbf{z})}$, and $n(\mathbf{z})$ is normal at a boundary point $\mathbf{z}$. The final pressure field at the listener position $\mathbf{x}_0$ due to the directional source $s(\mathbf{x}, \mathbf{y})$ is computed as a linear combination of the pressure fields of the elementary SH sources as

$$p(\mathbf{x}_0) = \sum_l \sum_m a_{lm} p_{lm}(\mathbf{x}_0). \tag{4.16}$$

Pressure derivative evaluation In order to compute spatial sound, we need to determine the coefficients of the plane wave decomposition using pressure derivatives (Equation (4.15)). The first-order derivative of the pressure field due to elementary source is computed by differentiating the functions involved analytically:

$$\frac{\partial p_{lm}(\mathbf{x})}{\partial x} = \int_S \left[\frac{\partial G(\mathbf{x}, \mathbf{z}, \omega)}{\partial x} q_{lm}(\mathbf{z}) - \frac{\partial F(\mathbf{x}, \mathbf{z})}{\partial x} v_{lm}(\mathbf{z}) \right] dS(\mathbf{z}) + \frac{\partial s_{lm}(\mathbf{x})}{\partial x}.$$

The first-order derivative of the total pressure field at the listener position $\mathbf{x}_0$ is computed as

$$\frac{\partial p(\mathbf{x}_0)}{\partial x} = \sum_l \sum_m a_{lm} \frac{\partial p_{lm}(\mathbf{x}_0)}{\partial x}. \tag{4.17}$$

Higher-order derivatives of the pressure field at the listener position can be computed in a similar manner.

4.4.2 Equivalent source method (ESM)

The integration of the source and directivity approach with the equivalent source method for sound propagation (Chapter 3) can be performed as follows:

Preprocessing Assume a scene composed of κ objects, A_1, A_2, ..., A_κ with sound source positioned at $\mathbf{y}$. During the preprocessing stage, we express the elementary SH source $s_{lm}(\mathbf{x}, \mathbf{y})$ in terms of the incoming equivalent sources of these objects. Next, we perform the global solve step to compute the strengths of outgoing equivalent sources for all the objects $(C_{lm})_{A_1}$, $(C_{lm})_{A_2}$, ..., $(C_{lm})_{A_\kappa}$. These outgoing equivalent source strengths represent an efficient encoding of the pressure field for the scene. This step is repeated for all the elementary SH sources $s_{lm}(\mathbf{x}, \mathbf{y})$ in the source representation (Equation (4.8)), and the computed equivalent source strengths are stored. These equivalent sources are then used at runtime to compute pressure and its high-order derivatives.

Runtime The equivalent source strengths are used to compute the pressure fields $p_{lm}(\mathbf{x})$ for the elementary SH sources $s_{lm}(\mathbf{x})$:

$$p_{lm}(\mathbf{x}) = \sum_{j=1}^{\kappa} (C_{lm})_{A_j}^{tr} \Phi_{A_j}^{out}(\mathbf{x}) + s_{lm}(\mathbf{x}), \tag{4.18}$$

where $\Phi^{out}(\mathbf{x})$ is the equivalent source basis functions (Equation 3.22). The total pressure field at the listener position due to the directional source $s(\mathbf{x}, \mathbf{y})$ is computed as before.

The derivative of the pressure field due to elementary source $\partial p_{lm}(\mathbf{x})/\partial x$ is computed by analytically differentiating the functions involved: the equivalent source basis functions $\Phi^{out}(\mathbf{x})$ and the source field $s_{lm}(\mathbf{x})$.

$$\frac{\partial p_{lm}(\mathbf{x})}{\partial x} = \sum_{j=1}^{\kappa} (C_{lm})_{A_j}^{tr} \frac{\partial \Phi_{A_j}^{out}(\mathbf{x})}{\partial x} + \frac{\partial s_{lm}(\mathbf{x})}{\partial x}. \tag{4.19}$$

The derivative of the total pressure field is computed as before.

4.5 Implementation

A 64-bit FASTBEM implementation of the boundary element method was used for BEM (`www.fastbem.com`). For ESM, the preprocessing code is implemented in MATLAB. The runtime code is implemented in C++ and has been integrated with Valve's Source game engine. The timing results for the BEM and ESM precomputation were measured on a 64-node CPU cluster (Xeon 5560 processor nodes, 8 cores, 2.80 GHz, 48 GB memory). The precomputation for each frequency is performed in parallel over all the nodes of the cluster. The timing results of the runtime system were measured on a single core of a 4-core 2.80 GHz Xeon X5560 desktop with 4 GB of RAM and NVIDIA GeForce GTX 480 GPU with 1.5 GB memory. Acoustic responses are computed up to the maximum simulation frequency of 1 kHz.

Measurement data The source directivity data used in the system is extracted from real-world measurement data provided by Meyer et al. (PTB, 1978). They recorded the directivity of various musical instruments belonging to the brass, woodwind and string categories, as well as the human voice. Measurements were collected on 36 longitudinal lines that span from 'front' to the 'back'. The count direction of these longitudinal lines continues counterclockwise at 10° steps. On each longitudinal line, samples are placed every 10° resulting in 19 samples per line. This data is magnitude-only, averaged over the frequencies in each octave band, and is provided in terms of relative sound pressure levels, and is normalized with respect to the front, which is at the reference level of 0 dB. The data was first "un-normalized" and converted to linear scale. Then SH coefficients were computed by solving the over-determined linear system as described in (Equation (4.12)). SH order $L = 3 - 4$ was used for the source representation, resulting in error less than $10 - 15\%$, which is a typical error threshold used for auralization purposes in interactive applications (James et al., 2006; Raghuvanshi et al., 2010). Error is defined as

$$\text{Error}(D, D_{SH}) = \sqrt{\frac{\sum_i \sum_j \|D(\theta_i, \phi_j) - D_{SH}(\theta_i, \phi_j)\|^2}{\sum_i \sum_j \|D(\theta_i, \phi_j)\|^2}}, \tag{4.20}$$

where D is the measured directivity data and D_{SH} is our SH-based source directivity representation (Section 4.3.1). For SH decomposition of the directivity function, we did not observe any significant ringing for the first four octave bands. For the last two octave bands, ringing was resolved by the

standard technique of windowing the truncated SH coefficients using Lanczos sigma factors (Sloan, 2008). For spatial sound, the HRTF dataset provided by (Algazi et al., 2001) was used, particularly the KEMAR dummy dataset. The SH coefficients of this HRTF were computed using standard SH projection (Equation (4.12)).

HRTF modeling The listener directivity approach supports any kind of SH-based HRTF models (Rafaely and Avni, 2010; Pollow et al., 2012). In practice, the SH-based HRTF model proposed by (Rafaely and Avni, 2010) is used. A detailed evaluation for studying the effect of HRTF's SH order on spatial perception was conducted using various metrics such as inter-aural time difference (ITD), inter-aural level difference (ILD) and inter-aural cross-correlation coefficient (IACC) (Rafaely and Avni, 2010; Avni, 2010). These results indicate that a SH order of 1-2 is sufficient for spatial perception of frequencies up to 1 kHz; a SH order of 3 suffices up to 2 kHz; and a SH order of 3-6 suffices up to 8 kHz. Higher SH orders result in better spatial resolution, but computing the higher-order derivatives of pressure field for plane-wave decomposition also increases the computational cost. Therefore, the SH order may be determined based on the performance-accuracy trade-off.

Auralization Pressure fields for elementary SH sources, precomputed using the equivalent source technique and the SH coefficients of the HRTF, are loaded into Valve's game engine upon startup. At runtime, the acoustic responses of the elementary SH sources are evaluated and extrapolated to the output frequency (22 kHz). SH coefficients of the source directivity are computed and used to generate the total pressure field and its derivatives at the listener position. As the listener (player) moves through the scene, the computed pressure and pressure derivatives are combined with the SH coefficients of the HRTF to compute binaural frequency responses, producing spatial sound at the listener. These frequency-response computations are performed asynchronously from both the visual-rendering pipeline and the audio-processing pipeline. As shown in Table 4.2, this technique can update binaural frequency responses at a rate of 10-15 Hz or more, which is sufficient for audio applications (IASIG, 1999), while the game itself is able to perform visual rendering at 60 frames per second (or more). Audio processing is performed using FMOD (with Fourier transforms computed using Intel MKL), and rendered in frames of 1024 samples.

4.6 Results

Scenarios The scenarios used to demonstrate the effects of source and listener directivity are as follows:

Crowd This scene shows the effect of source directivity in an everyday scenario. In this first scenario, there are two people talking in a noisy, crowded street. As the listener moves from behind the person talking to his front, the voice becomes louder and clearer. This demonstrates the directional behavior of human voice. Sound in front of the head is more pronounced than that towards the back of the head.

In the second scenario, two people are talking on an empty street. In front, both low and high frequency sounds (vowels and consonants) are audible. However, when the listener is behind the person, only the low frequency sounds (vowels) are audible due to diffraction. In both these scenarios, the measured data for human voice directivity (PTB, 1978) was used. These are simple scenarios where the sound propagation is in free space.

Musical instruments Cello and bass have prominent directivity at low frequencies. This scene demonstrates the technique's ability to handle rotating directional sources at runtime. Both the instruments are rotated to show the change in propagated sound at the listener position. The measured directivity has been expressed using the SH-based source representation at varying SH order ($L = 1, 2, 3$). As the SH order increases, our source representation is able to capture the directivity with better accuracy. The sound propagation for this scene also happens in free space.

Empty living room This scenario demonstrate the effect of source directivity on sound propagation. A comparison is shown between the sound propagation for an omnidirectional source and a directional source.

Furnished living room This scene shows the sound propagation due to a television (a typical directional source) in the living room setting. It also demonstrates low-passing of diffracted sound behind the doors.

Wall The previous scenes used a directional source with a monaural listener. In other words, the sound was not spatialized: the listener received no cues about the direction from which the sound came. This scene demonstrates the importance of spatial sound. First, a walkthrough of the entire scene is performed with spatial sound. It demonstrates that our listener directivity approach enables

the listener to localize the direction of the helicopter. Next, there is a rotating dipole source in the scene. As the lobe of the dipole sweeps past the listener counterclockwise, sound can be heard on the left channel. But as the lobe turns to the wall, strong wall reflections are received from the right which dominates the sound. Lastly, it shows the dynamic source directivity caused by the motion of an obstacle in front of the talking person.

Reservoir (from Half-Life 2) This scene demonstrates sound propagation due to directional sound sources and listener. It is a huge outdoor terrain with many rocks and a building. and two sound sources - a helicopter and a cello. For this case, a quadrupole ($l = 2, m = 2$) directivity is assigned to the helicopter, and measurement data-based directivity is assigned to the cello. The technique has been integrated with Valve Source^{TM} game engine to generate realistic acoustic effects arising from directional sources and listeners.

Christmas town This is a snow-covered town with many houses, a church, tall buildings, and a bell tower. It demonstrates the source directivity of bell tower.

Amphitheater This is a huge outdoor amphitheater with musical instruments playing at the center of the stage. It demonstrates propagated sound due to musical instruments (directional sources) at various listener positions around a real-world environment.

Timing Results In Table 4.1, the precomputation cost of BEM and ESM techniques for performing sound propagation for directional sources is shown. The three independent computations (for all sound sources; the elementary SH sources; and the different frequencies) can be easily performed in parallel. Table 4.2 shows the runtime performance and memory requirements of the interactive ESM technique for computing the pressure field due to directional source at a moving listener. It shows timing results with and without spatial sound. The spatial sound computation requires computation of pressure field derivatives. Runtime timings of BEM for the empty room and the furnished room at the listener position are 3 and 5 seconds, respectively. The runtime memory cost for the BEM pressure fields is 716 MB and 1.4 GB for the empty and furnished room, respectively.

Figure 4.4 demonstrates the directivity of various musical instruments and a human singer represented in the proposed SH-based source directivity formulation. Figure 4.2 shows that with increasing frequency of sound sources, higher-order terms in the SH representation are required. As a general trend, sound source directivity becomes sharper (more prominent) with increasing frequency,

ESM scenes	air volume	surface area	#objs. /#srcs	# freq. / L	ESM-sim (wall-clk)
Crowd	$85^3 m^3$	–	0/1	250/4	–
Music	$85^3 m^3$	–	0/1	250/3	–
Rotating	$85^3 m^3$	$71 m^2$	1/1	250/3	165 min
Parallel	$85^3 m^3$	$142 m^2$	2/1	250/3	195 min
Amphitheater	$180^3 m^3$	$220 m^2$	1/3	500/4	305 min
Reservoir	$180^3 m^3$	$950 m^2$	5/2	500/3	829 min
Christmas	$180^3 m^3$	$2952 m^2$	5/1	500/3	894 min

BEM scenes	air volume	surface area	#srcs	# freq. / L	BEM-sim (wall-clk)
Empty room	$24 m^3$	$60 m^2$	1	250/4	45 min
Furnished room	$66 m^3$	$142 m^2$	1	250/4	700 min

Table 4.1: **Precomputation cost:** Abbreviations are as follows- "#objs." number of ESM objects in the scene, "#srcs" number of directional sound sources, "#freq." number of frequency samples in the range (0-1 kHz), "L" is the SH order, and "ESM-sim" or "BEM-sim" is the total wall clock time to compute pressure fields using the ESM or BEM propagation techniques, respectively. The sound-propagation computations are performed in parallel for all the elementary SH sources for all the frequencies on a 64-node cluster with 8 cores per node. 'Crowd' and 'Music' scenes have no precomputation time, since only free-space propagation is performed.

Scene	L	# eq. srcs	eval. (no spatial, per src)	eval. (spatial, per src)	storage (total, fixed + per src)
Crowd	4	–	0.14 ms	0.34 ms	–
Music	3	–	0.14 ms	0.34 ms	–
Rotating	3	0.1 M	12.6 ms	16.7 ms	(1 + 29) MB
Parallel	3	0.2 M	17.9 ms	24 ms	(2 + 58) MB
Amphitheater	4	0.1 M	14 ms	18.6 ms	(1 + 51) MB
Reservoir	3	0.8 M	86 ms	115 ms	(10 + 230) MB
Christmas	3	0.7 M	90.6 ms	119.5 ms	(8 + 202) MB

Table 4.2: **Runtime performance:** "L" is the SH order, and "# eq. srcs" is the number of equivalent sources (in millions). For each scene, the runtime performance "eval." with and without spatial sound and the storage requirement "storage" are shown. Storage numbers indicate fixed cost for equivalent source positions and the total cost for storing the strengths for all elementary SH sources up to SH order L. 'Crowd' and 'Music' scenes have no equivalent sources, since only free-space propagation is performed.

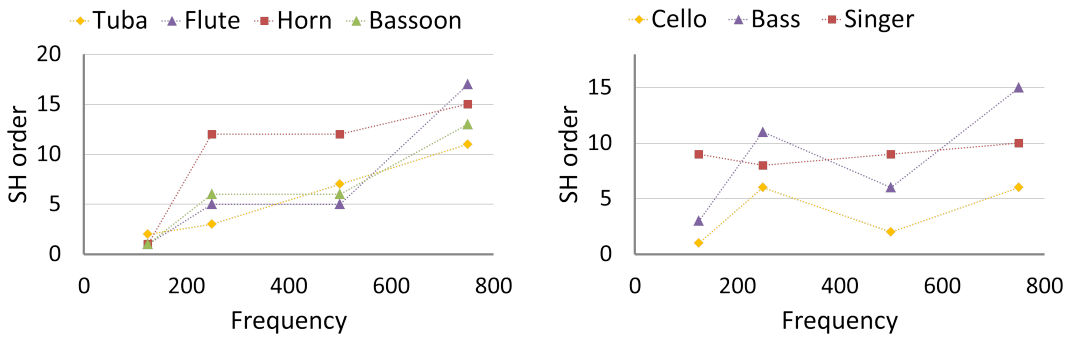


Figure 4.2: **Scaling with frequency**: Directivity of sound sources becomes sharper with frequency. To approximate the directivity function at higher frequencies while maintaining a constant error threshold of 5%, higher-order SH source representation is required.

requiring higher-order SH basis functions. Figure 4.3 demonstrates how directivity affects sound propagation.

4.7 Conclusion

We present a novel approach for incorporating dynamic source and listener directivity in a general frequency-domain, wave-based sound propagation technique. This approach can automatically model wave-effects from low-frequency directional sources and can handle analytical, data-driven, rotating or dynamic directivity at runtime. We have also described an efficient plane-wave decomposition approach to handle HRTF-based listener directivity in order to perform spatial sound rendering at interactive rates.

Limitations and Future Work Directional sources with sharp directivity patterns (delta functions) would be expensive to handle with our current approach since it would require a very high-order SH expansion. It would be interesting to explore other basis functions, such as wavelets, to handle sharp directivities. We would also like to add support for artist-controlled directivity, allowing real-time feedback on the effect of directivity on propagated sound. The current implementation uses magnitude-only directivity data, but the approach can easily support complex data (magnitude and phase), which we plan to test in the future. The source formulation can handle both near- and far-field sound radiation by directional sources. However, near-field directivity requires a set of dense measurements of complex frequency responses very close to the source at twice the Nyquist

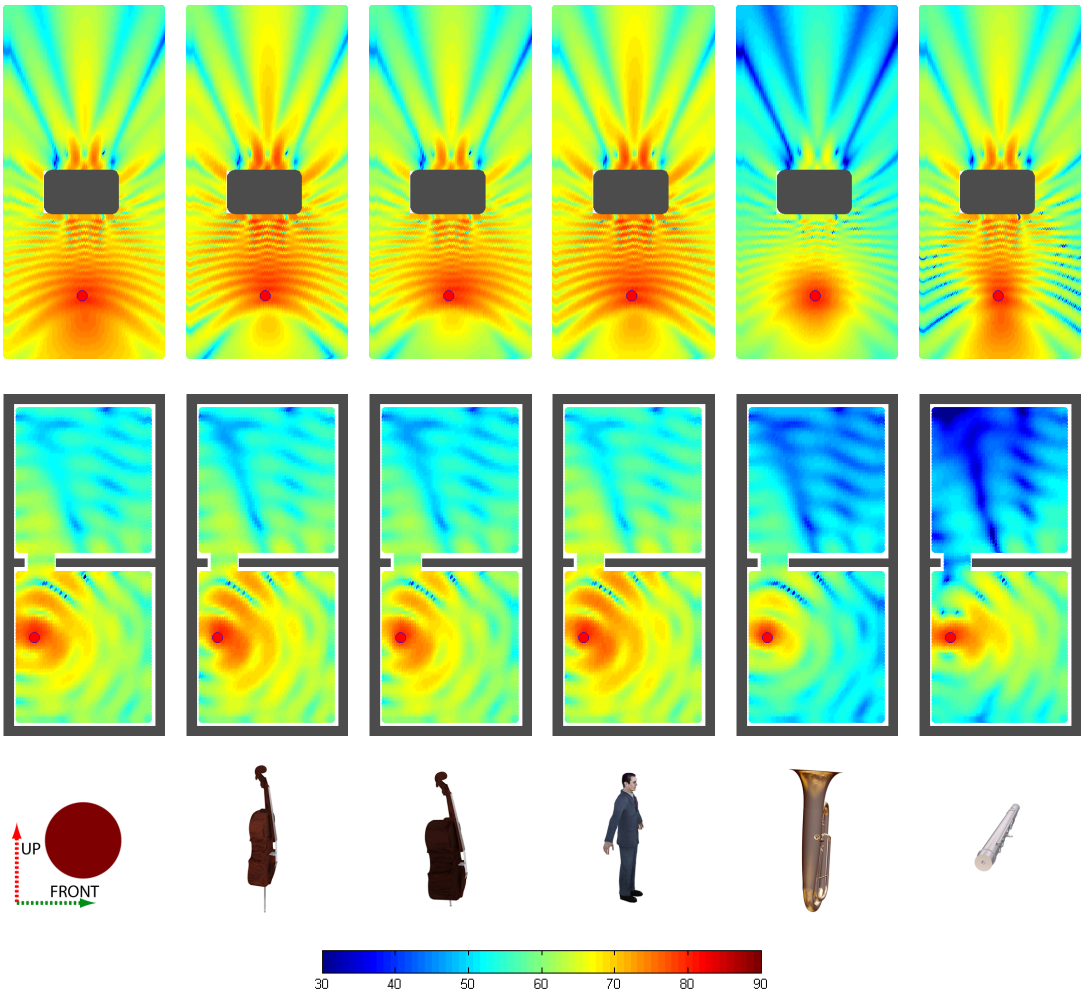


Figure 4.3: **Effect of directivity on propagated sound field**: Magnitude of pressure field (in dB) on a grid of listener positions for the single building and the empty room scene at 360 Hz with different directional sources. Source position is shown with a red dot. Front axis points towards the building in the single building and away from the left wall in the empty room. Up axis points upwards perpendicular to the listener grid.

rate, which is currently unavailable. We hope that such a dataset becomes available in the near future. Lastly, since wave-based approaches are computationally limited to a few kHz frequency, we would like to explore hybridization of wave-based techniques with geometric approaches to handle directional sources over the complete audible frequency range.

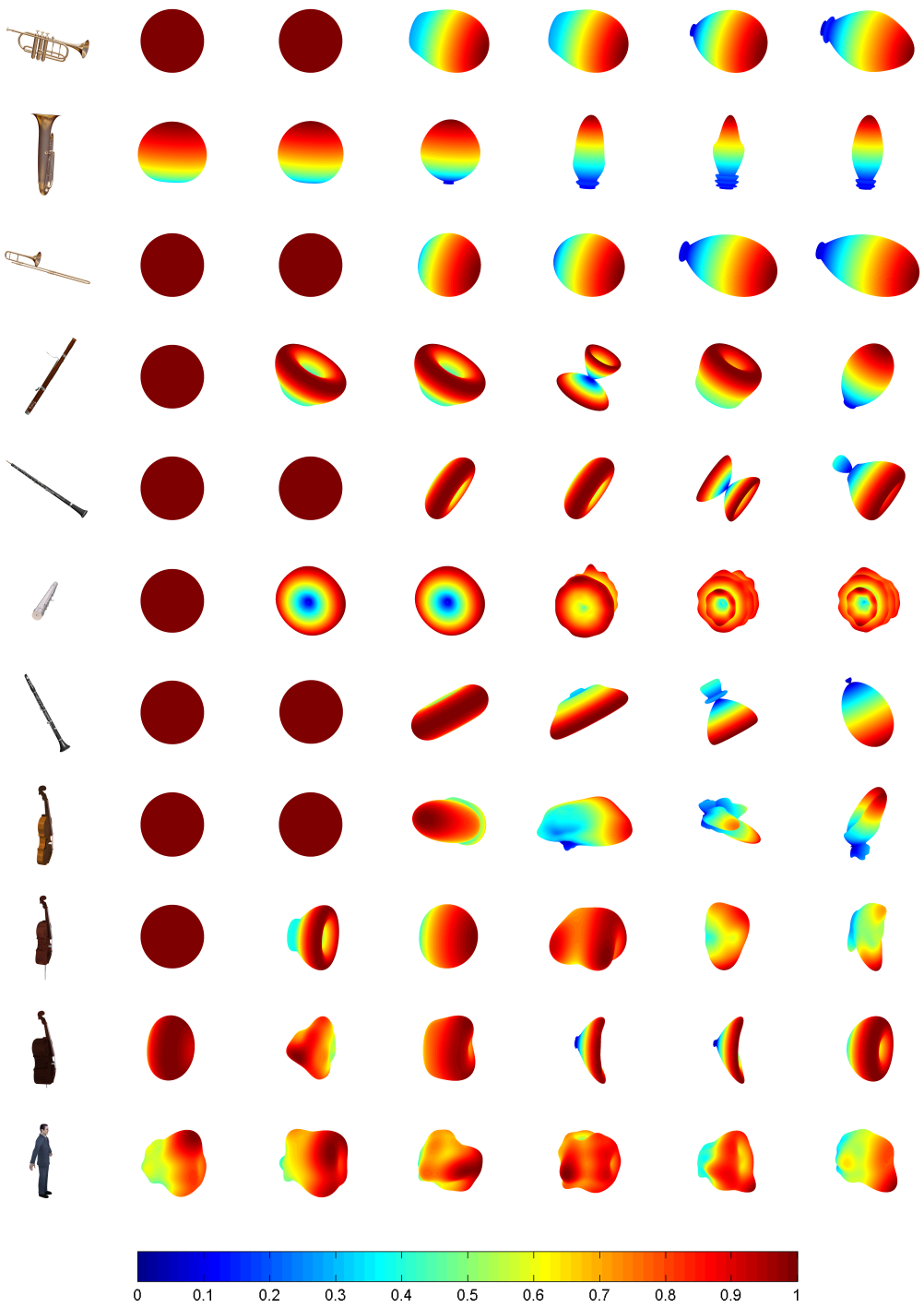


Figure 4.4: **Directivity varies with frequency**: Source directivity of musical instruments and a human singer corresponding to different frequency octaves: $[0-176]$ Hz, $[177-354]$ Hz, $[355-710]$ Hz, $[710-1420]$ Hz, $[1420-2840]$ Hz and $[2840-5680]$ Hz. These directivity patterns are visualized by fitting our spherical harmonic-based source representation (error threshold 5%) on the real-world measurement data provided by Meyers et al.(PTB, 1978). All directivity values have been normalized to the range $[0, 1]$. These visualizations are plotted by distorting a sphere where each point is scaled radially and colored by the directivity function value (color bar shown).

CHAPTER 5: PARALLEL ADAPTIVE RECTANGULAR DECOMPOSITION

5.1 Introduction

The frequency range of human hearing can be roughly split into three parts: (a) low frequencies ($<$ few hundred Hz), (b) mid frequencies ($>$ few hundred Hz and $<$ few kHz), and (c) high frequencies ($>$ few kHz). Sound propagation for high frequencies of sound can be easily performed using the geometric techniques. In this range, the wavelength of sound is much smaller than the size of everyday objects, and therefore the geometric approximation is valid. Thus, geometric techniques can accurately generate propagation effects in this frequency range. At low and mid frequencies of sound, wave effects dominate since the wavelength (ranging from tens of centimeters to few meters) is of the same size as everyday objects. Therefore, in these frequency ranges, accurate sound propagation requires the use of wave-based techniques. These techniques solve the acoustic wave equation (or the Helmholtz equation) of sound propagation and can model all the wave effects, such as diffraction and interference, accurately. Interactive wave-based propagation techniques split the computation into two parts, a preprocessing step and a runtime step. The preprocessing step runs the wave simulation for the scene and should take tens of minutes to a few hours of computation time. The simulation results are then stored compactly for runtime use. The runtime step uses this precomputed data to compute the propagated sound field at the listener position interactively. The runtime computation should be lightweight and memory requirements should be low.

The computational requirements of the precomputation step of the wave-based techniques vary as the fourth power of the frequency. These techniques can perform the wave simulation for the low frequency range in tens of minutes to a few hours. However, the computation time for handling mid-frequencies varies from tens of hours to days. This can be problematic for wave-based techniques especially if they are being used in interactive applications such as games, architectural acoustics, or VR simulations. If it takes days to run the acoustic wave simulation every time the game designer or

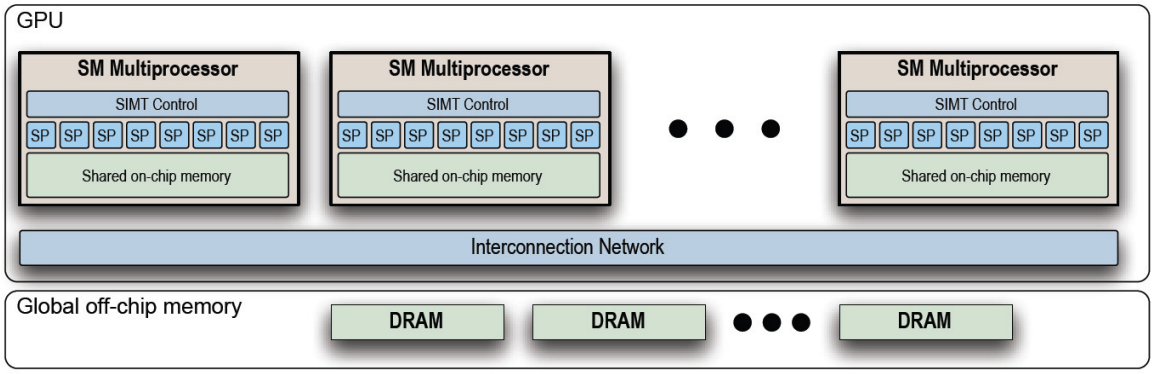


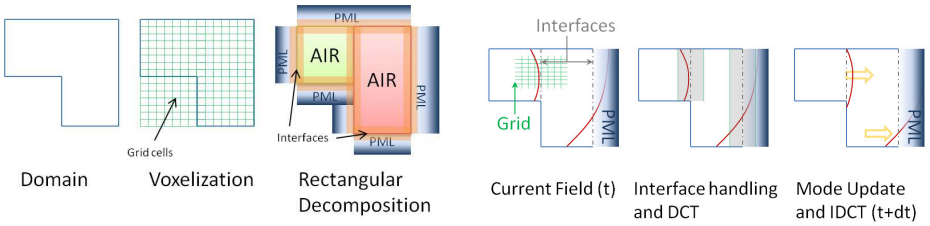
Figure 5.1: The graphics processing unit (GPU) architecture (image © Savioja (Savioja, 2010)): Current generation GPUs have many streaming multiprocessors(SMs), each containing several streaming processors(SPs), a fast on-chip shared memory and a single-instruction multiple-thread (SIMT) control unit. All the multiprocessors are connected to each other and to a larger off-chip global memory via a fast interconnection network.

architect makes a small change to the design, this would make the design process time-consuming and impractical.

In this Chapter, we present an efficient GPU-based wave solver that can perform wave-simulation for the mid-frequency range in tens of minutes. This technique is based on the adaptive rectangular decomposition approach and uses analytical solutions of wave equation inside rectangular domains. It is shown that by parallelizing all the components of the solver to match the parallel processing capabilities of graphics processors, significant improvement in performance can be achieved.

5.2 Background

GPU Architecture The architecture consists of a scalable array of **streaming multiprocessors** (SMs), each of which consists of a group of **streaming processors** (SPs), a fast (but small) on-chip **shared memory** and a SIMT control unit (see Figure 5.1). All the multiprocessors are connected to a large off-chip **global memory** via a **interconnection network**. In order to effectively solve a problem on a GPU, first it has to be partitioned into coarse sub-problems that can be solved independently in parallel by blocks of threads. These thread blocks are enumerated and distributed to the available SMs. Each sub-problem is further partitioned in smaller sub-sub-problems that can be solved on SPs cooperatively in parallel by all the threads within the block. The SM schedules and executes these threads in groups of parallel threads called *warps*. All the threads of a warp execute a



(a) Preprocessing (b) Simulation

Figure 5.2: **Stages of adaptive rectangular decomposition method:** (a) In the preprocessing stage, the input domain is voxelized into grid cells and adaptively decomposed into rectangular partitions. Artificial interfaces and PML absorbing layers are created between neighboring partitions and on the scene boundary respectively. (b) During the simulation stage, the simulator starts the current field and performs interface handling between neighboring partitions to compute forcing terms. Next, it transforms the forcing terms to the cosine spectral basis through DCT. These are then used to update the spectral coefficients to propagate waves within each partition. Lastly, the field is transformed back from spectral to spatial domain using IDCT to yield the updated field.

single common instruction at a time. The first or the second half of a warp is called a *half-warp*. GPU memory access pattern is based on half-warps. A parallel task is executed on the GPU by writing functions called *kernels* which are launched by the host-CPU and execute in parallel on the GPU. The GPU API provides the ability to create local and global thread barriers. In a local thread barrier, all the threads in a block must wait until every thread of the block has finished execution whereas in a global thread barrier all the threads on the GPU must wait until every thread has finished execution. The use of these barriers to synchronize the threads is called *thread synchronization* (Nickolls et al., 2008). For more details on parallel computing on GPUs, please read (NVIDIA, 2010; Kirk and Hwu, 2010; Savioja et al., 2010).

5.3 Adaptive Rectangular Decomposition

In this section, an overview of the adaptive rectangular decomposition (ARD) solver (Raghuvanshi et al., 2009b) is provided. We will also highlight its benefits over prior solvers for the acoustic wave equation for a uniform medium. The GPU-based wave equation solver is built upon the ARD solver.

5.3.1 ARD Computation Pipeline

ARD has two primary stages, *Preprocessing* and *Simulation*. In the preprocessing stage, the input scene is voxelized into grid cells at grid resolution h determined by the relation $h = \lambda_{\min}/s = c/(\nu_{\max}s)$ where $\lambda_{\min}$ is the minimum simulation wavelength, s is number of samples per wavelength, c is the speed of sound and $\nu_{\max}$ is the maximum usable simulation frequency[1]. This is followed by a rectangular decomposition step in which grid cells generated during voxelization are grouped into rectangles (see Figure 5.2(a)). These rectangles are called *air partitions*. Partitions created for the perfectly matched layer (PML) absorbing layer are referred to as *PML partitions*. PML absorbing layers are created to model both partially absorbing surfaces as well as complete absorption in open scenes. Both air and PML partitions have the same grid resolution h. Next, artificial interfaces are created between adjacent air-air and air-PML partitions. This one-time precomputation step takes 1-2 minutes for most scenes. During the simulation stage, the global acoustic field is computed in a time-marching scheme. The computation at each time-step is as follows (see Figure 5.2(b)):

1. **For all interfaces:** Interface handling to compute force f within each partition (Equation 5.8).
2. **For all air partitions:**
    (a) Discrete Cosine Transform (DCT) of force f to spectral domain $\tilde{f}$ (Equation 5.3).
    (b) Mode update for spectral coefficients $\tilde{p}$ (Equation 5.5).
    (c) Inverse Discrete Cosine Transform (IDCT) of $\tilde{p}$ to pressure p (Equation 5.3).
    (d) Normalize pressure p by multiplying it with a normalization constant.
3. **For all PML partitions:** Update pressure field.

During step 1, the coupling between adjacent partitions (air-air and air-PML) is computed to produce forcing values. In steps 2 and 3, these forcing values are used to update the pressure fields within the air and PML partitions, respectively. While air partitions are updated in the spectral domain, transforming to and from spatial domain using IDCT and DCT, PML partitions employ

[1] By maximum usable frequency of X Hz, we mean that our simulation results have no dispersion error and minimal other numerical errors till X Hz. Therefore, they can be directly used to compute impulse response for auralization and produce sound field visualization. So $\nu_{\max} = X$ kHz means that the useful range of the result is from 0 Hz till X kHz and the excitation is broadband, containing frequencies from 0 to X kHz

a finite-difference implementation of a fictitious, highly dissipative wave equation (Rickard et al., 2003) to perform absorption. DCT and IDCT steps are implemented using a generalized 3D FFT.

5.3.2 Mathematical Background

Rectangular domain In case of rectangular scenes, ARD achieves high accuracy for both spatial and temporal derivatives within rectangular volumes, nearly eliminating numerical dispersion. This is done by employing the eigendecomposition for the wave equation on rectangular domains, which, assuming spatially constant speed of sound, can be computed analytically, incurring no numerical error, as follows –

$$\begin{aligned} \nabla^2 \Phi_{i,j,k} &= -k^2_{i,j,k} \Phi_{i,j,k}, \\ \Phi_{i,j,k} &= \cos\left(\pi \frac{i}{l_x} x\right) \cos\left(\pi \frac{j}{l_y} y\right) \cos\left(\pi \frac{k}{l_z} z\right), \\ k^2_{i,j,k} &= \pi^2 \left(\frac{i^2}{l_x^2} + \frac{j^2}{l_y^2} + \frac{k^2}{l_z^2}\right). \end{aligned} \tag{5.1}$$

Note that the eigen-functions, $\Phi_{i,j,k}$, coincide with the basis functions for the 3D Discrete Cosine Transform. This follows from assuming sound-hard boundary conditions for the volume. Therefore, transformation to and from the spectral basis can be performed by leveraging memory and compute-efficient Discrete Cosine Transform (DCT) and inverse Discrete Cosine Transform (IDCT). The pressure and forcing fields are expressed in this basis as –

$$\begin{aligned} p\left(x,y,z,t\right) &= \sum_{i,j,k} \tilde{p}_{i,j,k}\left(t\right) \Phi_{i,j,k}\left(x,y,z\right), \\ f\left(x,y,z,t\right) &= \sum_{i,j,k} \tilde{f}_{i,j,k}\left(t\right) \Phi_{i,j,k}\left(x,y,z\right). \end{aligned} \tag{5.2}$$

The above equations are equivalent to –

$$\begin{aligned} \tilde{p}_{i,j,k}(t) &= DCT(p\left(x,y,z,t\right)), \\ \tilde{f}_{i,j,k}(t) &= DCT(f\left(x,y,z,t\right)), \\ p\left(x,y,z,t\right) &= IDCT(\tilde{p}_{i,j,k}(t)), \\ f\left(x,y,z,t\right) &= IDCT(\tilde{f}_{i,j,k}(t)). \end{aligned} \tag{5.3}$$

Substituting equation (5.2) into the wave equation leads to a independent set of Ordinary Differential Equations –

$$\frac{d^2\tilde{p}_{i,j,k}}{dt^2} + \omega_{i,j,k}^2 \tilde{p}_{i,j,k} = \tilde{f}_{i,j,k}, \text{ where } \omega_{i,j,k} = ck_{i,j,k}. \tag{5.4}$$

By recognizing that this is the equation of a forced simple harmonic oscillator having solutions of the form, $\tilde{p}(t) = \alpha e^{i\omega t} + \bar{\alpha} e^{-i\omega t}$ and assuming f is constant over a time-step, the following update rule is obtained (subscripts have been suppressed and are (i,j,k) for all terms) –

$$\tilde{p}^{(n+1)} = 2\tilde{p}^{(n)} \cos(\omega \Delta t) - \tilde{p}^{(n-1)} + \frac{2\tilde{f}^{(n)}}{\omega^2} (1 - \cos(\omega \Delta t)). \tag{5.5}$$

The above update rule is derived from the analytical solution by requiring time-symmetry and reversibility. In the absence of forcing terms, this scheme incurs no numerical errors.

Non-Rectangular domain For handling non-rectangular scenes with ARD, the scene's air volume is decomposed into a disjoint set of coordinate axis-aligned rectangular partitions that touch each other at artificial interfaces, as illustrated in Figure 1. Interface handling is used to ensure sound propagation between the partitions. The derivation is performed for the 1D case but this analysis extends straightforwardly to 3D all the partition boundaries, and thus interfaces, are axis aligned.

Consider two partitions in 1D, $[-\infty, 0]$ and $[0, \infty]$, with an interface lying on the origin. The boundary condition assumed for the internal solution within each partition is $\frac{\partial p}{\partial x}\Big|_{x=0} = 0$ (sound-hard boundary condition assumption), which results in full reflections from the origin. Consider the right partition: the local solution corresponds to a discrete differential operator, ∇^2_{local}, that satisfies the mentioned boundary condition. Representing the global (correct) operator by ∇^2_{global}, the acoustic wave equation can be re-written as –

$$\begin{aligned} \frac{\partial^2 p}{\partial t^2} - c^2 \nabla^2_{global} p &= f(\mathbf{x}, t), \\ \frac{\partial^2 p}{\partial t^2} - c^2 \nabla^2_{local} p &= f(\mathbf{x}, t) + f_I(\mathbf{x}, t), \\ f_I(\mathbf{x}, t) &= c^2 \left(\nabla^2_{global} - \nabla^2_{local} \right) p = c^2 \nabla^2_{res} p. \end{aligned} \tag{5.6}$$

In this way, the actual global operator ∇^2_{global} is expressed as the sum of an operator local to the partition ∇^2_{local} and a residual operator $\nabla^2_{res} = \left(\nabla^2_{global} - \nabla^2_{local} \right)$. The latter is accounted for in

the forcing term on the right hand side. At each step, the forcing term is computed as the sum of source terms $f(\mathbf{x}, t)$ and interface contributions $f_I(\mathbf{x}, t)$, and the remaining computation is identical to what was described in equations (5.1) through (5.5). All that remains is the form of the interface operator discussed above.

Interface operator

Denoting $x_i = \left(i + \frac{1}{2}\right) h$, where h is the cell size for the Cartesian grid, the forcing terms (denote $f_I(x_j, t)$ with $f_I(x_j)$) for the right partition for perfect, error-free interfacing is given by –

$$
\begin{aligned}
f_I(x_j) &= \sum_{i=-\infty}^{-1} p(x_i) s[j-i] - \sum_{i=0}^{\infty} p(x_i) s[j+i+1] \\
&\qquad \text{where } j \in [0, \infty), \\
s[i] = sinc''(ih) &= \frac{1}{h^2} \times \begin{cases} \frac{-\pi^2}{3} & i = 0 \\ (-1)^{i-1} \frac{2}{i^2} & i \neq 0, i \in Z \end{cases} \\
sinc(x) &= \begin{cases} \frac{\sin\left(\frac{\pi}{h} x\right)}{\left(\frac{\pi}{h} x\right)} & x \neq 0 \\ 1 & x = 0 \end{cases}
\end{aligned}
\tag{5.7}
$$

This exact operator is highly compute-intensive owing to its non-compact support. For partitions with N cells, its computational complexity is $O(N^2)$. Therefore, approximate interface handling is performed by using a sixth-order accurate finite-difference scheme. This leads to a compact operator, thus allowing faster computation, which is given as follows –

$$
\begin{aligned}
f_I(x_j) &= \sum_{i=j-3}^{-1} p(x_i) s[j-i] - \sum_{i=0}^{2-j} p(x_i) s[i+j+1] \\
&\qquad \text{where } j \in [0, 1, 2], \\
f_I(x_j) &= 0, j > 2, \\
s[-3...3] &= \frac{1}{180h^2} \{2, -27, 270, -490, 270, -27, 2\}.
\end{aligned}
\tag{5.8}
$$

Its computational complexity depends on the number of cells lying on the interface a) 1D : $O(1)$ b) 2D : $O(N^{1/2})$ c) 3D : $O(N^{2/3})$. This approximate operator results in low-amplitude fictitious reflections from the interface which are roughly 40 dB below the incident sound-field, thus making them inaudible (Raghuvanshi et al., 2009b). Lower errors could be obtained by optimized compact finite difference schemes, or even directly using the exact operator described above.

5.3.3 Accuracy and computational aspects

A direct performance comparison of the FDTD and ARD technique for the same amount of error is difficult since both techniques introduce different kinds of errors. Since the final goal in room acoustics is to auralize the sounds to a human listener, it is natural to set these error tolerances based on their auditory perceivability. This is complicated by the absence of systematic listening tests for perceivable errors with both FDTD and ARD. However, it is possible to compare them by assuming *conservatively low* errors with both the techniques. We briefly discuss how the parameters in both techniques are set for keeping the errors conservatively low and then present a theoretical comparison to motivate why ARD is more compute and memory efficient than FDTD.

In recent work, (Sakamoto et al., 2008) show that FDTD calculations of room-acoustic impulse responses on a grid with $s = 6 - 10$ agree well with measured values on a real scene in terms of room acoustic parameters such as reverberation time. Remember that grid size $h = \lambda_{\min}/s$. This mesh resolution is also commonly used with the finite difference method applied to electromagnetic wave propagation to control phase errors resulting from numerical dispersion (Taflove and Hagness, 2005) . Motivated from these applications, the mesh resolution for the FDTD is set conservatively at $s = 10$ throughout this paper, assuming that this safely ensures that numerical dispersion errors are inaudible in auralizations. ARD results in fictitious reflection errors at the artificial interfaces. As shown by Raghuvanshi et al. (Raghuvanshi et al., 2009b), using $s = 2.6$ with ARD, the fictitious reflection errors can be kept at a low level of -40 dB average over the whole usable frequency range by employing a sixth-order finite difference transmission operator. This means that for a complex scene with many interfaces, the global errors stay 40 dB below the level of the ambient sound field, rendering them imperceptible as demonstrated in the auralizations (Raghuvanshi et al., 2009b,a, 2010). Therefore, we assume sampling of $s = 2.6$ for ARD.

Table 5.1 shows the performance and memory comparison of FDTD and ARD. The update cost for sixth-order accurate FDTD in 3D is about 55 FLOP per cell per step including the cost of PML boundary treatment for a stencil width of 7. The total cost for ARD per step can be broken down as: DCT and IDCT (assuming a DCT and IDCT take $2N \log_2 N$ FLOP count each) $= 4N \log_2 N$, mode update $= 9N$, interface handling $= (300 \times 6N^{2/3})$ and PML boundary treatment $= (390 \times 6N^{2/3})$. The $6N^{2/3}$ term approximates the surface area of the scene by that of a cube with equivalent volume.

	s	# cells $N = V/h^3$	# steps $S = t/\Delta t$	FLOP count (TeraFLOP)	Total FLOP count (TeraFLOP)	FLOP per cell per step
FDTD	10	254 M	17000	FDTD : 221.55, PML : 15.95	∼**237**	55
ARD	2.6	4.5 M	4500	Interface : 0.22, DCT+IDCT : 1.79, Mode update : 0.18, PML : 0.29	∼**2.5**	120

Table 5.1: Floating-point operation (FLOP) count comparison of FDTD vs ARD on a scene of volume $V = 10,000m^3$ with maximum usable frequency $\nu_{max} = 1$ kHz (minimum wavelength $\lambda_{\min} = c/\nu_{\max} = 34cm$) for the simulation of duration $t = 1$ sec. The number of cells (in Millions M) with either technique is given by $N = V/h^3$ where $h = \lambda_{\min}/s$ is the grid size and s is number of samples per wavelength. The simulation time-step is restricted by the CFL condition $\Delta t \leq h/c\sqrt{3}$ with smaller cell sizes requiring proportionally smaller time-steps. Theoretically, ARD which uses $s = 2.6$ is nearly hundred times more compute efficient and 50 times more memory efficient than FDTD ($s = 10$) on account of using a much coarser grid. "FLOP per cell per step" is defined as the ratio of the total FLOP count and the total number of cells N times the number of steps S.

Due to the cartesian grid, this estimate is the lower bound of the surface area. PML boundary treatment cost per cell is the same for both FDTD and ARD. As can be seen in the table, theoretically ARD is nearly 100 times more compute efficient and 50 times more memory efficient than FDTD. In practice, the CPU-based ARD is $50 - 75$ times faster than FDTD implementation. Since ARD is highly memory efficient, an order of magnitude more than FDTD, this makes it possible to perform simulations on much larger scenes than FDTD without overflowing main memory or GPU memory. GPUs can easily become memory-bound, in which case the performance is dictated by memory bandwidth rather than the FLOP numbers. In these cases as well, ARD, on account of being more memory efficient, is more suitable for the GPUs.

5.4 GPU-based ARD Solver

In previous sections, the computational efficiency and mathematical background of the ARD technique were discussed. In this section, we describe the parallel GPU-based acoustic solver built on top of ARD. We discuss the key features of the approach and some of the issues that arise in parallelizing it on many-core GPU architecture.

5.4.1 Our GPU approach

Two levels of parallelism The ARD technique exhibits two levels of parallelism (a) a *coarse-grained* and (b) a *fine-grained*. Coarse grained parallelism is due to the fact that each of the partitions

(air or PML) solves the wave equation independently of each other. Therefore, each partition can be solved in parallel at the same time. Fine grained parallelism is achieved because within each partition all the grid cells are independent of each other with regards to solving the wave equation at a particular time-step. For solving the wave equation at the current time-step, a grid cell may use $p, f, \tilde{p}, \tilde{f}$ values of its neighboring cells computed at the previous time-step but is completely independent of their $p, f, \tilde{p}, \tilde{f}$ values at the current time-step. In other words, because ARD uses explicit time-stepping, there is no need for solving a linear system. Therefore within each partition, all the grid cells can run in parallel exhibiting fine grained parallelism. The GPU-based acoustic solver exploits both these levels of parallelism. It launches as many tasks in parallel as there are partitions. Each task is responsible for solving the wave equation for a particular partition. Within each task, each grid cell corresponds to a thread and the solver creates as many threads as the number of grid cells in that partition. All these threads are grouped into blocks and scheduled by the runtime environment on the GPU.

Avoiding host-device data transfer bottleneck The host-device data link between CPU and GPU via PCI express or Infiniband, is a precious resource that has a limited bandwidth. Many prior GPU-based numerical solvers were based upon the hybrid CPU-GPU design where few steps of the algorithm were performed in the GPU and the remaining on the CPU. This design suffers from data-transfer bottleneck as it has to transfer large amounts of data between host (CPU) and device (GPU) at each simulation step. The GPU-based solver ensures that the data-transfer between the CPU-host and GPU-device is minimal. In this case, the hybrid CPU-GPU approach is avoided and instead the entire ARD technique is parallelized on the GPU. The only host-device data transfer that is required is for storing the pressure grid p after each simulation step. Recent work on interactive auralization has shown that storing and processing the simulation results on a subsampled spatial grid (every fourth, eighth or sixteenth sample), can be used for convincing auralizations for moving sources and listener, after interpolation (Raghuvanshi et al., 2010). This results in a memory reduction by a factor of $1/4^3, 1/8^3, 1/16^3$ of the original size respectively, resulting in negligible overall cost for transferring the simulation results from GPU to CPU.

To provide an intuition of host-device data transfer, consider a room of air volume $10,000m^3$ for which the wave equation needs to be solved at $\nu_{\max} = 2$ kHz. A hybrid CPU-GPU system

by (Raghuvanshi et al., 2009a) is considered where only the DCT/IDCT steps of the technique are parallelized on GPU. In this case, at each time-step, the grid f is transferred from CPU to GPU for DCT, $\tilde{f}$ is returned back by the GPU, $\tilde{p}$ is transferred from CPU to GPU for IDCT and the final pressure p is returned to the CPU. An important point to note here is that the $p, f, \tilde{p}, \tilde{f}$ grids cannot be subsampled and transferred in this hybrid CPU-GPU system because the steps of the algorithm that reside on the CPU and GPU require the values on the complete grid to solve the wave equation. Since the size of $p, f, \tilde{p}, \tilde{f}$ is equal to the number of grid cells, the total data transfer cost is 4 x # grid cells x sizeof(*float*) $= 4V\left(\frac{s\nu_{\max}}{c}\right)^3$ x 4 bytes $= 4\text{x}10000\text{x}\left(\frac{2.6\text{x}2000}{340}\right)^3$ x 4 bytes = 145 MB **per time-step**. On the other hand, in the proposed technique, since all the computational steps are performed on the GPU, $p, f, \tilde{p}, \tilde{f}$ grids need not be transferred to the CPU for the purpose of the simulation. The only transfer that is required is of the subsampled pressure grid p from GPU to CPU for storage on the disk, perhaps for auralization later. For visualization applications, one might not need to perform *any* transfer at all because the data is already present on the GPU and can be displayed directly to the screen. As explained above, for the purpose of auralization, the subsampling of pressure grid is usually done at a lower resolution ($1/8^3$). Thus, the data transfer is = $1/8^3$ x # grid cells x 4 bytes = 3 kB per time-step. For such a small size, data-transfer is almost immediate (< 1 msec).

Computationally optimal decomposition Rectangular decomposition proposed by (Raghuvanshi et al., 2009b) uses a greedy heuristic to decompose the voxelized scene into rectangular partitions. Specifically, they place a random seed in the scene and try to find the largest fitting rectangle that can be grown from that location. This is repeated until all the free cells of the scene are exhausted. The cost of DCT and IDCT steps implemented using FFT depends on the number of grid cells in each partition. FFT operations are known to be extremely efficient if the grid dimensions are powers of 2. The proposed heuristic may produce partitions with irregular dimensions of the grids (not necessarily powers of 2) significantly increasing the cost of the DCT and IDCT operations.

In the GPU-ARD solver, a new approach is used to perform the rectangular decomposition that takes into account the computational expenditure of FFTs and its efficiency with powers of 2. Specifically, while performing rectangular decomposition, it imposes the constraint that the dimensions of the grid in each partition should be a power of 2. Similar to the original approach, it tries to fill the largest possible rectangle that could fit within the remaining air volume of the

scene. But instead of using the rectangle directly, it shrinks the size of rectangle in each dimension to the nearest power of 2, and declares the remaining cells as free. This step is repeated until all the free cells of the scene are exhausted. This increases the efficiency of the FFT computations and results in a speedup of 3 times in the running time of DCT and IDCT steps. For typical scenes, this optimized rectangular decomposition approach produces a higher number ($2-3$ times) of rectangular partitions, but since the total number of grid cells in the entire volume of the domain remains the same ($N = V/h^3$), it does not increase the FLOP count of the ARD computational steps except for the interface handling step. Since more partitions result in larger interface area, the interface handling cost increases by 25-30%. But since on the CPU, DCT and IDCT are the most time-consuming steps of the ARD technique compared to the cost of interface handling (Figure 5.6(a): CPU time), the gain achieved by faster powers-of-two DCT and IDCT far outweighs this higher interface handling cost.

5.4.2 Details

Among ARD's two main stages, the pre-processing is performed only once in the beginning and its contribution to the total running time is negligible ($1-2$ minutes) compared to the cost of the simulation step. Therefore, this stage is kept on the CPU itself and the simulation stage is parallelized on the GPU. Thus, the voxelization and rectangular decomposition is performed on the CPU. Once the rectangular partitions are available, the pressure p, force f, spectral pressure $\tilde{p}$ and spectral force $\tilde{f}$ data-structures are created on the GPU. The simulation stage has 6 main steps (see Section 5.3.1) and each of them is performed one after the another. The parallelization of all these steps on the GPU is discussed in detail.

Interface handling This step is responsible for computing forcing terms f at the artificial interfaces between air-air and air-PML partitions. These forces account for the sound propagation between partitions by applying a finite-difference stencil (Equation (5.8)). The overall procedure consists of iterating over all interfaces, applying the finite difference stencils to compute forcing values, and additively accumulating them at the affected cells. This step is data parallel – to compute the forcing term at a cell, only values in its spatial neighborhood are read. Thus, all interfaces could potentially be processed in parallel as long as there are no collisions and no two interfaces update the forcing value at the same cell. This can happen at corners (as shown below).

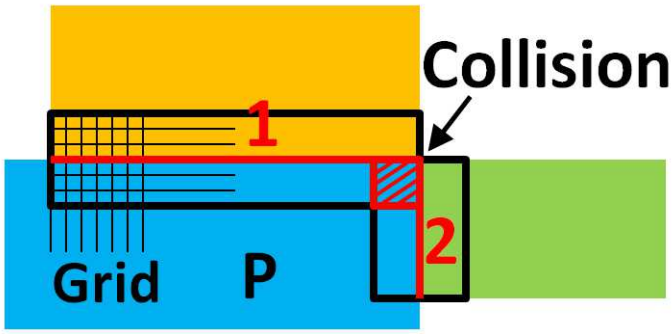


Figure 5.3: Interfaces 1 and 2 update forcing values of cells lying in their neighboring partitions. There is a Concurrent Write (CW) hazard in the hatched corner region (labeled "Collision ").

Interfaces 1 and 2 both update the forcing values 3-cells deep of their shared partitions. However, for partition P, cells lying in the hatched region (marked "Collision") are updated by *both* interfaces 1 and 2. These corner cases need to be addressed to avoid race conditions and concurrent memory writes. The GPU and its runtime environment places the burden of avoiding concurrent write hazards on the programmer. Fortunately, collisions can be avoided completely by using a conceptually simple technique. All interfaces are grouped into 3 batches consisting of interfaces with their normals in the X, Y and Z directions, respectively. Since all partitions are axis-aligned rectangles, every interface has to fall into one of these batches. By processing all interfaces within each batch in parallel and separating batches by a synchronization across all threads, all collisions in the corners are completely avoided. This approach is more general and well-supported on all GPUs as compared to using GPU-atomic operations.

DCT(f) The DCT step converts the force f from the spatial domain to the spectral domain $\tilde{f}$. DCTs are efficiently computed using FFTs. Typical FFT libraries running on GPU are an order of magnitude faster than optimized CPU implementations (Govindaraju et al., 2008). Since DCT and IDCT steps are among the slowest steps of the ARD technique (see Figure 5.6(a)), parallelization of these steps results in a great improvement in the performance of the entire technique.

Mode update $\tilde{p}$ The Mode update step uses the pressure and force in the spectral domain $\tilde{p}$, $\tilde{f}$ of the previous time-step to calculate $\tilde{p}$ at the current time-step. This step consists of linear combinations of $\tilde{p}$, $\tilde{f}$ terms (see Equation 5.5) and is highly parallelizable[2].

[2] By highly parallelizable, we mean that there should be no dependence between the threads, each thread has a very local and small memory access pattern and all of them can be computed in parallel.

Pressure normalize p This step multiplies a constant value to the pressure p, which is also highly parallelizable.

IDCT($\tilde{p}$) This step converts the pressure in the spectral domain $\tilde{p}$ back to pressure in spatial domain p. Similar to DCTs, the IDCTs are also efficiently computed using FFTs on GPU.

PML absorption layer The PML absorbing layer is responsible for sound absorption by the surfaces and walls of the 3D environment. It is applied on a 5-10 cell thick partition depending on the desired accuracy (Rickard et al., 2003; Raghuvanshi et al., 2009b). A 4th order finite-difference stencil (5 cell thickness) is used for PML computation (see Equation 2.3). Based upon the distance of the grid cell from the interface, PML performs different computations for different grid cells. Due to this, there are a lot of inherent conditionals in the algorithm. An efficient implementation of PML depends on minimizing the effect of these conditionals, as discussed in the next section.

5.4.3 Optimization

The performance of the GPU-based ARD algorithm described above can be improved by means of the following optimizations:

Batch processing Interface handling, DCT, IDCT, Mode update and Pressure normalize steps form the main components of the GPU-based solver, where each step corresponds to a GPU-function called kernel. Kernels are functions that are executed in parallel on the GPU. To run simulation steps on all the partitions and interfaces, one possible way is to launch a new kernel for each individual air partition, PML partition and interface. In typical scenes, there are thousands of partitions and interfaces (see Table 5.2). Since each kernel launch has an associated overhead, launching thousands of kernels can have a drastic impact on the overall runtime. To avoid this overhead, partitions and interfaces are grouped together into independent groups (also called *batches*) and a kernel is launched for each batch, also called *batch processing*. Therefore, instead of launching $P + I$ kernels where P is the number of partitions and I is the number of interfaces, as many kernels are launched as there are the number of batches. This grouping of partitions into batches depends on the number of independent groups that can be formed. If all the partitions are independent, they can be grouped into a single batch. For DCT and IDCT kernels, partitions are grouped into batches by using the BATCH FFT scheme of the GPU-FFT library (Govindaraju et al., 2008). Mode update and Pressure normalize

steps have no dependency between different partitions, and are grouped in a single batch resulting in just one kernel launch each. For the PML step also, the PML partitions can be grouped into a single batch and a single kernel can be launched. But to minimize the effect of conditionals, more than one kernel is launched, as discussed later. For interface handling, the interfaces are grouped into 3 separate independent batches as discussed in Section 5.4.2. A kernel launch for each batch is followed by a call to synchronize all the threads.

Maximizing coalesced memory access The global memory access pattern of the GPU can have a significant impact on its bandwidth. GPU accesses memory in group of threads called a half-warp. Global memory accesses are most efficient when memory accesses of all the threads of a half-warp can be *coalesced* in a single memory access. The $p, f, \tilde{p}, \tilde{f}$ data-structures and their memory access patterns for the Mode update and Pressure normalize kernels are organized in a way such that each thread of index i accesses these data-structures at position i itself. Thus, the memory access pattern of a half-warp is perfectly coalesced. DCT and IDCT kernels are based upon the FFT library (Govindaraju et al., 2008) which also uses memory coalescing. The PML kernel for thread i accesses memory at locations $\alpha + i$ where α is constant. This type of access results in a coalesced memory access on current generation GPUs (NVIDIA, 2010). The interface handling step can access p, f from many partitions and therefore achieving coalesced memory access for this kernel is difficult.

Minimizing path divergence The impact of conditionals (if/else statements) on the performance of a GPU kernel can be very severe. The PML step has conditionals that are based upon the distance of the grid cells from the interface and special cases like outer edges and corners. In the implementation, special care is taken to minimize the effect of conditional branching. Instead of launching a single kernel with conditional branching, separate small kernels are launched corresponding to different execution paths of the code. The number of different execution paths is limited and can be reformatted in 2-3 unique paths. Therefore, the increase in the number of kernel launches is minimal (2 or 3). These additional kernel launches do not adversely impact the performance.

5.5 Implementation

The original CPU-based ARD solver used a serial version of the FFTW library for computing DCT and IDCT steps. The CPU code uses two separate threads - one for air partitions and the other for

Scene	Air/Total Volume (m^3)	$\nu_{\max}$ Hz	# partitions (air+pml)	# cells (air+pml)
L-shaped room	6998/13520	1875	424+352	(22+5)M
Cathedral	7381/15120	1650	6130+12110	(16+6)M
Walkway	6411/9000	1875	937+882	(20+6)M
Train station	15000/83640	1350	3824+4945	(17+8)M
Living room	5684/7392	1875	3228+4518	(18+5)M
Small room	124/162	7000	3778+5245	(20+5)M

Table 5.2: "Total volume" is volume of the bounding box of the scene whereas "Air volume" is volume of the air medium in which the simulation is performed. $\nu_{\max}$ is the maximum usable simulation frequency. Number of partitions counted are generated using our computationally optimal decomposition. Number of pressure values updated at each time-step is equal to the number of grid cells (in millions M).

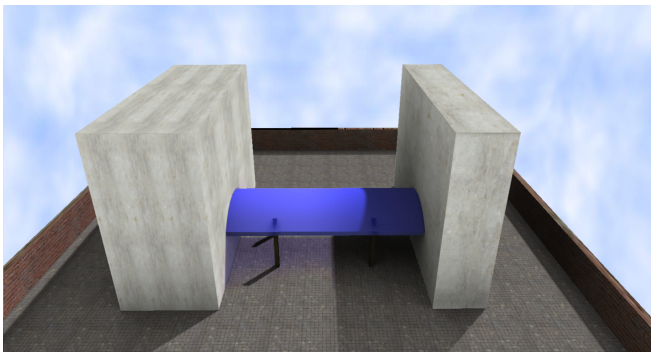
PML partitions, and performs both these computations in parallel. For simplicity of comparison with our GPU-based implementation, the sequential performance of the CPU-based solver is measured by using only a single thread. The CPU-based ARD code has been demonstrated to be sufficiently accurate in single precision (Raghuvanshi et al., 2009a,b). Since the calculations performed in the GPU-based solver are the same as the CPU-based solver, the results of the GPU-based solver match the CPU-based solver up to single-precision accuracy. The GPU-based solver was implemented using NVIDIA's parallel computing API, CUDA 3.0 with minimum compute capability 1.0. The following compiler and optimization options are used for the GPU code –

`nvcc CUDA_v3.0 : Maximize Speed (`$/O2$`).`

The DCT and IDCT kernels are based upon the FFT library developed by Govindaraju et al. (Govindaraju et al., 2008). The CUDA routine `cudaThreadSynchronize()` is used for synchronizing threads.

(a) Cathedral (35m x 16m x 27 m)

(b) Train station (34m x82m x 30m)



(c) Walkway (30m x 30m x 10m)

(d) Living room (22m x 28m x 12m)

Figure 5.4: The performance of the GPUARD technique is demonstrated on the following scenes.

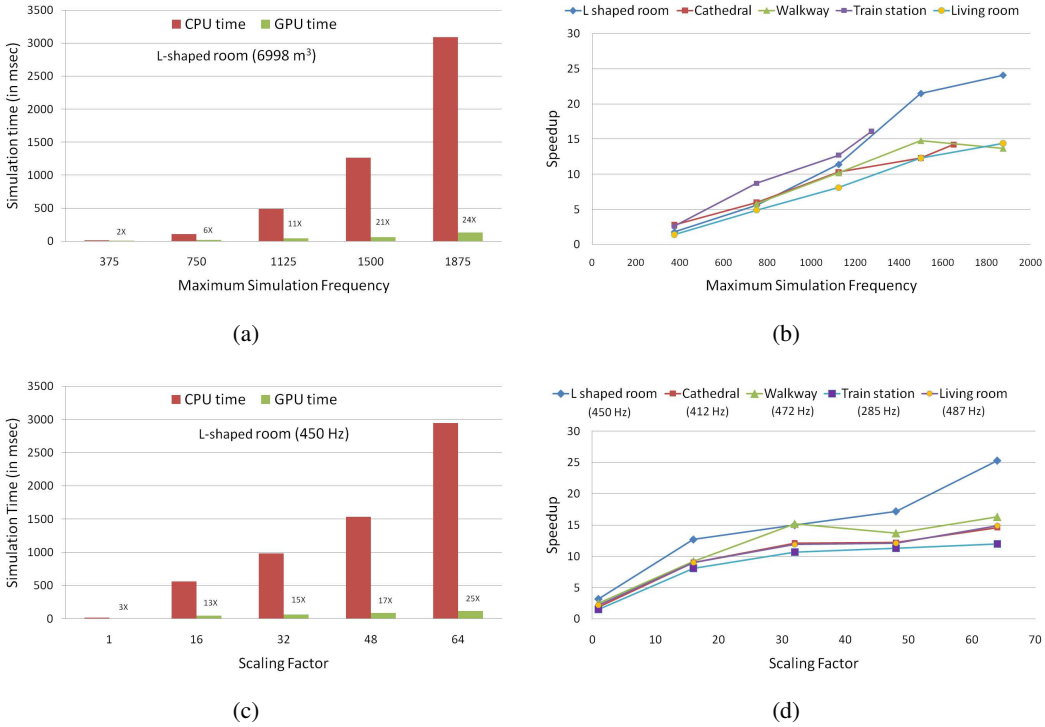


Figure 5.5: The performance of ARD solver is investigated with varying $\nu_{\max}$ and scene volume. a) Simulation time per time-step of CPU-based and GPU-based ARD solver with varying $\nu_{\max}$ for the L-shaped room scene. Note that the GPU-based solver is 24 times faster at highest $\nu_{\max}$. b) Speedup (=CPU time/GPU time) achieved by the GPU-based ARD solver over the CPU-based solver with varying $\nu_{\max}$ for the different test scenes. For higher $\nu_{\max}$, a speedup of $15-25$ times is achieved. c) Simulation time per time-step of both ARD solvers with varying scene volume for L-shaped room scene. The original volume of the test scenes was scaled by the *Scaling Factor*. Note that the GPU-based solver is 25 times faster at highest scaling factor. d) Speedup (=CPU time/GPU time) achieved by the GPU-based ARD solver over the CPU-based solver with varying scene volume for the different test scenes. As the scene volume increases, a higher speedup is achieved. For 64 times the original volume, the speedup becomes $12-25$ times.

5.6 Results

The GPU-based solver is compared with the well-optimized CPU implementation provided by the authors of the ARD (Raghuvanshi et al., 2009b). NVIDIA Geforce GTX 480 graphics card was used with a core clock speed of 700 MHz, graphics memory of 1.5 GB with 480 CUDA processors (also called *cores*). CPU timings are reported for an Intel Xeon X5560 (8M Cache, 2.80 GHz) machine. A single core for the CPU-based implementation was employed. Timings are reported by running the simulation over 100 time-steps and taking the average. Overall, 5 benchmark scenes were used varying in both size and complexity (see Table 5.2 and Figure 5.4).

In Figure 5.5(a), the performance of the CPU-based solver is compared with our GPU-based solver on the L-shaped room benchmark with varying $\nu_{\max}$. Figure 5.5(b) shows the speedup achieved by our GPU-based solver on different benchmarks. For smaller frequencies, the amount of work available is considerably less resulting in under-utilization of GPU and nominal speedup. But for higher frequencies[3], all the cores of the GPU are fully utilized. The GPU-based solver becomes a lot faster and outperforms its CPU counterpart by a factor of $15 - 25$ times on different scenes. The performance of the GPU-based solver is analysed with varying scene volume. The volume of the benchmark scenes are scaled uniformly in the range of $1 - 64$ times. In Figure 5.5(c), it is observed that as the amount of work increases with increasing scene volume, the performance of the GPU-based solver scales better. Speedup achieved by the GPU-based solver for varying scene volume also shows a similar behavior (see Figure 5.5(d)). As the scaling factor reaches 64 times, the speedup of $12 - 25$ times is achieved on different scenes. For simple scenes like the L-shaped room, rectangular decomposition gives fewer air partitions (see Table 5.2 column 4) resulting in fewer DCT and IDCT batches. Since each batch corresponds to a kernel call, fewer batches mean fewer kernel calls reducing the total overhead of kernel launches. Fewer batches also mean that an individual batch is of larger size. For each batch, the GPU gets fully utilized and the DCT and IDCT kernels based on GPU-FFT are much more efficient resulting in higher speedups for simpler scenes.

Figure 5.6(a) shows the breakdown of the time spent on various steps of the simulation stage. In the original CPU-based ARD solver, the DCT/IDCT and the PML steps heavily dominate the computation time. But for the GPU-based solver, as can be seen, all the steps of the simulator are more or less balanced except Mode update, Pressure normalize and PML, whose costs become negligible compared to other steps. The DCT and IDCT kernels implemented using the FFT library (Govindaraju et al., 2008), give a speedup of 14 times on the GPU. PML boundary treatment, Mode update and Pressure normalize achieve a higher speedup of 30 times, 28 times and 16 times respectively. The last stage of ARD, interface handling, involves a lots of uncoalesced memory accesses resulting in a nominal speedup of 3 times. But since the contribution of interface handling to the overall running time is far less than DCT/IDCT steps, it does not become a bottleneck.

[3] The amount of work increases with increasing frequency (number of grid cells $N \propto {\nu_{\max}}^3$).

The scalability analysis is performed on the GPU-based solver for four different NVIDIA GPUs with different number of CUDA cores : GeForce 9600M GT, GeForce 8800GTX, Quadro FX 5800 and Geforce GTX 480, each with 32, 128, 240 and 480 CUDA cores respectively. Figure 5.6(c) and 5.6(d) shows the performance of the solver on the cathedral and the small room scene as the number of CUDA cores increase. As can be seen, the GPU-based solver scales linearly with the number of cores. Increasing the number of CUDA cores 4 times from 32 to 128 results in a speedup of $3-4$ times, from 32 cores to 240 cores (7.5 times) gives $7-7.5$ times speedup and from 32 to 480 cores (15 times) gives a speedup of $14-15$ times. As the amount of work increases with increasing $\nu_{\max}$, the performance scaling becomes perfectly linear. This shows that the GPU-based ARD solver is compute-bound rather than limited by memory bandwidth. In the future, as GPU's continue their super-Moore's law growth (Owens et al., 2007; Govindaraju et al., 2006), the GPU-based solver will exhibit super-exponential performance improvement.

A performance comparison of CPU-based FDTD solver, CPU-based ARD solver and our GPU-based ARD solver is performed with varying $\nu_{\max}$. The CPU-based FDTD solver is based upon the FDTD work proposed by Sakamoto et al. (Sakamoto et al., 2008). As can be seen in Figure 5.6(b), the CPU-based solver achieves a maximum speedup of $50-75$ times over the CPU-based FDTD solver. The GPU-based ARD solver achieves a speedup of over 1100 times over the CPU-based FDTD solver for the same scene. Since FDTD runs out of memory for $\nu_{\max} > 3750Hz$, the timings below $3750Hz$ were used to calculate the projected timings for FDTD above $3750Hz$ (using the relation that the simulation time varies as fourth power of $\nu_{\max}$).

5.7 Conclusion

In this chapter, we have presented an efficient GPU-based time-domain solver for the acoustic wave equation. This solver provides more than three orders of magnitude improvement over prior FDTD-based solvers. Moreover, the use of GPUs can accelerate the computation by more than an order of magnitude as compared to the CPU-based ARD solver. We have also shown that the technique scales linearly with the number of GPU processors.

Limitations and Future Work The approach has some limitations. The current implementation assumes that the entire spatial decomposition fits into GPU memory and is based on single precision

arithmetic. In terms of future work, given a reformulation of the BEM-FMM solution technique in the time-domain, a very interesting possibility would be to combine the ARD approach with BEM-FMM – utilizing FMM based solutions for partitions with large volume and the current domain-based ARD method for smaller partitions. Comparing detailed impulse response measurements of full-sized 3D concert halls against wave-based numerical simulation is a very new and exciting method of investigation, which has opened up because of the increased computational power and memory on today's computers. The present work opens up the possibility of doing such detailed comparisons on a desktop computer in the mid-high frequency range (1-4 kHz) in the near future, along with visualizations of the propagating wavefronts. It would also be interesting to apply this approach to more complex acoustic spaces such as CAD models and large outdoor scenes, and extend it to multi-GPU clusters as well.

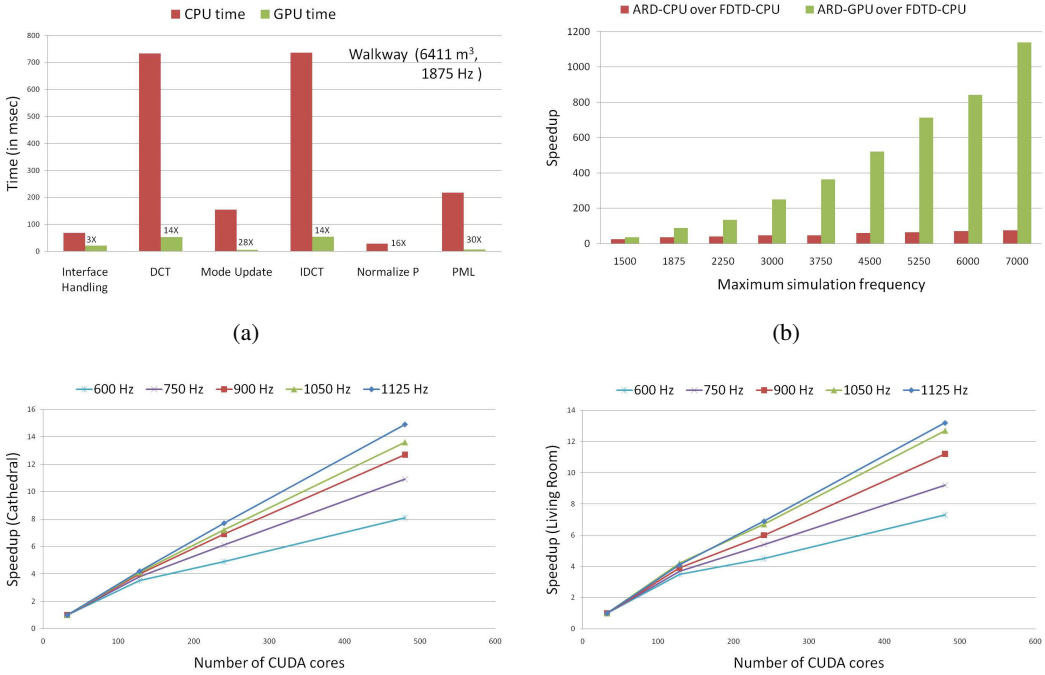


Figure 5.6: a) Simulation steps - Interface handling, DCT, Mode update, IDCT, Pressure normalize and PML, and the corresponding time spent in the CPU-based and GPU-based ARD solver for the Walkway scene. Speedups achieved by individual steps of the GPU-based ARD over the CPU-based simulator - PML(30 times), Mode update(28 times), Pressure normalize(16 times), DCT(14 times), IDCT(14 times) and interface handling(3 times). b) The speedup achieved by CPU-based and the GPU-based ARD solver over CPU-based finite-difference time-domain (FDTD) solver is plotted with varying $\nu_{\max}$ for the small room benchmark scene. The CPU-based FDTD solver is based upon the work proposed by Sakamoto et al. (Sakamoto et al., 2008). The CPU-based ARD solver achieves a maximum speedup of 75 times over CPU-based FDTD whereas the GPU-based ARD solver achieves a maximum speedup of 1100 times. c & d) Simulations were run on 4 different NVIDIA GPU's with different number of CUDA processors (also called *cores*) - GeForce 9600M GT (32 cores), GeForce 8800GTX (128 cores), Quadro FX 5800 (240 cores) and Geforce GTX 480 (480 cores). Speedup on GPU with X cores = (Simulation time on 32-cores GPU)/(Simulation time on X-cores GPU). Linear scaling in performance is achieved at higher values of $\nu_{\max}$.

CHAPTER 6: VALIDATION

6.1 Introduction

In this Chapter, we discuss the validation of the three techniques proposed in this dissertation: (a) Equivalent Source Method (Chapter 3), (b) Directional Sources and Spatial Audio framework (Chapter 4), and (c) Parallel Adaptive Rectangular Decomposition (Chapter 5), by comparing their results with the state-of-art numerical solvers and real-world measurements.

In case of the equivalent source technique, pressure fields generated by this technique are compared with those generated by the boundary element method (BEM) and fast-multipole boundary element method over a grid of listener positions at different frequencies. The spectral extrapolation employed by this technique is also validated by comparing the results with a wide-band signal generated by the Biot-tolstoy-Medwin (BTM) technique.

For directional sources and the spatial audio framework, the spherical harmonic-based source modeling is validated by increasing the order of the spherical harmonics and computing the error between the measured directivity function and the spherical harmonic approximation. The source directivity framework has been integrated with both the state-of-the-art BEM technique and the interactive ESM technique. The pressure fields generated by both the techniques for the same scenario are compared to each other at a grid of listener positions at different frequencies. Also, the audio generated by this technique is validated by comparing the results with the audio generated by the BTM technique at different source-listener positions.

Finally, for the parallel ARD technique, the validation is performed in a two step process: (a) by comparing the parallel ARD results in scenarios that have analytical solutions (b) by comparing the parallel ARD results with real-world measurements.

6.2 Background

Analytical Solutions of the Acoustic Wave Equation The wave equation has analytical solutions for certain simple cases. such as spherical wave scattering by a rigid sphere (Hasegawa et al., 1980). The spherical wave is produced by a monopole source. The sphere acts as a scatterer with a rigid boundary condition (i.e. normal velocity $= 0$). The derivation of the analytical solution is based on (ANSOL, 2014)[1]. The analytical solution can be derived as follows:

Assume a rigid sphere of radius a centered at the origin $(0, 0, 0)$ and a monopole sound source placed at a point $\mathrm{r}_s = (r_s, \theta_s, \phi_s)$. The sound field p_{inc} produced by the source at a point $\mathrm{r} = (r, \theta, \phi)$ is

$$p_{inc} = -\varpi \, G_\infty(\mathrm{r} - \mathrm{r}_s),$$

where $G_\infty = \frac{e^{ikR}}{4\pi R}$ is the free-space Green's function and $R = \|\mathrm{r} - \mathrm{r}_s\|_2$. Expanding the free-space Green's function in spherical harmonics, we get

$$p_{inc} = \frac{ik}{2\pi} \sum_{l=0}^{\infty} \sum_{m=0}^{+l} \epsilon_m \cos m(\phi - \phi_s) \; p_l^m(\cos\theta_s) \; p_l^m(\cos\theta) \; j_l(kr_<) h_l(kr_>), \tag{6.1}$$

where j_l is the spherical Bessel function, h_l is the spherical Hankel function of first kind, p_l^m is the normalized associate Legendre function of the first kind, $i = \sqrt{-1}$, and,

$$\begin{aligned} r_< &= \min(r, r_s) \\ r_> &= \max(r, r_s) \\ \epsilon_m &= \begin{cases} 1 & m = 0 \\ 2 & m \neq 0 \end{cases} \end{aligned}$$

The scattered pressure field p_{sca} of the rigid sphere should satisfy both the Helmholtz equation and Sommerfeld radiation condition, and therefore, can be represented as:

$$p_{sca} = \sum_{l=0}^{\infty} \sum_{m=0}^{+l} h_l(kr) \; p_l^m(\cos\theta) \; \left[A_{l,m} \cos m\phi + B_{l,m} sinm\phi\right],$$

[1] There are certain mathematical inconsistencies in that solution which have been fixed here.

where $A_{l,m}$ and $B_{l,m}$ are the coefficients.

The coefficients of the scattered field can be solved by satisfying the rigid boundary condition at the surface of the sphere:

$$\frac{\partial p_{inc}}{\partial r} + \frac{\partial p_{sca}}{\partial r} = 0, \quad \text{at } r = a$$

The coefficients are:

$$\begin{aligned} A_{l,m} &= \epsilon_m \varpi \frac{ik}{2\pi} \left[\frac{l j_{l-1}(ka) - (l+1) j_{l+1}(ka)}{l h_{l-1}(ka) - (l+1) h_{l+1}(ka)} \right] \cos m\phi_s \, p_l^m(\cos\theta_s) \, h_l(kr_s) \\ B_{l,m} &= \epsilon_m \varpi \frac{ik}{2\pi} \left[\frac{l j_{l-1}(ka) - (l+1) j_{l+1}(ka)}{l h_{l-1}(ka) - (l+1) h_{l+1}(ka)} \right] \sin m\phi_s \, p_l^m(\cos\theta_s) \, h_l(kr_s) \end{aligned}$$

Substituting the values of $A_{l,m}$ and $B_{l,m}$, we get the scattered field

$$p_{sca} = \varpi \frac{ik}{2\pi} \sum_{l=0}^{\infty} \sum_{m=0}^{+l} \epsilon_m \left[\frac{l j_{l-1}(ka) - (l+1) j_{l+1}(ka)}{l h_{l-1}(ka) - (l+1) h_{l+1}(ka)} \right] \cos m(\phi - \phi_s) \, p_l^m(\cos\theta_s) \, p_l^m(\cos\theta) \, h_l(kr_s) h_l(kr) \tag{6.2}$$

The total sound field is the sum of incident and scattered pressure fields:

$$p_{total} = p_{inc} + p_{sca} \tag{6.3}$$

Dynamic Time Warping (DTW) This is a standard technique used in the signal processing community for pattern matching between two time-domain signals which can handle non-linear transformations of the time axis. It has been widely used in the areas of speech processing (Sakoe and Chiba, 1978), acoustics (Masterson et al., 2009), bioinformatics (Aach and Church, 2001), and medicine (Caiani et al., 1998). The main idea behind this technique is to use dynamic programming to warp the time axis of one signal over another to minimize the difference between them. This technique has two variations– symmetric and asymmetric. In the symmetric DTW, the time axes of both the signals are transformed into a temporarily defined time axis. In contrast to this, the asymmetric DTW transforms the time axis of one signal onto that of another.

Given two time signals, expressed as discrete samples, of length I and J:

$$\begin{aligned} A &= a_1, \, a_2, \, ----, a_i, ----, a_I \\ B &= b_1, \, b_2, \, ----, b_j, ----, b_J \end{aligned}$$

The mapping between these two signals can be depicted by a sequence of points $c = (i, j)$ as:

$$F = c(1),\ c(2),\ ----,\ c(k), ----, c(K),$$

where $c(k) = (i(k), j(k))$. This sequence maps the time axis of signal A onto the signal B and is therefore called the warping function.

The difference between the two signals after applying the warping F becomes

$$E(F) = \sum_{k=1}^{K} d(c(k))\ w(k),$$

where $d(c(k)) = d(i(k), j(k)) = \|a_{i(k)} - b_{j(k)}\|$ and $w(k)$ is the weight function.

The DTW algorithm uses a dynamic programming scheme to find the optimal warping function that minimizes the difference between the two signals. Restrictions such as monotonicity, continuity, adjustment window, and slope constraints, are applied on the warping function to constrain the solution to the feasibility space of the application. Too small or too large a warping may not be physically feasible for the application. For more details on Dynamic Time Warping, readers are referred to the seminal paper by (Sakoe and Chiba, 1978).

6.3 Validation of Equivalent Source Method

In this section, validation results for the equivalent source technique proposed in Chapter 3 are presented.

6.3.1 Validation of Pressure Fields

Figure 6.1 compares the results of the equivalent source technique with that of the reference wave solvers: BEM (Cheng and Cheng, 2005) and FMM-BEM (Liu et al., 2009; Gumerov and Duraiswami, 2009), on a spatial grid of listeners at different frequencies. The scene considered for this experiment is the two parallel walls scene (Figure 3.6). Even though the scene looks simple geometrically, it is acoustically complex due to the presence of multiple orders of interactions (reflections, diffractions, etc) happening between the two walls.

The sound source is positioned to the left of the first wall. For each frequency, the per-object and inter-object transfer functions are computed for both the walls and the global linear system is solved to to compute the strength of the equivalent sources. These equivalent sources are then used to compute the pressure field over a grid of listener positions at the given frequency. The error thresholds used for the per-object and inter-object transfer functions are 15% and 1%, respectively. The state-of-the-art FastBEM simulator (`http://www.fastbem.com`) is used for generating the pressure fields for the BEM and FMM-BEM techniques. BEM error tolerance and order of multipole expansion were set to 1% and 6, respectively. For this scene, the FASTBEM simulator can handle frequency only up to the maximum frequency possible (358 Hz) on a single desktop machine. Therefore, the comparison is restricted to frequencies up to 358 Hz.

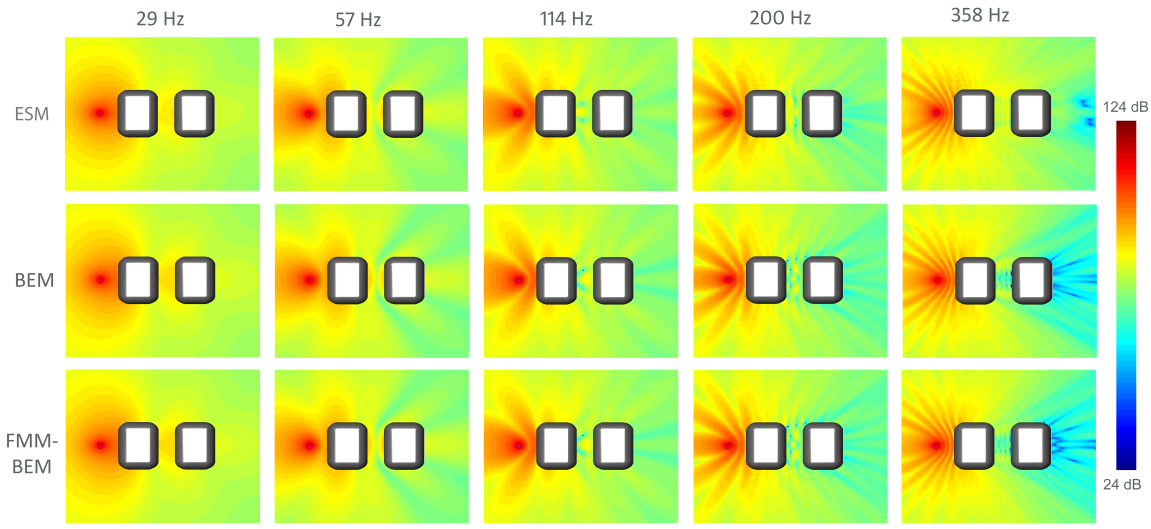


Figure 6.1: Comparison between the magnitude of the total pressure field (in Sound Pressure Level SPL, units dB) computed by the ESM and the reference BEM and FMM-BEM techniques for the two parallel walls scene on a XY cutview grid of listeners. The red point denotes the position of the sound source. The error between the ESM-BEM fields is $< 5\%$ and ESM-FMM BEM fields is $< 5\%$ for the frequencies shown.

Based on visual inspection, the pressure fields produced by the ESM technique match well with the pressure fields produces by the BEM and FMM-BEM technique. For quantitative comparison, error is defined as error $= \|P_{ref} - P_{ESM}\|^2 / \|P_{ref}\|^2$, where P_{ref} is the pressure produced by the reference wave-solver (either BEM or FMM-BEM) and the P_{ESM} is the pressure field produced by the proposed equivalent source technique. The error between the ESM-BEM and the ESM-FMM BEM pressure fields is less than 5%. The ESM technique can generate pressure fields at a listener

position at interactive rates ($8ms$ per listener position) whereas the BEM and FMM-BEM techniques are offline techniques that require a few seconds per listener position. The runtime memory required by the ESM technique is $14MB$ whereas the BEM and FMM-BEM techniques require $700MB$ (see Table 3.4). Even higher accuracy can be achieved with the ESM technique at the cost of more runtime memory and reduced runtime performance by further reducing the user-defined error thresholds for per-object and inter-object transfer functions. This allows for flexible performance-to-accuracy tradeoffs based on the application.

6.3.2 Validation of Spectral Extrapolation

The spectral extrapolation technique used in Chapter 3 is evaluated by comparing the audio quality of its results with wide-band spectrum (0-22 kHz) produced by the Biot-Tolstoy-Medwin (BTM) technique for the single, finite-edge scenario. The scene chosen for this comparison is a right-angled wall. In the BTM method, edge diffraction impulse responses are computed by evaluating a time-domain line integral over the finite length of the edge. This is essentially based on Huygens theory where a diffracting sound wave is modeled as a superposition of an infinite number of secondary point sources situated along the diffracting edge, each with different strengths and directivities. BTM has been shown to converge to the exact analytical solution for a simple scene like the right-angled wall (Svensson et al., 1999). The MATLAB-based edge diffraction toolbox (`http://www.iet.ntnu.no/~svensson/software/index.html`) is used to generate the BTM results. The final auralized audio generated by both the techniques for this scene sound similar.

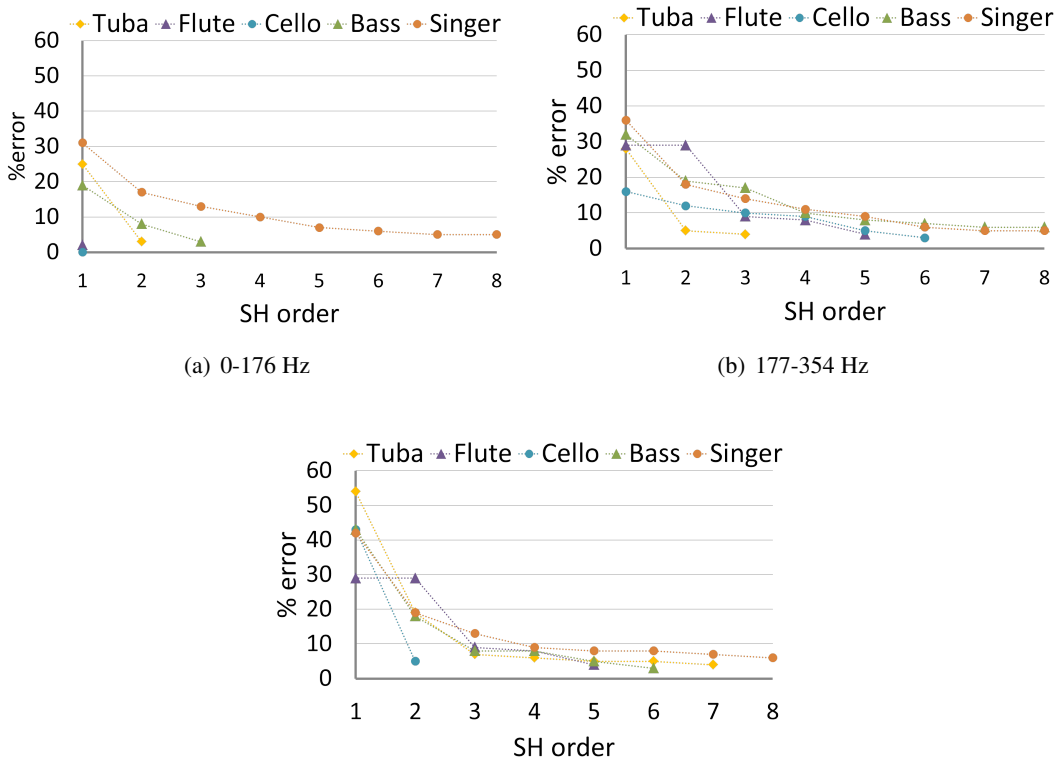
Note that the spectral extrapolation technique is approximate which becomes exact in a specific, single-edge diffraction configuration. It does not guarantee accuracy on general scenes at high frequencies. While single-edge diffraction arises frequently in outdoor scenes, many other complex configurations also occur, such as double-diffraction and diffracted-reflection. The proposed extrapolation approach would be accurate in such cases only if the acoustic response is dominated by diffraction from a single edge. In other cases, this extrapolation technique has been shown to generate plausible results. A general spectral extrapolation approach for band-limited acoustic responses with guarantees on extrapolation error for arbitrary scenes, is an important area for future research.

6.4 Validation of Directivity Formulation

In this section, the validation of the directivity presented in Chapter 4 is presented.

6.4.1 Validation of Source Directivity

Figure 6.2 shows that the proposed SH-based source representation converges to the measured directivity with increasing SH order. Due to the different directivity of sound sources (cello, flute, singer), a different SH order is required to achieve the same error threshold.



(a) 0-176 Hz

(b) 177-354 Hz

(c) 355-710 Hz

Figure 6.2: **Convergence**: Our SH-based source representation converges to the measured directivity data with higher SH orders, for different directional sources. Error metric defined in Equation (4.20).

6.4.2 Validation of Sound propagation

In order to validate the integration of our directivity framework with a sound propagation technique, we compare the pressure fields of ESM and BEM techniques integrated with our proposed directivity formulation. Figure 6.3 shows the magnitude of the pressure field computed by the BEM simulation and the ESM technique for directional sources. BEM error tolerance and order of multipole expansion was set to 1% and 6, respectively. ESM error thresholds for scattering and interactive matrices are 15% and 1%, respectively. ESM error thresholds were chosen for interactive performance and can be further reduced to achieve higher accuracy. The pressure fields produced by the two techniques agree within error of $< 5 - 10\%$. The offline BEM method has a much higher computational and memory overhead, but the resulting pressure fields are more accurate.

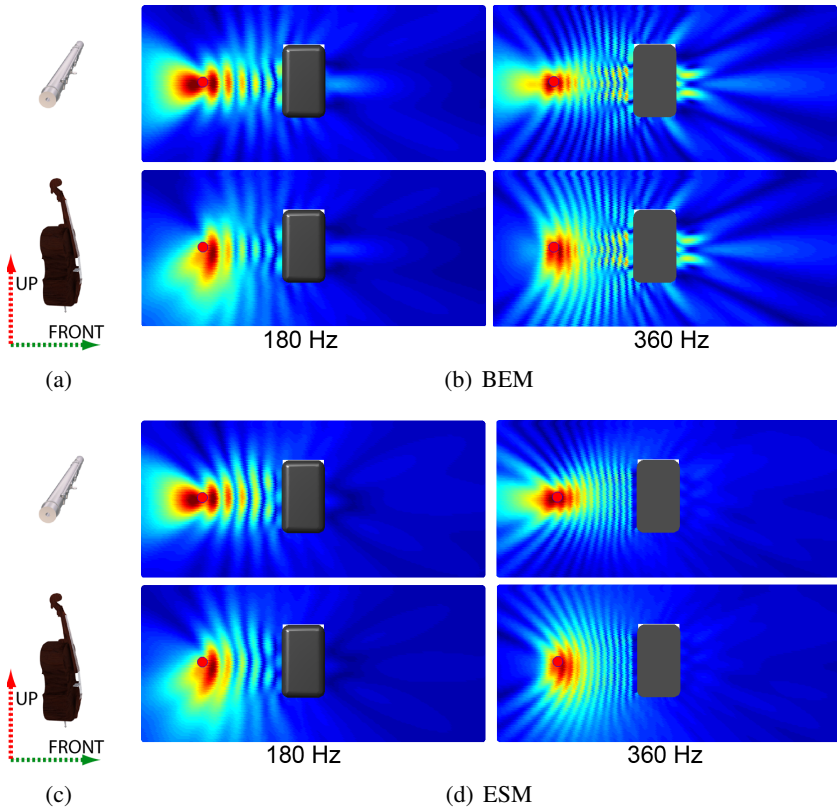


Figure 6.3: **Integration of source directivity with frequency-domain sound propagation techniques:** Magnitude of pressure field (in Pa, normalized $[0, 1]$) using offline BEM and interactive ESM technique for the single wall scene at 180 Hz and 360 Hz. Source position is shown with a red dot. Front axis points towards the building. Up axis points upwards perpendicular to the listener grid.

We also perform validation experiments with the Biot-Tolstoy-Medwin (BTM) method (Svensson et al., 1999; Lokki et al., 2002). This is an offline technique that can incorporate source directivities and provides a wide-band reference solution with accurate diffraction in simple scenarios. We integrated our SH-based directivity formulation with the BTM Toolbox for MATLAB (`www.iet.ntnu.no/~svensson/software/index.html#EDGE`). The formulation to evaluate source directivity using BTM is based on the prior work on BTM line integrals (Stanton et al., 2007), which can be extended to incorporate source directivities as well as handle arbitrary directivities (Lokki et al., 2002). The BTM approach models edge diffraction by placing several secondary sources at positions sampled along the diffracting edges. Their intensities are determined only by the intensity of the sound source and the distance between the sound source and the secondary source. We scale the strengths of each secondary source by the SH-based directivity function $D(\theta, \phi)$, where (θ, ϕ) are determined by the direction vector from the source to the secondary source. We observed that BTM integral calculations were significantly slowed down for directional sources due to the need to evaluate source directivity for each integration sample along the edge. We performed BTM simulations over three fixed-receiver positions and a fixed source position in the wall scene. Simulations for receivers 1, 2, and 3 took 57, 27, and 18 minutes, respectively. The final audio produced by our ESM-integrated directivity approach and BTM-integrated directivity approach produce similar sounding audio.

6.5 Validation of Parallel Adaptive Rectangular Decomposition

In this section, the parallel adaptive rectangular decomposition technique proposed in Chapter 5 is validated with analytical solutions and real-world measurements.

6.5.1 Validation with Analytical Solutions

The validation of the parallel ARD technique is performed on two benchmark test-cases: (a) spherical wave scattering by a rigid sphere and (b) edge-diffraction from a right-angled rigid wall. In the first case, the acoustic wave equation has known analytical solution (Section 6.2). The scene setup is as follows: a sphere of radius $a = 1m$, surrounded by air with speed of sound $343m/s$ and mean density of $1.21kg/m^3$, is centered at origin $(0, 0, 0)$. A spherical sound source (monopole source) is placed at position $(0, 0, -3)$. The spherical wave emitted by the source is scattered by the

rigid sphere. The total field (incident+scattered field) is computed using the analytical solution of the wave equation at an angular distribution of listener positions at distance of $1.5m$ (Equation (6.1) and (6.2)). The analytical solutions are compared against the simulation results at different wave numbers k as shown in Figure 6.4. The results are plotted versus the polar angle θ where $\theta = 180\,^{\circ}$ corresponds to the front end of sphere with respect to the incoming spherical wave. The comparison between the analytical expressions and the ARD simulation results show very good agreement.

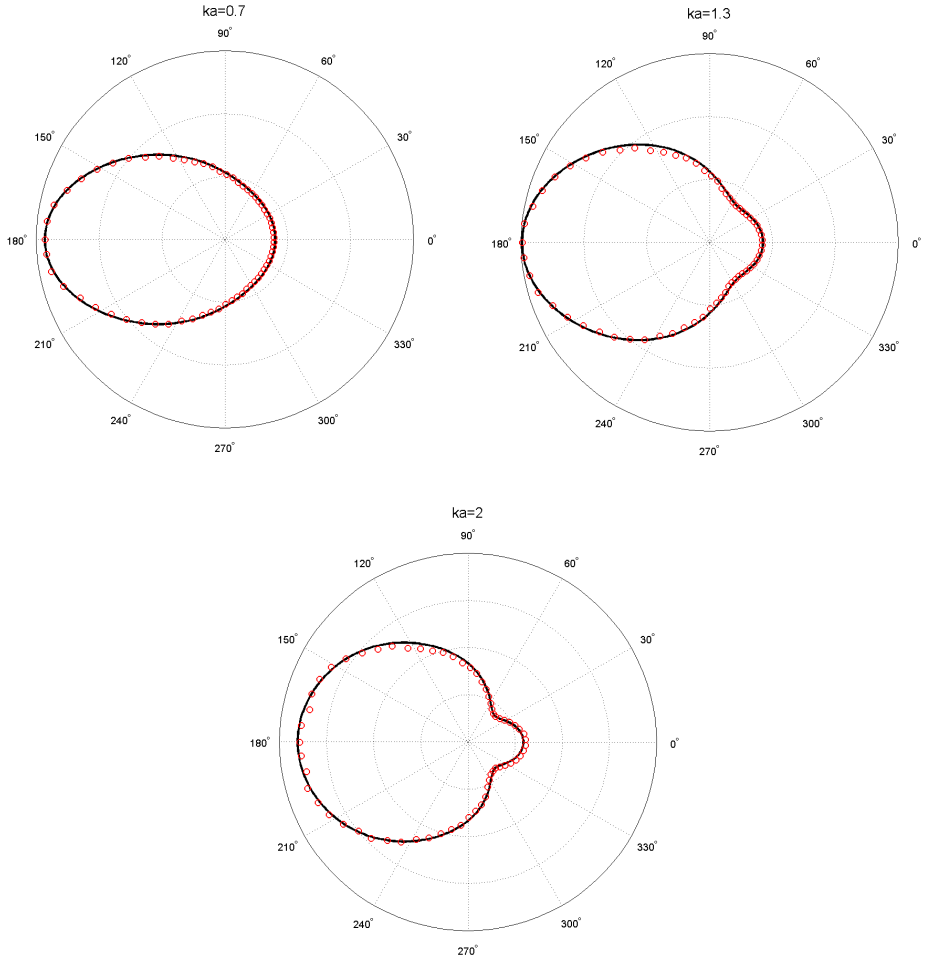
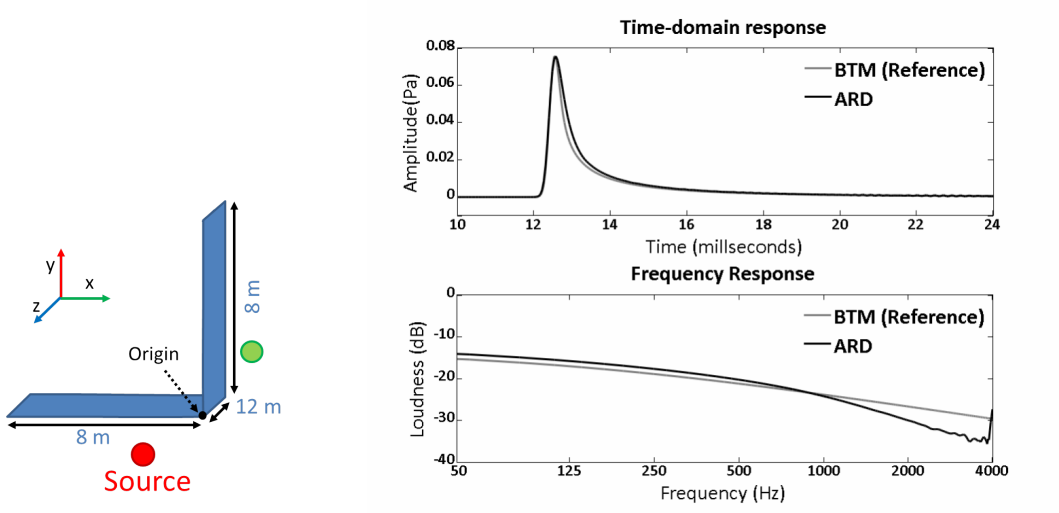


Figure 6.4: Validation of the ARD simulation results (dots) with analytical expressions (curves) for the scattering of a spherical wave by a rigid sphere. Normalized pressure is plotted in the radial axis. The radius of the sphere $a = 1m$ and wave numbers k considered are 0.7, 1.3 and $2\ m^{-1}$, respectively.

In the second case, validation of the ARD technique is performed by comparing it against the edge diffraction model proposed by (Svensson et al., 1999). This model is an extension of

Biot-Tolstoy-Medwin solution (Medwin et al., 1982) to finite edges. The scene setup is as follows: a right-angled rigid wall of dimension $8m$ x $12m$ is considered with the longer edge being the diffraction edge. Source and receiver are placed at symmetric positions with respect to the wall at $(-1.8, -0.9, -6.0)$ and $(0.9, 1.8, -6.0)$, respectively. The time- and frequency-domain responses are computed using the BTM finite-edge diffraction model and compared against the results of the ARD simulation. As shown in Figure 6.5, the agreement between the responses of two techniques is very good.



(a) (b)

Figure 6.5: Time and frequency responses produced by the Biot-Tolstoy-Medwin diffraction model (reference) and the ARD simulation for a right-angled rigid wall.

6.5.2 Validation with Measurements

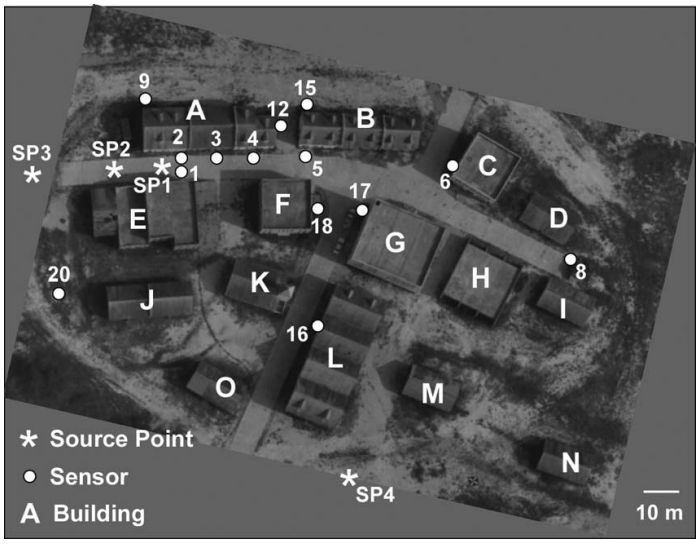
Measurements The real-world measurements used in the validation of the numerical simulation are discussed here. This dataset was presented in the work of Albert and Liu (2010).

Scene Layout: The experiment was conducted in an artificial village spanning a 150 x 150 m^2 area with 15 buildings and two cross-streets: "Main street" running nearly horizontal, and "Church street" running in an approximately vertical direction. Figure 6.6(a) shows the 2D layout (top view) of the urban scene. The buildings in the village were two or three stories tall and made up of concrete blocks. The ground areas consisted of streets, grass areas, and hard-packed soil.

Weather Conditions: The experiment was conducted over two sunny days with temperature, wind and relative humidity variation between 8 to 19° C, 2 to 5 m/s and 30%-50% respectively.

Sources: Acoustic pulses were produced by using small explosives of 0.57 kg of C4 suspended at a height of 1.5 m from the ground. The measurements were recorded for four source positions SP1-SP4.

Receivers: Sensors were placed at 14 different receiver positions spread throughout the scene, in both line-of-sight (LOS) and non-line-of-sight (NLOS) positions. These sensors were connected to digital seismographs that recorded the pressure signal at a sampling rate of 5 or 8 kHz.



(a)

(b)

Figure 6.6: a) Top view of the urban scene used in the experimental study. b) An approximate 3D model of the scene constructed based on the 2D layout, photographs of the scene and heights of the buildings corners and roof tops.

ARD Simulation The details of ARD simulation parameters used for this scenario is as follows:

Parameters: The source function used to model the explosive blast signal for calculations in the ARD simulator is described in (Liu and Albert, 2006). Figure 6.7 shows the corresponding source function with peak pressure normalized to 1.

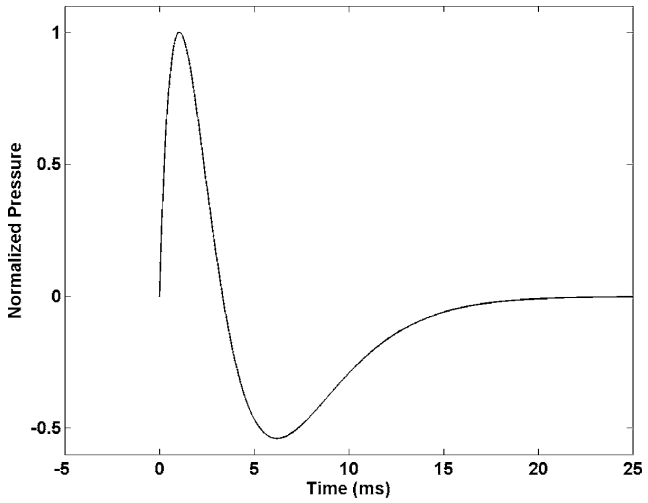


Figure 6.7: The source pulse used for modeling the blast signal produced in the experiment as calculated from Eq.(5) (Liu and Albert, 2006).

In order to run a 3D numerical simulation, a virtual 3D model of the scene is required. However, a detailed 3D model cannot be constructed due to the lack of availability of the architectural blueprints or a laser-scanned point cloud of the site. Therefore, a simplified 3D model of the scene is constructed using a 2D layout of the village, photographs, and the heights of corners and rooftops of buildings. This 3D model is an approximation to the actual geometry of the scene, since it lacks particular geometric details such as peaked roofs, door/window locations, facade details, and extraneous geometry such as cars and a fountain. The dimensions of the simulation domain are 175m $\times$ 140m $\times$ 14 and the heights of the buildings are between $6 - 9$m. Therefore, depending on the building, a vertical space of $5 - 8$m is available above the roof between the rooftop to the top of the simulation domain, allowing for correct simulation of rooftop diffraction. Figure 6.6(b) is a textured rendering of the 3D model. Based on the type of material present (e.g. concrete, grass, soil, etc.), the appropriate absorption coefficients are assigned to the surfaces of the 3D model.

The ARD simulation was run with an acoustic wave velocity of $375ms^{-1}$ and an air density of $1.2kgm^{-3}$. The high value for the acoustic wave velocity comes from the propagation of the

Parameters	Values
simulation frequency	450Hz
grid size	175m x 140m x 14m
grid spacing	0.31 m
# grid points	11 million
time step size	385 μs
# time steps	2000
simulation length	0.77 s

Table 6.1: Parameters used for the ARD simulation.

high-amplitude acoustic pulse generated by the C4 explosive used as the sound source. In case of concrete, the acoustic wave velocity is $2950ms^{-1}$ and density is $2300kgm^{-3}$, which results in a reflection coefficient of 0.99 for concrete. These parameters correspond to the values used in Albert and Liu (2010) for the finite-difference simulation.

The ARD simulator is run to propagate the acoustic pulse from each given sound source position, one by one. The response at the specified receiver positions are recorded and the results are compared with the recorded measurement data.

Simulation: The ARD simulations were run for the four source positions SP1 to SP4 shown in Figure 6.6(a). The parameters used in the simulator are given in Table 6.1. The total simulation time for each source was 20 mins for the CPU-based ARD implementation (Raghuvanshi et al., 2009b) and $1-2$ mins for the GPU-based ARD implementation (Chapter 5). The CPU-based implementation is in C++ and the GPU-based implementation uses NVIDIA's CUDA programming language. The timing results were measured on a single core of a 4-core 2.80 GHz Xeon X5560 desktop machine with 4 GB of RAM and on an NVIDIA GeForce GTX 480 GPU with 448 CUDA cores and 1.5 GB memory.

The ARD responses were computed for four source positions, up to a maximum frequency of 450 Hz. The measured waveforms were low-passed to 450 Hz for source positions 2 and 3 to compare with the calculated ARD waveforms. For source position 1 and 4, both the ARD and the measured responses were low-passed to 200 Hz, since the 2D FDTD waveforms are available only up to 200 Hz.

In this scene, the simulation frequency is less than $500Hz$ and the propagation distances are of the order of hundreds of meters, for which the intrinsic absorption of the atmosphere is negligible. Therefore, atmospheric absorption is ignored during the simulation.

Figure 6.8 shows the visualization of the time-domain ARD simulation at specific time-steps. These visualizations show the propagation of wave fronts in the scene and how they are modified by the multiple reflections and diffractions from the buildings, and reflections from the ground. These can be helpful in guiding engineering modifications to the scene.

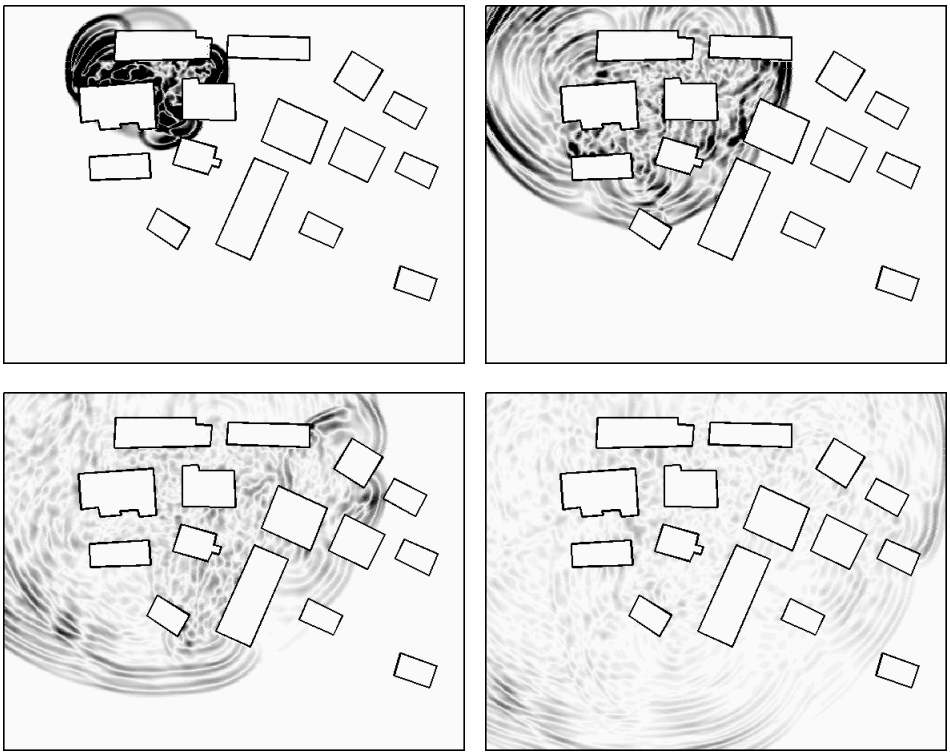


Figure 6.8: Calculated pressure field for the source position SP1 in the artificial village scene using the ARD technique. Simulated wavefields are shown at times t = 75, 150, 225, and 300 ms.

Varying propagation speed: In the training village scene considered in this study, the sound source used is a C4 explosive. The high-amplitude explosion caused by this source generates a varying sound-speed profile. Figure 5 in Liu and Albert (2010) shows the measured values: over 400 m/s a meter or two away and 375 m/s at about 20 m distance. The speed varies with the amplitude of the traveling pulse, reducing with distance. To make things more complicated, the amplitude of a wave that diffracts around a building corner is reduced, and that diffracted wave travels slower than waves at the same propagation distance propagating in line-of-sight paths. This generates a

complicated wave-speed profile that is harder to model. The peak arrival times can get shifted in the measured waveform as compared to the calculated waveforms, which assume a constant speed of sound. Also, the individual peaks can get stretched due to decreasing wave speed. Therefore, there are complicated changes in acoustic wave speed even for different waves at the same location.

Similar to the 2D finite difference simulation of Albert and Liu (2010), the 3D ARD simulation also did not allow us to track these changes with amplitude, one would have to follow the individual wavefronts to do that correctly, or modify the code to explicitly include non-linear effects. Instead, a constant wave speed was chosen to get the best waveform fit to the measured data. Also, similar to Albert and Liu (2010), time-shifting is performed to the response of linear simulation to align the first arrival peak with the measured waveform. In addition to that, kinematic errors due to the varying sound speed are taken into account with the "robust" error metric discussed later.

3D vs 2D wave simulation: The advantages of the 3D wave simulator over the 2D wave simulator for acoustic pulse propagation are discussed. Firstly, a 3D wave simulation incorporates propagation paths over the top of walls or buildings, as well as wave diffraction from the upper edges, both of which are completely ignored by a 2D simulation. Secondly, in a 3D simulation, the sound reflection from the ground terrain is handled accurately for all frequencies. For a 2D simulation, the pressure is simply doubled to approximate ground reflections, which is accurate only for frequencies up to $600Hz$, as discussed in Liu et al.(2006). Lastly, the results of a 2D simulation must be renormalized by an additional factor of $1/\sqrt{r}$ to account for 3D geometric spreading. This normalization is valid only for large kr (where k is the wave number and r is distance to the source). A 3D simulation requires no such normalization.

Results In this section, the results of the parallel ARD technique are compared to the measurements and the prior FDTD technique.

Error Metrics In order to perform a quantitative evaluation between the measured and the calculated waveforms, two types of error metrics are used in the comparison: the spectrogram difference metric (SDM) and the average decibel metric (ADM). The SDM metric is computed on the spectrograms $SPEC$ of the time-domain pressure signals $\{a\}_{i=1}^{N}$ and $\{b\}_{i=1}^{N}$:

$$SDM(a,b) = \frac{\sum_{j=1}^{M}\sum_{k=1}^{T}\|SPEC(a)_{jk} - SPEC(b)_{jk}\|^2}{\sum_{j=1}^{M}\sum_{k=1}^{T}\|SPEC(a)_{jk}\|^2}, \tag{6.4}$$

where $M = \{N/2$ if N is even or $N/2 + 1$ if odd$\}$ and T is the number of time segments in spectrogram.

The average decibel metric (ADM) is computed on the pressure signals in the decibel (dB) scale $\{dB(a)\}_{i=1}^{N}$ and $\{dB(b)\}_{i=1}^{N}$ as follows:

$$ADM(a,b) = \sum_{i=1}^{N} \|dB(a)_i - dB(b)_i\|/N. \tag{6.5}$$

The error metrics defined above are very sensitive to the time of arrival in the waveforms. In scenarios where the speed of sound is constant, these metrics perform well. However, in this case where the speed of sound varies, the arrival times can be off by few milliseconds. Even though this small time-shift might not cause a big difference in individual waveform characteristics such as shape, frequency content, etc, it can generate a large error with these error metrics. One can think of using regular cross correlation to remedy this situation. However, regular cross correlation will match the peak arrival but will misalign the rest of the measured waveform.

The above error metrics are changed to make them more robust to the small time-shifts and peak stretching caused by varying sound speeds. The proposed change is based on the idea of Dynamic Time Warping (DTW), which is a standard tool used in the signal-processing community to handle non-linear transformations in the time axis. In the DTW technique, the two input time signals are allowed to shift, stretch, or contract, in a limited manner to generate the optimal matching between the signals.

In the proposed solution, the first arrival peaks of the calculated and measured waveforms are aligned first. This removes any time-shift that happens before the arrival of the first peak. Next, the DTW technique are applied to these signals to align the remaining part, thus taking into account small time shifts and stretching. Finally, the above error metrics are applied to these aligned signals to give a quantitative measure of the error; at the same time the *confidence* in the resulting measure is also defined. The confidence measure is based on the intuition that the less DTW warping required to align the signals, the more the confidence on the similarity of the signals.

$$\text{Confidence} = (1 - d_w/l_o) \times 100 \quad (\text{in percent}),$$

where $d_w = |l_w - l_o|$ is the difference in the length of the warped signal l_w and the original signal l_o. For all the results shown in the paper, a warping length change of only $5 - 10\%$ is allowed, resulting in a confidence measure of $90 - 95\%$.

Comparison with measurements in time domain: In this section, the waveforms calculated using the ARD simulation are compared with the measured waveforms for different source and receiver positions. Note that waveform modeling requires a strong agreement between the amplitudes and phases of the calculated and measured waveforms, making it a stringent test for any acoustic pulse propagation technique.

Figure 6.9 compares the ARD waveforms and the measurements for the source SP2 and the receiver positions. The upper traces in each panel correspond to the calculated ARD waveforms, and the lower traces correspond to the measurements. The source is located to the left of the narrow street canyon formed between buildings A and E (see Figure 6.6(a)). For all the line-of-sight (LOS) positions (R01 thru R06), an excellent match between the calculated and measured waveforms for the direct sound (first arrival) and the subsequent reflections is observed. The main characteristic of LOS responses is the strong first arrival which dominates the signal, followed by (comparatively) weaker reflections. For receivers very close to the source (R01 thru R04), the high amplitude of direct sound completely dominates the later reflections, whereas for receivers far away (R05, R06), the reflections have comparable energy.

For non-line-of-sight (NLOS) positions, the waveform characteristic is position-dependent and more complex. In the case of receiver R20, the propagation paths mainly consist of diffraction around and from the top of building E. For receiver R08, the diffracted first arrival is followed by high-order reflections from buildings C, H, D and I. In the case of receiver R15, diffracted arrivals around and from the top of buildings A and B are followed by high-order reflections and diffractions from A, F and B. Similarly, for R09, the first arrival corresponds to diffraction from the rooftop of building A followed by high-order reflections and diffractions from buildings A and E. For receivers R17 and R18 the first arrival is via diffraction, followed by multiple reflections trapped between buildings F and G. In the case of R17, the diffraction angle is 10-15 degrees, resulting in a high amplitude diffraction peak; in R18, by contrast, the diffraction angle is 90 degrees, which results in a low amplitude diffraction peak.

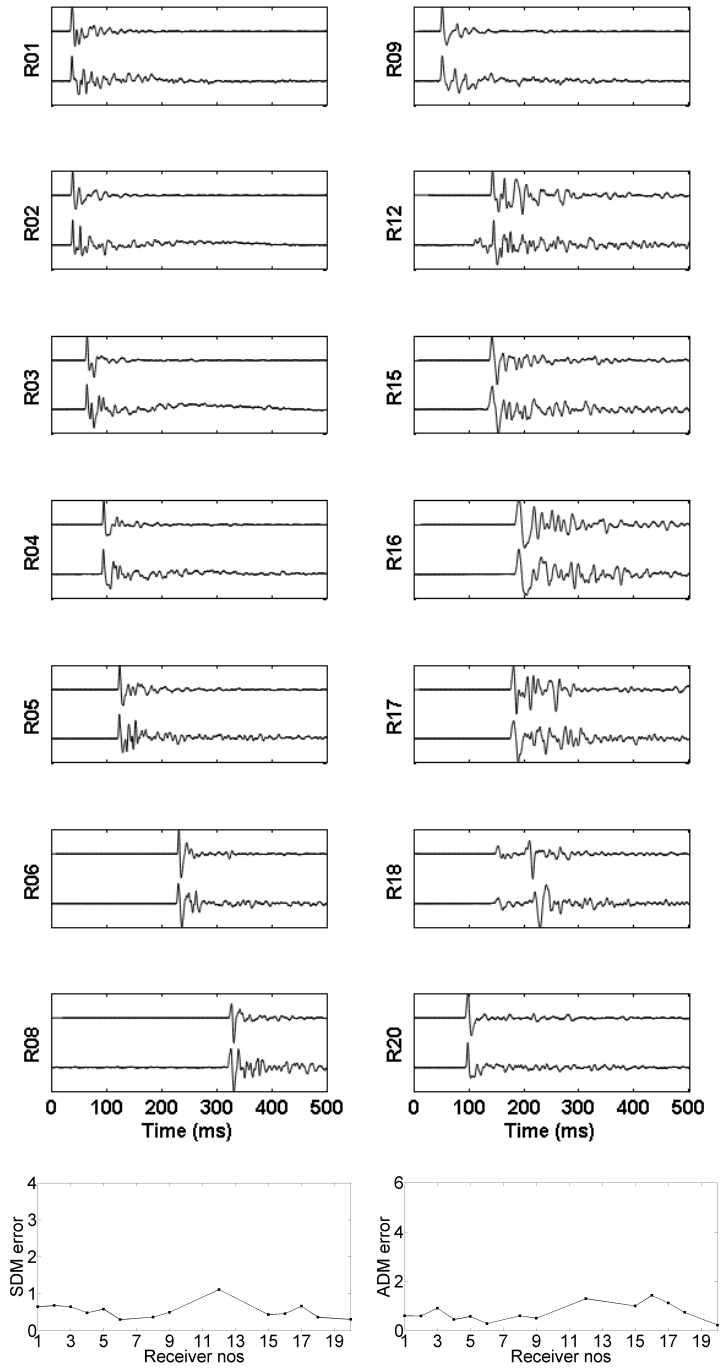


Figure 6.9: Waveforms calculated by the 3D ARD simulator (upper) and measured (lower) in the artificial village for source position 2 at 14 receiver positions. All the waveforms have been individually normalized and low-passed to the maximum frequency of 450 Hz. The error between the 3D ARD simulation and measurement has been calculated using the DTW-based SDM and ADM metric.

As can be seen in Figure 6.9, the calculated waveforms incorporate all these features and match with the measured waveforms to a high degree of accuracy. The biggest mismatch between the

waveforms is for sensor R12, which is a NLOS position around the corner of the building A on Main Street. In this case, the ground floor room at the corner had two open windows facing Main Street, and one open window around the corner between Main Street and the sensor at R12. This resulted in a shorter path through that room to the sensor for sound coming from source positions SP2. This is probably what can be seen before the large arrival, which presumably diffracted around the building corner itself. The open windows had no glass at all and were about 2 x 4 feet in area, so this small size (compared to a wavelength of about 7 meters at the source) would reduce the amount of energy traveling on that shorter "indoor" path. Some of the additional high frequency arrivals later on in the measured waveform may be caused by reverberation inside that room. Due to lack of availability of window positions data, these were not included in the virtual 3D model constructed for the ARD simulation. Therefore, these early arrivals are not modeled in the simulation results. Same behavior is observed for source positions SP1 and SP3.

In terms of basic error metric (not shown in the figure), the highest value for this source simulation occurs at receiver R18 (basic SDM error=1.67) due to a decrease in the wave speed after the first diffraction from building F. This causes subsequent strong reflection from the opposite building G to arrive much later in time than the calculated waveform (which assumes a constant sound speed $375ms^{-1}$). This time stretching cannot be modeled by a constant time shift applied at the beginning of the signal. Thus, the measured waveforms for R18 appear to be the stretched equivalents of the calculated waveforms; this results in a higher error using the basic metric. The DTW-based robust error metric takes this stretching into account, correctly predicting a low error as shown in the figure. Similar behavior can be observed for receivers R16 and R17.

In Figure 6.10, the same comparison between calculated and measured waveforms is performed for the source position SP3. All the LOS positions (R01 to R06, and R20) exhibit excellent matches between the calculated and measured waveforms. For NLOS positions involving first-order diffraction (i.e. R09), the calculated responses incorporate diffraction paths from both around and on top of the buildings to generate the correct waveform. For R08, the first diffraction arrival is weaker than the later reflection; this behavior is corrected modeled by the calculated ARD response. As described before, the waveforms of receivers R16, R17, R18 get stretched in time by the variable acoustic speed; this stretching results in high error with the unwarped metrics and low error with DTW-based metrics.

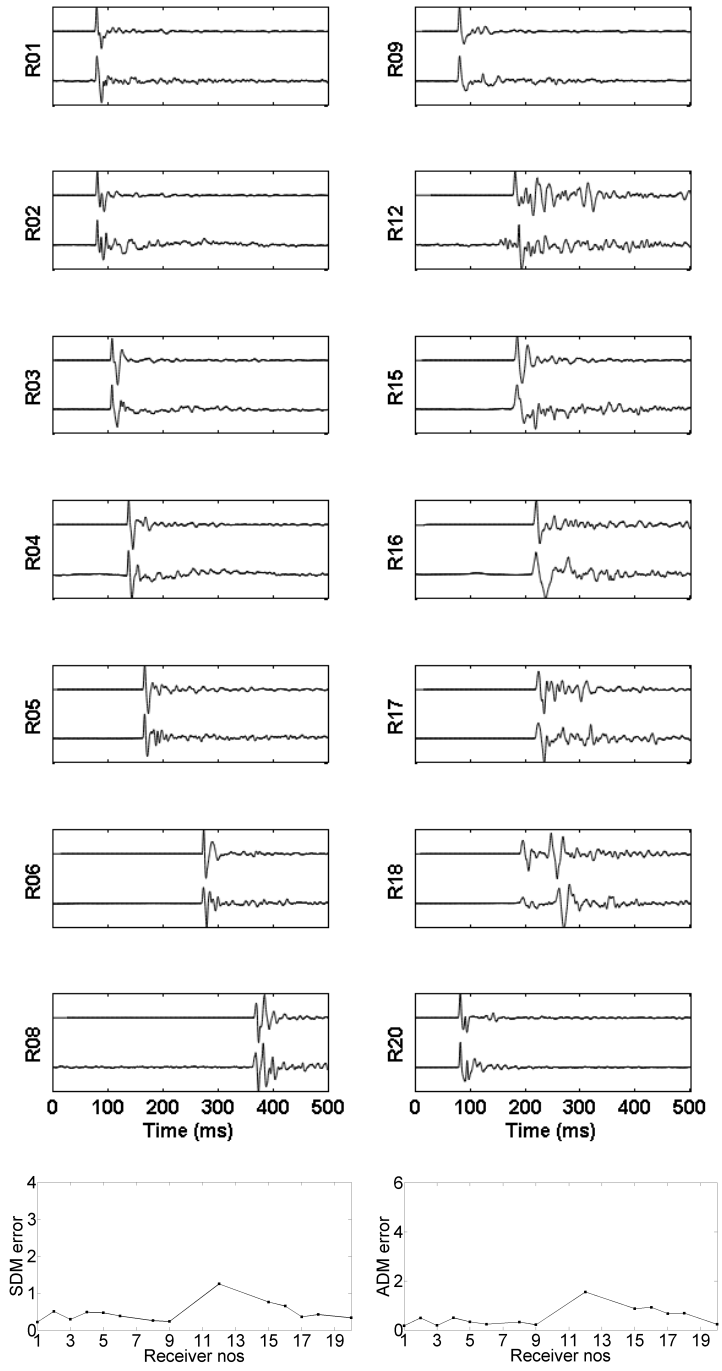


Figure 6.10: Waveforms calculated by the 3D ARD simulator (upper) and measured (lower) in the artificial village for source position 3 at 14 receiver positions. All the responses have been individually normalized and low-passed to the maximum frequency of 450 Hz. Error between the 3D ARD simulation and measurement has been calculated using the DTW-based SDM and ADM metric.

Aside from sensor R12, the biggest mismatch between the waveforms is for receiver R15 for which the calculated waveform shows two distinct peaks as compared to only one peak for the

measured waveform. One possible explanation is that the arrival times between the two diffraction peaks shown in the calculated response is smaller in the real scene due to varying speed of sound, resulting in a constructive interference and peak merging in the measured waveform.

Comparison with 2D FDTD: In this section, the calculated ARD and FDTD waveforms are compared to the measured waveforms for two source positions, SP1 and SP4. The ARD waveforms are calculated by running a 3D simulation on the 3D model. The FDTD waveforms are calculated by running a 2D finite-difference simulation on a 2D grid as described in Liu and Albert (2006).

Figure 6.11 shows the calculated and measured waveforms for the source position SP1 and its receiver positions. The upper traces in each panel correspond to the calculated ARD response, the middle trace to the measured waveform, and the lower traces to the calculated FDTD responses. In case of LOS positions (R01 to R06), the dominant propagation happens in the XY plane containing the sources and receivers. Therefore, the waveforms calculated using the 3D ARD simulation and the 2D FDTD simulation match equally well to the measured waveforms. The main difference between a fully 3D and a 2D simulation arises in cases where the sound waves diffract from the rooftops of the buildings, resulting in shorter propagation paths and higher energy (as illustrated in Figure 6.13). For receiver R09, the diffraction path from the top of building A is the shortest path and corresponds to the first arrival. This shortest path is modeled correctly by the 3D ARD simulation and not by the 2D FDTD simulation. In the case of receiver R20, the secondary arrival peak corresponds to the rooftop diffraction path, which is missing in the 2D simulation waveform. For receiver R15, the energy from rooftop diffraction paths is missing from the 2D simulation but not from the 3D simulation. Therefore, for these NLOS cases, the 3D simulation waveforms match far better with the measured waveforms than the waveforms from the 2D simulation.

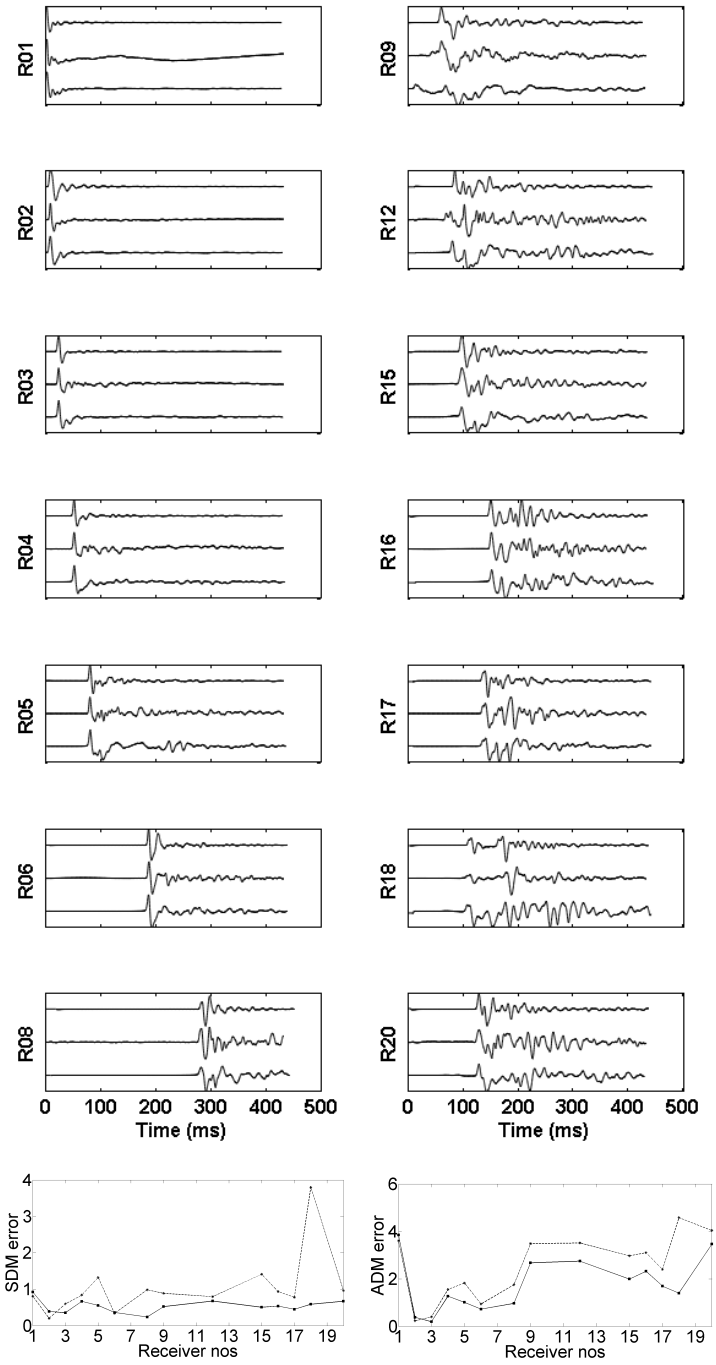


Figure 6.11: Waveforms calculated by the 3D ARD (upper) and 2D FDTD (lower) simulations and measured (middle) in the artificial village for source position 1 at 14 receiver positions. All the responses have been individually normalized and low-passed to the maximum frequency of 200 Hz. The error between the 2D FDTD simulation and measurement (dashed line) and 3D ARD simulation and measurement (solid line) has been calculated using the DTW-based SDM and ADM metric.

In Figure 6.12, a similar comparison is performed for source position SP4. This source is positioned outside the main village compound, and most of the receiver positions are non-line-of-sight. The only LOS position is receiver R20, where both the calculated waveforms match well with the measured waveforms. For NLOS positions, the measured waveforms are again stretched as compared to the calculated waveforms, as discussed before. The speed of sound reduces significantly after diffraction, resulting in high-order propagation peaks to arrive later in the measured waveforms than in ARD and FDTD waveforms (which assume constant speed of sound). In the case of receivers R04 to R06 and R15 to R18, the correct modeling of rooftop diffraction with a 3D simulation (ARD) results in a better match with the measured waveforms. The 2D FDTD simulation ignores these paths, resulting in a lower first arrival energy than in the measured data. The measurement data for receiver position R1 is not available for the source position SP4. As described in Liu and Albert (2010), this location was fitted with the high-pressure blast sensor from the previous measurement, when it was measuring the pressure from the nearby explosive charge at SP1. This high-pressure sensor was unable to detect the low pressure waveform produced by the distant source SP4.

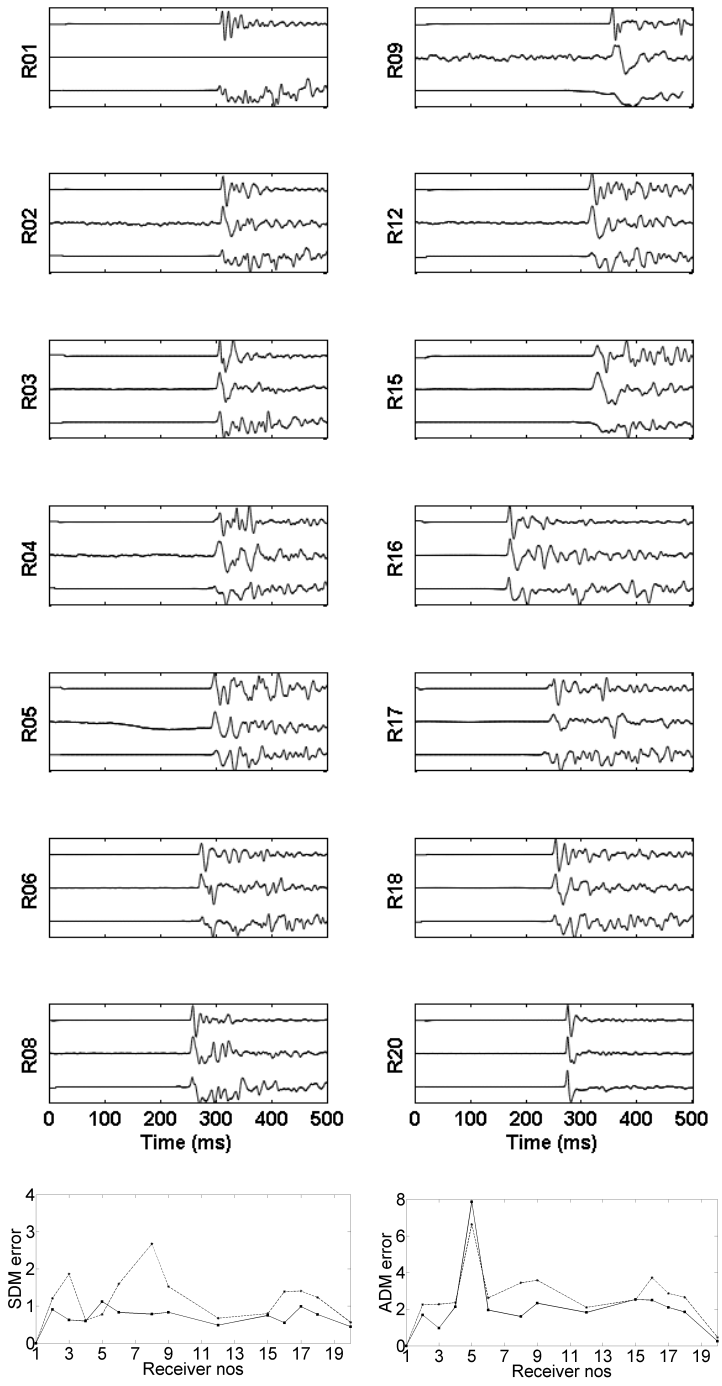


Figure 6.12: Waveforms calculated by the 3D ARD (upper) and 2D FDTD (lower) simulations and measured (middle) in the artificial village for source position 4 at 14 receiver positions. All the responses have been individually normalized and low-passed to the maximum frequency of 200 Hz. The error between the 2D FDTD simulation and measurement (dashed line) and 3D ARD simulation and measurement (solid line) has been calculated using the DTW-based SDM and ADM metric.

Comparison with measurements in spatial domain: One of the primary advantages of a time-domain wave simulation (FDTD or ARD) is the ability to save snapshots of the pressure field at any time step in the simulation. These snapshots can be assembled into a movie to elucidate wave-field evolution in time as the acoustic pulse travels through the environment. This movie can serve as a useful tool for studying in detail the complex wave-interactions involved in the acoustic pulse propagation.

As an example, wave-field snapshots validate the presence of rooftop diffraction paths in both measured and 3D ARD waveforms for sensor R09 and source SP1 (see Figure 6.14). In Figure 6.13 (upper trace), the first arrival for 3D ARD and the measured waveforms happens at $t = 61$ ms. The wave-field snapshot in Figure 6.14 at $t = 61$ ms shows that the corresponding propagation path is a rooftop diffraction path from the top of building A, followed by a diffraction from the side of the building at $t = 69$ ms. The 2D FDTD simulation cannot model 3D rooftop propagation paths and misses the energy corresponding to this path. Late arrivals in the waveform correspond to high-order reflections between building A and E that get diffracted around the side or the rooftop of building A before reaching the sensor. These correspond to peaks at $t = 83, 104$ and 177 ms, which are marked by circles on top the waveforms in Figure 6.13. These peaks are correctly modeled by the 3D ARD simulation, as shown by the wave-field snapshots for these times in Figure 6.14. The wave-field snapshots and movies can thus provide a more thorough understanding of acoustic pulse propagation.

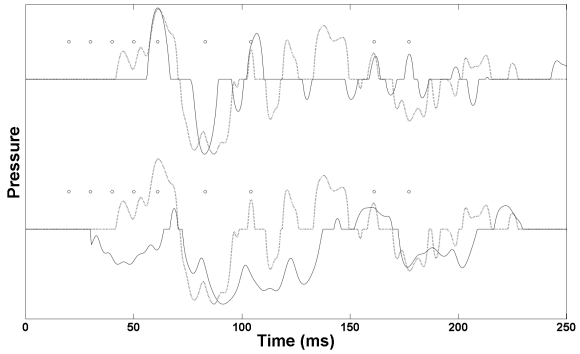


Figure 6.13: Comparison between the calculated and measured waveforms for the source position SP1 and receiver R09 behind building A. The upper trace (solid line) corresponds to the 3D ARD waveform whereas the lower trace (solid line) corresponds to the 2D FDTD waveform. The measured waveform is drawn as dotted line. Note that the 2D FDTD simulation cannot model the diffraction path from the building's rooftop resulting in the missing first arrival at 60 ms.

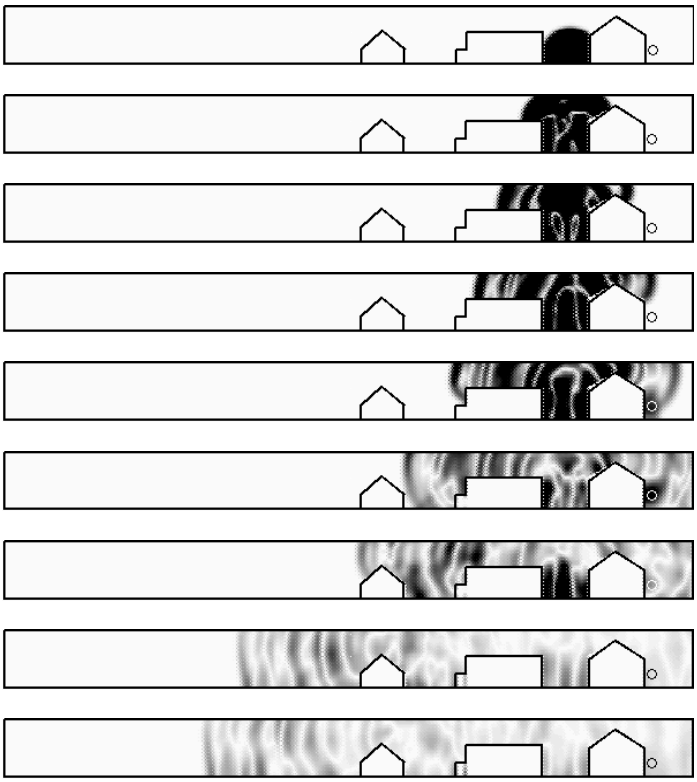


Figure 6.14: **Rooftop diffraction:** Calculated acoustic response for the source position SP1 in the artificial village scene using the ARD technique. Receiver position R09 is marked by a white circle on the right of the first building (from right). Simulated wavefields are shown at times t = 20, 30, 40, 50, 61, 83, 104, 161 and 177 ms corresponding to marked points in Figure 6.13.

Figure 6.15 shows the variation of error with distance between the source-receiver positions. The error seems to be independent of the source-receiver distance. Note that error values are much higher for source position SP4 than others. This is due to the presence of more NLOS positions in SP4, which typically have more complex propagation characteristics than LOS positions. The top three sensors with consistently high errors across all sources are R12, R16 and R15. As discussed before, R12 has high errors due to open windows in building A, resulting in shorter propagation paths that are not modeled in the simulation. As for sensor R16 and R15, these are NLOS positions for all sources, which typically have higher errors.

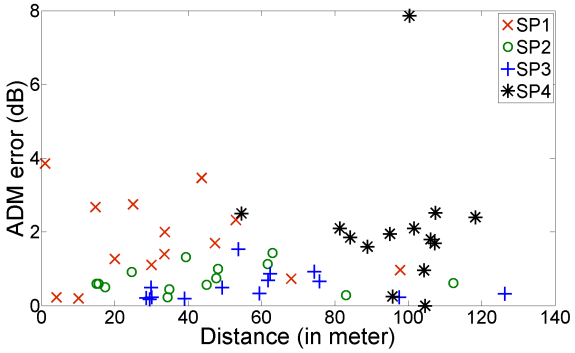


Figure 6.15: Variation of error with distance between the source-receiver positions.

6.6 Conclusion

In this Chapter, we have presented validation results for the three techniques proposed in this dissertation: (a) Equivalent Source Method for sound propagation, (b) directivity formulation to handle direction sources, and (c) Parallel Adaptive Rectangular Decomposition technique. These results produced by the techniques are validated by comparing them with analytical solutions and state-of-the-art offline numerical simulation techniques (BEM and FMM-BEM).

Limitations and Future Work: The ESM technique for sound propagation was validated on the parallel walls scene which had only two objects. In the future, we would like to validate the ESM technique on larger scenes with tens to hundreds of objects. This would require a scalable ESM, BEM or FMM-BEM simulators for generating the pressure field solutions. With regards to validation of the directivity formulation, we would like to perform quantitative as well as qualitative validation of our directivity formulation with real-world measurements conducted in an indoor scene

with real sound sources (for e.g. musical instruments). It would be interesting to explore the SH order required by our technique to match the measured data quantitatively and qualitatively (via listening tests). As for the parallel ARD technique, the current validation is performed up to a maximum frequency of 500 Hz. In the future, we would like to extend the frequency range of validation up to the mid-frequencies (few kiloHertz). We would also like to perform a qualitative evaluation of the SDM and ADM metric to understand the effect of these metrics on sound perception.

CHAPTER 7: USER EVALUATION

7.1 Introduction

In this Chapter, we present the results of the user evaluations conducted to study the effect of the equivalent source technique (Chapter 3) and the directivity framework (Chapter 4) on a user's sense of immersion in the virtual environment.

In the first study, the user is presented with a simple task of locating the sound source in the virtual environment. The user task performance is assumed as an indirect metric to measure the sense of presence and immersion of a user in the virtual world. Several propagation techniques are evaluated for sound propagation by the source in the environment. The comparison is performed between a geometric technique without spatial audio, a wave-based technique without spatial audio, and a wave-based technique with spatial audio. The task performance results demonstrate that the users were able to use wave-based propagation effects and spatial audio cues to locate the sound source faster than without these cues.

The second study deals with the effect of source and listener directivity on the sense of realism of the audio and audio-visual correlation in the virtual environments. The comparison cases are sound propagation without (source and listener) directivity and sound propagation with directivity. Based on the results, the users rated the virtual world with integrated directivity to have higher realism and audio-visual correlation than without directivity.

7.2 User Evaluation of Task Performance

A preliminary user study is conducted to evaluate the effect of wave-based sound propagation and spatial audio integration on a user's sense of immersion in the virtual environment. This evaluation is performed by measuring a user's task performance in the virtual environment. Such a study would

be extremely useful for someone designing a new VR application and needed to make a decision regarding which propagation system should be used.

Study Design Two experiments were conducted to evaluate the performance of the wave-based propagation system for navigation in virtual environments. In the first experiment, the performance of the wave-based approach is compared to an interactive geometric acoustics approach. The second experiment compares the wave-based approach with and without spatial audio integrated. We conducted between-group experiments. Experiments were conducted with three groups corresponding to three conditions (geometric, wave, wave+spatial). Subjects were randomly and independently sorted into the three groups and each subject performed only one condition. Each subject in the group was asked to perform the same navigation task: attempt to locate a sound source in a VR environment. Starting positions of source and listener were kept the same for all the subjects. The source was kept stationary. The task-completion time was measured as the elapsed time from the moment the subject starts walking to the subject's arrival at the sound source.

In the first experiment, the main goal was to quantify the improvement in user's task performance w.r.t. acoustic effects generated by the wave-based propagation method vs. a pure geometric propagation method. To avoid the effect of spatial audio on the results, we used monaural audio for both the methods in this experiment. In the second experiment, the performance of the wave-based approach with and without spatial audio rendering was compared to demonstrate the benefit of wave-based spatial audio in source localization and navigation. The between-groups independent variables were the sound propagation systems (geometric/wave-based) in the first experiment and wave-based system with spatial audio rendering (active/inactive) in the second experiment. The dependent variable was the task completion time. Between-group study design was chosen instead of within-group to avoid any learning effects in the subjects. Since each subject performed only one task, there was no effect of condition order or learning.

System Details The equivalent source method proposed in Chapter 3 was used as the wave-based approach and the commercial geometric acoustic system, Acoustect SDK, developed by Impulsonic^{TM} was used as the geometric acoustic approach in this study. Acoustect SDK is a sound propagation system developed for interactive applications (Chandak et al., 2012). For the first group, Acoustect SDK was chosen as the sound propagation technique and monaural audio was delivered to the

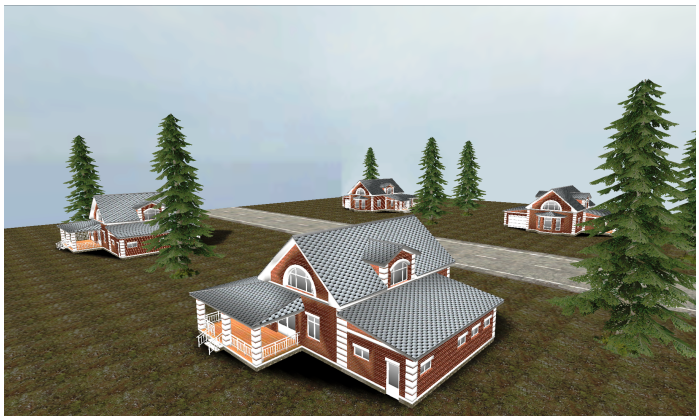


Figure 7.1: Benchmark scene used for the user study.

subjects. In case of the second group, the equivalent source method without spatial audio rendering was used and monaural audio was delivered to the subjects. For the third group, the spatial audio rendering technique discussed in Chapter 4 was used along with the equivalent source technique and binaural audio was delivered to the subjects.

We created a 3D scene in the shape of a typical suburban street with houses (named the "suburban scene", see Figure 7.1). The houses act as obstacles for sound, and generate multiple propagation effects, including reflections, diffraction, and scattering. The scene was set up in the Valve's Source^{TM} game engine. A subject's character is spawned behind one of the houses, and the sound source (a radio playing music) is placed behind another house. The same audio signal is used for all the three groups.

The subject has the first-person view of the VR environment and the rendering of the virtual environment was delivered to the subject using the Oculus Rift Development Kit-1 HMD (see Figure 7.2). The movement is controlled using the Xbox 360 controller. Audio is delivered using the Bayerdynamic DT 990 headphones. Task completion time was measured by triggering start and finish events in the game engine code and using timing counters to measure the difference.

The Acoustect system uses the image source method (Borish, 1984) for simulating reflections in an environment and a ray tracing algorithm to reduce the number of image sources that must be created. In addition to reflections, Acoustect SDK also incorporates edge diffraction based on the Uniform Theory of Diffraction (Tsingos et al., 2001). In order to maintain a similar runtime performance as our WAVE system, the following parameters were chosen for the Acoustect system: number of primary rays = 1024, number of secondary rays = 32, maximum order of reflections =

Figure 7.2: User performing the user study in a VR environment. The system tracks player's interactions using the Oculus Rift HMD and the Xbox 360 controller and delivers spatial audio using headphones.

4, and maximum order of diffractions = 2. A simplified model of the 3D scene was used for the geometric system. This simplification was performed to ensure ray approximation (object size $>>$ wavelength) remains valid.

Research Hypothesis The main research hypotheses of this study were: (1) In the first experiment, the performance of the subjects using the wave-based propagation approach would be considerably higher than those using the geometric acoustic method. This would imply that the acoustic effects produced by our system result in better task performance in the VR environment than the geometric acoustics-based propagation system. (2) In experiment two, for the wave-based approach, subjects using the spatial audio rendering would outperform those using the monaural audio rendering. This would mean that the users were able to perceive and utilize the directional cues inherent in wave-based spatial audio to navigate to the source position faster.

Procedure The study was conducted with a total of 30 subjects, all between the age of 19 and 34. There were 27 males and 3 females, and the mean age of the group is 25.5 years. All the subjects had normal hearing and normal or corrected-to-normal vision. 13 subjects had prior experience with VR environments either with HMDs or CAVE. Before starting the study, the subjects filled in a background questionnaire, and were given detailed instructions. All the subjects completed the study.

The subjects participated in two training sessions– first, to make them comfortable with HMDs and navigation using the Xbox controller, and second, to familiarize them to spatial audio. In the first training session, the subjects were asked to walk around in the virtual environment. The subjects were allowed as much time as required for them to be able to navigate fluidly. In the second training session, the subjects were asked to navigate in the direction from which they perceive the sound is coming from. It was ensured that the subjects were "sufficiently trained" by making them navigate a training map and measuring the time it takes for the users to navigate the complete map. If the time taken is less than the base-case time, the subject was asked to repeat the training step. It took the subjects 2-3 iterations to achieve the base-case time. The base-case time was computed as the average time taken by skillful users who are adept at using game controllers and HMD. Next, each subject performed the main task and their task completion time was recorded. The subjects were allowed to take as much time as needed.

Evaluation Figure 7.3 shows that the average task completion time for different sound simulation approaches used in the study: Acoustect system, the equivalent source technique without the spatial audio, and the equivalent source technique with spatial sound framework integrated. In case of experiment 1, there was a 27% increase in the task performance, which is consistent with hypothesis 1. In case of experiment 2, subjects that used the wave-based approach with spatial audio had a mean time of 43sec, which outperformed those with non-spatialized audio (mean time of 60sec). This demonstrates a further 28% increase in the task performance. This is consistent with hypothesis 2. Independent t-tests were conducted for both the experiments. For experiment one, t-ratio equal to +1.82 and p-value of 0.043 were found. In case of experiment 2, the t-ratio was +1.77 with a p-value of 0.047. Along with the independent t-tests, ANOVA was performed: F(2,27)=6.45, p-value = 0.005132. This shows the statistical significance of the results for both the experiments.

In the first experiment, the goal was not to compare performance of geometric and wave system vis-à-vis binaural audio but to quantify improvement in task performance w.r.t acoustic effects generated by a wave-based approach vs. a pure geometric approach. A typical geometric system cannot model wave effects accurately creating a discontinuity in the perceived sound field as compared to a wave-based system. In many cases, no sound is produced by the geometric system in the occluded regions. This resulted in participants having difficulties in locating the source with the

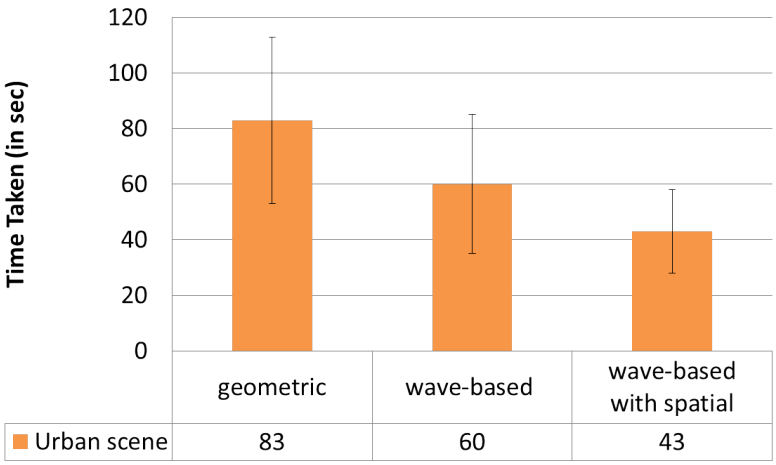


Figure 7.3: Average task completion time and standard deviation of the subjects' navigation timings for the Acoustect system, the ESM technique with no spatial sound, and the ESM technique with spatial sound. The user's performance increases as we go from Acoustect geometric system to the ESM technique without spatial sound and then to the ESM technique with spatial sound as exemplified by reduction in task completion time.

geometric system resulting in reduced task performance. In the second experiment, spatial audio added directional cues to the propagated sound making it easier to localize the source.

Note that even though the wave-based approaches have higher precomputation costs than the geometric approaches, the main idea was to compare performance of a wave-based and geometric technique with *similar* runtime performance. Higher precomputation cost is necessary to generate accurate wave-effects (diffraction, scattering, and interference), which cannot be computed accurately by current geometric systems. These acoustic effects are necessary in VR to generate better sense of presence (Larsson et al., 2008). Such precomputation-based methods are widely used for visual effects as well as in games and VR.

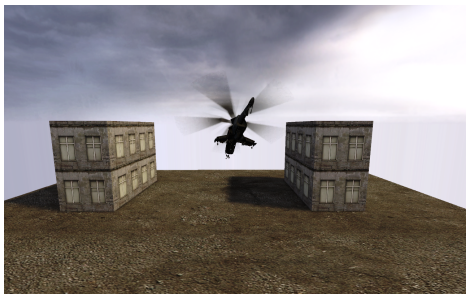
7.3 User Evaluation of Realism of Audio and Audio-Visual Correlation

A preliminary user evaluation is conducted to study the effect of source directivity and spatial audio on the realism of audio and audio-visual correlation in the virtual environments. In this study, a comparison is made between the sound generated by (a) wave-based sound propagation technique without source directivity and spatial audio (called *base* method) and (b) wave-based sound propagation technique with integrated source directivity and spatial audio framework (called *our* method). For this experiment, the equivalent source technique (ESM) discussed in Chapter 3 is

chosen as the wave-based sound propagation technique and the directivity framework proposed in Chapter 4 is used to integrate directional sources and spatial audio.

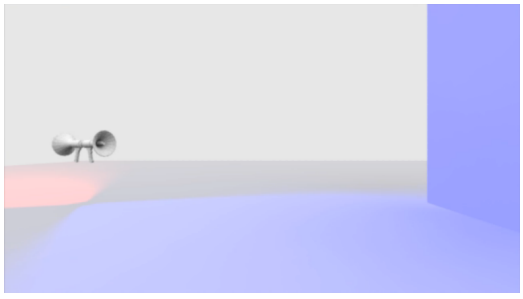
Study Design The study was designed for four comparison cases: *base vs. base*, *our vs. our*, *base vs. our* and *our vs. base*, which were tested over three benchmark scenes: parallel walls, reservoir, and rotating (see Figure 7.4). For each comparison case and each benchmark scene, a pair of videos is generated with identical visuals but different audio as computed by the two sound propagation techniques corresponding to the comparison case. In total, twelve such video pairs are generated (4 comparison cases x 3 benchmarks).

The benchmark scenes used in this study were chosen based on the number and types of sound effects demonstrated be each scenario. The simplest scene "rotating" demonstrated rotating source, reflection, source directivity and spatial audio. The next scene "reservoir" showed a listener rotating in place and multiple sources along with sound effects such as diffraction and scattering on top of the effects described before. The last scene "parallel walls" incorporated moving source, high-order reflections and diffraction low passing along with source directivity and spatial audio.



(a) walls

(b) reservoir



(c) rotating

Figure 7.4: Benchmark scenes used in the user study.

The study was conducted in form of an online survey where each subject was shown the 12 video pairs in random order. For each video pair, users were asked two questions: (a) "*Which of the two videos contains more realistic audio?*" and (b) "*Which of the two videos contains audio that matches better with the visuals shown?*".

Figure 7.5: Online survey used for the user study. Users were shown 12 pairs of videos with each pair having identical visuals but different audio depending on the propagation technique used. The users were asked to rate the realism of audio and audio-visual correlation on a scale of 1 to 11.

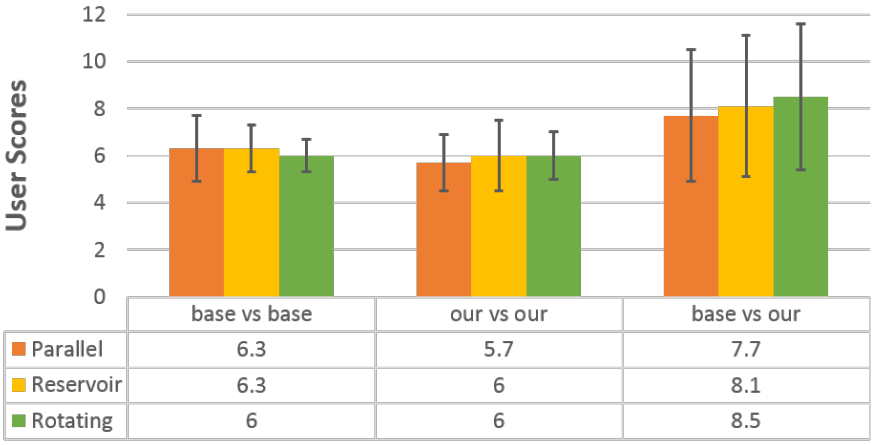
Research Hypotheses Our research hypotheses were: 1) Sound produced by *our* method will result in more realistic audio and better audio-visual correlation than that produced by the *base* method. 2) The level of increase in realism and audio-visual correlation for *our* method as compared to the *base* method would depend upon the type of scene.

Procedure The study was conducted for a total of 43 subjects, all between the age of 18 and 48, made up of 27 males and 16 females. The mean age of the group is 27.8, and all of them had

normal hearing and normal or corrected-to-normal vision. The average number of hours per week the subjects listened to music was 16.5. Before starting the study, the subjects were given detailed instructions and filled out a background survey questionnaire. Since the videos contained stereo audio, the subjects were required to use either headphones or earphones. The subjects were also asked to make sure they were wearing the headphones/earphones in correct orientation (left channel in left ear and so on). The next stage was the volume calibration session, in which the subjects were asked to play a test audio clip and change the volume of the audio system until the audio clip is barely audible. Subjects then started the study, in which they were presented with 12 video pairs in random order and asked to rate each pair on a scale of 1 to 11 on two separate questions. A 1 response meant a strong preference for the first clip; 6 meant equal preference; and 11 meant a strong preference for the second clip. The subjects were allowed as much time as needed and were free to take a break at any time. The study concluded after the subject finished rating all the 12 videos. After completing the study, the subjects were encouraged to give feedback and thanked for their time and efforts.

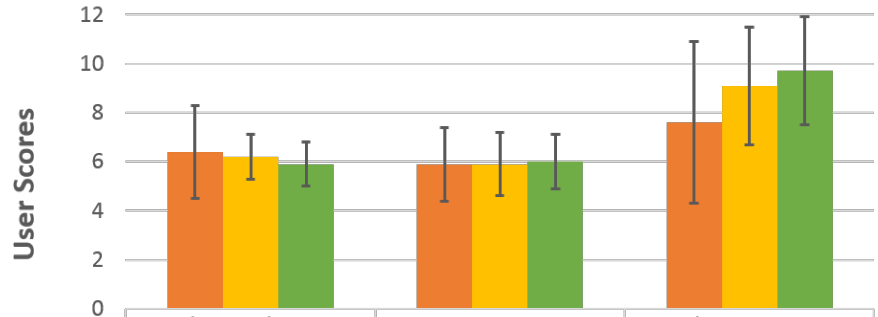
Evaluation Figure 7.6 shows the mean and standard deviation of the subjects' scores for the two questions. The scores of *our vs. base* comparison case were reversed and combined with *base vs. our*. For the *base vs. base* and *our vs. our* case, the subjects showed equal preference, as expected, exemplified by the mean scores ranging from 5.7 to 6.4. However, for the *base vs. our* case, the subjects showed an increasing preference for *our* method as exemplified by the higher mean scores (ranging from 7.6 to 9.7). This is consistent with hypothesis 1. In addition, the mean scores of *base vs. our* case increased from parallel to the rotating scene-type, consistent with hypothesis 2. This may be due to the fact that in the parallel wall scene, which contains multiple propagation effects (diffraction low-passing, moving source, rotating listener, source and listener directivity) all happening at the same time, the effect of directivity was masked out to some extent. In the rotating scene, source and listener directivity were the dominant propagation effects present in the scene, resulting in comparatively higher scores.

The scores were analysed with a two-way analysis of variance (ANOVA). For the question regarding audio-visual correlation, significant effects were found for both the comparison case ($F(2,84)=67.36$, $p<0.0001$) and scene-type ($F(2,84)=3.72$, $p=0.0284$). A significant interaction between the case and scene-type factors was found as well ($F(4,168)=8.18$, $p<0.0001$). This is



	base vs base	our vs our	base vs our
Parallel	6.3	5.7	7.7
Reservoir	6.3	6	8.1
Rotating	6	6	8.5

(a) "Level of realistic audio"

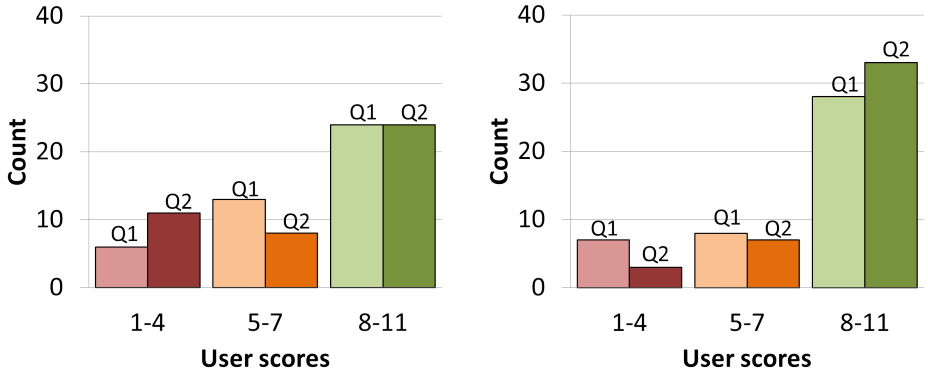


	base vs base	our vs our	base vs our
Parallel	6.4	5.9	7.6
Reservoir	6.2	5.9	9.1
Rotating	5.9	6	9.7

(b) "Audio-visual correlation"

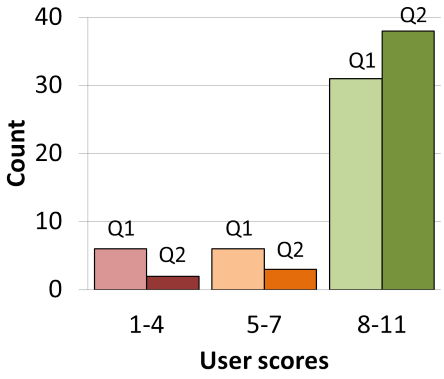
Figure 7.6: Mean and standard deviation of subject's responses are tabulated for the two questions regarding "realistic audio" and "audio-visual correlation". The term **base** refers to the wave-based sound propagation without directivity; **our** refers to wave-based propagation with our directivity approach integrated.

demonstrated in Figure 7.6(b) where the variation of the scores with scene-type changes for different comparison-cases. For the question regarding realistic audio, a significant effect was found for the comparison case ($F(2,84)=35.78$, $p<0.0001$). However, no significant effect was found for the scene-type ($F(2,84)=0.86$, $p=0.42$) and the interaction between case and scene-type ($F(4,168)=1.05$, $p=0.38$). In this case, even though the scores for the *base vs. our* case is higher than for the other two cases, the variation of the scores with scene-type does not change as significantly as before for the different comparison cases.



(a) Parallel

(b) Reservoir



(c) Rotating

Figure 7.7: Histogram of subject scores for the comparison between the *base* method (without directivity) and *our* method (with directivity). Q1 and Q2 refers to the question regarding the level of realistic audio and the level of audio-visual correlation respectively.

In Figure 7.7, the histograms of the scores are plotted for the *base vs. our* case for the three scenes. The scores are distributed in three bins: (1-4) means preference for *base* method, (5-7) means equal preference and (8-11) means preference for *our* method. For the question regarding realism, 56%, 65% and 76% of the participants preferred *our* method for the three scenes respectively. Furthermore, for audio-visual correlation, 56%, 76% and 88% of the participants preferred *our* method. These results show that the majority of subjects found the sound generated using our directivity approach to be more realistic and that it correlated better with the visuals as compared to sound generated without directivity. This is a preliminary user study and we plan to conduct a more extensive user evaluation of our technique in the future.

7.4 Conclusion

In this Chapter, we performed user evaluation to study the effect of a wave-based sound propagation and source and listener directivity on user's sense of immersion in the virtual environment. In the first case, we compared wave-based sound propagation technique with a geometric sound propagation technique and demonstrated that wave-effects help increase task performance in virtual environments. We also showed that wave-based spatial audio further increased task performance in such environments. In the second case, we compared a user's evaluation of the realism of audio and audio-visual correlation for a wave-based sound propagation technique with and without directivity. We demonstrated that directivity helps increase both these metrics in the users.

Limitation and Future Work: The user studies discussed in this chapter are preliminary studies and we plan to conduct extensive user evaluation in future. We would like to conduct a detailed user study with more scene configurations and additional conditions. We would also like to incorporate additional evaluation strategies, such as a presence questionnaire, and conduct user evaluations with a larger number of subjects to better assess our method's qualitative benefits. In addition to this, we would like to study each sound effect, such as reflection, diffraction, interference, reverberation, in isolation and in different combinations with each other to understand the impact of sound effect on a user's sense of presence and immersion in the VR environment.

CHAPTER 8: CONCLUSIONS

In this dissertation, we have presented techniques that enable wave-based sound propagation for interactive applications. These techniques solve the many challenges faced by prior wave-based propagation techniques. These techniques can perform wave-based sound propagation in large environments, handle time-varying source and listener directivity, and solve the wave equation efficiently for mid-frequencies. The computational complexity and runtime memory requirements of these techniques is significantly lower than for prior wave-based techniques enabling interactive performance. We have validated the accuracy of these techniques using state-of-the-art offline numerical solvers, analytical solutions, and real-world measurements. We have also performed preliminary user evaluations to study the effect of these techniques on a user's perceived sense of immersion in the VR environments. Our results indicate that the wave-based sound propagation and accurate modeling of source and listener directivity increases a user's task performance and sense of realism in the virtual environments.

8.1 Summary of Results

We have presented a novel equivalent source method (ESM) for wave-based sound propagation in large open scenes. In contrast with prior volumetric or surface-based work, this is an object-centric method for performing sound propagation. The computational complexity and memory requirements of the technique scales with the number of objects in the scene rather than volume or surface area. This makes the proposed ESM technique very attractive for large open scenes with high volume or surface area but with a sparse collection of objects. The runtime memory requirement of this technique is three orders of magnitude smaller than for prior wave-based approaches. This technique can perform wave-based sound propagation in large scenes spanning hundreds of meters within a computational budget of tens of milliseconds and memory budget of tens of megabytes. This

technique has been integrated with the Valve's Source^{TM} game engine to demonstrate realistic acoustic effects such as diffraction, low-passed sound behind obstructions, focusing, scattering, high-order reflections, and echoes, on a variety of scenes.

Secondly, we have proposed a source and listener directivity framework to incorporate time-varying directivity in interactive wave-based sound propagation techniques. This framework is based on a spherical harmonic representation of the directivity function that can handle analytical, measured or simulated directivity. We have also derived a novel theoretical relationship that relates the coefficients of a plane-wave decomposition of the sound field with the derivatives of the sound field. This enables spatial audio computations for moving listeners at interactive rates. The effect of the source and listener directivity on sound propagation is evaluated in multiple scenarios such as people talking on the street, loudspeakers between buildings, a television in a living room, a helicopter in a rocky outdoor terrain, a bell tower in a snow-covered town, a rotating siren, and musical instruments in an amphitheater.

Lastly, we have presented a parallel GPU-based time-domain solver for the acoustic wave equation to perform efficient precomputation for wave-based sound propagation in mid-frequencies. It is based on the adaptive rectangular decomposition algorithm that uses analytical solutions of the wave equation inside cuboidal regions. The precomputation time of this parallel GPU-based ARD solver is an order of magnitude smaller than the corresponding CPU-based solver and three orders of magnitude smaller than for the CPU-based FDTD solver. The performance of this solver scales linearly with the number of GPU cores. We have demonstrated that by carefully mapping all the steps of the algorithm to the parallel processing capabilities of the GPU, significant improvement in performance can be achieved.

8.2 Limitations

In this section, we discuss the limitations of the proposed techniques. The equivalent source method is an object-centric approach to wave-based sound propagation whose computational and memory complexity scales with the number of objects in the scene. This technique performs well for large scenes with well-separated objects such as deserts or villages. However, for scenes where all the objects are in close vicinity to each other (such as the downtown of cities or indoor spaces), the ESM

technique does not hold computational advantage over other techniques. In such scenarios, since all the objects are in close-proximity, they have to be treated as a single object and the simulation domain extends to the entire region. Secondly, due to the high cost of computation for the inter-object transfer functions, the current ESM technique only supports scenes with static objects. Therefore, the current ESM technique cannot handle dynamic objects such as moving cars. Also, the current runtime system does not model Doppler shift caused by the motion of fast-moving sources.

Our sound and listener directivity framework is based on spherical harmonic functions. These functions can efficiently represent smooth directivity functions over the sphere. In case of sharp directivities, the number of SH coefficients grows significantly making the current approach computationally inefficient for sound propagation for sources with sharp directivities. The current listener directivity framework represents the head-related transfer functions (HRTFs) as a spherical harmonic expansion which is only valid for HRTFs measured in far-field. In case of near-field, the HRTFs have a distance dependence which is not handled by the current approach.

The current GPU-based ARD solver is based on the shared-memory architecture. This means that the simulator assumes that the simulation data structures fit in the main memory of the GPU. Even though the GPU main memory has increased significantly over the years (up to 12 GB), the memory requirement to run acoustic simulations on large scenes (hundreds or thousands of meters) or high frequencies can easily reach Terabytes. This requires a distributed memory implementation on a CPU cluster. Therefore, the current GPU-ARD solver is restricted to medium-sized scenes spanning tens of meters and maximum simulation frequency of 1-2 kHz. Secondly, the ARD algorithm uses analytical solutions of wave equation inside cuboidal regions to solve the acoustic wave equation. These analytical solutions are based on the homogenous media assumption i.e. spatially-constant speed of sound. This limitation is also applicable to the GPU-ARD solver. Therefore, the GPU-ARD technique cannot be used to perform sound propagation in cases where the speed of sound changes with distance, for e.g. underwater acoustics.

8.3 Future Work

There are many avenues of future work. In order to support dynamic objects in the equivalent source method, it would be interesting to explore the Fast Multipole Method to enable fast evaluations

of inter-object transfer functions. Currently, the ESM technique for wave-based sound propagation is computationally limited to the low and mid-frequency range. Geometric techniques become accurate for the high frequency range (greater than few kHz). It would be interesting to explore novel hybrid techniques that combine wave-based and geometric techniques to compute sound propagation for the entire range of human hearing. The current ESM technique takes MBs of memory for scenes of size spanning hundreds of meters. However, for massive scenes spanning kilometers, the ESM technique would require Gigabytes of memory. We hope that by using far-field approximations of the pressure field generated by equivalent sources, we can further reduce the memory requirements of this technique. We would also like to tightly couple the sound propagation technique with sound synthesis and radiation techniques to enable a coupled sound synthesis-propagation system. This would enable the use of mechanical models of object vibration along with pre-recorded audio clips as sound sources.

In the case of our source and listener directivity framework, we want to explore other basis functions which might be more appropriate for sharp directivities such as wavelets. We would also like to develop a spatial audio formulation that supports near-field HRTFs by using single-point multipole expansion to express propagating sound fields as well as the HRTFs. The current directivity results are based on magnitude-only directivity data. In the future, we want to record and utilize both magnitude and phase data to represent the pattern of complex directional sound sources. Our listener directivity formulation supports the use of personalized HRTFs for an individual. However, measuring a personalized HRTF requires specialized equipment and an anechoic chamber. We want to explore HRTF personalization approaches based on directly running a numerical simulation on the 3D meshes of the head and torso model of the user.

We want to develop a parallel ARD solver based on a distributed-memory architecture to enable wave-based sound propagation for massive scenes and higher frequencies. We want to develop a scalable solver which can be parallelized on thousands or tens of thousands of cores available on a CPU cluster. This would involve developing approaches to handle load balancing and communication cost hiding. Another direction for future work is to extend the ARD solver to support non-homogenous media which consists of layers of homogenous regions for e.g. outdoor acoustics and underwater acoustics. This would require developing new boundary conditions to couple the solutions in the two regions with different speeds of propagation.

APPENDIX A: EQUIVALENT SOURCE METHOD FOR SOUND PROPAGATION

A.1 Two-object Steady State Field Solution

We describe in detail the way we compute the equivalent source strengths for a scene composed of two objects. Consider a scene with objects A and B and a sound source s. Let the incoming field multipoles for A and B be Φ_A^{in} and Φ_B^{in}, respectively. Similarly, let the multipoles for the outgoing field for A and B be Φ_A^{out} and Φ_B^{out}, respectively. The scattering matrices for A and B are T_A and T_B, respectively. Let the interaction matrices for the objects be G_A^B and G_B^A, respectively. First of all, we express the incoming field produced by sound source s on objects A and B in terms of their incoming field multipoles :

$$\begin{aligned} s_A^{in} &= \sum_{i=1}^{\mathbb{Q}} \sum_{k=1}^{M^2} a_{ik} \varphi_{ik}^{in} = S_A^{tr}\ \Phi_A^{in}; \\ s_B^{in} &= \sum_{i=1}^{\mathbb{Q}} \sum_{k=1}^{M^2} b_{ik} \varphi_{ik}^{in} = S_B^{tr}\ \Phi_B^{in}. \end{aligned}$$

Now assume that the steady state outgoing field of object A and B is P_A^{out} and P_B^{out} respectively.

$$P_A^{out} = \sum_{j=1}^{\mathbb{P}} \sum_{h=1}^{N^2} c_{jh}^A \varphi_{jh}^{out} = C_A^{tr} \Phi_A^{out}; \tag{A.1}$$

$$P_B^{out} = \sum_{j=1}^{\mathbb{P}} \sum_{h=1}^{N^2} c_{jh}^B \varphi_{jh}^{out} = C_B^{tr} \Phi_B^{out}. \tag{A.2}$$

The outgoing field of one object becomes the incoming field for the other object. Exploiting the linearity of the inter-object transfer function and (3.17), we find the incoming field for B produced by the outgoing field of A as

$$\widehat{P}_B^{in} = g_A^B(C_A^{tr} \Phi_A^{out}) = C_A^{tr}\ G_A^B\ \Phi_B^{in}.$$

Similarly, we find the incoming field for A produced by the outgoing field of B as

$$\widehat{P}_A^{in} = g_B^A(C_B^{tr} \Phi_B^{out}) = C_B^{tr}\ G_B^A\ \Phi_A^{in}.$$

The total incoming fields on objects A and B are given by

$$\begin{aligned} P_A^{in} &= s_A^{in} + \widehat{P}_A^{in} = S_A^{tr}\ \Phi_A^{in}\ +\ C_B^{tr}\ G_B^A\ \Phi_A^{in}; \\ P_B^{in} &= s_B^{in} + \widehat{P}_B^{in} = S_B^{tr}\ \Phi_B^{in}\ +\ C_A^{tr}\ G_A^B\ \Phi_B^{in}. \end{aligned}$$

Applying the linearity of per-object transfer function f and using (3.12), we get outgoing pressure P_{out}^A and P_{out}^B due to the scattering of incoming fields by the objects as

$$P_A^{out} = f(P_A^{in}) = (S_A^{tr}\ T_A\ +\ C_B^{tr}\ G_B^A\ T_A)\ \ \Phi_A^{out}, \tag{A.3}$$

$$P_B^{out} = f(P_B^{in}) = (S_B^{tr}\ T_B\ +\ C_A^{tr}\ G_A^B\ T_B)\ \ \Phi_B^{out}. \tag{A.4}$$

In steady state, this outgoing pressure should match the outgoing pressure we started with. Equating (A.3) with (A.1), and (A.4) with (A.2), we get

$$\begin{aligned} C_A^{tr} &= S_A^{tr}\ T_A\ +\ C_B^{tr}\ G_B^A\ T_A; \\ C_B^{tr} &= S_B^{tr}\ T_B\ +\ C_A^{tr}\ G_A^B\ T_B. \end{aligned}$$

Combining the above two equations, and rearranging, we obtain

$$\begin{bmatrix} C_A \\ C_B \end{bmatrix} = \begin{bmatrix} T_A^{tr} & 0 \\ 0 & T_B^{tr} \end{bmatrix} \left(\begin{bmatrix} 0 & (G_B^A)^{tr} \\ (G_A^B)^{tr} & 0 \end{bmatrix} \begin{bmatrix} C_A \\ C_B \end{bmatrix} + \begin{bmatrix} S_A \\ S_B \end{bmatrix} \right).$$

In other words,

$$(\mathbf{I} - \mathbf{TG})\mathbf{C} = \mathbf{TS}, \tag{A.5}$$

which is a linear system $Ax = b$. We solve this linear system to get the outgoing equivalent source strengths $\mathbf{C}$. At runtime, the outgoing scattered field at any listener position $\mathbf{x}$ is given by

$$p(\mathbf{x}) = C_A^{tr}\ \Phi_A^{out}(\mathbf{x})\ +\ C_B^{tr}\ \Phi_B^{out}(\mathbf{x}). \tag{A.6}$$

The total pressure field becomes

$$p(\mathbf{x}) = C_A^{tr}\ \Phi_A^{out}(\mathbf{x})\ +\ C_B^{tr}\ \Phi_B^{out}(\mathbf{x})\ +\ s(\mathbf{x}). \tag{A.7}$$

A.2 Multiple Objects Steady State Field Solution

For a scene with κ objects, $A_1, A_2, ..., A_\kappa$, equation (A.5) remains the same except the vectors and matrices are generalized for κ objects,

$$(\mathbf{I} - \mathbf{TG})\mathbf{C} = \mathbf{TS}, \tag{A.8}$$

where

$$\mathbf{S} = \begin{bmatrix} S_{A_1} \\ S_{A_2} \\ . \\ . \\ S_{A_g} \end{bmatrix}, \mathbf{C} = \begin{bmatrix} C_{A_1} \\ C_{A_2} \\ . \\ . \\ C_{A_g} \end{bmatrix}, \mathbf{T} = \begin{bmatrix} T_{A_1}^{tr} & 0 & . & . & 0 \\ 0 & T_{A_2}^{tr} & 0 & . & . \\ . & . & . & . & . \\ . & . & . & 0 & T_{A_g}^{tr} \end{bmatrix}, \mathbf{G} = \begin{bmatrix} (0 & (G_{A_2}^{A_1})^{tr} & . & . & (G_{A_g}^{A_1})^{tr} \\ (G_{A_1}^{A_2})^{tr} & 0 & . & . & (G_{A_g}^{A_2})^{tr} \\ . & . & . & . & . \\ (G_{A_1}^{A_g})^{tr} & (G_{A_2}^{A_g})^{tr} & . & . & 0 \end{bmatrix}.$$

The total pressure field becomes

$$p(\mathbf{x}) = \sum_{j=1}^{\kappa} C_{A_j}^{tr}\ \Phi_{A_j}^{out}(\mathbf{x})\ +\ s(\mathbf{x}). \tag{A.9}$$

A.3 Computational Complexity

BEM The storage requirements of BEM depends on the total surface area S of the objects in the scene and the number of frequency samples $\nu_{\max}/\Delta\nu$. Assuming BEM places τ samples per wavelength (usually $\tau = 12$), the number of BEM elements placed on the object's surface at frequency sample ν_i is equal to $S\tau^2\nu_i^2/c^2$. The total number of BEM elements for all the frequency samples is equal to $S\tau^2\nu_{\max}^3/(3c^2\Delta\nu)$, where each element is specified by its position (3 floats) and four complex amplitudes corresponding to pressure and its gradient (2 floats each). Total memory

requirement of storing the simulation results becomes

$$11S\tau^2\nu_{\max}^3/(3c^2\Delta\nu).$$

ARD and FDTD The runtime memory requirements of ARD and FDTD are equal to the number of grid cells in the spatial discretization of the entire volume of the scene and the number of timesteps in the simulation. Assuming volume of the scene to be V, the grid size h, the maximum frequency $\nu_{\max}$, the speed of sound c, and the number of samples per wavelength τ (equal to 3 for ARD and 10 for FDTD), the number of grid cells are $(\tau\nu_{\max}/c)^3V$. The total number of time samples to store is at least twice the number of samples in the frequency domain. The total memory requirement of storing the simulation results for these techniques is thus

$$2\tau^3\nu_{\max}^4V/(c^3\Delta\nu).$$

BIBLIOGRAPHY

Aach, J. and Church, G. M. (2001). Aligning gene expression time series with time warping algorithms. *Bioinformatics*, 17(6):495–508.

Abdou, A. and Guy, R. W. (1996). Spatial information of sound fields for room acoustics evaluation and diagnosis. *The Journal of the Acoustical Society of America*, 100(5):3215–3226. Available from: http://scitation.aip.org/content/asa/journal/jasa/100/5/10.1121/1.417205.

Abramowitz, M. and Stegun, I. (1964). *Handbook of Mathematical Functions*. Dover, New York, fifth edition.

Ahnert, Wolfgang; Feistel, S. (2005). The significance of phase data for the acoustic prediction of combinations of sound sources. In *Audio Engineering Society Convention 119*. Available from: http://www.aes.org/e-lib/browse.cfm?elib=13368.

Ahrens, J. and Spors, S. (2012). Wave field synthesis of a sound field described by spherical harmonics expansion coefficients. *The Journal of the Acoustical Society of America*, 131(3):2190–2199. Available from: http://link.aip.org/link/?JAS/131/2190/1.

Algazi, V., Duda, R., Thompson, D., and Avendano, C. (2001). The CIPIC HRTF database. In *Applications of Signal Processing to Audio and Acoustics, 2001 IEEE Workshop on the*, pages 99 –102.

Alghamdi, A., Ahmadia, A., Ketcheson, D. I., Knepley, M. G., Mandli, K. T., and Dalcin, L. (2011). Petclaw: A scalable parallel nonlinear wave propagation solver for python. In *Proceedings of the 19th High Performance Computing Symposia*, HPC '11, pages 96–103, San Diego, CA, USA. Society for Computer Simulation International. Available from: http://dl.acm.org/citation.cfm?id=2048577.2048590.

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

ANSOL (2014). Spherical Wave Scattering by Rigid Sphere (date last viewed 07/27/2014). *http://ansol.us/Products/Coustyx/Validation/MultiDomain/Scattering/SphericalWave/HardSphere*.

Avni, A. (2010). Spaciousness of sound fields captured by spherical microphone arrays. Master's thesis, Bengurion University of the Negev Faculty of Engineering Sciences.

Bao, H., Bielak, J., Ghattas, O., Kallivokas, L. F., O'Hallaron, D. R., Shewchuk, J. R., and Xu, J. (1998). Large-scale simulation of elastic wave propagation in heterogeneous media on parallel computers. *Computer Methods in Applied Mechanics and Engineering*, 152(1?2):85 – 102. Containing papers presented at the Symposium on Advances in Computational Mechanics. Available from: http://www.sciencedirect.com/science/article/pii/S0045782597001837.

Begault, D. R. (1994). *3D Sound for Virtual Reality and Multimedia*. Academic Press.

Bernacki, M., Fezoui, L., Lanteri, S., and Piperno, S. (2006). Parallel discontinuous galerkin unstructured mesh solvers for the calculation of three-dimensional wave propagation problems. *Applied Mathematical Modelling*, 30(8):744 – 763. Parallel and distributed computing for computational mechanics. Available from: http://www.sciencedirect.com/science/article/pii/S0307904X0500212X.

Bhandarkar, M. and Kal, L. (2000). A parallel framework for explicit fem. In Valero, M., Prasanna, V., and Vajapeyam, S., editors, *High Performance Computing ? HiPC 2000*, volume 1970 of *Lecture Notes in Computer Science*, pages 385–394. Springer Berlin Heidelberg. Available from: http://dx.doi.org/10.1007/3-540-44467-X_35.

Borish, J. (1984). Extension of the image model to arbitrary polyhedra. *The Journal of the Acoustical Society of America*, 75(6):1827–1836. Available from: http://scitation.aip.org/content/asa/journal/jasa/75/6/10.1121/1.390983.

Boyd, J. P. (2001). Chebyshev and Fourier Spectral Methods. *Dover Publications, NY, USA,*, pages 1–668.

Caiani, E., Porta, A., Baselli, G., Turiel, M., Muzzupappa, S., Pieruzzi, F., Crema, C., Malliani, A., and Cerutti, S. (1998). Warped-average template technique to track on a cycle-by-cycle basis the cardiac filling phases on left ventricular volume. In *Computers in Cardiology*, pages 73–76.

Chadwick, J. N., An, S. S., and James, D. L. (2009). Harmonic shells: a practical nonlinear sound model for near-rigid thin shells. NY, USA. ACM. Available from: http://doi.acm.org/10.1145/1661412.1618465.

Chandak, A., Antani, L., and Manocha, D. (2012). Ipl sdk: Software development kit for efficient acoustics simulation. In *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, volume 2012, pages 7938–7949. Institute of Noise Control Engineering.

Chandak, A., Lauterbach, C., Taylor, M., Ren, Z., and Manocha, D. (2008). Ad-frustum: Adaptive frustum tracing for interactive sound propagation. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1707–1722.

Cheng, A. and Cheng, D. (2005). Heritage and early history of the boundary element method. *Engineering Analysis with Boundary Elements*, 29(3):268–302. Available from: http://dx.doi.org/10.1016/j.enganabound.2004.12.001.

Colton, D. and Kress, R. (2013). *Integral Equation Methods in Scattering Theory*. Society for Industrial and Applied Mathematics, Philadelphia, PA. Available from: http://epubs.siam.org/doi/abs/10.1137/1.9781611973167.

Dadoun, N., Kirkpatrick, D. G., and Walsh, J. P. (1985). The geometry of beam tracing. In *Proceedings of the First Annual Symposium on Computational Geometry*, SCG '85, pages 55–61, New York, NY, USA. ACM. Available from: http://doi.acm.org/10.1145/323233.323241.

Damian, K. (2010). The Next Big Steps In Game Sound Design, http://www.gamasutra.com/view/feature/4257/the_next_big_steps_in_game_sound_.php. Available from: http://www.gamasutra.com/view/feature/4257/the_next_big_steps_in_game_sound_.php.

Diekmann, R., Dralle, U., Neugebauer, F., and Rmke, T. (1996). Padfem: A portable parallel fem-tool. In Liddell, H., Colbrook, A., Hertzberger, B., and Sloot, P., editors, *High-Performance Computing and Networking*, volume 1067 of *Lecture Notes in Computer Science*, pages 580–585. Springer Berlin Heidelberg. Available from: http://dx.doi.org/10.1007/3-540-61142-8_599.

Doicu, A., Eremin, Y. A., and Wriedt, T. (2000). *Acoustic and Electromagnetic Scattering Analysis Using Discrete Sources*. Academic Press, 1st edition. Available from: http://www.worldcat.org/isbn/0122197402.

Dolby (2010). Dolby Axon, https://axon.dolby.com/in_games.php. Available from: https://axon.dolby.com/in_games.php.

Funkhouser, T., Carlbom, I., Elko, G., Pingali, G., Sondhi, M., and West, J. (1998). A beam tracing approach to acoustic modeling for interactive virtual environments. In *ACM SIGGRAPH*, pages 21–32.

Funkhouser, T., Tsingos, N., and Jot, J.-M. (2003). Survey of methods for modeling sound propagation in interactive virtual environment systems. *Presence and Teleoperation*. Available from: http://www-sop.inria.fr/reves/Basilic/2003/FTJ03.

Govindaraju, N., Gray, J., Kumar, R., and Manocha, D. (2006). GPUTeraSort: high performance graphics co-processor sorting for large database management. In *ACM SIGMOD '06*, pages 325–336, New York, NY, USA. ACM.

Govindaraju, N. K., Lloyd, B., Dotsenko, Y., Smith, B., and Manferdelli, J. (2008). High performance discrete Fourier transforms on graphics processors. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, SC '08, pages 1–12, Piscataway, NJ, USA. IEEE Press. Available from: http://dx.doi.org/10.1145/1413370.1413373.

Green, R. (2003). Spherical Harmonic Lighting: The Gritty Details. *Archives of the Game Developers Conference*. Available from: http://www.research.scea.com/gdc2003/spherical-harmonic-lighting.pdf.

Guiffaut, C. and Mahdjoubi, K. (2001). A parallel fdtd algorithm using the mpi library. *Antennas and Propagation Magazine, IEEE*, 43(2):94–103.

Gumerov, N. A. and Duraiswami, R. (2009). A broadband fast multipole accelerated boundary element method for the three dimensional Helmholtz equation. *The Journal of the Acoustical Society of America*, 125(1):191–205. Available from: http://scitation.aip.org/getabs/servlet/GetabsServlet?prog=normal&id=JASMAN000125000001000191000001&idtype=cvips&gifs=yes.

Hacihabiboglu, H., Gunel, B., and Kondoz, A. (2008). Time-domain simulation of directive sources in 3-d digital waveguide mesh-based acoustical models. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(5):934–946.

Hasegawa, T., Ochi, M., and Matsuzawa, K. (1980). Acoustic radiation pressure on a rigid sphere in a spherical wave field. *The Journal of the Acoustical Society of America*, 67(3):770–773.

Hidaka, T., Beranek, L. L., and Okano, T. (1995). Interaural cross-correlation, lateral fraction, and low- and high-frequency sound levels as measures of acoustical quality in concert halls. *The Journal of the Acoustical Society of America*, 98(2):988–1007. Available from: http://scitation.aip.org/content/asa/journal/jasa/98/2/10.1121/1.414451.

Hobson, E. W. (1955). *The Theory of Spherical and Ellipsoidal Harmonics*. Cambridge University Press, New York, NY, USA.

Hughes, D. E., Thropp, J., Holmquist, J., and Moshell, J. M. (2006). Spatial perception and expectation: Factors in acoustical awareness for mout training. *Proceedings of the 24th US Army Science Conference: Transformational Science and Technology for the Current and Future Force*, pages 339–343.

IASIG (1999). Interactive audio special interest group : Interactive 3d audio rendering guidelines, level 2.0. Available from: http://www.iasig.org/pubs/3dl2v1a.pdf.

James, D. L., Barbič, J., and Pai, D. K. (2006). Precomputed acoustic transfer: output-sensitive, accurate sound generation for geometrically complex vibration sources. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, pages 987–995, New York, NY, USA. ACM. Available from: http://doi.acm.org/10.1145/1179352.1141983.

Jers, H. (2005). Directivity of singers. *The Journal of the Acoustical Society of America*, 118(3):2008–2008. Available from: http://link.aip.org/link/?JAS/118/2008/2.

Kino, G. (1987). *Acoustic waves: devices, imaging, and analog signal processing*. Prentice-Hall Signal Processing Series. Prentice Hall PTR. Available from: http://books.google.com/books?id=hcsYAQAAIAAJ.

Kirk, D. B. and Hwu, W.-m. W. (2010). *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition.

Komatitsch, D., Erlebacher, G., Gddeke, D., and Micha, D. (2010). High-order finite-element seismic wave propagation modeling with {MPI} on a large {GPU} cluster. *Journal of Computational Physics*, 229(20):7692 – 7714. Available from: http://www.sciencedirect.com/science/article/pii/S0021999110003396.

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

Kowalczyk, K. and van Walstijn, M. (2010). Room acoustics simulation using 3-D compact explicit FDTD schemes. *IEEE Transactions on Audio, Speech and Language Processing*.

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

Larsson, P., Västfjäll, D., and Kleiner, M. (2008). Effects of auditory information consistency and room acoustic cues on presence in virtual environments. *Acoustical Science and Technology*, 29(2):191–194.

Lauterbach, C., Chandak, A., and Manocha, D. (2007). Interactive sound rendering in complex and dynamic scenes using frustum tracing. *IEEE Transactions on Visualization and Computer*

Graphics, 13(6):1672–1679. Available from: http://dx.doi.org/10.1109/TVCG.2007.70567.

Liu, Q. H. (1997). The PSTD algorithm: A time-domain method combining the pseudospectral technique and perfectly matched layers. *The Journal of the Acoustical Society of America*, 101(5):3182. Available from: http://dx.doi.org/10.1121/1.419176.

Liu, Y. (2009). *Fast Multipole Boundary Element Method: Theory and Applications in Engineering*. Cambridge University Press. Available from: http://books.google.com/books?id=UnRmPgAACAAJ.

Liu, Y., Shen, L., and Bapat, M. (2009). Development of the fast multipole boundary element method for acoustic wave problems. In *Recent Advances in Boundary Element Methods*, pages 287–303. Springer Netherlands. Available from: http://dx.doi.org/10.1007/978-1-4020-9710-2_19.

Lokki, T., Svensson, P., and Savioja, L. (2002). An efficient auralization of edge diffraction. In *Architectural Acoustics and Sound Reinforcement Conf.* Available from: http://www.aes.org/e-lib/browse.cfm?elib=11207.

Malham, D. G. (1999). Higher order ambisonic systems for the spatialization of sound. *ICMC Proceedings*.

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*.

McNamara, D., Pistorius, C., and Malherbe, J. (1990). *Introduction to the Uniform Geometrical Theory of Diffraction*. Antennas and Propagation Library. Artech House, Incorporated. Available from: http://books.google.com/books?id=bn6pQgAACAAJ.

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

Mehra, R., Raghuvanshi, N., Antani, L., Chandak, A., Curtis, S., and Manocha, D. (2013). Wave-based sound propagation in large open scenes using an equivalent source formulation. *ACM Trans. Graph.* Available from: http://doi.acm.org/http://dx.doi.org/10.1145/2451236.2451245.

Mehra, R., Raghuvanshi, N., Savioja, L., Lin, M. C., and Manocha, D. (2012). An efficient gpu-based time domain solver for the acoustic wave equation. *Applied Acoustics*, 73(2):83 – 94.

Menzies, Dylan; Al-akaidi, M. (2007). Ambisonic synthesis of complex sources. *J. Audio Eng. Soc*, 55(10):864–876. Available from: http://www.aes.org/e-lib/browse.cfm?elib=14175.

Meyer, J. and Elko, G. (2002). A highly scalable spherical microphone array based on an orthonormal decomposition of the soundfield. In *ICASSP*, volume 2, pages II–1781–II–1784.

Meyer, J. and Hansen, U. (2009). *Acoustics and the Performance of Music (Fifth edition)*. Lecture Notes in Mathematics. Springer. Available from: http://books.google.com/books?id=ruA3AQAAIAAJ.

Nickolls, J., Buck, I., Garland, M., and Skadron, K. (2008). Scalable parallel programming with cuda. *Queue*, 6(2):40–53.

Noisternig, N. and Katz, B. (2009). Reconstructing sound source directivity in virtual acoustic environments. In *International workshop on the Principles and Applicaions of Spatial Hearing (IWPASH)*.

NVIDIA (2010). CUDA Programming Guide (date last viewed 09/01/2010). *http://developer.download.nvidia.com/compute/cuda/3_1/toolkit/docs/NVIDIA_CUDA_C_ProgrammingGuide_3.1.pdf*, pages 114–116.

Ochmann, M. (1995). The source simulation technique for acoustic radiation problems. *Acustica*, 81:512–527.

Ochmann, M. (1999). The full-field equations for acoustic radiation and scattering. *The Journal of the Acoustical Society of America*, 105(5):2574–2584. Available from: `http://link.aip.org/link/?JAS/105/2574/1`.

Otondo, F. and Rindel, J. H. (2004). The influence of the directivity of musical instruments in a room. *Acta Acustica*, 90(6):1178–1184. Available from: `http://www.ingentaconnect.com/content/dav/aaua/2004/00000090/00000006/art00017`.

Owens, J. D., Luebke, D., Govindaraju, N., Harris, M., Krger, J., Lefohn, A. E., and Purcell, T. J. (2007). A Survey of General-Purpose Computation on Graphics Hardware. *Computer Graphics Forum*, 26(1):80–113.

Park, M. and Rafaely, B. (2005). Sound-field analysis by plane-wave decomposition using spherical microphone array. *The Journal of the Acoustical Society of America*, 118(5):3094–3103. Available from: `http://link.aip.org/link/?JAS/118/3094/1`.

Pavic, G. (2006). A technique for the computation of sound radiation by vibrating bodies using multipole substitute sources. *Acta Acustica united with Acustica*, 92:112–126(15). Available from: `http://www.ingentaconnect.com/content/dav/aaua/2006/00000092/00000001/art00012`.

Pellerin, C. (2011). Immersive technology fuels infantry simulators. *American Forces Press Service*.

Pierce, A. D. (1989). *Acoustics: An Introduction to Its Physical Principles and Applications*. Acoustical Society of America. Available from: `http://www.worldcat.org/isbn/0883186128`.

Pollow, M., Nguyen, K.-V., Warusfel, O., Carpentier, T., Muller-Trapet, M., Vorlander, M., and Noisternig, M. (2012). Calculation of head-related transfer functions for arbitrary field points using spherical harmonics decomposition. *Acta Acustica united with Acustica*, 98(1):72–82. Available from: `http://www.ingentaconnect.com/content/dav/aaua/2012/00000098/00000001/art00007`.

PTB (1978). http://www.ptb.de/cms/en/fachabteilungen/abt1/fb-16/ag-1630/room-acoustics/directivities.html (last viewed Oct. 23, 2012). Available from: `http://www.ptb.de/cms/en/fachabteilungen/abt1/fb-16/ag-1630/room-acoustics/directivities.html`.

Rafaely, B. and Avni, A. (2010). Interaural cross correlation in a sound field represented by spherical harmonics. *The Journal of the Acoustical Society of America*, 127(2):823–828. Available from: http://link.aip.org/link/?JAS/127/823/1.

Raghuvanshi, N., Lloyd, B., Govindaraju, N. K., and Lin, M. C. (2009a). Efficient Numerical Acoustic Simulation on Graphics Processors using Adaptive Rectangular Decomposition. In *EAA Symposium on Auralization*.

Raghuvanshi, N., Narain, R., and Lin, M. C. (2009b). Efficient and Accurate Sound Propagation Using Adaptive Rectangular Decomposition. *IEEE Transactions on Visualization and Computer Graphics*, 15(5):789–801. Available from: http://dx.doi.org/10.1109/TVCG.2009.28.

Raghuvanshi, N., Snyder, J., Mehra, R., Lin, M. C., and Govindaraju, N. K. (2010). Precomputed Wave Simulation for Real-Time Sound Propagation of Dynamic Sources in Complex Scenes. *SIGGRAPH 2010*, 29(3).

Rickard, Y. S., Georgieva, N. K., and Huang, W.-P. (2003). Application and optimization of PML ABC for the 3-D wave equation in the time domain. *Antennas and Propagation, IEEE Transactions on*, 51(2):286–295. Available from: http://dx.doi.org/10.1109/TAP.2003.809093.

Rindel, J. and Otondo, F. (2005). *The interaction between room and musical instruments studied by multi-channel auralization*.

Rizzo, A., Graap, K., Mclay, R. N., Perlman, K., Rothbaum, B. O., Reger, G., Parsons, T., Difede, J., and Pair, J. (2007). Virtual iraq: Initial case reports from a vr exposure therapy application for combat-related post traumatic stress disorder. In *Virtual Rehabilitation, 2007*, pages 124–130.

Sakamoto, S., Nagatomo, H., Ushiyama, A., and Tachibana, H. (2008). Calculation of impulse responses and acoustic parameters in a hall by the finite-difference time-domain method. *Acoustical Science and Technology*, 29(4).

Sakamoto, S., Ushiyama, A., and Nagatomo, H. (2006). Numerical analysis of sound propagation in rooms using finite difference time domain method. *Journal of the Acoustical Society of America*, 120(5):3008. Available from: http://scitation.aip.org/getabs/servlet/GetabsServlet?prog=normal&id=JASMAN000120000005003008000003&idtype=cvips&gifs=yes.

Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(1):43–49.

Savioja, L. (2010). Real-Time 3D Finite-Difference Time-Domain Simulation of Mid-Frequency Room Acoustics. In *13th International Conference on Digital Audio Effects (DAFx-10)*.

Savioja, L., Manocha, D., and Lin, M. (2010). Use of GPUs in room acoustic modeling and auralization. In *Proc. Int. Symposium on Room Acoustics*, Melbourne, Australia.

Schissler, C., Mehra, R., and Manocha, D. (2014). High-order diffraction and diffuse reflections for interactive sound propagation in large environments. *ACM Trans. Graph.*, 33(4):39:1–39:12. Available from: http://doi.acm.org/10.1145/2601097.2601216.

Sloan, P.-P. (2008). Stupid spherical harmonics tricks. In *Game Developers Conference*.

Sloan, P.-P., Kautz, J., and Snyder, J. (2002). Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph.*, 21(3):527–536. Available from: http://doi.acm.org/10.1145/566654.566612.

Southern, A. and Murphy, D. (2009). Low complexity directional sound sources for finite difference time domain room acoustic models. In *Audio Engineering Society Convention 126*. Available from: http://www.aes.org/e-lib/browse.cfm?elib=14960.

Southern, A., Siltanen, S., and Savioja, L. (2011). Spatial room impulse responses with a hybrid modeling method. In *Audio Engineering Society Convention 130*. Available from: http://www.aes.org/e-lib/browse.cfm?elib=15852.

Stanton, T. K., Chu, D., and Norton, G. V. (2007). Acoustic diffraction by deformed edges of finite length: Theory and experiment. *The Journal of the Acoustical Society of America*, 122(6):3167–3176. Available from: http://link.aip.org/link/?JAS/122/3167/1.

Svensson, U. P., Fred, R. I., and Vanderkooy, J. (1999). An analytic secondary source model of edge diffraction impulse responses. *The Journal of the Acoustical Society of America*, 106(5):2331–2344.

Sypek, P., Dziekonski, A., and Mrozowski, M. (2009). How to render fdtd computations more effective using a graphics accelerator. *Magnetics, IEEE Transactions on*, 45(3):1324–1327.

Taflove, A. and Hagness, S. C. (2005). *Computational Electrodynamics: The Finite-Difference Time-Domain Method, Third Edition*. Artech House Publishers, 3 edition. Available from: http://www.worldcat.org/isbn/1580538320.

Taraldsen, G. and Jonasson, H. (2011). Aspects of ground effect modeling. *The Journal of the Acoustical Society of America*, 129(1):47–53. Available from: http://link.aip.org/link/?JAS/129/47/1.

Taylor, M. T., Chandak, A., Antani, L., and Manocha, D. (2009). Resound: Interactive sound rendering for dynamic virtual environments. In *Proceedings of the 17th ACM International Conference on Multimedia*, MM '09, pages 271–280, New York, NY, USA. ACM. Available from: http://doi.acm.org/10.1145/1631272.1631311.

Thompson, L. L. (2006). A review of finite-element methods for time-harmonic acoustics. *The Journal of the Acoustical Society of America*, 119(3):1315–1330. Available from: http://dx.doi.org/10.1121/1.2164987.

Thompson, L. L. and Pinsky, P. M. (2004). *Acoustics*. John Wiley & Sons, Ltd. Available from: http://dx.doi.org/10.1002/0470091355.ecm046.

Torres, R. R., Svensson, U. P., and Kleiner, M. (2001). Computation of edge diffraction for more accurate room acoustics auralization. *The Journal of the Acoustical Society of America*, 109(2):600–610. Available from: http://scitation.aip.org/content/asa/journal/jasa/109/2/10.1121/1.1340647.

Tsingos, N., Dachsbacher, C., Lefebvre, S., and Dellepiane, M. (2007). Instant Sound Scattering. In *Rendering Techniques (Proceedings of the Eurographics Symposium on Rendering)*. Available from: http://www-sop.inria.fr/reves/Basilic/2007/TDLD07.

Tsingos, N., Funkhouser, T., Ngan, A., , and Carlbom, I. (2001). Modeling acoustics in virtual environments using the uniform theory of diffraction. In *Computer Graphics (SIGGRAPH 2001)*.

Vigeant, M. C. (2008). Investigations of incorporating source directivity into room acoustics computer models to improve auralizations. *The Journal of the Acoustical Society of America*, 124(5):2664–2664. Available from: `http://link.aip.org/link/?JAS/124/2664/2`.

Ward, D. B. and Abhayapala, T. (2001). Reproduction of a plane-wave sound field using an array of loudspeakers. *IEEE Transactions on Speech and Audio Processing*, 9(6):697–707.

Warusfel, O. and Misdariis, N. (2004). Sound source radiation syntheses: From performance to domestic rendering. In *Audio Engineering Society Convention 116*. Available from: `http://www.aes.org/e-lib/browse.cfm?elib=12793`.

Whitted, T. (1980). An improved illumination model for shaded display. *Commun. ACM*, 23(6):343–349. Available from: `http://doi.acm.org/10.1145/358876.358882`.

Yee, K. (1966). Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions on Antennas and Propagation*, 14(3):302–307. Available from: `http://dx.doi.org/10.1109/TAP.1966.1138693`.

Yu, W., Liu, Y., Su, Z., Hunag, N.-T., and Mittra, R. (2005). A robust parallel conformal finite-difference time-domain processing package using the mpi library. *Antennas and Propagation Magazine, IEEE*, 47(3):39–59.

Zheng, C. and James, D. L. (2010). Rigid-body fracture sound with precomputed soundbanks. In *SIGGRAPH '10: ACM SIGGRAPH 2010 papers*, pages 1–13, New York, NY, USA. ACM.

Zienkiewicz, O. C., Taylor, R. L., and Nithiarasu, P. (2006). *The finite element method for fluid dynamics*. Butterworth-Heinemann, 6 edition. Available from: `http://www.worldcat.org/isbn/0750664312`.

Zotkin, D. N., Duraiswami, R., and Gumerov, N. A. (2010). Plane-wave decomposition of acoustical scenes via spherical and cylindrical microphone arrays. *IEEE Trans. Audio, Speech and Language Processing*, 18.