Performance Tradeoffs in HRTF Interpolation Algorithms for Object-Based Binaural Audio

Matthew Skarha

Music Technology Area, Department of Music Research
Schulich School of Music
McGill University
Montréal, Québec, Canada

September 2021

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of Arts

Abstract

A popular method for rendering the direct acoustic path in object-based binaural audio is to convolve the source signal with a Head-Related Transfer Function (HRTF) that corresponds to the source angle. HRTFs, however, are only measured for a discrete set of angles. In order to simulate arbitrary source angles, then, HRTF interpolation must be performed. In this thesis, we present a comparison of two HRTF interpolation techniques: bilinear interpolation of the four closest (BI4C) and N-th order spherical harmonic decomposition (SHD-N). The two metrics used to perform this comparison are reconstruction error and computational cost in a real-time auralization engine. Reconstruction error is analyzed with an error function computed on a perceptually-informed frequency axis. Computational cost is measured via benchmarking efficient C++ implementations of each algorithm. A higher truncation order N in SHD-N will result in lower reconstruction error at the cost of being more expensive to compute. On the other hand, the reconstruction error and computational cost of BI4C is only a function of the measurement lattice chosen and can be regarded as having fixed cost.

Results show BI4C generally outperforms SHD-N in terms of reconstruction error if only sparse measurement grids are available. If dense grids are available, the superiority of either algorithm is a function of the grid and truncation order used. BI4C computational cost benchmarks outperform SHD-1 for a low number of sources (≤ 5) but higher order SHD-N have a lower marginal source cost than BI4C as the number of sources approaches 500.

Abrégé

Une méthode populaire de rendu du chemin acoustique direct pour l'audio binaural à approche objet consiste à convoluer le signal source avec une fonction de transfert de tête (HRTF) liée à l'angle de la source. Cependant, les HRTFs ne sont mesurées que pour un ensemble d'angles fini. Une interpolation des HRTFs doit alors être effectuée afin de considérer des angles de source arbitraires. Dans cette thèse, nous présentons une comparaison de deux techniques d'interpolation des HRTFs: l'interpolation bilinéaire selon les quatre plus proches voisins (BI4C) et la décomposition en harmoniques sphériques d'ordre N (SHD-N). Les deux métriques considérées pour effectuer cette comparaison mesurent l'erreur de reconstruction et le coût de calcul dans un dispositif temps-réel d'auralisation. L'erreur de reconstruction est analysée avec une fonction d'erreur calculée sur un axe fréquentiel de nature perceptive tandis que le coût de calcul est mesuré en comparant des implémentations en C++ efficaces de chaque algorithme. Un ordre de troncature N plus élevé pour l'interpolation SHD-N entraînera une erreur de reconstruction plus faible au prix d'un calcul plus coûteux. À l'inverse, l'erreur de reconstruction et le coût de calcul de l'interpolation BI4C ne dépendent que du maillage de mesure choisi et peuvent être considérés comme ayant un coût fixe.

Les résultats montrent que BI4C surpasse généralement SHD-N en termes d'erreur de reconstruction lorsque seules des grilles de mesure parcimonieuses sont disponibles. Cependant, si des grilles de mesure denses sont disponibles, la supériorité de l'un ou l'autre des algorithmes d'interpolation dépend de la grille et de l'ordre de troncature utilisés. Les coûts de calcul de BI4C surpassent ceux de SHD-1 pour un faible nombre de sources (≤ 5), tandis qu'une interpolation SHD-N d'ordre supérieur a un coût marginal de source inférieur à celui de l'interpolation BI4C lorsque le nombre de sources approche les cinq cents.

Acknowledgements

I would like to extend many thanks to my supervisor, Prof. Gary Scavone, for his continued guidance and patience during this process, which was significantly complicated by the pandemic. I am deeply grateful for him being so accommodating to my research interests when I was choosing my topic and for introducing me to my co-supervisor Dr. Esteban Maestre. I would like to also recognize and thank Dr. Maestre for his superior commitment to putting and keeping me on the right track through numerous emails and Zoom discussions. There is no question that I could not have done this without his continued support.

Thank you very much to others who have made my time at McGill very special: Prof. Marcelo Wanderley, Prof. Philippe Depalle, Darryl Cameron, Yves Méthot, and the folks at the Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT). The unique combination of resources, funding opportunities, coursework, technical support, and collaborators who have backgrounds in a variety of disciplines makes Montréal a very special place to study music technology.

Lastly, I extend my heartfelt gratitude to a number of others who believed in me and have provided support in one way or another: my parents, my sister, Vincent Cusson, Christian Frisson, John Sullivan, Samuel Waranch, Julian Vanasse, Emily Highkin, and Alex Gallo.

Contents

List of Figures

List of Acronyms

ACN	Ambisonic Channel Order.
AVR	Acoustic Virtual Reality.
BI4C	Bilinear Interpolation of the Four Closest.
C2R	Complex-to-Real.
CBA	Channel-Based Audio.
CPU	Central Processing Unit.
DFT	Discrete Fourier Transform.
DSP	Digital Signal Processing.
FDN	Feedback Delay Network.
FFT	Fast Fourier Transform.
FIR	Finite Impulse Response.
HOA	Higher-Order Ambisonics.
HRIR	Head-Related Impulse Response.
HRTF	Head-Related Transfer Function.
IFFT	Inverse Fast Fourier Transform.
IIR	Infinite Impulse Response.
ILD	Interaural Level Difference.
ISFT	Inverse Spherical Fourier Transform.
ITA	Institute of Technical Acoustics.
ITD	Interaural Time Difference.
KEMAR	Knowles Electronics Manikin for Acoustic Research.
LTI	Linear, Time-Invariant.
MAA	Minimum Audible Angle.
OBA	Object-Based Audio.
OLA	Overlap-Add.
OLS	Overlap-Save.
PCA	Principle Component Analysis.

R2C	Real-to-Complex.
SADIE	Spatial Audio for Domestic Interactive Environment.
SBA	Scene-Based Audio.
SDK	Software Development Kit.
SFT	Spherical Fourier Transform.
SH	Spherical Harmonic(s).
SHD	Spherical Harmonic Decomposition.
SHD-N	N-th Order Spherical Harmonic Decomposition.
SMD	Spectral Magnitude Decay.
SVD	Singular Value Decomposition.

Chapter 1

Introduction

Sound is inherently spatial. As humans, we are constantly experiencing the relationship between sound and space. This can exist as one or many sound sources interacting with the acoustics of a given space before arriving at our ears. It is also reflected in the evolutionary origin of the ear and is evident at several stages of the auditory system [1]. The importance of space in regards to music has been understood for ages, especially with respect to composition, instrument building, performance practice, concerts, and music playback. The complicated radiation characteristics of a musical instrument might be exploited by a composer to create a particular sense of space in a composition, for example.

Acoustic Virtual Reality (AVR) attempts to use computers to simulate the acoustics of a non-existent world [1]. A listener is presented with digitally processed auditory stimuli with the intent of immersing them in a virtual environment. In order to be successful in this immersion, the stimuli must confirm our everyday understanding of the laws of physics such as our perception of the volume, timbre, distance, and location in 3D space. The content of these virtual environments can be drawn from reality (e.g. recorded acoustic musical performance), digitally synthesized, or a combination of the two.

The practice of creating audible sounds based on a description of an acoustic scene is known as *auralization* or acoustic rendering. At the core of this process is audio signal processing. To create an interactive virtual acoustic space, auralization must be performed in real-time since it must react to various inputs (e.g. head-tracking in a virtual reality headset). This requirement often creates a trade-off between computational cost and sound field accuracy. Therefore, it is important to pay special attention to the algorithms used during auralization and optimize them to match the needs or limits of the platform being created.

In the free field, a sound radiated from a point in space will reach the two ears after interacting with the head, torso, and pinnae. If we recorded the air pressure levels just before they entered each ear canal, they would contain various cues that tell our brain where to localize this sound in 3D space. Mathematically, the incorporation of these localization cues can be expressed as an overall filtering operation on the original sound. These filters are known as Head-Related Transfer Functions (HRTFs).

The HRTF is a key component in virtual acoustic rendering because it describes how a listener's head, ears, and torso affect the acoustic propagation of sound sources arriving from various directions [1]. Typically, HRTFs are obtained for an individual by taking a series of discrete Head-Related Impulse Response (HRIR) measurements in an anechoic chamber. HRTFs are then computed as the Fourier transform of these HRIRs. Ideally, we would be able to create the illusion of sound emanating from an arbitrary direction and include resiliency to head movements. However, due to the discrete measurement requirement of HRTFs, we are required to choose the HRIR closest to the source angle we wish to simulate, unless we interpolate between known HRIRs. The task, then, is to form a continuous functional representation of an HRTF by using various mathematical techniques.

This thesis will explore two popular methods for building continuous HRTF models in order to spatially interpolate an HRTF from its individual measurements: interpolation via weighted averaging and spherical harmonic decomposition. In particular, we are interested in comparing their interpolation quality and computational cost in a real-time auralization engine. Because interpolation quality is directly related to the accuracy of the resulting sound field, it is useful to know how well these techniques reconstruct new HRTFs. Moreover, we must consider the computational complexity of these algorithms in order to validate their use for a platform with limited computational resources.

1.1 Motivation

The democratization of development tools for gaming, audio, film, and smartphone apps has created a need for efficient, high-quality spatial audio to render immersive experiences over headphones. Accordingly, a number of open- and closed-source software development kits (SDKs) for audio spatialization have been released over the years to fulfill this need

[2] [3] [4] [5]. Each of these tools have chosen schemes for rendering the direct path (sound that travels directly from source to receiver), the near-field (complex interference patterns at short distances), and the far-field (reverberation). It is generally agreed upon that using HRTFs to simulate the direct path is a good choice as it is an efficient yet accurate approach. However, there is still research to be done to determine how to best compute or retrieve these HRTFs to make the experience as realistic as possible while maintaining the real-time requirement.

Since the localization cues embedded in HRTFs are individual dependent, HRTFs would ideally be personalized for everyone interested in consuming spatial audio over headphones. Of course this is an extremely expensive and time-consuming process, especially to get the number of measurements required to at least match human localization accuracy. Accordingly, lots of research focuses around adapting generic HRTFs to individuals based on anatomic data, such as photos of a listener's head and ears. Still, the question of the minimum number of HRTF measurements required to build an accurate, continuous HRTF model remains to be seen. This question is of interest in this thesis and will be explored more in-depth in Chapter 2.

There has been a renewed interest in recent years around Ambisonics with the increased accessibility of B-format microphones [6]. These microphones use multiple capsules to record 3D sound field information and encode the signals into the Ambisonics domain, where it can be then decoded onto an arbitrary configuration of microphones or loudspeakers. Ambisonic technologies are desirable for their high level of adaptability to different environments including binaural and surround sound setups. The downside, however, is the requirement for the choice of an Ambisonic order, where higher orders provide more accurate sound fields but are expensive to compute.

The mathematical framework underlying Ambisonic technologies involves projecting a sound field (sampled at many points on a sphere) onto a set of special functions called the *spherical harmonics*. This process is referred to as the spherical Fourier transform (SFT). Because HRTFs can be considered as a sound field sampled at many points on a sphere, it is possible to encode HRTFs into spherical harmonics (i.e. Ambisonics) via the SFT and spatially interpolate this representation to compute HRIRs at arbitrary angles. This, then, is the motivation for spherical harmonic decomposition of HRTFs, which will be studied in-depth in this thesis.

A more traditional method for performing interpolation involves taking weighted averages of known points of a function to approximate some intermediate point. This can be done by fitting a line, polynomial, or spline function between the two known points to estimate a third. Here, we are interpolating the function in question directly rather than interpolating a set of physically-informed functions that approximate the function in question, as is the case in spherical harmonic decomposition. Direct interpolation of HRTF coefficients can be accomplished by using, for example, the bilinear method. Bilinear interpolation is linear interpolation (fitting a line) along two orthogonal axes. For example, if we want to compute an HRIR at some query angle, we can take a linear combination of the four HRIRs closest to that angle, where each HRIR is weighted by a factor inversely proportional to its distance to the query angle. In this way, we can estimate an HRIR anywhere on the sphere.

The incorporation of both of these approaches into an auralization engine involves convolution since the source signals must be filtered by the interpolated HRTFs. Time-domain convolution, however, is a computationally expensive process [7]. Therefore, for real-time filtering, convolution is accomplished via element-wise multiplication of frequency domain coefficients, where the discrete Fourier transform of each signal is computed via the Fast Fourier Transform (FFT). In this thesis, we will explore these FFT-based fast convolution techniques in the context of the two HRTF interpolation algorithms described above.

As of yet, while they are two of the most popular HRTF interpolation algorithms, there has been no in-depth comparison of spherical harmonic decomposition and interpolation via weighted averaging. This thesis will attempt to provide this comparison. Two metrics will be used in this analysis: reconstruction error and computational cost. The goal of this research is to provide a developer who is building an auralization engine with the data necessary to better inform their choice of interpolation technique. The findings of this research will also prove to be beneficial to those interested in the question of minimum HRTF measurement density.

1.2 Thesis Overview

In Chapter 2, an overview of virtual acoustic rendering is presented, followed by explorations of the theory behind spherical harmonic decomposition and interpolation via weighted averaging. These will also include comments on the advantages and shortcomings of each algorithm. Additionally, a discussion of fast convolution algorithms for real-time auralization is presented.

In Chapter 3, we motivate and present the results of the reconstruction error analysis, showing how well each algorithm is able to interpolate according to our error function.

In Chapter 4, we describe our computational cost benchmark procedure, test system, and its performance data.

Finally, in Chapter 5, we conclude by summarizing the observations drawn from the performance of each algorithm and their suitability for real-time applications. Future directions of this research are also discussed.

Chapter 2

Background

In this chapter, we give an overview of how humans localize sound in order to motivate virtual acoustic rendering and interpolation of Head-Related Transfer Functions. We then explain considerations relevant to auralization and binaural technology. Next, we describe in detail each of the interpolation algorithms considered in this thesis. We conclude by giving an overview of FFT-based convolution algorithms for real-time auralization.

2.1 Spatial Listening

Human hearing includes the ability to perceive loudness, pitch, and timbre as well as evaluate the position of the sound source in space. From an evolutionary perspective, the ability to localize sounds has helped humans identify and evade danger for thousands of years. Many studies have measured our ability to locate a sound source and have found it depends on many factors. These vary from person to person and include primarily the source's direction and acoustical properties. On average, humans can localize sounds to a precision $\Delta\theta$ of about 1° to 3° in front of us. $\Delta\theta$ is known as the *minimum audible angle* (*MAA*) or *localization blur* and refers to the smallest change in source position that we can perceive. $\Delta\theta$ is about three times larger in lateral directions and twice as large in rear directions. The MAA is also frequency-dependent; we can best localize frequencies below 1.1 kHz [1].

When in the presence of two or more sound sources, our brain can often locate all the sources based on the superposition of each source's pressure in our ears. If many sources have the same onset in time, we will sometimes perceive the sounds as a fused auditory event coming from a single virtual sound source. In fact, this is the basis of stereophonic and multi-channel surround sound reproduction.

Audio engineers can alter the amplitudes of loudspeaker channels that are playing the same source through panning in order to create a sense of space or movement. However, in traditional stereophonic headphones for example, this sense of space is limited to a straight line connecting the entrances of the two ear canals. Our ability to determine the location of an auditory event along this line is referred to as *lateralization*, whereas *localization* refers to locating an event in three dimensions. For an enclosed space with reflective surfaces, sound will reflect of those surfaces and arrive at the ears in addition to the direct path. It is through these reflections that the listener can form a spatial impression of both the environment and sound source. Often, audio engineers will imitate these reflections by applying reverberation to sounds in order to enhance the sense of space created with amplitude panning.

2.1.1 Sound Localization Cues

Sound localization involves determining the apparent or perceived position of a sound source in space in terms of the direction and distance relative to the listener. Our brain localizes sounds by aggregating various localization cues. Psychoacoustic studies have identified these as *interaural time difference* (*ITD*), *interaural level difference* (*ILD*), *spectral cues*, and *dynamic cues*. Here, we will give a brief overview of each of these.

Interaural time differences describe the time difference of arrival of sound waves between the left and right ears:

$$ITD(s) = \tau_L(s) - \tau_R(s),$$

where $\tau_L(s)$ and $\tau_R(s)$ are the source-dependent onsets in time of sound waves at the left and right ears, respectively. When a source is directly in front or behind you, the ITD is theoretically zero since the sound reaches both ears at the same time. As sources deviate from this plane, the acoustic path length will be shorter for the ear closer to the source and longer for the ear farthest from the source. Thus, finding the difference between these two path lengths gives a non-zero cue that helps the brain localize the source. Additional psychoacoustic experiments have shown that *interaural phase delay difference*, the difference in phase delays between the two ears, is important for localizing sounds below 1.5 kHz [1]. Various methods exist for calculating these values based on different approximations of the head.

Interaural level differences denote the difference in sound pressure amplitude at the two ears. For a source off to one side, the sound pressure at the farther ear will be attenuated due to the shadowing effect of the head, especially at the high frequencies. This difference in amplitudes is frequency-dependent and can be written as

$$ILD(s, f) = 20 \log_{10} \left| \frac{P_r(s, f)}{P_l(s, f)} \right| \tag{2.1}$$

where $P_l(s, f)$ and $P_r(s, f)$ are the frequency-domain sound pressures at the left and right ears, respectively, generated as the result of some sound source s. At low frequencies, ILDs are small regardless of the source direction because the head-shadowing effect is negligible for these frequencies in the far field. Above 1.5 kHz, both ILDs and ITDs contribute to localization, with ILDs becoming gradually more dominant above 4 kHz.

Spectral cues refer to the direction-dependent spectral filtering performed by the head and pinnae. When direct and reflected acoustic paths enter the pinnae, they interfere with waves reflected within the pinnae in complex patterns, leading to resonances at certain frequencies, especially above 5-6 kHz. Many reseachers have tried to explain the relationship between localization and the spectral peaks and notches caused by the pinnae. Although we know spectral cues are important for vertical localization and clarifying front-back confusion, the quantitative reasoning for this is still incomplete [1].

Dynamic cues simply describe how humans improve localization ability (especially on the vertical plane) when the head is not stationary. By aggregating many of these cues over time for even tiny head movements, the brain can build a much more complete picture of the acoustic environment.

2.1.2 Head-Related Transfer Functions

As we have mentioned, sound enters the auditory system after interacting with various anatomical structures like the head, torso, and pinnae. Therefore, the sound pressures at the two ears (binaural pressures) have these localization cues embedded within them. If we wish to model the process of localizing sound sources for virtual acoustic rendering, we can simply focus on these signals, as opposed to physically modeling and aggregating each localization cue. In signal processing terms, the filtering effect resulting from the propagation of sound from a source to the two ears can be regarded as a linear, time-invariant (LTI) process. These filters are referred to as *Head-related transfer functions* (*HRTFs*). They can be measured by comparing the binaural sound pressures resulting from some source with the corresponding sound pressure in the free-field (i.e. if no head, body were there). For some arbitrary source position located in spherical coordinates at (ϕ, θ, r), the HRTFs for the left and right ears can be defined as

$$H_L = H_L(\phi, \theta, r, f) = \frac{P_L(\phi, \theta, r, f)}{P_0(r, f)} \tag{2.2}$$

$$H_R = H_R(\phi, \theta, r, f) = \frac{P_R(\phi, \theta, r, f)}{P_0(r, f)} \tag{2.3}$$

where P_L and P_r are the complex-valued frequency-domain sound pressures at the left and right ears, respectively. P_0 is the complex-valued frequency-domain sound pressure in the free field at the center of the head with the head absent. These functions are also dependent on the anatomical features of the individual measured.

Because P_0 is the result of a simple free-field acoustic wave propagation, it can be written as

$$P_0(r, f) = j\frac{k\rho_0 c Q_0}{4\pi r} e^{j(\omega t - kr)} \tag{2.4}$$

where ρ_0 is the density of air, $c = 343$ m/s is the speed of sound, Q_0 is the intensity of the point sound source, r is the source distance, t is time, and $k = 2\pi f/c = \omega/c$ is the wave number.

2.2 Auralization and Binaural Technology

As we have discussed, binaural signals (sound pressure signals recorded at the entrances of the human ear canal) contain most of the information we need to localize sounds in 3D space. The most straightforward way to obtain a binaural signals is to place small microphones at each ear and record the acoustic pressure signals. Therefore, the localization cues introduced by the head, torso, and pinnae are preserved and can be reproduced over headphones, for example. Traditionally, binaural recordings are made with a manikin. These manikins are typically designed based on the average dimensions of a certain population and have been used to measure HRTFs, as well.

One early method for auralizing spatial audio is the *binaural recording and playback* *system*. This simply involves recording binaural signals (often with a manikin), applying some equalization and amplification, and playing them back over headphones to a listener. In this way, one can experience the sound field surrounding the manikin during the recording. Because this method requires the reproduced sound field to exist during the recording, it is limited in the type of acoustic scenes that can be rendered. In the next section, we will see a technique for rendering arbitrary acoustic environment through the use of signal processing.

2.2.1 Coordinate Systems

Before discussing auralization via synthesis of binaural signals with computers, we must review spatial coordinate systems. In most spatial listening research, a sound source is located in terms of its direction and distance in relation to the listener's head. Typically, the origin of these coordinate systems is chosen as the midpoint of the line segment connecting the listener's ear canals.

In a three-dimensional Euclidean space (i.e. $\mathbb{R}^3$), pairing the orthogonal basis vectors in the x, y, and z dimensions specify three perpendicular planes. They are defined relative to the direction the listener is facing and are referred to as the *horizontal plane* (front/back and left/right), *median plane* (front/back and up/down), and *lateral plane* (up/down and left/right). In the literature, different coordinate systems for sound localization have arisen based on various conventions. For this thesis, we will exclusively use the anti-clockwise spherical coordinate system.

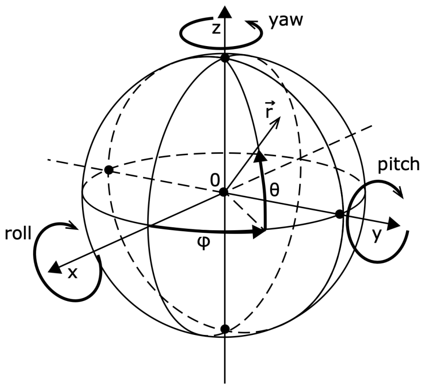

Figure 2.1: Anti-clockwise spherical coordinate system [8].

In the anti-clockwise spherical coordinate system, all points in 3D space are described by the ordered triple (ϕ, θ, r). ϕ is called the azimuth angle and describes the angle between the front direction ($\phi = 0°$) and the projection of a vector onto the horizontal plane. It can take on values $0° \leq \phi < 360°$ which increase as they move anti-clockwise (left) about the up-down axis. The elevation angle θ is the angle between a vector and the horizontal plane, relative to the horizontal plane ($\theta = 0°$). It can take on values $-90° \leq \theta \leq 90°$, where $-90°$ is down and $90°$ is up. The source distance with respect to the origin is denoted by r and can range from $0 \leq r < \infty$.

This coordinate system is preferable compared to (x, y, z) Euclidean systems due to the spherical symmetry of acoustic wave propagation in an isotropic medium. In the next section, we will see why this is important in auralizing virtual acoustics.

2.2.2 Virtual Acoustic Rendering

In order to construct virtual acoustic scenes that might not exist in the real world, we must synthesize binaural signals using audio signal processing. This idea was pioneered in the early 1980s by Morimoto and Ando [9] and later applied to headphones by Wightman and Kistler [10]. Binaural synthesis for virtual acoustic rendering works by filtering monophonic source signals by a pair of HRTFs, where the source location is determined by the location of the measured HRTF. Given left and right ear HRTFs $H_L(\phi, \theta, r, f)$ and $H_R(\phi, \theta, r, f)$, the free-field binaural signals resulting from a mono discrete-time source signal $e_o[n]$ with frequency-domain representation $E_0(f)$ can be expressed in the frequency domain as:

$$E_L(\phi, \theta, r, f) = H_L(\phi, \theta, r, f)E_0(f) \quad \text{and} \quad E_R(\phi, \theta, r, f) = H_R(\phi, \theta, r, f)E_0(f) \tag{2.5}$$

Equivalently, in the discrete time domain, they can be expressed as

$$e_L[\phi, \theta, r, n] = h_L[\phi, \theta, r, n] * e_0[n] \quad \text{and} \quad e_R[\phi, \theta, r, n] = h_R[\phi, \theta, r, n] * e_0[n] \tag{2.6}$$

where the head-related impulse responses (HRIRs) h_L and h_R are the inverse Fourier transforms of H_L and H_R, respectively. Here, $a * b$ denotes the circular convolution of two N-point periodic signals a and b and is defined by

$$(a * b)_N \triangleq \sum_{m=0}^{N-1} a[m]b[n-m]. \tag{2.7}$$

where n is the time sample index and $n = 0, 1, 2, \ldots, N-1$.

When the binaural signals e_L and e_R are displayed over a pair of headphones, the listener will hear sound pressures comparable to those generated by the ideal point source localized at (ϕ, θ, r). This, then, is the main idea behind virtual acoustic rendering (virtual auditory display).

Robust binaural synthesis engines will go beyond the simple convolution of sources with HRIRs to account for other aspects of the environment. This might involve simulating reverberation, source directivity, occlusion, reflections off obstacles, or frequency-dependent distance attenuation. Reverberation simulation can be divided into two main approaches: convolution-based or synthetic [11]. In convolution-based reverberation, the binaural signals are convolved with impulse responses of the environment to be simulated (called Binaural Room Impulse Responses). In synthetic reverberation, many approaches exist including ray tracing, Feedback Delay Network (FDN), or Spectral Magnitude Decay (SMD) techniques, each with their own advantages and disadvantages [12]. Simulation of source directivity refers to the way many sound sources, especially musical instruments, sound different based on our location relative to it. Occlusion happens when sound waves emanating from source to listener are blocked by objects between them [13]. Techniques also exist for simulating reflections off obstacles as well as modeling distance attenuation beyond the traditional -6 dB with every doubling of distance. For example, some spatializers address the effect of air absorption at high frequencies for large distances.

The techniques described here that improve upon basic anechoic path simulation to create a more realistic virtual acoustic environment are out of the scope of this thesis. Rather, we are solely focused on how to best render the anechoic (direct) or early reflection wavefronts for arbitrary source angles when only sparse HRTFs are available.

2.2.3 Channel-, Object-, and Scene-Based Audio

In general, audio spatialization techniques fall into one of three categories: channel-based, scene-based, or object-based. These distinctions refer to how individual sound sources are aggregated and mapped onto loudspeakers for presentation. Channel-based audio currently accounts for the large majority of audio rendering whereas scene- and object-based audio are often referred to as the “next generation” of audio rendering [14].

In channel-based audio (CBA), the audio material (i.e. individual sources) is mixed for a specific loudspeaker layout (e.g. stereo or 5.1) by a mixing engineer. Listeners would ideally have their loudspeakers placed in the same location as the mixing engineer's loudspeakers so the spatialization is heard as the mixing engineer intended.

In object-based audio (OBA), the material is stored as unmixed audio components (i.e. objects) that each have associated metadata. The metadata contains information that characterizes each object such as its spatial location in the scene and gain level. It is intended to be adaptable to the specific loudspeaker configuration available and can allow for real-time adjustment of audio object characteristics. In binaural OBA, retrieving the HRIR that will spatialize each object is a nontrivial problem when we only have sparsely measured HRTFs. In this thesis, two solutions for this problem will be developed and compared with each other.

Scene-based audio (SBA) describes the use of Higher-Order Ambisonics (HOA) to render sound scenes. HOA involves constructing a 3D sound field in the Ambisonics domain by mixing sources that have been projected onto a set of special functions called spherical harmonics. The audio material is easily transported because an arbitrary number of sources are encoded onto a fixed number of Ambisonic channels, which can then be decoded onto an arbitrary configuration of loudspeakers. The quantity of channels depends on the Ambisonic *order*, which is a quantity that specifies how many spherical harmonic functions we are to include in our decomposition. For an Ambisonic order N, $(N + 1)^2$ coefficients are required per sample, each representing the contribution of a given spherical harmonic (SH) function to the overall sound field. The sound field is then defined as a linear combination of these SH functions weighted by their corresponding coefficient. SH functions are structured hierarchically, meaning we can always choose to include more functions to create a more precise sound field (i.e. increase order) at the cost of computing more coefficients. Typically, when rendering SBA for headphones (binaural SBA), the Ambisonic-encoded sound field is decoded onto a set of virtual loudspeakers, which are spatialized as static virtual sources via convolution with the corresponding HRIR. This convolution can be performed per Ambisonic channel (per SH function) or per virtual loudspeaker in the time domain. For convolution per Ambisonic channel, the HRTFs must be also projected onto spherical harmonics (i.e. encoded into Ambisonics). This projection is identical to the SH decomposition of HRTFs for OBA that is central to this thesis but is executed with a different purpose. In SBA, the purpose is simply to decrease overall computation by only convolving once per channel whereas in OBA, the purpose is to also retrieve arbitrary source angles (i.e. interpolate).

2.2.4 Measurement of HRTFs

As we have seen, the process of sound radiating from a point source to a listener's left and right ears can be regarded as a linear, time-invariant (LTI) system. In signal processing, LTI systems are characterized entirely by a single function called the impulse response. In the frequency domain, this function is called the frequency response. Numerous methods exist for acoustically measuring impulse responses or frequency responses that typically involve exciting the system with a signal and recording its output.

HRTF measurement is no different. Most commonly, small measurement microphones are placed at the ear canal entrances of human or artificial subjects and excitation signals are presented from various directions. Ideally, the measurement takes place in an anechoic chamber to avoid any coloration from the environment. Three main techniques have arisen for HRTF measurement [1].

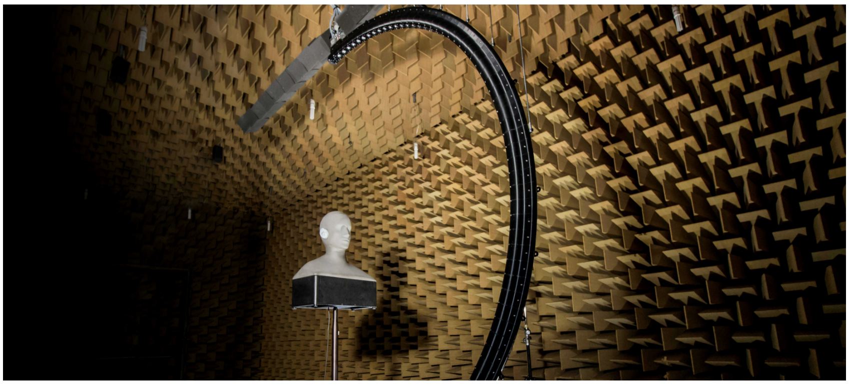

Figure 2.2: Measurement setup for the ITA-HRTF database. A dummy head is rotated on a turn table to achieve varying measurement angles relative to the fixed speaker arc. Taken from [15].

In the impulse method, the system is excited by an approximation of an ideal Dirac distribution $\delta(t)$, which is a deterministic signal with a flat magnitude spectrum and linear phase. Generating these impulse signals with computers, however, introduces a tradeoff between the length of the rectangular signal (necessary for good signal-to-noise ratio) and the bandwidth (necessary for understanding the response of the system to a broadband of frequencies). Additionally, the highly transient sound pressure can cause nonlinear effects in the air. Therefore, the impulse method is seldom used for acoustic measurement and is perhaps best understood as a theoretical basis for impulse response measurement. The Fourier analysis and correlation methods are then practical realizations of this theoretical basis.

In the Fourier analysis method, a discrete Fourier transform (DFT) is applied to the input signal $x[n]$ and output signal $y[n]$ and divided for each ear:

$$H(f) = \frac{Y(f)}{X(f)}. \tag{2.8}$$

To retrieve $h[n]$, an inverse DFT can be applied to $H(f)$. Excitation signals for this method should cover the largest bandwidth possible. Therefore, swept sinusoidal signals are commonly used.

In the correlation method, a random-phase white or pink spectrum signal (i.e. noise) is chosen as an input signal $x[n]$ because its N-point autocorrelation function approximates the unit sampling sequence $\delta[n]$. The autocorrelation function is used to describe how similar a function is with a delayed version of itself. Upon recording the output signal $y[n]$, the cross-correlation function of $x[n]$ and $y[n]$ is used to compute the impulse response $h[n]$. Cross-correlation describes how correlated two functions are with each other. Often, excitation signals such as Maximum Length Sequences or Golay Codes are substituted due to their ability to optimize signal features such as the crest factor (ratio of peak to RMS values of the waveform).

2.2.5 Properties of HRTFs

Observing time domain plots of HRIRs on the horizontal plane ($\theta = 0°$) with a radius of 1.2m as in Figure 2.3, we notice a few features. First, the samples are roughly zero for the beginning part of each IR. This section corresponds to the time it takes for the sound to propagate from the source to each ear. We also notice the non-zero amplitudes for the left ears begin before and have larger amplitudes than the corresponding right ears for azimuths $\phi = 45°, 90°, 135°$. These time differences of arrival between the left and right ears are the Interaural Time Differences (ITDs) while the differences in amplitudes are the Interaural Level Differences (ILDs). Since the azimuths are increasing in an anti-clockwise direction, the left ear is closer to the sound source, meaning the sound will reach it earlier than and be louder than the right ear.

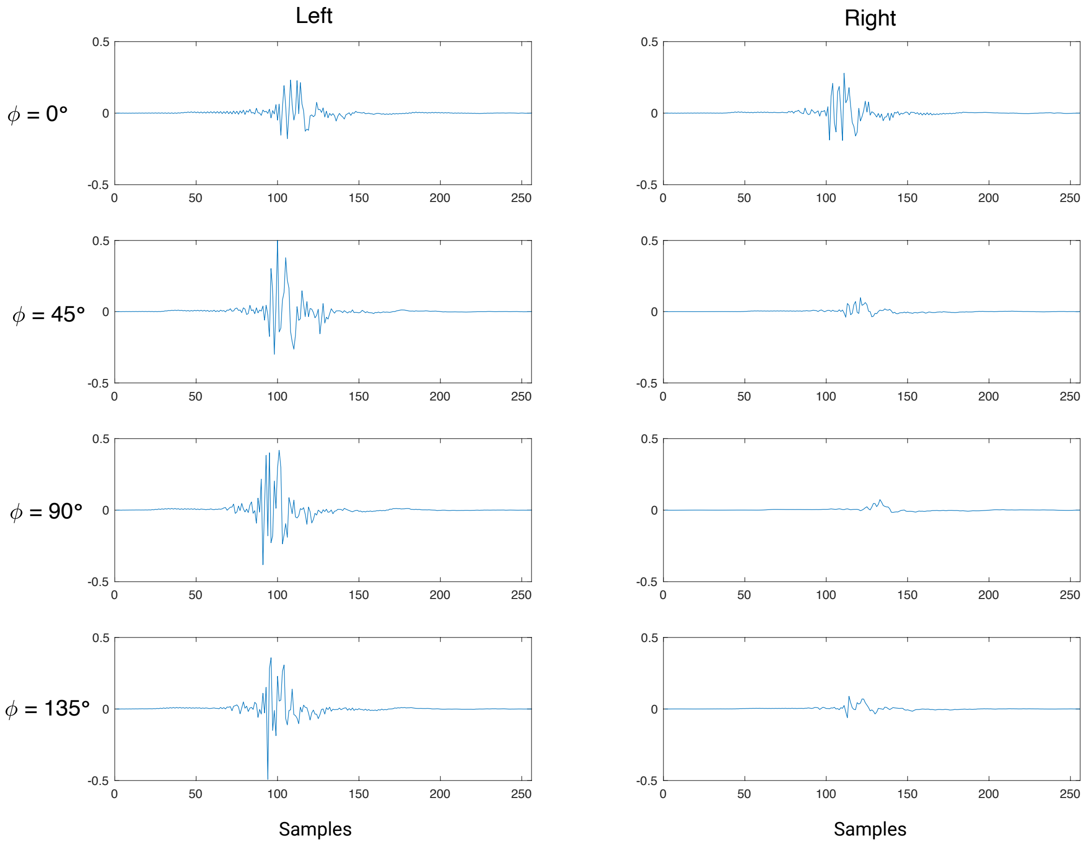

Figure 2.3: Left ear and right ear HRIRs of the KEMAR mannequin from SADIE II Database [16] for 4 azimuths on the horizontal plane from a radius of 1.2m.

Figure 2.4 shows plots of the HRTF magnitude responses for the same azimuths on the horizontal plane. Inspecting these graphs, we notice the responses are roughly flat near 0 dB for frequencies less than 800 Hz. This is because the scattering and shadowing effects of the head are negligible since the head is smaller than the half wavelength of these sound waves. However, as frequency increases, the interactions between the head, torso, and pinnae become much more complicated. Again, we notice the ILDs between the left and right ears as the left ear has larger magnitude for azimuths $\phi = 45°, 90°, 135°$. We mentioned in Section 2.1.1 how pinna anatomy creates spectral notches in HRTFs, which are crucial for localization above 5-6 kHz. While we can see in these plots that these notches exist, further signal processing techniques, such as those described in [17], would need to be performed to identify and map spectral notch frequencies.

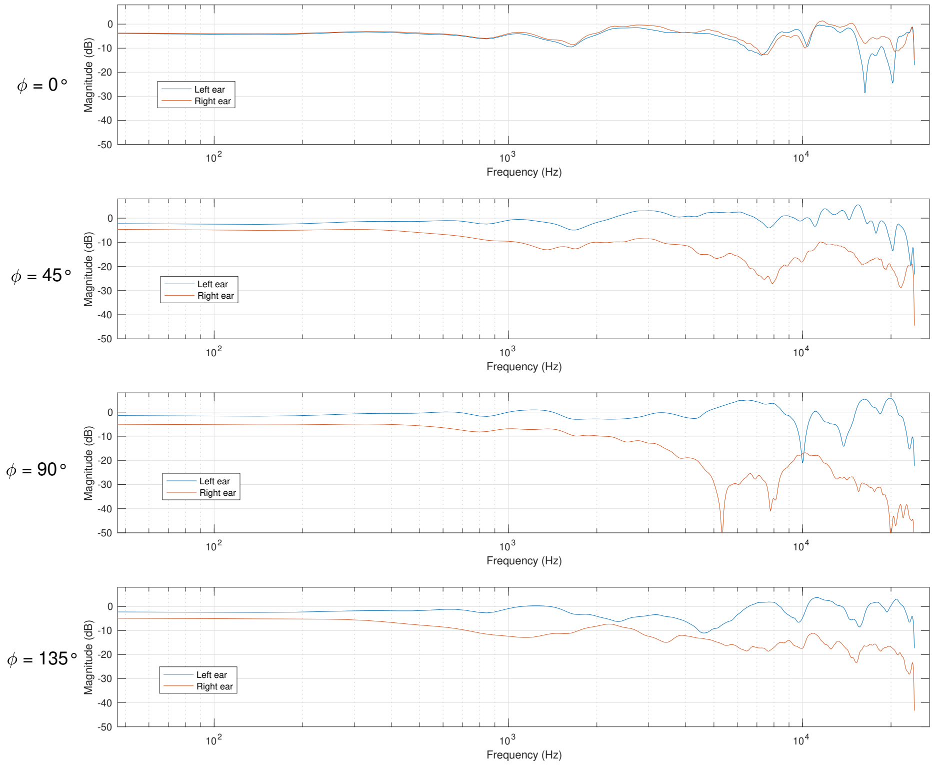

Figure 2.4: Left ear and right ear HRTF magnitude responses of the KEMAR mannequin from SADIE II Database [16] for 4 azimuths on the horizontal plane.

Another property of HRTFs is their predominately minimum-phase characteristic. In 1977, Mehrgardt and Mellert showed that HRTFs are approximately minimum-phase below 10 kHz [18]. They showed that since HRTFs are complex-valued functions of frequency, they can be written as the product of a *minimum-phase* function and an *all-pass* function:

$$H(e^{j\omega}) = H_{min}(e^{j\omega})H_{ap}(e^{j\omega}) \tag{2.9}$$

Generally speaking, any transfer function that describes a causal LTI system can be written as a product of a minimum-phase component and an all-pass component [19]. Minimum-phase transfer functions have all of their poles and zeros inside the unit circle of the z-plane. All-pass functions are characterized by a unit magnitude response. In 1995, Kulkarni et al. argued that subtle variations in the phase component of the all-pass factor of an HRTF are perceptually irrelevant when binaural synthesis is performed from a fixed direction [20]. In that case, it is justified to model an HRTF as a minimum-phase function plus a pure-delay component. One method for obtaining minimum-phase components is done by computing the complex cepstrum (the inverse Fourier transform of the log of the original spectrum) and reflecting anticausal components across the time = 0 axis to make them causal [21]. In other words, zeros and poles lying outside the unit circle are shifted to their conjugate reciprocals which lie inside the unit circle. Then, the all-pass component can be computed as the ratio of the original spectrum and minimum-phase component.

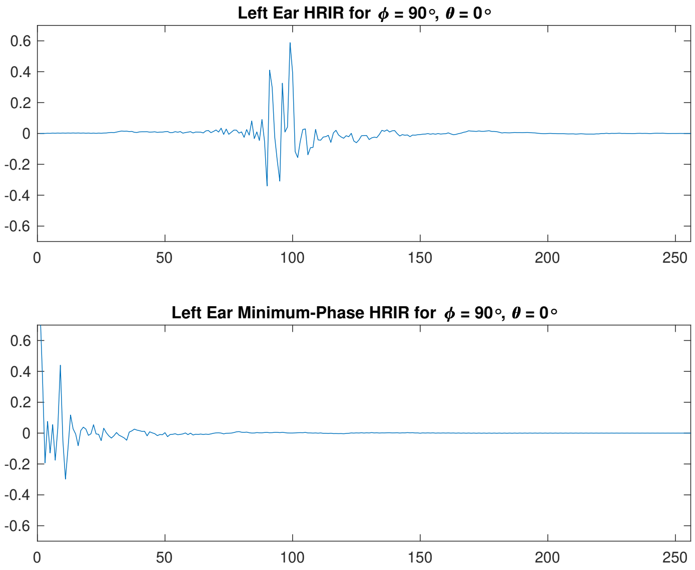

Figure 2.5: A left ear HRIR and its minimum-phase version.

Given that ITDs are pure delays (it takes longer for sound to reach the farther ear), the minimum-phase component is an approximation of the HRTF with the ITD removed (i.e. both ears are time-aligned). In 1992, Kistler and Wightman showed that localization error improved when reconstructing HRTFs by using minimum-phase approximations cascaded with simple delays [22]. Since then, many techniques for estimating ITDs with minimum-phase approximations have arisen. For example, Nam et. al [23] proposed an ITD estimator based on the value $\tau \in \mathbb{Z}$ that maximizes the cross-correlation of an HRIR $h[n]$ with its minimum-phase version $h_{min}[n]$:

$$\hat{\tau} = \underset{\tau}{\operatorname{argmax}} \left\{ \sum_{n} h[n-\tau] h_{min}[n] \right\} \tag{2.10}$$

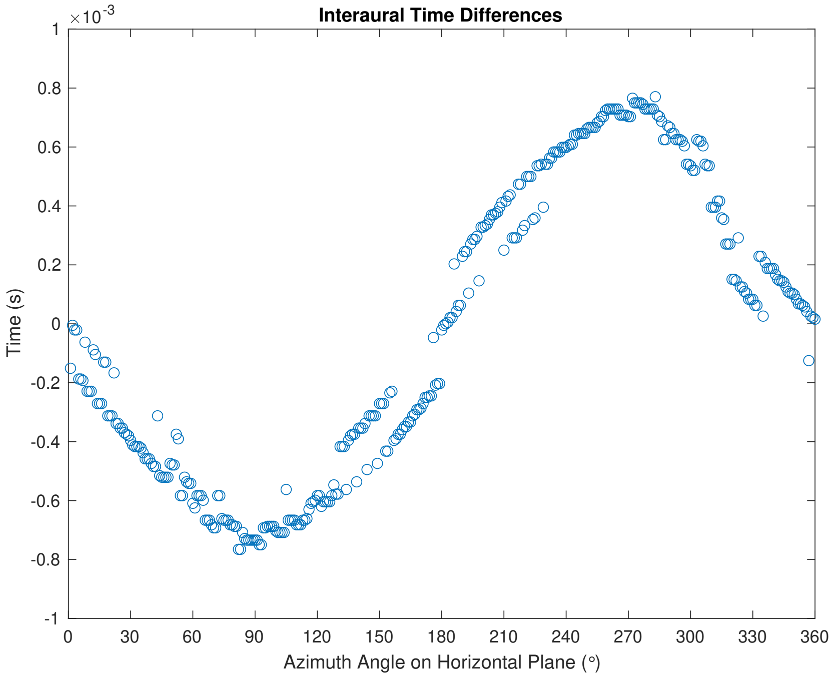

Figure 2.6: KEMAR mannequin ITDs estimated with Nam et al. technique [23] for the horizontal plane.

Once $\hat{\tau}$ is known for both the left and right ears, they can be subtracted to obtain the interaural time difference ITD:

$$ITD = \hat{\tau}_L - \hat{\tau}_R. \tag{2.11}$$

Figure 2.5 is a plot of ITDs for the KEMAR mannequin returned from this estimator for many azimuthal angles on the horizontal plane. Note the longest ITDs occur along the lateral plane while the shortest occur along the median plane. Also note the discontinuities and deviations from the overall sinusoidal shape of the plot. This is a result of various non-smooth anatomical features of the mannequin head as well as limitations of this ITD estimation approach.

Once an accurate estimation of ITDs is achieved, they can be simulated with fractional delay lines. Constructing good quality fractional delay lines of time-varying length (for example, with Lagrange interpolation [24]), however, is itself a computationally expensive process, making it costly to perform in an online scenario. For this reason, approximations of HRTFs as minimum-phase functions cascaded with all-pass functions for binaural rendering are left out of the scope of this thesis.

2.2.6 Functional Modeling of HRTFs

In practice, HRTFs are typically measured for an individual at discrete locations on a sphere. The density and uniformity of this measurement grid varies across HRTF databases and is often chosen according to the measurement equipment and resources available. As we have discussed, we ideally would be able to render sources coming from arbitrary angles, not only the angles that were used to measure the HRTF. Therefore, it is desirable to use the HRIRs available to build a continuous, functional model in order to construct HRIRs at unmeasured directions. This general process is sometimes referred to as *spatial interpolation*, which should not be confused with more precise definitions of interpolation (the act of fitting a function between two or more known points to estimate an intermediate point).

Many methods exist for spatially interpolating HRTFs. In [1], Bosun Xie grouped these into three categories: linear techniques, spectral shape basis function techniques, and spatial basis function techniques. This thesis will explore bilinear interpolation of HRTFs (which falls under linear techniques) and spherical harmonic decomposition of HRTFs (which falls under spatial basis function techniques). For completeness, we will mention some other popular approaches before analyzing these two in-depth in the following sections.

Linear techniques can most generally be described as estimating unknown HRIRs/HRTFs via a linear combination of known HRIRs/HRTFs. If we maintain a constant source distance $r = r_0$, this can be written in the time domain as:

$$\hat{h}_{\phi,\theta}[n] = \sum_{i=0}^{M-1} w_i h_{\phi_i,\theta_i}[n] \tag{2.12}$$

where the index $i = 0, 1, ..., M-1$ spans the total number of spatial samples M and w_i are weights associated with each HRIR. Due to the linearity property of the discrete Fourier transform, we can write the same equation in the frequency domain:

$$\hat{H}_{\phi,\theta}(e^{j\omega}) = \sum_{i=0}^{M-1} w_i H_{\phi_i,\theta_i}(e^{j\omega}) \tag{2.13}$$

The weights w_i and HRIRs/HRTFs are chosen based on the selected interpolation scheme. As a basic example, one could perform *adjacent linear interpolation*. Given two adjacent azimuths (ϕ_0, θ_0) and (ϕ_1, θ_0) and their corresponding measured HRIRs $h_{\phi_0,\theta_0}[n]$ and $h_{\phi_1,\theta_0}[n]$, an intermediate HRIR $\hat{h}[n]$ located halfway between these two azimuths could be calculated as

$$\hat{h}[n] = 0.5 * h_{\phi_0,\theta_0}[n] + 0.5 * h_{\phi_1,\theta_0}[n]. \tag{2.14}$$

This technique is generalizable to arbitrary intermediate points where the weights are inversely proportional to the distance to each measured HRIR. *Bilinear interpolation* and *barycentric interpolation* are two-dimensional extensions of adjacent linear interpolation. Bilinear interpolation will be explored more in-depth in Section 2.4.

It is also possible to interpolate HRTFs with a recursive IIR structure, although issues of stability often arise. In 1995, Jot et al. gave a method for pairing and ordering HRIRs that allow for IIR structures to be easily imposed [25]. The four possibilities for IIR filter representation given in that paper are:

- direct-form coefficients of the cascaded second-order sections,
- magnitudes and log-frequencies of the poles and zeros,
- measurement grid filter coefficients,
- log area ratios of the measurement grid filter coefficients.

In IIR representation, the act of dynamically updating filter coefficients can create transients (heard as audible clicks) that must be compensated for with cross-fading [26].

Decomposition of HRTFs with spectral shape basis functions involves linear combinations of frequency-dependent basis functions that constitute various HRTF spectral shapes [1]. The most common scheme for selecting the basis functions and weights is called Principal Component Analysis (PCA). PCA was first applied to HRTFs in 1987 by Martens [27]. The technique involves eliminating correlations among HRTFs in order to reduce dimensionality. Many researchers have implemented and published different versions of PCA [22] [28] [29] [30], but here we will review the basic principle.

In PCA, eliminating correlations involves subtracting the mean HRTF across source directions from each of the Q measured HRTFs. Then, a Hermitian matrix is constructed from this matrix and its eigenvectors are computed. These eigenvectors are then the spectral shape basis vectors. Finally, the weights are obtained by checking the orthonormality of the uncorrelated HRTFs with each spectral shape basis vector.

Other methods for performing decomposition of HRTFs with spectral shape basis functions are PCA via singular value decomposition (SVD) [31] and subset selection of HRTFs [32].

In the next section, we will review in-depth the theory of spatial basis function decomposition of HRTFs using the spherical harmonic basis functions as an example.

2.3 Spherical Harmonic Decomposition of HRTFs

This section presents the technique for representing HRTFs in the spherical harmonics (SH) domain. We then describe spatial interpolation of SH-encoded HRTFs.

2.3.1 The Spherical Fourier Transform

The precise definition of spherical harmonics expansion varies greatly depending on author inclinations or application-specific conventions, so care must be taken when comparing equations. The coordinate system used in this thesis is the spherical coordinate system suggested by [8] (shown in Figure 2.7) which specifies locations (ϕ, θ, r). This is also known as the right-handed vertical-polar coordinate system. Here, $0° \leq \phi < 360°$ is the azimuthal angle measured counterclockwise from the positive x-axis and $-90° \leq \theta \leq 90°$ is the elevation angle measured relative to the x-y (horizontal) plane.

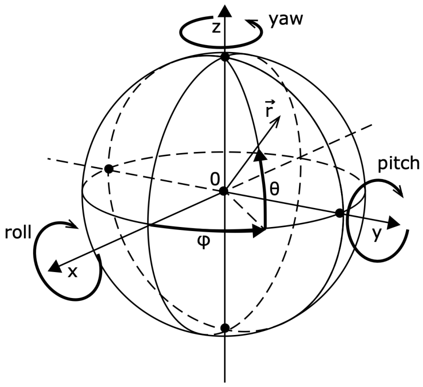

Figure 2.7: Spherical coordinate system used, as illustrated in [8].

SH decomposition of HRTFs can be considered as an analysis/synthesis process, where we first analyze a sound field with a projection onto SH basis functions, then re-synthesize that sound field using these basis functions. The analysis step is executed using the *spherical Fourier transform*, which is given by

$$H_{nm}^{L,R}(k) = \int_{\phi=0}^{2\pi} \int_{\theta=-\pi/2}^{\pi/2} H^{L,R}(\phi, \theta, k) Y_n^m(\phi, \theta)^* \sin(\theta) d\theta d\phi \tag{2.15}$$

Here, we are considering decomposition of HRTFs for both the left $H^L(\phi, \theta, k)$ and right $H^R(\phi, \theta, k)$ ears, where each of these are idealized continuous functions in the spherical angles $0° \leq \phi < 360°$ and $-90° \leq \theta \leq 90°$; $k = 2\pi f/c$ is the wave number with frequency f and speed of sound c. The complex spherical harmonics Y_n^m are defined by

$$Y_n^m(\phi, \theta) = \sqrt{\frac{2n+1}{4\pi} \frac{(n-m)!}{(n+m)!}} P_n^m(\cos(\theta)) e^{jm\phi} \tag{2.16}$$

where n is the spherical harmonic *order*, m is the spherical harmonic *degree*, and $j = \sqrt{-1}$. $(\cdot)^*$ denotes complex conjugation. The spherical harmonics are the solution to the wave equation (the Helmholtz equation) in spherical coordinates [33]. The P_n^m terms are the associated Legendre polynomials, which represent standing spherical waves for the elevation angle θ, whereas the term $e^{jm\phi}$ represents traveling spherical waves for the azimuth angle ϕ. The integral $\int_0^{2\pi} \int_{-\pi/2}^{\pi/2} \sin(\theta) d\theta d\phi$ is a surface integral over the entire unit sphere.

The SHs are orthonormal with respect to each other, i.e.:

$$\int_0^{2\pi} \int_{-\pi/2}^{\pi/2} Y_{n'}^{m'}(\phi, \theta) Y_n^m(\phi, \theta)^* \sin(\theta) d\theta d\phi = \delta_{n-n'} \delta_{m-m'} \tag{2.17}$$

where δ is the Kronecker delta function and n' and m' are an arbitrary SH order and degree, respectively.

In Equation 2.15, $H_{nm}^{L,R}(k)$ is then referred to as the spherical Fourier transform of $H^{L,R}(\phi, \theta, k)$ and represents the contributions of the basis functions Y_n^m. To re-synthesize the HRTFs from these weighted basis functions, we perform the inverse spherical Fourier transform, which is given by

$$H^{L,R}(\phi, \theta, k) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} H_{nm}^{L,R}(k) Y_n^m(\phi, \theta) \tag{2.18}$$

where $H_{nm}^{L,R}(k)$ is as in Equation 2.15 and $Y_n^m(\phi, \theta)$ is as in Equation 2.16.

In practice, however, the HRTFs are not continuous in ϕ, θ but are sampled at Q

directions given by $\Omega_1, \Omega_2, \ldots \Omega_Q$ where each of these is an ordered pair (ϕ, θ). Additionally, the SH representation is truncated at an order $N < \infty$ which results in $(N+1)^2$ total SH coefficients.

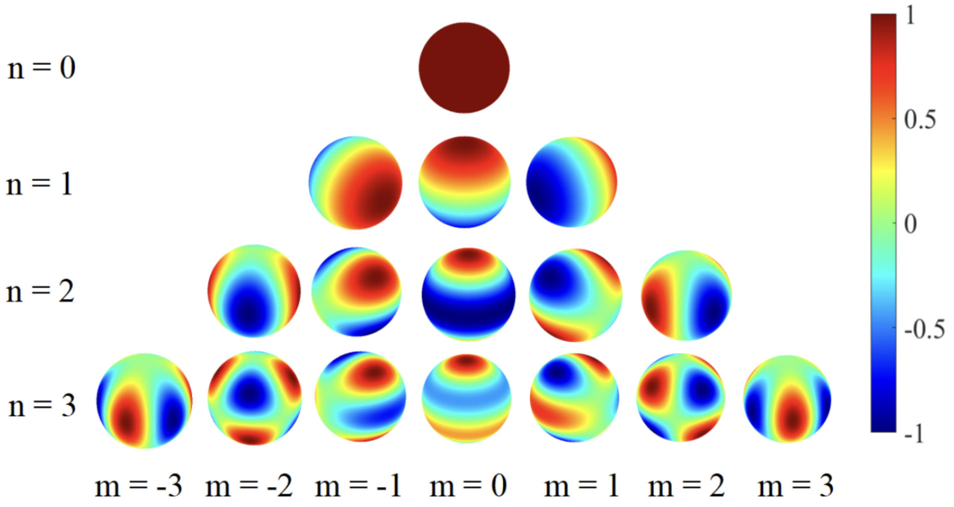

Figure 2.8: Projection of several real spherical harmonic functions onto the surface of the sphere. As order n increases, the inclusion of more SHs permit higher spatial resolution during decomposition of functions such as HRTFs. From [34].

All HRTFs in a measured HRTF set can be given by the length-Q space-domain column vector $\boldsymbol{H} = [H(\Omega_1, k), H(\Omega_2, k), \ldots, H(\Omega_Q, k)]^T$. Then, Equation 2.18 can be rewritten as the matrix multiplication

$$\boldsymbol{H} = \boldsymbol{Y_n^m} \boldsymbol{H_{nm}}, \tag{2.19}$$

where $\boldsymbol{Y_n^m}$ is the Q x $(N+1)^2$ SH transformation matrix given by

$$\boldsymbol{Y_n^m} = \begin{bmatrix} Y_0^0(\Omega_1) & Y_{-1}^1(\Omega_1) & \ldots & Y_N^M(\Omega_1) \\ Y_0^0(\Omega_2) & Y_{-1}^1(\Omega_2) & \ldots & Y_N^M(\Omega_2) \\ \vdots & \vdots & \ddots & \vdots \\ Y_0^0(\Omega_Q) & Y_{-1}^1(\Omega_Q) & \ldots & Y_N^M(\Omega_Q) \end{bmatrix} \tag{2.20}$$

and $\boldsymbol{H_{n,m}}$ is a length-$(N+1)^2$ row vector containing the contributions of each basis function. In audio, a convention has been adopted for mapping the coupled indices n, m onto a single index. This index is called the Ambisonic Channel Number (ACN). Thus, the rows of $\boldsymbol{Y}_{\boldsymbol{n}}^{\boldsymbol{m}}$ span the $(N+1)^2$ ACNs while the columns span the locations of the Q HRTF measurements. Here, we have omitted the superscript L, R for simplicity.

Likewise, we can write Equation 2.15 as

$$\boldsymbol{H}_{\boldsymbol{nm}} = \boldsymbol{Y}^{\dagger}\boldsymbol{H} \tag{2.21}$$

where $\boldsymbol{Y}^{\dagger} = (\boldsymbol{Y}^{H}\boldsymbol{Y})^{-1}\boldsymbol{Y}^{H}$ is the Moore-Penrose pseudoinverse of the SH matrix $\boldsymbol{Y}$. In this way, we attempt to find the unique least-squares estimate of the SH weights $\boldsymbol{H}_{\boldsymbol{nm}}$. This estimate is only possible if there are more spatial HRTF samples than SH coefficients to be calculated, i.e. $Q > (N+1)^2$. In the next section, we will see that this bound is larger in practice.

If we have our HRTF represented with $\boldsymbol{H}_{\boldsymbol{nm}}$, we can perform spatial interpolation, i.e. calculate the HRTF at any desired angle. Let $\boldsymbol{L}$ represent a set of query angles at which we wish to estimate individual HRTFs. Then,

$$\boldsymbol{H}_{\boldsymbol{L}} = \boldsymbol{Y}_{\boldsymbol{L}}\boldsymbol{H}_{\boldsymbol{nm}} \tag{2.22}$$

is the inverse spherical Fourier transform (ISFT) where the query SH transformation matrix $\boldsymbol{Y}_{\boldsymbol{L}}$ is equal to Equation 2.20 with $\Omega_1, \Omega_2, \ldots$ corresponding to each angle in $\boldsymbol{L}$. Accordingly, $\boldsymbol{H}_{\boldsymbol{L}}$ will result in a vector with each element consisting of an HRTF localized at the corresponding angle in $\boldsymbol{L}$. In this way, we can spatially interpolate an HRTF.

2.3.2 Spatial Aliasing

In the same way microphones sample time-domain sound pressures, the locations of fixed-radius HRTF measurements spatially sample the surface of a sphere. This sampling requires limited bandwidth to prevent spatial aliasing. Moreover, the uniformity of this sampling with respect to the surface of the sphere is highly important because it is directly related to the number of measurements Q required to perform the least-squares estimation. When considering SH decomposition of HRTFs, we are constrained by measurement apparatuses, which often do not uniformly sample the sphere. In this section, we will consider SH-encoded HRTFs as mode-limited functions and discuss this effect on the minimum SH truncation order.

As seen in the previous section, HRTFs can be robustly approximated if we choose a sufficiently large SH truncation order N:

$$H^{L,R}(\phi,\theta,k) \approx \sum_{n=0}^{N} \sum_{m=-n}^{n} H_{nm}^{L,R}(k) Y_n^m(\phi,\theta) \tag{2.23}$$

This approximation can be rewritten using the Jacobi-Anger expansion into a linear combination of a product of three different types of separable basis functions weighted by the SH coefficients $H_{nm}^{L,R}(k)$. For a complete overview of this expansion, see Section III of [35]. These three types of basis functions are the spherical harmonics, the spherical Bessel function, and the spherical Hankel function of the first kind. Each of these are hierarchically organized based on a "mode" index n, for which we will consider truncations at N. In this way, HRTFs can be considered as mode-limited functions. The spherical harmonics and the spherical Hankel function of the first kind represent the HRTF spatial variations whereas the spherical Bessel function and the SH coefficients $H_{nm}^{L,R}(k)$ represent the HRTF spectral components.

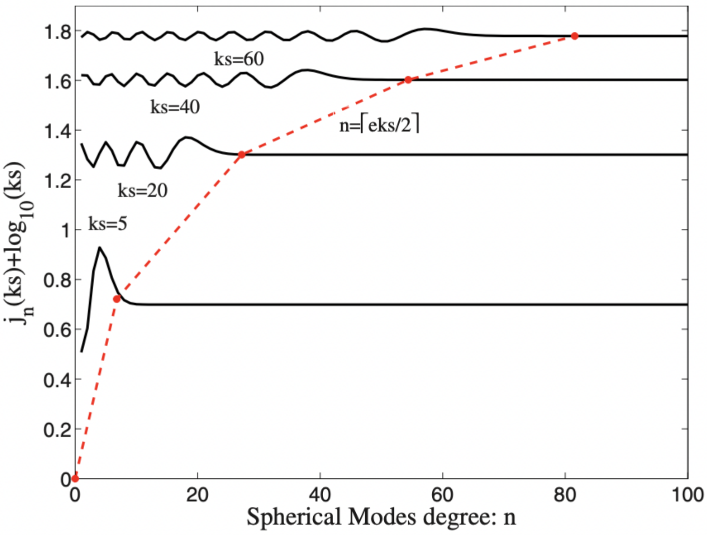

Figure 2.9: Dependence of the point of decay of the spherical Bessel function $j_n(ks)$ on n for a few values of ks, shown on the vertically shifted curves. From [35].

The lower bound of the truncation order N is necessary in order to accurately represent the spatial variations of the HRTF (i.e. prevent spatial aliasing). For a given frequency, the spherical Bessel function will oscillate up to some N then quickly decay. To fully include the HRTF spatial variations, we must not truncate the decomposition order until the spherical Bessel function decays. The length of this oscillation is proportional to the frequency we are trying to represent such that higher frequencies require larger N. This relationship can be described as

$$N = \lceil eks/2 \rceil \tag{2.24}$$

where e is Euler's number, $k = 2\pi f/c$ is the wave number, f is the frequency, $c = 343$ m/s is the speed of sound, and s is the radius of the smallest sphere surrounding an average head. A commonly cited value for s is 8.75 cm [36]. This relationship is depicted in Figure 2.9. The minimum number of HRTF measurements Q required to retrieve HRTFs corresponding to all directions, then, is given by

$$Q \geq (N+1)^2 = (\lceil eks/2 \rceil + 1)^2. \tag{2.25}$$

Thus, if we are trying to represent a bandwidth of 20 Hz to 20 kHz (i.e. the range of audible frequencies), a truncation order $N = 46$ and number of HRTF measurements on the sphere $Q = 2209$ are required. If we do not have enough measurements to satisfy this criteria, we are still able to decompose the HRTFs at the cost of inaccurate representation of the higher frequencies.

2.3.3 Spatial Sampling on the Sphere

Schemes for sampling a sphere are studied in a variety of fields [37] [38]. Even within audio and acoustics, there are many sampling schemes which have desirable properties for a specific application. For example, T-designs of platonic solids [39] and Fliege nodes [40] are particularly useful in spherical microphone array processing [41] while Lebedev grids [42] find application in Ambisonics [43].

In the analysis step of spherical harmonic decomposition of HRTFs, the sampling scheme chosen has a large impact on the SH representation of the HRTFs. In the previous section, we showed that at least $Q \geq (N+1)^2$ HRTF measurements are required to accurately construct arbitrary HRTFs. In practice, however, this lower bound is complicated by a factor $\lambda \geq 1$, i.e.:

$$Q \geq \lambda(N+1)^2 \tag{2.26}$$

where λ represents an *oversampling* factor intrinsic to sampling schemes that do not spatially sample the sphere in a uniform fashion. We must consider λ because HRTF measurement apparatuses typically do not permit a uniform spatial sampling of the sphere. In this section, we will present a few schemes for spatially sampling a sphere and discuss their advantages and disadvantages.

Equiangular sampling is the process of sampling a sphere with equal spacing in both azimuth ϕ and elevation θ:

$$\phi_i = \frac{2\pi i}{\sqrt{Q}},\ i = 0, 1, \ldots, \sqrt{Q} - 1 \tag{2.27}$$

$$\theta_j = \frac{\pi j}{\sqrt{Q}} - \frac{\pi}{2},\ j = 0, 1, \ldots, \sqrt{Q} - 1 \tag{2.28}$$

where we have Q total samples and have defined $0 \leq \phi < 2\pi$ and $-\pi/2 \leq \theta \leq \pi/2$. The regular angle differences make this a natural sampling scheme, especially in hardware measurement setups where a fixed step mechanical rotation in ϕ and θ is simple. Moreover, table lookups can be accomplished in $\mathcal{O}(1)$ time with basic modulo and rounding operations. Equiangular sampling, however, suffers from a denser grid near the poles, which means we are oversampling near the poles and undersampling near the equator (i.e. not spatially sampling the sphere in the minimum number of samples necessary for the least-squares estimation). Driscoll and Healy [44] showed that the oversampling factor $\lambda = 4$ for equiangular sampling of bandlimited functions, requiring $2N + 2$ samples in both azimuth and elevation.

Because the least-squares SFT will be biased and therefore inaccurate if we have more samples near the poles, we can compensate for this by appling Voronoi weights to each measurement. Voronoi weights are coefficients obtained by performing a spherical Voronoi tesselation of the sampling grid. A spherical Voronoi tesselation is a partition of the surface area of a sphere into regions defined by the spatial samples on the sphere where the vertices of each region are the samples closest to each other. The weights, then, are proportional to the area of these regions (given as a ratio relative to the total surface area $4\pi r^2$) such that groups of samples far apart will be assigned more weight than groups of samples close together. By using a weighted least-squares SFT, we can compensate for a non-uniform sampling.

A common sampling scheme used in publicly available HRTF databases is the *mixed* equiangular grid. In mixed equiangular grids, the equal spacing between consecutive angles in azimuth differs from the equal spacing between consecutive angles in elevation. If we have Q_ϕ samples in ϕ and Q_θ samples in θ,

$$\phi_i = \frac{2\pi i}{Q_\phi},\ i = 0, 1, \ldots, Q_\phi - 1 \tag{2.29}$$

and

$$\theta_j = \frac{\pi j}{Q_\theta} - \frac{\pi}{2},\ j = 0, 1, \ldots, Q_\theta - 1 \tag{2.30}$$

where $Q = Q_\phi Q_\theta$ is the total number of measurement points. The weighted least-squares SFT is then given by

$$H_{nm}^{L,R}(k) = \sum_{j=0}^{Q_\theta - 1} \sum_{i=0}^{Q_\phi - 1} \alpha_i H^{L,R}(\phi_i, \theta_j, k) Y_n^m(\phi_i, \theta_j)^\dagger \tag{2.31}$$

where α_j are the Voronoi weights and $Y_n^m(\phi_j, \theta_k)^\dagger$ is the Moore-Penrose pseudoinverse of Equation 2.20. A further extension of the mixed equiangular grid is to vary the number of azimuths depending on the elevation such that elevations near the poles have less samples than elevations near the equator [45].

Lebedev sampling is a scheme characterized by the construction of quadratures that are rotationally-invariant on the sphere [43]. The motivation for this is to find a set of grid points and corresponding weights that enforce exact integration of the spherical harmonics up to some order N while maintaining a near-uniform distribution and keeping the grid small [42]. The sample points and weights can be found in [42] and the references within. In Lebedev grids, the oversampling factor $\lambda = 1.3$, meaning we can reconstruct the sound field with far less HRTF measurements than equiangular grids.

2.4 HRTF Interpolation via Weighted Averaging

Another method for estimating arbitrary HRIRs/HRTFs for rendering the anechoic path in object-based binaural audio is by direct interpolation of the HRIR/HRTF coefficients. This approach is more simple than using basis functions since it only involves weighted averages of the nearest HRIRs/HRTFs on the measurement grid. Many techniques exist for directly interpolating coefficients. These are generally grouped into bilinear or barycentric methods. Bilinear interpolation of the four closest points (BI4C) was first suggested by [46] while a simplified version using just three points was given by [47]. Gamper proposed a method for using barycentric interpolation in azimuth, elevation, and distance by considering the 3D tetrahedron surrounding the query point [48]. Most recently, Cuevas-Rodriguez et al. used 2D barycentric interpolation of HRIRs to render the anechoic path in the open-source auralization engine 3D Tune-In Toolkit [3]. In this section, we will review the theory of the bilinear method.

2.4.1 Bilinear Method

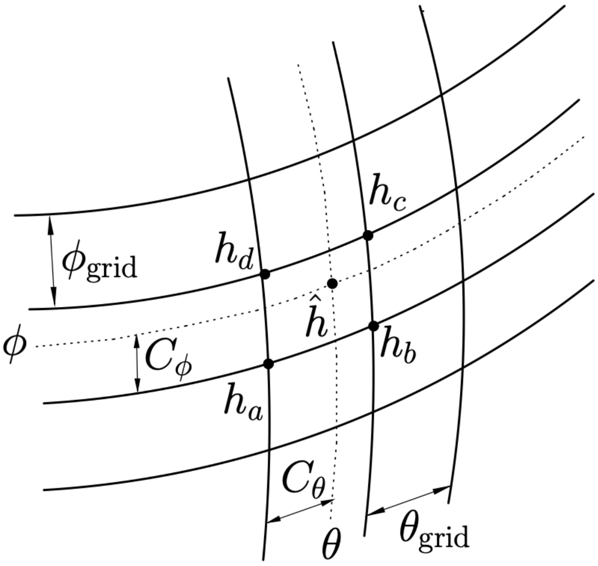

Figure 2.10: Bilinear interpolation of the four closest involves a weighted averaging of the HRIRs/HRTFs that construct the vertices of the rectangle surrounding the query angle. From [49].

Bilinear interpolation is only applicable if our HRTF measurement grid has a regular basis. If this is the case, the bilinear method is simply linear interpolation along two orthogonal axes.

Consider a mixed equiangular grid with azimuthal spacing ϕ_{grid} and elevation spacing θ_{grid}, as in Figure 2.10. Given some query angle (ϕ, θ), we begin by locating the rectangle in which this angle exists, where the vertices of the rectangle are defined by known HRIRs/HRTFs. In Figure 2.10, these four HRIRs are given by h_a, h_b, h_c, and h_d. The interpolated HRIR $\hat{h}[n]$ is computed as a weighted sum of these four closest HRIRs:

$$\hat{h}[n] = (1 - C_\theta)(1 - C_\phi)h_a[n] + C_\theta(1 - C_\phi)h_b[n] + C_\theta C_\phi h_c[n] + (1 - C_\theta)C_\phi h_d[n], \quad (2.32)$$

where $C_\phi = \phi - \phi_a$ and $C_\theta = \theta - \theta_a$. Here, (ϕ_a, θ_a) is the location of the HRIR h_a.

2.5 FFT-Based Convolution for Real-Time Auralization

In order to properly compare N-th order SH decomposition (SHD-N) of HRTFs with bilinear interpolation of the four closest (BI4C) in terms of computational cost, we must consider their implementation in an auralization engine. A fair computational cost comparison should attempt the most efficient implementation of both interpolation techniques, which should include the convolution step (convolution of the interpolated HRIRs with each source signal). As such, in this section, we'll review fast convolution techniques and their incorporation in a real-time auralization engine.

2.5.1 FFT-Based Fast Convolution

Ever since the legendary paper by Cooley and Tukey describing the fast Fourier transform (FFT) was published in 1965 [50], the FFT has become one of the most important and ubiquitous tools in signal processing. It is an algorithm for efficiently computing the discrete Fourier transform which is regarded as one of the 'top ten algorithms of the (20th) century' [51]. The method that applied the FFT to convolution was given by Stockham in 1966 [52]. Today, when people refer to *fast* convolution techniques, they are almost always referring to FFT-based convolution.

Convolution can be utilized in a variety of contexts. In this thesis, we are interested in the most efficient implementation of real-time FIR filtering using general purpose processors (i.e. not specialized DSPs). For real-time audio on these processors, we are constrained to block-based processing; that is, the real-time audio stream is partitioned into *blocks* (also called *frames*) of samples on which we perform signal processing tasks. Let B denote the *block length*. Typical values for B are powers of two such as 64, 128, 256, 512, or 1024 samples. At a sample rate of 44.1 kHz, these correspond to block durations T_B of 1.45 ms, 2.90 ms, 5.80 ms, 11.61 ms, and 23.22 ms, respectively. At the beginning of every audio cycle (i.e. block), a vector of B input samples is provided by some component. At the end of that cycle, a vector of B output samples is requested by the audio driver for playback. The time in-between the input and output transfers can then be used to perform signal processing tasks on the current block. It is important that the computational cost of the signal processing does not exceed the block duration in order to not cause dropouts in the playback. In practice, intermediate operations (such as the transportation of the blocks between the audio device and CPU) consume additional time so typically only 90-95% of the block duration can be spent on signal processing [7].

Consider the linear convolution of a length-M time-domain input block $x[n]$ with a length-N impulse response $h[n]$:

$$y[n] = (x * h)[n]. \tag{2.33}$$

Convolution can be accomplished via element-wise multiplication of coefficients in the frequency-domain:

$$y[n] = \mathcal{DFT}^{-1}_{(K)} \left\{ \mathcal{DFT}_{(K)} \left\{ x[n] \right\} \times \mathcal{DFT}_{(K)} \left\{ h[n] \right\} \right\} \tag{2.34}$$

where $\mathcal{DFT}_{(K)} \{ \cdot \}$ is a K-point discrete Fourier transform (DFT) operator. The transform size K must satisfy $K \geq M + N - 1$. To perform the K-point DFTs, both $x[n]$ and $h[n]$ are zero-padded to length K. Then, the steps involved in FFT-based convolution are K-point forward FFTs of both $x[n]$ and $h[n]$, K complex-valued multiplications, and a single K-point inverse FFT (IFFT). The first $M + N - 1$ samples of the IFFT output correspond to the linear convolution of $x[n]$ and $h[n]$.

The algorithms behind the state-of-the-art FFT libraries are outside of the scope of this thesis. It is pertinent, though, that the computational cost of an FFT is non-trivial relative to the other operations in the HRTF interpolation and anechoic path rendering algorithms.

For real-time FIR filtering, the input is partitioned into uniform blocks of length B, as previously discussed. However, using the above convolution algorithm on each individual block would cause artefacts at the edges of each output block. Therefore, we must rely on a set of algorithms designed to perform running convolutions on the partitioned input. In 2015, Wefers published a comprehensive overview of these algorithms, including benchmarks [7]. For the most efficient realization of real-time FIR filtering, the algorithm to choose depends on the length N of the impulse response relative to the block length B. Wefers suggests using either the Overlap-Add (OLA) [53] or Overlap-Save (OLS) [19] methods for short filters $N \leq B$. Uniformly partitioned convolution algorithms are suggested for $B < N < 20B$ while non-uniformly partitioned convolution algorithms are proposed for long filters $N \geq 20B$. Since HRIRs are typically shorter than common block lengths, we will only consider the conventional OLS and OLA methods for implementing FFT-based convolution. Moreover, both OLS and OLA require the same number of FFTs, IFFTs, and complex-valued multiplications. However, OLS avoids some extra additions that are necessary in OLA. For this reason, only OLS will be explored in the context of this thesis.

2.5.2 Overlap-Save Method

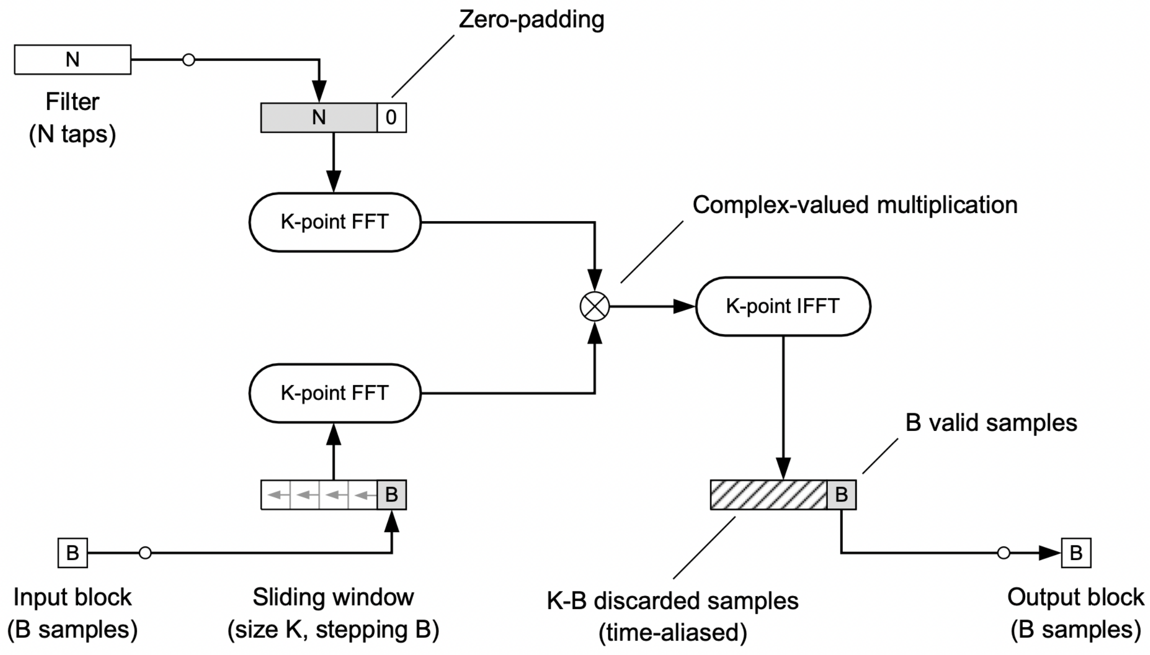

Figure 2.11: FFT-Based running convolution incorporating the Overlap-Save method for real-time FIR filtering. From [7].

The Overlap-Save Method is a technique for incorporating unpartitioned convolution techniques with partitioned input signals (e.g. block-based audio processing). The algorithm is shown in Figure 2.11. A length-K sliding window $s[n]$ of the input is constructed. At the beginning of each audio cycle, the contents of this window are shifted left by B samples, with the left-most B samples discarded. The input block is copied into the right-most B samples of the sliding window $s[n]$. A K-point real-to-complex (R2C) FFT of $s[n]$ is performed. Since all K input values are real-valued, the corresponding DFT spectrum $S(k)$ holds Hermitian symmetry [19]:

$$S(k) = S(K-k)^* \tag{2.35}$$

The presence of complex-conjugate symmetry means that half of the K DFT coefficients are redundant. Therefore, the entire DFT spectrum can be constructed from just C complex-conjugate symmetric coefficients where

$$C = \left\lceil \frac{K+1}{2} \right\rceil. \tag{2.36}$$

The impulse response $h[n]$ is zero-padded to length K and also FFTed. Then, each of the C coefficients in the spectra $S(k)$ and $H(k)$ are pairwise multiplied with complex-valued multiplication. A K-point complex-to-real (C2R) IFFT is performed and only the final B samples are saved and sent to the output block.

If the impulse response $h[n]$ can be zero-padded and FFTed offline, the entire computational cost of FFT-based convolution with the OLS method is then the cost of shifting and copying into the sliding window, twice the cost of a K-point R2C FFT, the cost of K complex-valued multiplications, the cost of a K-point C2R IFFT, and the cost of truncating $K-B$ time-aliased samples.

2.5.3 Filter Exchange Strategies

Ideally, a virtual acoustics auralization engine would be resilient to movement of sources in the space over time. Since the location of a source in the virtual space is simply given as the source signal convolved with the corresponding HRIR, moving sources corresponds to instantaneously exchanging HRIRs measured or interpolated at the locations we are interested in. However, it is not natural from a listener's perspective for sound objects to jump to different locations in a discontinuous fashion. Therefore, we need a way to apply a cross-fade between consecutive audio frames for each source in order to create the illusion of a sound moving smoothly across the space.

In 2014, Wefers and Vorlaender published a technique for cross-fading consecutive FIR filters based on operators working on the DFT spectra [54]. The technique can easily be incorporated in the OLA or OLS methods for real-time FFT-based convolution. The DFT operators are derived by considering the DFT of the time-domain sinusoidal amplitude envelopes

$$f_{in}[n] = \sin^2\left(\frac{\pi n P}{K}\right) \tag{2.37}$$

$$f_{out}[n] = \cos^2\left(\frac{\pi n P}{K}\right) \tag{2.38}$$

where the sum of these maintains unit amplitude:

$$f = f_{in}[n] + f_{out}[n] = \sin^2\left(\frac{\pi n P}{K}\right) + \cos^2\left(\frac{\pi n P}{K}\right) = 1. \tag{2.39}$$

Here, the transform size K must be an integer multiple P of double the block length B

$$P = \frac{K}{2B} \in \mathbb{N} \tag{2.40}$$

in order for the right-most B samples to coincide with a half-period of the envelopes. This is illustrated in Figure 2.12.

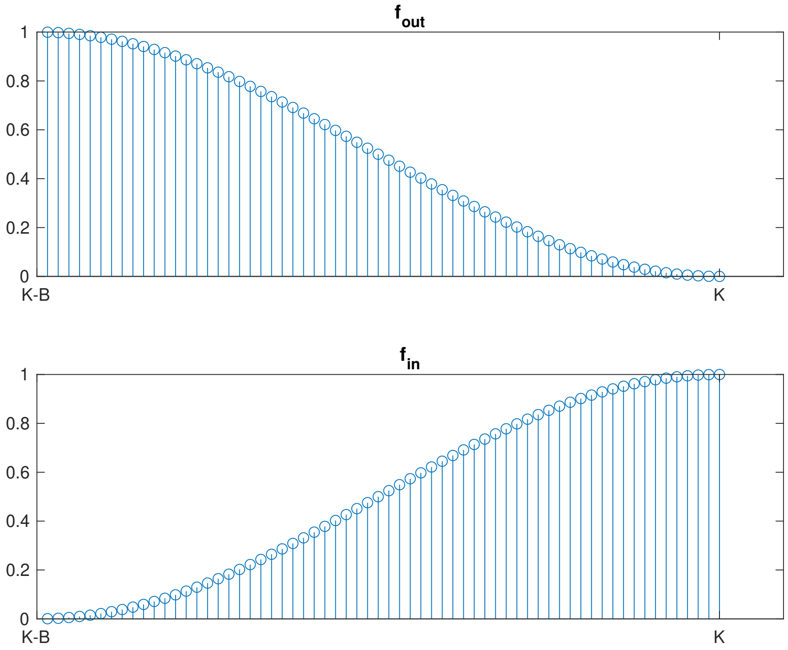

Figure 2.12: Fade out f_{out} and fade in f_{in} envelopes must coincide with the right-most B samples of the FFT buffer in order to be applied to the time-domain samples after the IFFT.

It is certainly possible to cross-fade by simply multiplying the amplitude envelopes in Figure 2.12 to the time-domain filtered signals and sending the sum to the output. In other words, if we are exchanging the current audio buffer filtered by the impulse response from the previous audio cycle $y_0[n]$ for the current audio buffer filtered by the impulse response from the current audio cycle $y_1[n]$, we could write

$$y[n] = f_{out}[n]y_0[n] + f_{in}[n]y_1[n]. \tag{2.41}$$

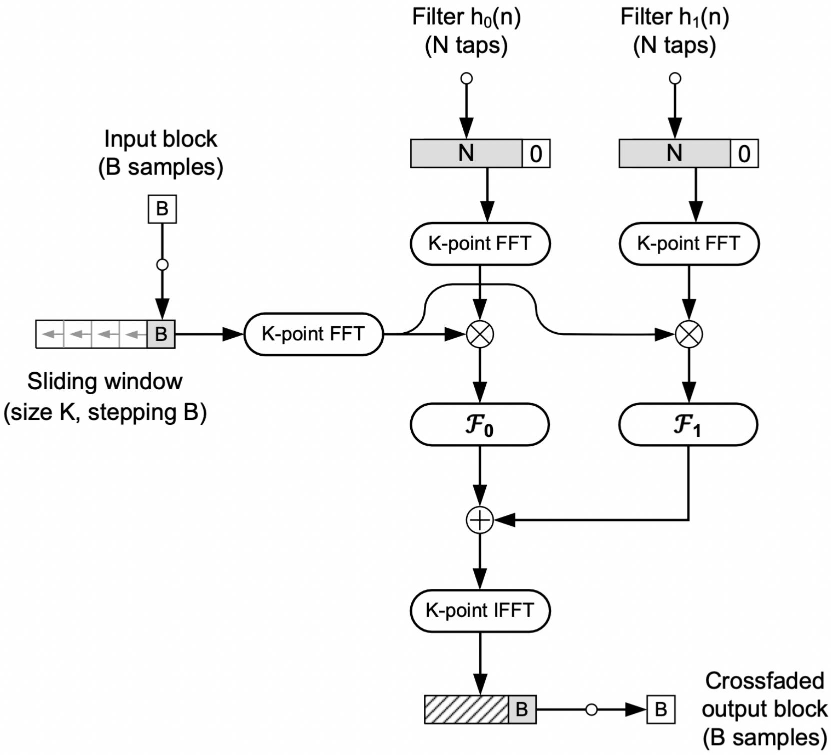

Figure 2.13: Overlap-Save FFT-based convolution with frequency-domain cross-fading. From [54].

But, Wefers showed via benchmarking that it is more efficient to apply these envelopes with frequency domain operations on the DFT spectra $Y_0(k)$ and $Y_1(k)$, which correspond to the current source buffer filtered by the previous impulse response and current impulse response, respectively. This is given as

$$Y(k) = K\left[Y_0\left\langle k\right\rangle_K + Y_1\left\langle k\right\rangle_K + \frac{1}{2}[Y_1\left\langle k+P\right\rangle_K - Y_0\left\langle k+P\right\rangle_K + Y_1\left\langle k-P\right\rangle_K - Y_0\left\langle k-P\right\rangle_K]\right], \tag{2.42}$$

where $\langle\cdot\rangle_K$ denotes the K-periodic continuation of each spectra. The incorporation of this cross-fading into FFT-based convolution with the OLS method is shown in Figure 2.13 where the fade-out and fade-in DFT operators $\mathcal{F}_0$ and $\mathcal{F}_1$ are

$$\mathcal{F}_0\{Y_0(k)\} = -\frac{K}{2}Y_0\left\langle k+P\right\rangle_K + KY_0\left\langle k\right\rangle_K - Y_0\left\langle k-P\right\rangle_K \tag{2.43}$$

and

$$\mathcal{F}_1\{Y_1(k)\} = \frac{K}{2}Y_1\left\langle k+P\right\rangle_K + KY_1\left\langle k\right\rangle_K + Y_1\left\langle k-P\right\rangle_K . \tag{2.44}$$

In Chapter 4, we will discuss how to incorporate bilinear interpolation of the four closest (BI4C) and N-th order spherical harmonic decomposition (SHD-N) into the algorithm shown in Figure 2.13 order to properly compare computational costs.

Chapter 3

Reconstruction Error Analysis

An ideal HRTF interpolation scheme should return a function identical to an HRTF measured at that location. Therefore, to measure the interpolation quality of SHD-N and BI4C, we can attempt to reconstruct measurements with each scheme. In BI4C, we delete the HRTF to be reconstructed from the HRTF set before interpolation in order to see how well that measurement could be constructed if it did not exist. In SHD-N, we do not need to delete measurements since interpolation is performed solely based on a linear combination of spatial basis functions. By comparing the reconstructed HRTF with the original measured HRTF with an error function, we can determine the quality of reconstruction as measured by our error function.

We will begin this chapter by presenting and motivating the error function and reconstruction locations. Next, we will validate SHD-N and BI4C by giving examples of reconstruction. Then, spherical harmonic matrix conditioning issues will be discussed. After that, we will show the spatial distribution of reconstruction error on the surface of the sphere. Finally, we will present and discuss the results of the comparison by showing how BI4C performs with respect to SHD-N for many N and for different measurement grids.

3.1 Reconstruction Error Function

3.1.1 Definition

Our reconstruction error is defined as

$$\epsilon = 10\log_{10}\frac{||\mathbf{H_L} - \mathbf{\hat{H}_L}||_2}{||\mathbf{H_L}||_2} \tag{3.1}$$

where $\mathbf{H_L}$ is the set of original HRTFs measured at $\mathbf{L}$ desired locations and $\mathbf{\hat{H}_L}$ is the set of interpolated HRTFs reconstructed at the $\mathbf{L}$ desired locations without using the original $\mathbf{L}$ measurements. In other words, a measured HRTF $H(\Omega_l, k)$ is deleted from the original set, then reconstructed using the remaining measurements to obtain $\hat{H}(\Omega_l, k)$. The average dB error ϵ_{avg} across directions was computed as a dot product with the Voronoi sampling weights to compensate for the non-uniform spherical distribution:

$$\epsilon_{avg} = \boldsymbol{\epsilon_q} \cdot \boldsymbol{w_q} \tag{3.2}$$

where ϵ_q is the dB reconstruction error for the q-th location and w_q is the Voronoi weight corresponding to that location.

The HRTF magnitude responses are evaluated for frequencies between 50 Hz and 20 kHz along a perceptually-informed frequency axis called the Bark scale. The Bark scale is a psychoacoustical scale proposed by Edward Zwicker in 1961 [55]. It is a frequency scale where each frequency corresponds to the center frequency of a critical band along the basilar membrane in the cochlea. Therefore, the distances between Bark frequencies agree with how we perceive differences in frequency.

3.1.2 Reconstruction Locations

For a fair comparison of reconstruction error, the interpolation should be performed at locations that theoretically correspond to the worst-case interpolation quality. We expect that reconstruction error is highest at points furthest from measurement points. For example, in equiangular grids, the reconstruction points should be located at the midpoint of the square created by the four closest grid points, as shown in Figure 3.1.

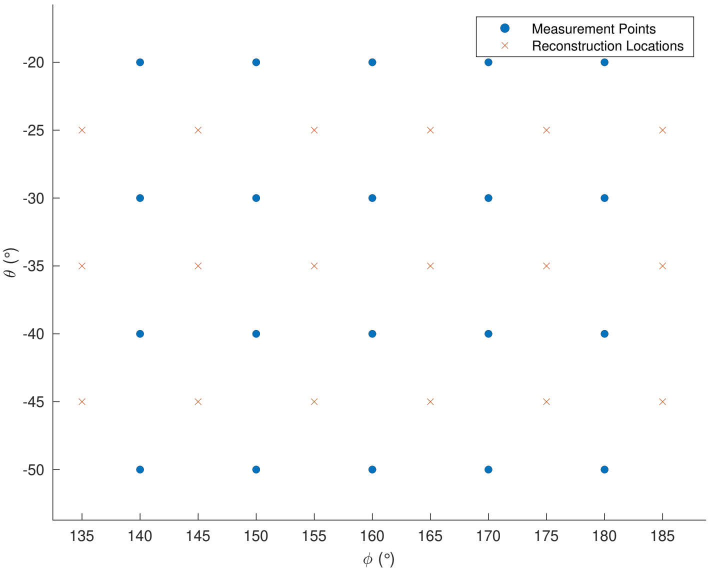

Figure 3.1: Zoomed-in example of equiangular measurement lattice and theoretically worst-case reconstruction locations. The measurement lattice has been mapped onto a Euclidean plane and uses 10° spacing in both azimuth and elevation.

In practice, however, this scheme is only applicable to the BI4C algorithm. In SHD-N, reconstruction quality is not a function of the reconstruction location on the sphere. That is, reconstruction will have similar quality at all points on the sphere. Therefore, for our analysis of reconstruction error, we will reconstruct halfway between grid points for BI4C and at grid points for SHD-N (since the reconstruction location does not matter for SHD-N).

3.2 Validation of Interpolation Techniques

In this section, we will introduce our analysis framework and provide examples of interpolated HRTFs relative to the original measurements for many SH orders and grid sizes. All reconstruction will be performed only for left ear HRTFs. Two HRTF databases will be used for their desirable measurement grids: The SADIE II Database from the Department of Electrical Engineering, University of York [16] and the ITA HRTF-database from the Institute for Hearing Technology and Acoustics at RWTH Aachen University [15].

The SADIE II Database includes HRIR data for 20 subjects (two mannequins and

18 humans) sampled at 15° elevation increments and variable azimuthal increments. Additionally, it includes measurements for various Ambisonic loudspeaker configurations: Octahedron (x3 orientations), Cube, Bi-Rectangle (x3 orientations), Icosehedron, 7-Design, 26pt Lebedev Grid, Pentakis Icosedodecahedron and 50pt Lebedev Grid. For the most densely sampled subjects (the two mannequins), azimuth is sampled at 1° increments, resulting in a total of 8802 measurements which contains a 1° × 15° mixed equiangular grid. In Section 2.3.3, we discussed how Lebedev spherical sampling schemes are preferable because they allow for decomposition of sound fields with fewer measurements relative to equiangular and mixed equiangular schemes. Therefore, in this thesis, we will be considering subsets of the 1° × 15° mixed equiangular grid as well as the 26pt and 50pt Lebedev grids.

The ITA HRTF-database includes HRIR data for 48 human subjects sampled on a 5°×5° equiangular grid. The advantage of this database is the finer resolution in elevation, allowing for BI4C on a 10° × 10° grid in order to compare reconstruction at the center of each 10° × 10° square. One disadvantage, however, is that the measurement apparatus did not permit measurements at elevations below $-65°$. Therefore, we cannot perform BI4C to compare reconstruction at these angles.

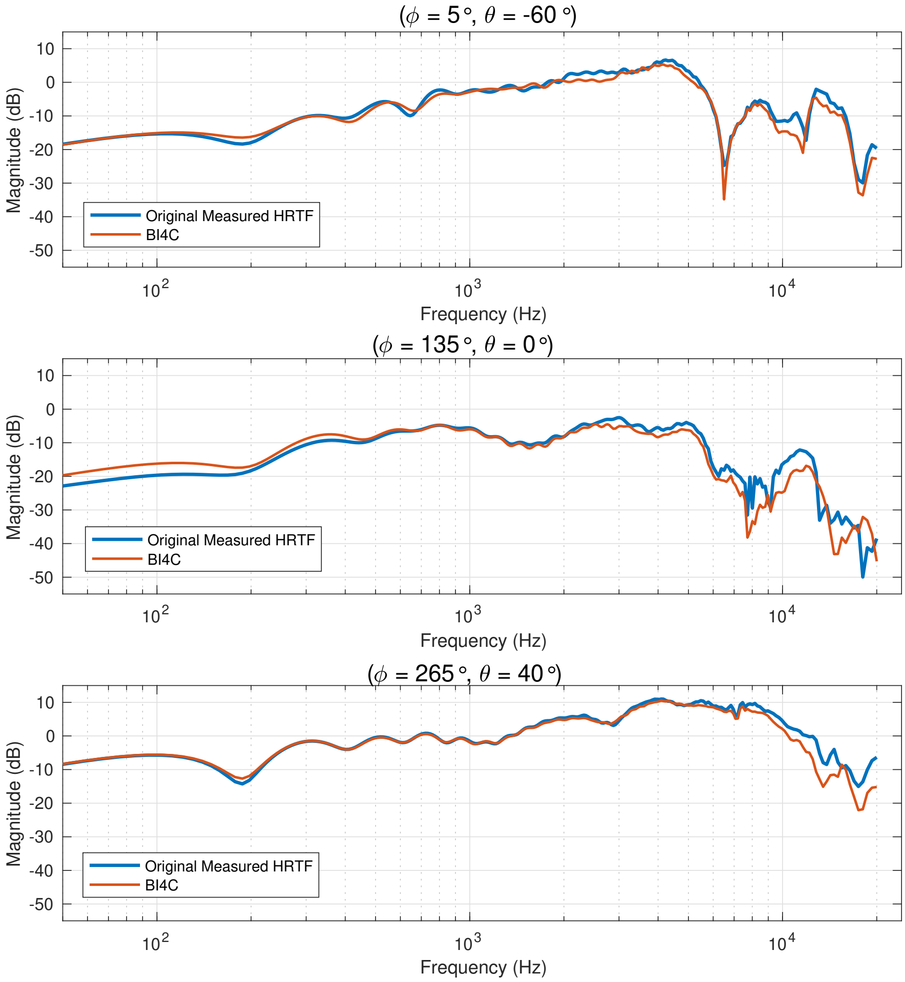

Figure 3.2: Examples of HRTFs spatially interpolated with bilinear interpolation of the four closest (BI4C) compared to the original measured HRTF for three locations on the sphere. The measurement lattice used is a 10° × 10° equiangular grid from the ITA HRTF-database [56]. Reconstruction locations are at the center of squares created by the grid points.

Figure 3.2 shows three examples of a measured HRTF and the same HRTF reconstructed via bilinear interpolation of the four closest. Although they are a small sample size, these plots indicate BI4C is highly precise for frequencies less than 3000 Hz with the potential for good reconstruction above this threshold, as seen in the bottom plot.

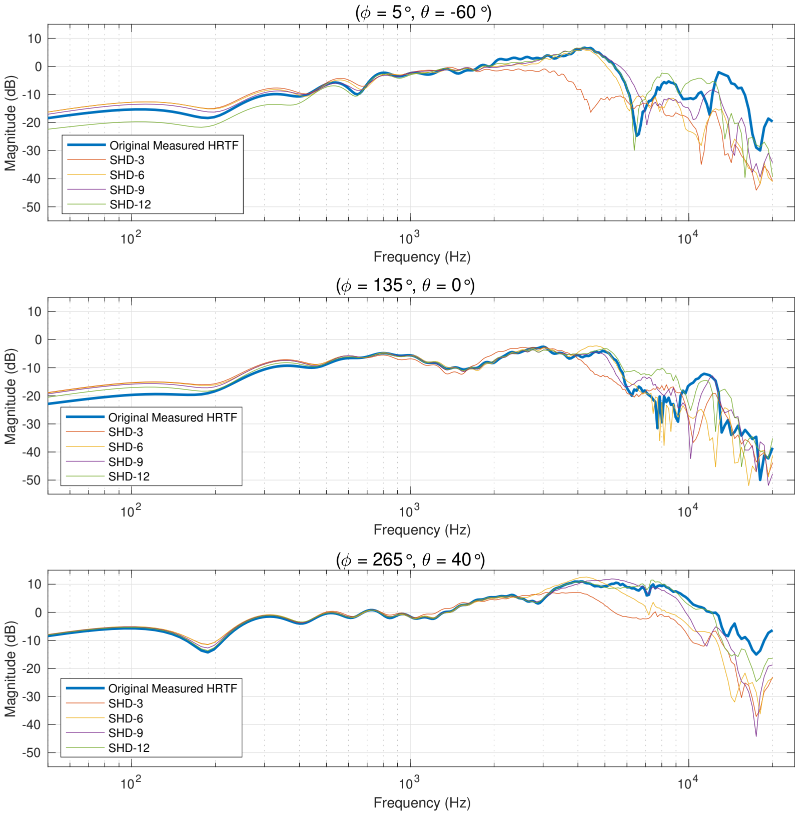

Figure 3.3: Examples of HRTFs spatially interpolated with 3rd, 6th, 9th, and 12th order SHD compared to the original measured HRTF for three locations on the sphere. The spherical harmonic basis functions were constructed with a weighted least-squares SFT of the 5° × 5° equiangular grid in the ITA HRTF-database [15].

Figure 3.3 shows three examples of the original measured HRTF compared with interpolation of that HRTF via 3rd, 6th, 9th, and 12th order spherical harmonic decomposition. The SHD-N was performed using the real spherical harmonics, a convention that has been adopted in Ambisonics [57]:

$$Y_n^m(\phi,\theta) = N_n^{|m|} P_n^{|m|} \sin(\theta) \begin{cases} \cos(|m|\phi) & \text{if } m \geq 0 \\ \sin(|m|\phi) & \text{if } m < 0 \end{cases} \tag{3.3}$$

where ϕ is the azimuth angle, θ is the elevation angle, n is the SH order, m is the SH degree, and $P_n^{|m|}$ are the associated Legendre polynomials with the Condon-Shortley phase undone. The Condon-Shortley phase is a factor $(-1)^m$ included in some quantum mechanical formulations of the spherical harmonics that inverts the relative polarity of every other SH function. In Ambisonics, we must undo it by applying it again in order to prevent distortions during rendering.

The normalization term $N_n^{|m|}$ adopted in Ambisonics is called SN3D and is computed as

$$N_n^{|m|} = \sqrt{(2-\delta_m)\frac{(n-|m|)!}{(n+|m|)!}}. \tag{3.4}$$

From these plots, we notice SHD-N struggles at reconstructing the higher frequencies, with error being lower for higher orders. At lower frequencies, BI4C and SHD-N are comparable in terms of reconstruction, with the quality of each being dependent on the spherical location.

3.3 Spherical Harmonic Matrix Conditioning Issues

In spherical harmonic decomposition of HRTFs, the spherical harmonic transformation matrix $\boldsymbol{Y_n^m}$ is applied to the HRTF measurements in order to build the basis functions $\boldsymbol{H_{nm}}$. This matrix is given as

$$\boldsymbol{Y_n^m} = \begin{bmatrix} Y_0^0(\Omega_1) & Y_{-1}^1(\Omega_1) & \dots & Y_N^M(\Omega_1) \\ Y_0^0(\Omega_2) & Y_{-1}^1(\Omega_2) & \dots & Y_N^M(\Omega_2) \\ \vdots & \vdots & \ddots & \vdots \\ Y_0^0(\Omega_Q) & Y_{-1}^1(\Omega_Q) & \dots & Y_N^M(\Omega_Q) \end{bmatrix} \tag{3.5}$$

where $\Omega_1, \Omega_2, \dots, \Omega_Q$ are the locations of the Q HRTF measurements on the sphere, specified by ordered pairs (ϕ_q, θ_q). The $\boldsymbol{Y_n^m}$ are computed as in Equation 3.3.

3.3.1 Precision Issues

To perform the Spherical Fourier Transform and retrieve the basis functions $\boldsymbol{H_{nm}}$, a weighted least-squares pseudoinverse is computed, as in Equation 2.21. A classic matrix inverse $\boldsymbol{Y}^{-1}$ only exists if $\boldsymbol{Y}$ is square and it's determinant is non-zero. Here, we have omitted the indices n and m for conciseness. Since the SH transformation matrix $\boldsymbol{Y}$ has dimensions $Q \times (N+1)^2$, we are interested in a least-squares solution to the linear system. The pseudoinverse defined as

$$\boldsymbol{Y}^{\dagger} = (\boldsymbol{Y}^H \boldsymbol{Y})^{-1} \boldsymbol{Y}^H \tag{3.6}$$

is a generalization of the classic inverse to rectangular matrices, where $\boldsymbol{Y}^H$ is the conjugate-transpose of $\boldsymbol{Y}$. Computing this pseudoinverse must require the term $\boldsymbol{Y}^H \boldsymbol{Y}$ to have a non-zero determinant (i.e. be non-singular) since a classic inverse of that term is computed.

The term $(\boldsymbol{Y}^H \boldsymbol{Y})^{-1}$ can be expanded to

$$(\boldsymbol{Y}^H \boldsymbol{Y})^{-1} = \frac{1}{\det(\boldsymbol{Y}^H \boldsymbol{Y})} \operatorname{adj}(\boldsymbol{Y}^H \boldsymbol{Y}) \tag{3.7}$$

where $\operatorname{adj}(\boldsymbol{Y}^H \boldsymbol{Y})$ is the adjugate matrix of $\boldsymbol{Y}^H \boldsymbol{Y}$. Here, we can see that the determinant of $\boldsymbol{Y}^H \boldsymbol{Y}$ must be non-zero. Moreover, increasing the SH order from $N-1$ to N results in the addition of $2N+1$ columns to $\boldsymbol{Y}$. Therefore, the term $\det(\boldsymbol{Y}^H \boldsymbol{Y})$ grows very quickly as we go to higher order. In double-precision arithmetic (e.g. the IEEE 754 standard), computers can perform computations on numbers that are within about 16 orders of magnitude with each other. As we increase SH order N, there is a critical point where the term $1/\det(\boldsymbol{Y}^H \boldsymbol{Y})$ becomes so small relative to the elements of $\operatorname{adj}(\boldsymbol{Y}^H \boldsymbol{Y})$ that the computer can no longer distribute it to those elements because it gets lost in precision noise. At this point, the computer rounds it to zero, the matrix becomes singular, and no inverse exists. It is therefore necessary to keep the SH order low enough such that the pseudoinverse does not involve computations with numbers more than 16 orders of magnitude apart from each other.

When computing a pseudoinverse in MATLAB, MATLAB will check for this issue by calculating the condition number of the matrix involved. The condition number κ of a matrix $\boldsymbol{Y}$ can be computed as the ratio of the maximum singular value to the minimum singular value of the matrix $\boldsymbol{Y}$:

$$\kappa(\boldsymbol{Y}) = \frac{\sigma_{max}(\boldsymbol{Y})}{\sigma_{min}(\boldsymbol{Y})} \tag{3.8}$$

where $\sigma_{max}(\boldsymbol{Y})$ and $\sigma_{min}(\boldsymbol{Y})$ represent the maximum and minimum singular values of the matrix $\boldsymbol{Y}$, respectively. Using the condition number as an indicator of matrix singularity is more robust than calculating a determinant because determinants of a non-singular matrix can be arbitrarily close to zero.

3.3.2 Measurement Error Robustness

Another property of condition numbers is that they can describe the sensitivity of the output of a matrix to perturbations of the input. In other words, it will represent the factor by which noise at the input will be amplified at the output. Perturbations of the spherical harmonic transformation matrix exist as small errors in the positioning of loudspeakers and microphones during the HRTF measurement stage. Because the spherical harmonics are essentially solutions to the wave equation in spherical coordinates, the expectation is that measurement speakers are placed exactly at the locations specified so that the wave equation can be utilized. However, if the condition number is low, the SH transformation matrix will be robust to these errors to some extent. Reddy and Hegde gave an optimization-based approach for minimizing the SH transformation matrix condition number by selecting rectangular sub-matrices of a dense grid with low condition numbers [58]. In this work, we are constrained to measurements available in publicly released HRTF datasets, which are typically measured on grids that are not optimized to have low condition numbers when $\boldsymbol{Y}$ is computed.

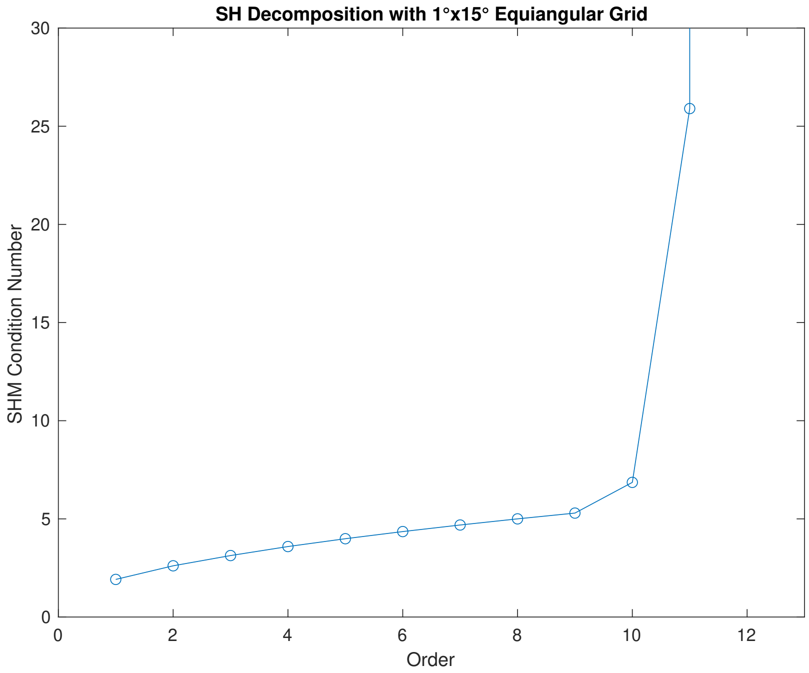

Figure 3.4: Plot of condition number $\kappa(N)$ for orders $N = 1, 2, \ldots, 12$. The order 12 condition number $\kappa(12) \approx 1.26 \times 10^{16}$. The point where the condition number disproportionately increases is a strong indicator of singularity.

3.4 Spherical Distribution of Reconstruction Quality

In this section, we will explore how BI4C and SHD-N differ with respect to the spatial distribution of reconstruction quality according to our error function. A key difference between the two interpolation techniques is their consistency of reconstruction quality with respect to locations on the sphere. This discrepancy is seen by comparing Figures 3.5 and 3.6.

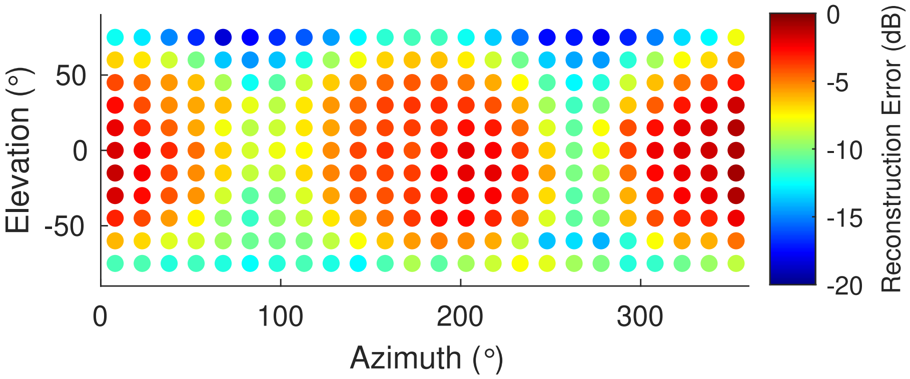

Figure 3.5: Spatial distribution of BI4C reconstruction error performed on a 15° × 15° equiangular grid, reconstructing halfway between grid points in azimuth. Weighted average reconstruction error: -6.24 dB. Weighted standard deviation: 3.59 dB.

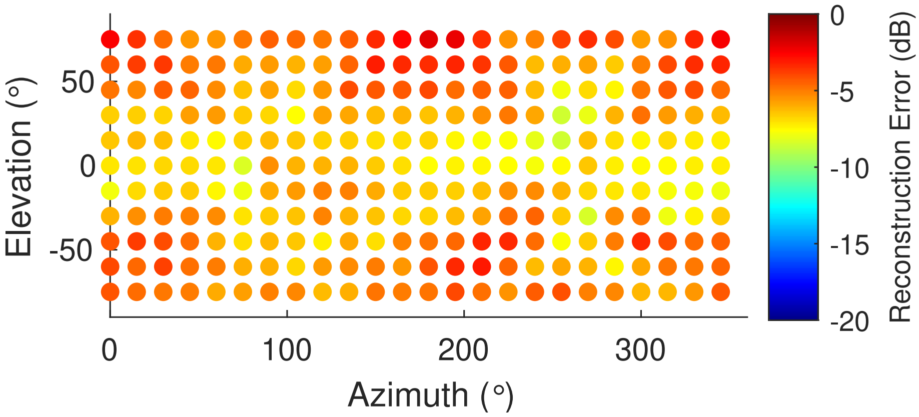

Figure 3.6: Spatial distribution of SHD-10 reconstruction error using basis functions computed with a weighted least-squares SFT of a 15° × 15° equiangular grid. Reconstruction performed at grid points. Weighted average reconstruction error: -5.91 dB. Weighted standard deviation: 1.35 dB.

In Figure 3.5, error is shown for BI4C performed on a 15° × 15° equiangular grid, reconstructing halfway between azimuth points. The reconstruction error is computed for each reconstruction location and plotted according to the color axis. Here, the sphere has been mapped onto a Euclidean plane for conciseness. The weighted average reconstruction error for this set is -5.05 dB. However, we notice that reconstruction is poor on the median plane (front/back and up/down) and good on the lateral plane (left/right and up/down). The weighted standard deviation of these errors is 3.59 dB.

In Figure 3.6, reconstruction via 10th order SHD is performed at the grid points with error shown on the color axis. The basis functions $\boldsymbol{H_{nm}}$ were computed with a weighted least-squares spherical Fourier transform of HRTFs measured on a 15° × 15° equiangular grid. The scaling of the color axis is identical to Figure 3.5. The weighted average reconstruction error for these points is -5.91 dB. The error in this case is more uniform with respect to spherical distribution, with a weighted standard deviation of 1.35 dB.

The two plots have similar average errors but differ in terms of the standard deviation. Therefore, if one values spatial consistency of reconstruction when choosing an interpolation technique, SHD might be preferred. It should be noted, however, that computing 10th order SHs to achieve as good reconstruction as BI4C for a given grid will result in higher computational costs relative to BI4C, as we will see in the next section. Figure 3.7 shows error resulting from BI4C on a 10° × 10° equiangular grid, reconstructing at the center of the squares specified by the grid points (as in Figure 3.1). As discussed earlier, we expect this is where reconstruction is worst. Since the ITA HRTF-database was used, elevations below $-65°$ were not available. These points give a weighted average reconstruction error of -5.69 dB with a weighted standard deviation of 1.74 dB. Note: the color axis scaling has changed.

In contrast, Figure 3.8 is reconstruction via 7th order SHD, using the same 10° × 10° equiangular grid for both the analysis points and reconstruction points. This set has a weighted average reconstruction error of -5.76 dB with a weighted standard deviation of 1.53 dB.

Figures 3.7 and 3.8 again have similar average errors but this time with similar standard deviations. The difference in standard deviation between Figures 3.6 and 3.8 can be explained by the distinct reconstruction locations. By introducing interpolation along the elevation axis in conjunction with the azimuth axis, the output is prone to more error. This fact combined with higher density of points near the poles permitting smaller HRTF spatial variations can account for the good reconstruction seen in Figure 3.8.

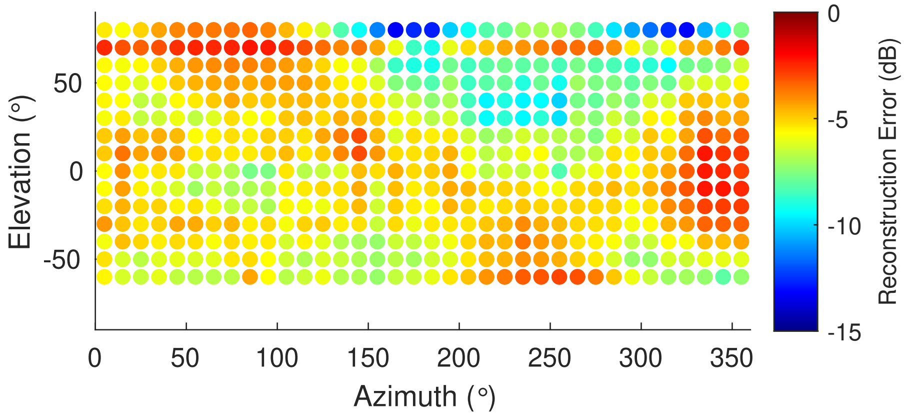

Figure 3.7: Spatial distribution of BI4C reconstruction error performed on a 10° × 10° equiangular grid, reconstructing at center of square formed by grid points. Weighted average reconstruction error: -5.69 dB. Weighted standard deviation: 1.74 dB.

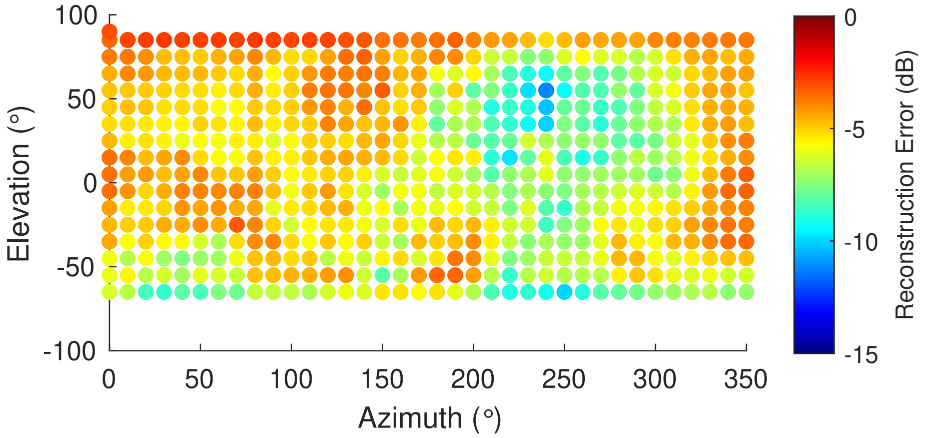

Figure 3.8: Spatial distribution of SHD-7 reconstruction error using basis functions computed with a weighted least-squares SFT of a 10° × 10° equiangular grid. Reconstruction performed at grid points. Weighted average reconstruction error: -5.76 dB. Weighted standard deviation: 1.53 dB.

Figure 3.9 shows reconstruction via SHD-5 on a 50pt Lebedev grid. Here, the weighted average reconstruction error is -4.51 dB with a weighted standard deviation of 1.39 dB. This plot indicates the ability to achieve as good reconstruction as a SHD-7 of a 10° × 10° grid but with only 50 measured HRTFs, as opposed to 630 (the number of grid points on a 10° × 10° grid).

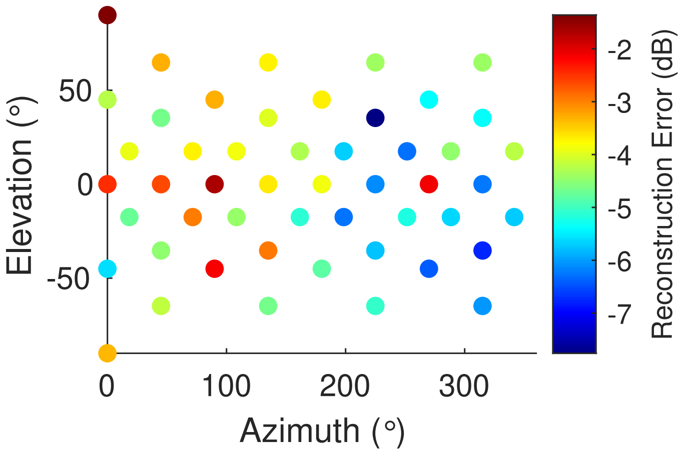

Figure 3.9: Spatial distribution of SHD-5 reconstruction error using basis functions computed with a weighted least-squares SFT of a 50pt Lebedev grid. Reconstruction performed at grid points. Weighted average reconstruction error: -4.51 dB. Weighted standard deviation: 1.39 dB.

3.5 Comparison of SHD-N and BI4C Reconstruction Error

This section presents results showing the relative quality of interpolation of the two techniques studied in this thesis. We are interested in showing how various truncation orders of spherical harmonic decomposition fare against BI4C performed on various grid sizes. By doing this, we are able to build a more complete picture of the performance tradeoffs of these two HRTF interpolation methods.

Our analysis was conducted for both equiangular and mixed equiangular grids. For all cases, the SH basis functions $\boldsymbol{H}_{nm}^{L,R}$ should be built on the most dense grid available to achieve the best reconstruction since there is no additional cost related to more measurements once encoded into SHs. Including additional measurements will add more resolution to the spatial variations of the HRTF, as well as allow for higher SH orders by keeping the condition number lower (as demonstrated in Section 3.3). In the context of this work, the most dense mixed equiangular grid is the 1° × 15° grid available in the SADIE II database [16] while the most dense equiangular grid is the 5° × 5° grid available in the ITA HRTF-database [15].

For BI4C, there are additional tradeoffs to be taken into account when arguing for more dense measurement grids. All these measurements ideally are stored in cache for quick access and denser grids could prevent this from happening. For this reason, we have shown relative error of reconstruction for a few grid densities. These grids along with their dB reconstruction error are found on the horizontal lines in Figures 3.10 and 3.11. Figure 3.10 gives relative errors of reconstruction for the mixed equiangular case while 3.11 is for the equiangular case.

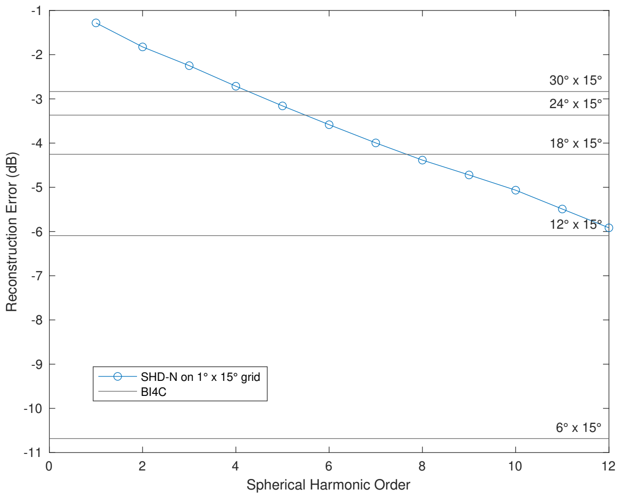

Figure 3.10: Relative reconstruction errors for BI4C performed with five mixed equiangular grids compared to 12 orders of SHD-N performed with a 1° × 15° mixed equiangular grid.

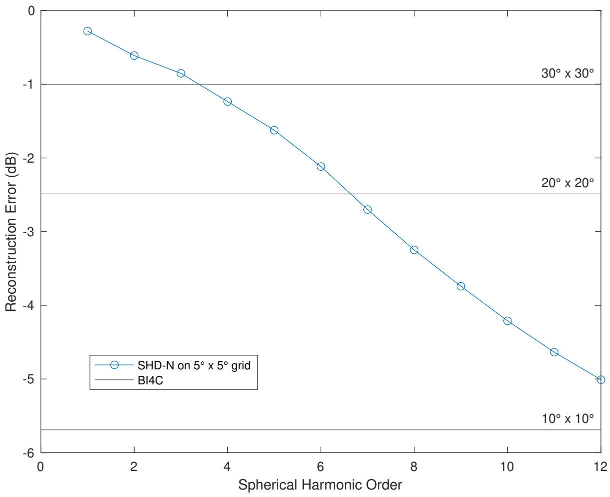

Figure 3.11: Relative reconstruction errors for BI4C performed with three equiangular grids compared to 12 orders of SHD-N performed with a 5° × 5° equiangular grid.

3.5.1 Discussion

Inspecting Figures 3.10 and 3.11, we notice that interpolation improves for both denser grids of BI4C and higher orders of SHD-N. This is expected, as in both cases there is more information available during the reconstruction, leading to better quality. It should be noted, though, that all data in these plots are computed as a weighted average reconstruction error over all grid points, where the weights are the Voronoi tessellation weights. If spherical consistency of reconstruction is desired, the results of the previous section show that SHD-N provides more consistent reconstruction with respect to location on the sphere. Future work should include subject-based listening tests to validate the importance of spherical reconstruction consistency. It is difficult to draw generalized conclusions about the superiority of either algorithm based on these plots, since it is entirely based upon the truncation order N and grid density chosen. The plots do indicate, however, that BI4C gives lower reconstruction error if only sparse measurement grids are available. For example, one would need to use 7th order SHD *and* have a 5° × 5° equiangular measurement grid to achieve as good reconstruction as BI4C with the much coarser 20° × 20° grid.

Chapter 4

Computational Cost Analysis

The second metric used in this thesis for comparing BI4C and SHD-N is the computational complexity of each algorithm. This research is motivated by an effort to understand the tradeoffs of each algorithm by studying quantitative benchmarks. While a purely theoretical analysis of each interpolation technique is possible, it would be difficult to make objective statements about their actual performance on a computer.

In this chapter, we will discuss the process of modeling computational cost via benchmarking, describe our test systems, explain and motivate the configurations used to measure computational cost, give implementation details, and show our results.

4.1 Modeling Cost with Benchmarks

Improvements in hardware design over the years have necessitated an empirical approach to studying computational cost. In the past, algorithms were often optimized according to the performance of the available hardware. For example, multiplications were executed much slower than additions, so one might try to build algorithms that minimize the number of multiplications. These days, advancements in computer architecture such as cache hierarchies, pipelining, and compiler optimizations have made theoretical cost models much more difficult. Still, as we will see, understanding the basic costs of the signal processing algorithms can help to understand benchmarks.

This thesis will attempt efficient C++ implementations of both BI4C and SHD-N, measuring the time it takes to execute them for different initial conditions (block lengths, number of sources, numerical precision). In this chapter, it is assumed that we are operating in an auralization framework that is providing some number of time-domain source signals with associated angle metadata corresponding to direct field or reflection wavefronts as input at the beginning of every audio cycle. An audio cycle, here, refers to a single buffer in a real-time audio thread. At the end of each cycle, the framework requests a stereo time-domain buffer corresponding to the aggregated sources auralized at their respective angles. Auralized, in this context, simply means rendering some acoustic path via convolution with an HRIR.

Both BI4C and SHD-N are characterized by a high number of arithmetic operations and consistent memory access patterns. Their main difference is how the number of convolutions scales with the number of sound sources to be rendered. The advantage of SHD-N is that we can render an arbitrary number of sources with $(N + 1)^2$ convolutions. Conversely, in BI4C, we must convolve per source. Since convolution is performed via multiplication in the frequency-domain, a major source of cost is the cost of the FFT. State-of-the-art FFT algorithms such as the modified split-radix technique given by Johnson and Frigo [59] can compute K-point FFTs in $\mathcal{O}(K \log K)$ in the average case. This complexity is substantial compared to the rest of the operations and therefore must be taken into account when considering these two interpolation techniques.

There are of course an infinite number of feasible parameter configurations we can choose from to study computational cost. The performance of each will be a function of a large number of factors such as hardware platform, algorithm implementation, FFT implementation, architecture-specific compiler optimizations, SIMD (single instruction, multiple data) vectorization instruction sets, arithmetic precision, and much more. A comprehensive overview of each of these would be nearly impossible. In this thesis, we are interested in the average use case: efficient, yet straightforward implementations on general-purpose processors. Section 4.4 will explore the test system more in-depth.

4.2 Pre-Processing vs. Real-Time Components

As discussed in Chapter 2, continuous HRTF representations are highly desirable due to their aid in simulating arbitrary source angles for virtual acoustic rendering. Continuous, in this sense, can be regarded in a few ways. One way would be to define it perceptually, i.e. the HRTF measurement grid is dense enough that our brain cannot discern distances between sources rendered with adjacent HRTF measurements. This would require the HRTFs to be measured or interpolated on a grid with resolution at least as fine as the *minimum audible angle* (the smallest change in source angle that we can perceive). This angle is direction dependent but can be as low as 1° [1]. An equiangular grid with 1° spacing in azimuth and elevation would require a total of 64,800 HRIRs. These could be measured or interpolated with any precise technique (e.g. very high order SHD-N or BI4C of minimum-phase HRIRs with customized ITDs added as delay lines) since the interpolation would happen offline. If each HRIR is 256 samples long, however, this table would be over 132 MB in double-precision format, well over the size of most caches. Looking up data from a table this large would be a slow process for most architectures and therefore likely not viable for real-time applications.

On the other hand, computing spherical harmonics is a computationally expensive process due to the various trigonometric and arithmetic functions involved, making it also difficult to do in real-time. Since N-th order spherical harmonic decomposition uses only $(N+1)^2$ SH coefficients to represent an arbitrary sound field, a dense lookup table of these coefficients could be stored in cache and the coefficients corresponding to the angle closest to the query angle could be retrieved in real-time to avoid the cost associated with computing them. Therefore, we are interested in algorithms that avoid costly operations while also keeping lookup tables small so that cache can be utilized. Of course, the trade-off between lookup table size and processing power will be unique to the given hardware platform and these observations may not be generalizable. Still, in order to provide quantitative benchmarks, we have chosen to perform SHD-N by retrieving the $(N+1)^2$ coefficients for a given source angle with a lookup table computed on a dense grid. This grid will be described in the next section.

4.3 Implementation Details

4.3.1 SHD-N

Fibonacci Lattice Lookup Table

As we have discussed, looking up SH coefficients in a table is more cost-effective than computing them in real-time. The task, then, is to select a spherical sampling scheme on which to build the SH coefficient lookup table that is suited for real-time applications. In this sense, suited for real-time means keeping this table small in order to keep it in cache while also attempting a perceptually-continuous HRTF representation. Moreover, lookups should happen in $\mathcal{O}(1)$ time. That is, we want to quantize the incoming source query angle to the lookup table grid points by quickly locating the nearest grid point and returning the set of coefficients corresponding to that point.

One possibility for this would be an equiangular grid. An advantage of equiangular grids is they have a regular basis, so lookups are a matter of simple rounding and modulo operations. Equiangular grids suffer from redundancies near the poles, however, meaning the table would be unnecessarily large. Another possibility is the M-point Lebedev grid. These sample the sphere more uniformly but it would be difficult to locate the nearest grid point in real-time as the table would not have a regular basis. Moreover, the table sizes could not be arbitrary since Lebedev grids only exist for a discrete set of grid sizes.

Instead, we have chosen to build the lookup table on a Fibonacci lattice. A Fibonacci lattice is yet another technique for sampling the sphere commonly used in mathematical geosciences [60]. This scheme is perhaps the most uniform sampling scheme since the area represented by each grid point is almost identical. It is characterized by a spiral tightly wound on the surface of the sphere, with each point fitted into the largest gap between the previous points. The packing efficiency of grid points is optimized by using the golden ratio to determine the spacing between points. More precisely, let S be any positive integer and let the integer i range from $-S$ to S. The spherical coordinates of the i-th point in radians are then given by

$$\phi = 2\pi i \Phi^{-1} \tag{4.1}$$

$$\theta = \sin^{-1}\left(\frac{2i}{2S+1}\right) \tag{4.2}$$

where $\Phi = (1+\sqrt{5})/2$ is the golden ratio. The total number of grid points is $2S+1$. Figure 4.1 shows an example of an equiangular grid and Fibonacci lattice of similar sizes.

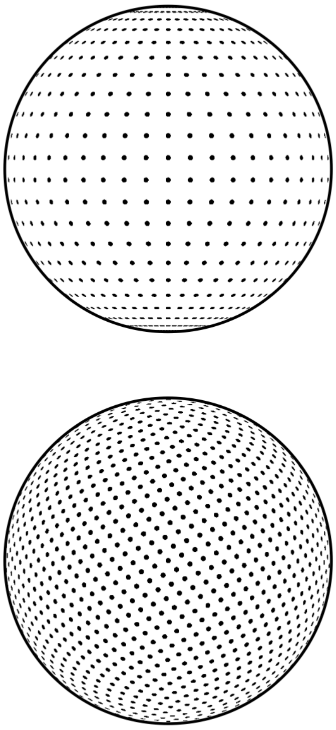

Figure 4.1: 1014 point equiangular lattice (*top*) and 1001 point Fibonacci lattice (*bottom*). The Fibonacci grid does not suffer from higher density near the poles. From [60].

Although the Fibonacci lattice does have a closed form expression, finding the nearest grid point given a query angle is still a difficult task which cannot be solved in $\mathcal{O}(1)$ time. Therefore, for this thesis, we have resampled a Fibonacci lattice onto a special mixed equiangular grid which promotes both uniformity with respect to the sphere as well as fast quantization and lookups. This resampling was performed by first dividing the elevation axis into Q_{el} bins of equal size such that the i-th bin contains elevations in the interval

$$\left[\frac{-\pi}{2} + \frac{i\pi}{Q_{el}}, \frac{-\pi}{2} + \frac{(i+1)\pi}{Q_{el}}\right) \tag{4.3}$$

where $i = 0, 1, \ldots, Q_{el} - 1$. Next, each Fibonacci lattice point was assigned to a bin if the elevation angle of that point falls in the interval specified by that bin. In this way, we can build a histogram of the distribution of the number of azimuths along the elevation axis, as shown in Figure 4.2. This sampling is similar to the grid chosen by the MIT KEMAR HRTF database [45]. Let each bin be identified by it's center angle, denoted as θ_i. Let the elevation-dependent number of azimuths be denoted by $Q_{az}(\theta_i)$. Finally, a grid is constructed by equally spacing $Q_{az}(\theta_i)$ azimuths at elevation θ_i. An example of this is shown in Figure 4.3.

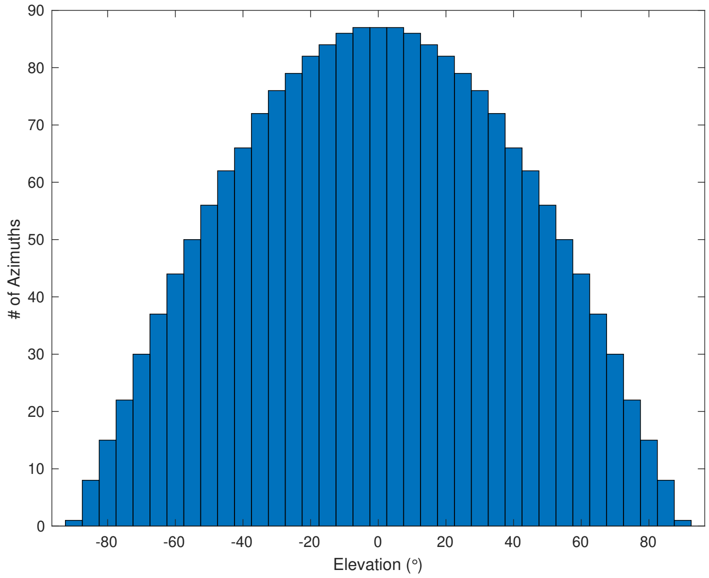

Figure 4.2: Distribution of number of azimuths for 37 equally-spaced elevation bins on a 2003 point Fibonacci lattice.

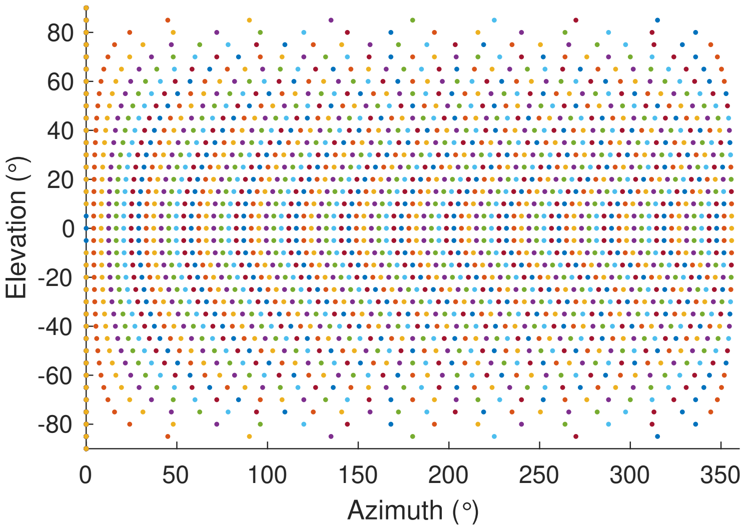

Figure 4.3: Example of a 2003 point equiangular grid with elevation-dependent azimuthal resolution sampled from a Fibonacci lattice.

Algorithm

This section describes how interpolation via SHD-N was incorporated into the framework for fast convolution with partitioned input signals (e.g. block-based audio processing). As discussed in Section 2.5, real-time FIR filtering of audio on general purpose processors is constrained to the block-based approach, where the input is partitioned into length-B blocks on which signal processing can be performed. Moreover, efficient FIR filtering is often implemented with FFT-based convolution. To incorporate FFT-based convolution with partitioned input signals, the overlap-save method can be applied. This method involves convolution using overlapped audio blocks in order to prevent artefacts at the edges of the output.

In Section 2.5.3, we discussed a method for crossfading consecutive audio buffers in order to create the illusion of sources moving throughout the virtual space in a continuous fashion. This method was based on frequency-domain operators acting on the spectra of adjacent output buffers [54]. A block diagram implementing FFT-based convolution with the overlap-save method and frequency-domain crossfading is shown in Figure 2.13. Algorithm 1 shows how SHD-N is incorporated into this process. Let K denote the size of all FFTs, B denote the length of each audio block, and numSources denote the total number of sources to be rendered. In general, terms with the subscript 0 refer to the previous audio cycle and terms with the subscript 1 refer to the current audio cycle. We need to store information about the HRTF interpolated from the previous audio cycle so that we can convolve the current source buffers with both the previous and current HRTFs in order to crossfade the two.

The $K \times (N+1)^2$ matrix containing the SH-encoded HRTFs (i.e. the frequency-domain basis functions $\boldsymbol{H_{nm}^{L,R}}$) are computed offline and stored in a table. Additionally, the SH transformation matrix lookup table $\boldsymbol{Y_{nm}}$ is computed offline according to the Fibonacci lattice lookup table described previously. For each source, we retrieve the angle at which this source is to be localized and find the nearest angle on the Fibonacci lookup table grid. This quantization is performed by rounding the source elevation to the nearest Fibonacci elevation, looking up the density of azimuths at that elevation, and rounding the source azimuth to the nearest azimuth. Next, for each of the $(N+1)^2$ Ambisonic channels (i.e. SH functions), we retrieve the SH coefficient c_0 corresponding to the quantized source angle from the previous audio cycle and multiply it by each sample of the source buffer. The resulting buffer, x_0, corresponds to the current source buffer encoded into the current Ambisonic channel. As we loop over the sources, we can sum (i.e. accumulate) each encoded source buffer into the same per-channel buffer, since SHD-N can store sound fields as functions independent of the number of sources. Then, we lookup the SH coefficient c_1 corresponding to the quantized source angle in the Fibonacci lookup table $\boldsymbol{Y_{nm}}$ and multiply it by each sample of the source buffer, again accumulating over the sources into a single buffer x_1. We need to be sure to update the saved coefficient c_0 so that it can be retrieved during the next audio cycle.

Now, we have x_0 and x_1, the accumulated sound sources encoded into $(N+1)^2$ Ambisonic channels localized at angles corresponding to the previous and current audio cycles, respectively. In order to convolve these with HRTFs, we must apply the overlap-save method. This is performed by constructing per-channel sliding windows y_0 and y_1 of the encoded inputs x_0 and x_1. These windows must hold the most recent K samples of each input. Next, K-point real-to-complex FFTs of both y_0 and y_1 are computed, resulting in Y_0 and Y_1, respectively. Finally, convolution is performed via per-channel element-wise complex multiplication of both Y_0 and Y_1 with the frequency-domain SH-encoded left and right HRTFs $H^{L,R}_{channel}$. These buffers, corresponding to the products of Y_0 and Y_1 with $H^{L,R}_{channel}$, are accumulated over the channels into the buffers acc$Y_0^{L,R}$ and acc$Y_1^{L,R}$, respectively.

acc$Y_0^{L,R}$, then, is the stereo frequency-domain buffer corresponding to each sound source auralized at angles specified by the previous audio cycle. acc$Y_1^{L,R}$ is the stereo frequency-domain buffer corresponding to each sound source auralized at angles specified by the current audio cycle. The last step is to crossfade these two buffers, in order to create the illusion of sounds moving continuously through the space. Crossfading of acc$Y_0^{L,R}$ and acc$Y_1^{L,R}$ is performed via the method described in [54], where shifted versions of each buffer are added such that sinusoidal fade-out and fade-in amplitude envelopes are applied once an inverse FFT is computed. This crossfading is described in Section 2.5.3 and is given by

$$\begin{aligned}\text{out}^{L,R} = K[&\text{acc}Y_0 \langle k \rangle_K + \text{acc}Y_1 \langle k \rangle_K + \frac{1}{2}\{\text{acc}Y_1 \langle k+P \rangle_K - \text{acc}Y_0 \langle k+P \rangle_K \\ &+ \text{acc}Y_1 \langle k-P \rangle_K - \text{acc}Y_0 \langle k-P \rangle_K\}] \qquad (4.4)\end{aligned}$$

where $\langle \cdot \rangle_K$ denotes the K-periodic continuation of each spectra and P is the integer multiplier that relates twice the block length B to the transform size K, i.e.:

$$P = \frac{K}{2B} \in \mathbb{N}. \tag{4.5}$$

Finally, a K-point complex-to-real inverse FFT of out^{L,R} is performed and the first $K - B$ samples are discarded since they are time-aliased. The remaining length-B stereo buffer is transmitted to the audio callback for output to headphones.

Algorithm 1 SHD-N using FFT-based overlap-save convolution with frequency-domain crossfading

```
for source= $1, 2, \dots,$numSources do
    quantize source angle to Fibonacci lookup table grid
    for channel=$1, 2, \dots, (N+1)^2$ do
        $c_0 \leftarrow$ get SH coefficient corresponding to source angle from previous audio cycle
        $x_0 \leftarrow$ multiply $c_0$ by each sample of the source buffer, accumulate over sources
        $c_1 \leftarrow$ lookup SH coefficient in $\boldsymbol{Y_{nm}}$ table with quantized source angle
        $x_1 \leftarrow$ multiply $c_1$ to each sample of the source buffer, accumulate over sources
        $c_0 \leftarrow c_1$, update coefficient
    end for
end for
for channel=$1, 2, \dots, (N+1)^2$ do
    $y_0 \leftarrow$ append $x_0$ to $y_0$ and slide window to hold last $K$ samples
    $Y_0 \leftarrow K$-point R2C FFT of $y_0$
    acc$Y_0^{L,R} \leftarrow$ bin-by-bin complex multiply $Y_0$ with $H_{channel}^{L,R}$, accumulate over channels
    $y_1 \leftarrow$ append $x_1$ to $y_1$ and slide window to hold last $K$ samples
    $Y_1 \leftarrow K$-point R2C FFT of $y_1$
    acc$Y_1^{L,R} \leftarrow$ bin-by-bin complex multiply $Y_1$ with $H_{channel}^{L,R}$, accumulate over channels
end for
out$^{L,R} \leftarrow$ crossfade acc$Y_0^{L,R}$ and acc$Y_1^{L,R}$ via Wefers, Vorlaender 2014 method [54]
out$^{L,R} \leftarrow K$-point C2R IFFT of out$^{L,R}$, only save last $B$ samples
```

4.3.2 BI4C

Since bilinear interpolation of the four closest (BI4C) involves a simple weighted averaging of the nearest HRTFs, it can be incorporated into block-based audio processing more easily. For this thesis, we chose to benchmark BI4C by interpolation of a 30° × 15° mixed equiangular grid of HRTF measurements, denoted as $\boldsymbol{H^{L,R}}$. Offline steps include loading these Q frequency responses into a $Q \times K$ table. The online process of performing both

BI4C and convolution with source signals is given in Algorithm 2. Here, we are also interested in crossfading consecutive audio cycles to create the illusion of a continuously moving source.

For each source, the first step is to determine the four nearest measured HRTFs stored in $\boldsymbol{H^{L,R}}$. This is done by simply rounding the query angle in four directions (up and down in both azimuth and elevation) such that the four resulting angles correspond to measurements in $\boldsymbol{H^{L,R}}$. The next step is to determine the relative position of the query angle in the rectangle specified by the four closest HRTFs. An efficient way to do this is to choose a coordinate system such that the four corners of the rectangle correspond to the Euclidean points $(0,0),(0,1),(1,0),(1,1)$ and locate the query angle in this unit square. If the relative position of the query angle in the unit square is given as $(q_x,q_y)\in[0,1]^2$, the scalar interpolation weights can be given as

$$w_{0,0}=(1-q_x)(1-q_y) \tag{4.6}$$

$$w_{0,1}=(1-q_x)q_y \tag{4.7}$$

$$w_{1,0}=q_x(1-q_y) \tag{4.8}$$

$$w_{1,1}=q_xq_y \tag{4.9}$$

and the interpolation is performed as

$$H_1^{L,R}=w_{0,0}H_{0,0}^{L,R}+w_{0,1}H_{0,1}^{L,R}+w_{1,0}H_{1,0}^{L,R}+w_{1,1}H_{1,1}^{L,R}, \tag{4.10}$$

where $H_{0,0}^{L,R}, H_{0,1}^{L,R}, H_{1,0}^{L,R}$ and $H_{1,1}^{L,R}$ are the stereo length-K four closest HRTFs located at the points given by the subscripts after the unit square coordinate system has been applied.

Once the interpolated frequency-domain HRTF is calculated, the overlap-save method with frequency-domain crossfading must be applied. In order to crossfade the current source auralized at angles specified by consecutive audio cycles, the interpolated HRTF from the previous audio cycle $H_0^{L,R}$ is retrieved. Next, the length-B source buffer is appended to a per-source stream of source samples x, which is windowed to hold the most recent K samples. A K-point real-to-complex FFT of x is performed, resulting in X. To convolve the overlapped FFTed source buffer X with the interpolated HRTFs $H_0^{L,R}$ and $H_1^{L,R}$, element-wise complex multiplication of X with both $H_0^{L,R}$ and $H_1^{L,R}$ is performed, resulting in $Y_0^{L,R}$ and $Y_1^{L,R}$, respectively. $H_1^{L,R}$ is then stored for use during the next audio cycle. The process of crossfading $Y_0^{L,R}$ and $Y_1^{L,R}$ is identical to the crossfading given in Equation 4.4, where accY_0 and accY_1 have been replaced by $Y_0^{L,R}$ and $Y_1^{L,R}$, respectively. As in Algorithm 1, a K-point complex-to-real inverse FFT of the crossfaded frequency-domain stereo output buffer out^{L,R} is computed and the last B samples are sent to the audio callback for binaural output.

Algorithm 2 BI4C using FFT-based overlap-save convolution with frequency-domain crossfading

```
for source= $1, 2, \ldots,$numSources do
    determine four nearest HRTFs by rounding query angle to four closest grid points
    determine relative position of query angle within this $30° \times 15°$ rectangle
    compute the four interpolation weights based on this relative position
    $H_1^{L,R} \leftarrow$ average the four HRTFs using their corresponding weights, as in Equation 4.32
    $H_0^{L,R} \leftarrow$ retrieve interpolated HRTF from previous audio cycle
    $x \leftarrow$ append source buffer to $x$ and slide window to hold last $K$ samples
    $X \leftarrow$ $K$-point R2C FFT of $x$
    $Y_0^{L,R} \leftarrow$ bin-by-bin complex multiply $H_0^{L,R}$ by $X$, accumulate over sources
    $Y_1^{L,R} \leftarrow$ bin-by-bin complex multiply $H_1^{L,R}$ by $X$, accumulate over sources
    $H_0^{L,R} \leftarrow H_1^{L,R}$, update stored HRTF
end for
out$^{L,R} \leftarrow$ crossfade $Y_0^{L,R}$ and $Y_1^{L,R}$ via Wefers, Vorlaender 2014 method [54]
out$^{L,R} \leftarrow$ $K$-point C2R IFFT of out$^{L,R}$, only save last $B$ samples
```

4.4 Test System

4.4.1 Hardware

The test system is a 2020 13-inch MacBook Pro with an Apple M1 chip. The M1 chip has an 8-core CPU with 4 performance cores and 4 efficiency cores with a maximum CPU clock rate of 3.2 GHz. The system is equipped with 8 GB of RAM. Benchmark data for an additional hardware platform, with an Intel i5 core, is given in Appendix A.

4.4.2 Software

The operating system is macOS Big Sur Version 11.1. The code is written in C++ and built using Xcode 12.5.1. Release mode binaries are generated with the flag -*Ofast*, which corresponds to fastest, aggressive optimizations. Standard C++ memory buffers are used as data structures where necessary. FFTs are computed using FFTW version 3.3.9. This library is preferred because it is well-documented [61], among the fastest, and the source code is available to the public.

4.5 Measurement Procedure

Both Algorithm 1 and Algorithm 2 were implemented in C++ on the test system described. A common approach to benchmarking is to measure the time it takes for the algorithm to complete for multiple, consecutive times in a loop. The runtime of the algorithm is then computed by dividing the cumulative runtime by the number of loop iterations. Usually, a fixed number of measurements at the beginning of each loop is discarded, as these can be thought of as the system "warming up" to the algorithm. Let N_{warm} denote the number of iterations allotted to let the system warm up to the algorithm and let N_{per} denote the number of iterations used to measure the performance of the algorithm. Here, iterations correspond with audio cycles if this was a real-time auralization engine. The benchmarking procedure used in this thesis is shown in Algorithm 3. All clock times were sampled using the `chrono::high_resolution_clock` function which is part of the C++ standard library.

Algorithm 3 Measurement procedure

```
for $i = 1, \ldots, N_{warm}$ do
    runAlgorithm();
end for
$t_1$ = getTime();
for $i = 1, \ldots, N_{per}$ do
    runAlgorithm();
end for
$t_2$ = getTime();
$t = (t_2 - t_1)/N_{per}$
```

Each algorithm was measured for three different block lengths and nine different numbers of sources:

$$B = 256, 512, 1024, \tag{4.11}$$

$$\text{numSources} = 1, 2, 5, 10, 20, 50, 100, 200, 500. \tag{4.12}$$

Additionally, SHD-N was measured up to order 12:

$$N = 1, 2, 3, \ldots, 12. \tag{4.13}$$

The number of performance iterations N_{per} is based on 10 seconds of block-based audio at a sample rate of 48 kHz:

$$N_{per} = \frac{10\text{s} \times 48\text{kHz}}{B} \tag{4.14}$$

where B is the block length. For this thesis, we have chosen FFT sizes K to always be twice the block length B:

$$K = 2B.$$

In practice, K can be any integer multiple of $2B$, as shown in Equation 2.40.

4.6 Results

The results of the experiments are given as the average of five measurements for each configuration. In order to interpret the benchmarks, we have defined a *Real-Time Factor* as the number of times faster a benchmark is than real-time. Real-time is defined as the maximum amount of time we can spend on signal processing within an audio cycle in order to finish in time for the next audio cycle. Since our performance loop count N_{per} is defined normalized by the block length, as in Equation 4.14, Real-Time Factor is given simply as the ratio of the simulation time to the computation time:

$$\text{Real-Time Factor} = \frac{\text{Simulation Time}}{\text{Computation Time}} \tag{4.15}$$

For example, if we are simulating 10 seconds of auralization and all of the computation necessary for this auralization is completed in 5 seconds, this would correspond to a Real-Time Factor of 2. Accordingly, higher Real-Time Factors agree with faster algorithms.

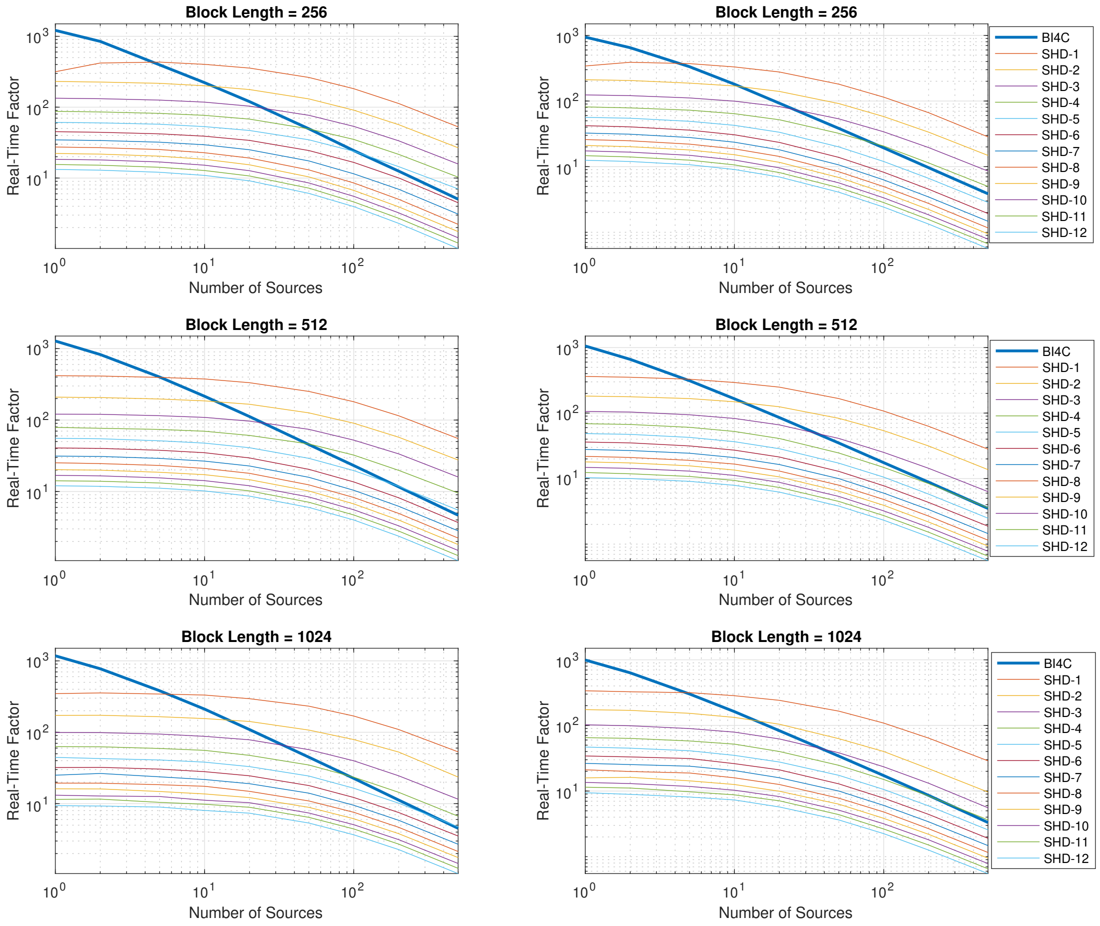

Figure 4.4: Computational performance of BI4C relative to SHD-N for $N = 1, 2, \ldots, 12$ on the 2020 M1 MacBook Pro for different numbers of total sources. Both algorithms use crossfading. The left column shows computation with single-precision arithmetic while the right column is with double-precision arithmetic.

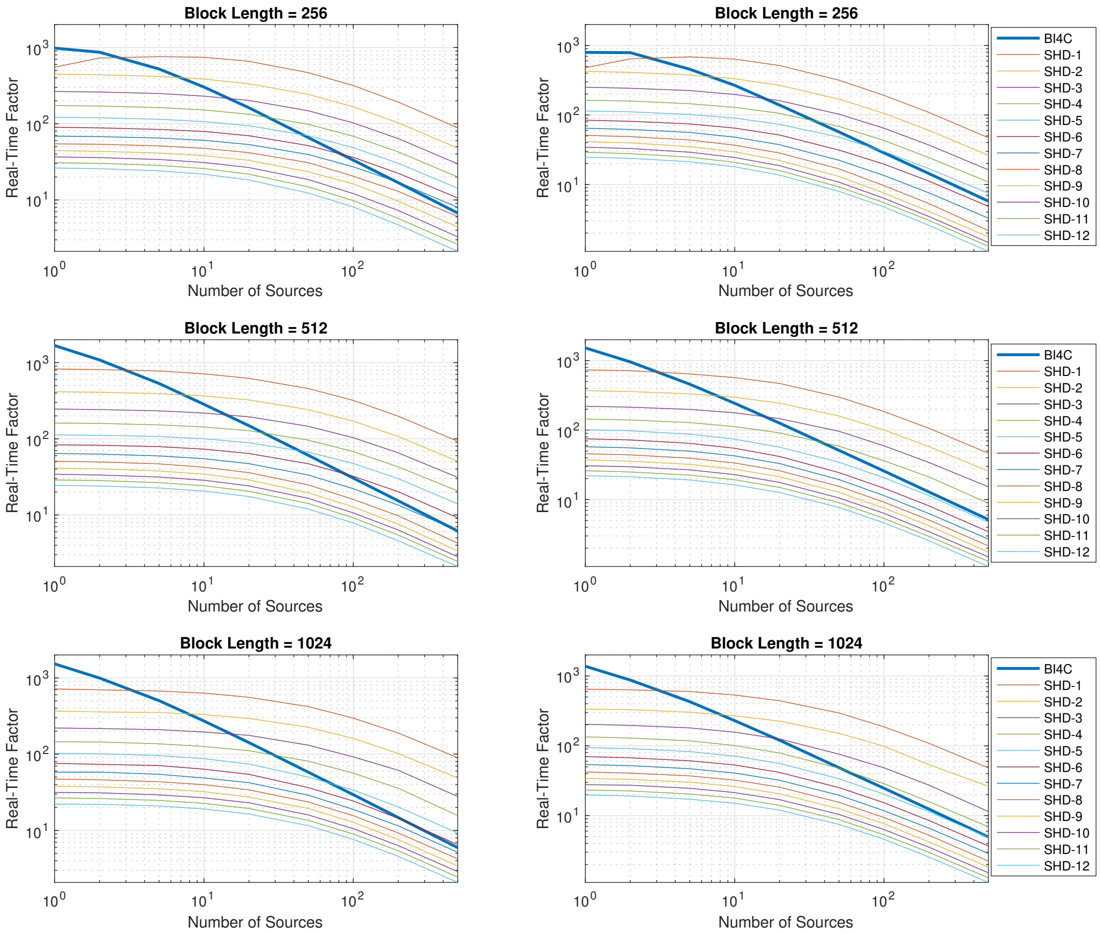

Figure 4.5: Computational performance of BI4C relative to SHD-N for $N = 1, 2, \ldots, 12$ on the 2020 M1 MacBook Pro for different numbers of total sources. Neither algorithm uses crossfading. The left column shows computation with single-precision arithmetic while the right column is with double-precision arithmetic.

Data for SHD-N and BI4C using FFT-based overlap-save convolution *with* frequency-domain crossfading is shown in Figure 4.4, where results using single-precision arithmetic are shown in the left column and results with double-precision arithmetic are shown in the right column. The same data measured *without* any crossfading in either implementation is shown in Figure 4.5.

4.7 Discussion

In this section, we will attempt to provide insight into the results presented in the previous section. While we have shown plots of many configurations of parameters, the basic structure of each plot is roughly the same.

4.7.1 SHD-N

First, we notice the computational cost of SHD-N increases monotonically with truncation order, as anticipated. This is expected because increasing order means distributing more SH transformation coefficients to each sample of the source signals, resulting in higher costs. Moreover, the spacing of the Real-Time Factor curves for SHD-N decreases as order increases. This can be explained by the fact that increasing the order from $N-1$ to N results in the inclusion of $2N+1$ additional basis functions. Since this quantity is a function of N, we would expect the computational cost relative to the total simulation time to be increasingly larger as N increases. The definition of Real-Time Factor as the ratio of simulation time to computation time, however, means that the benchmarks will be decreasingly less times faster than real-time.

Next, we notice these curves tend towards a line as the number of sources increases. For a numbers of sources less than 100, however, the marginal cost of adding a source is lower. This is due to the larger influence of computational operations that are independent of the number of sources. For example, since just $(N+1)^2$ convolutions are necessary for an arbitrary number of sources, the cost imparted by the FFTs is independent of the number of sources and dominates for lower numbers of sources. As we go to higher numbers of sources, the FFT-related costs remain constant while the cost of encoding additional sources into SHs begins to dominate.

4.7.2 BI4C

The computational cost to perform BI4C roughly increases with number of sources, which is expected. All operations except the crossfading and the inverse FFT are dependent on the number of sources. Therefore, the marginal cost of a source remains constant.

4.7.3 Initial Behavior

Inspecting the initial behavior of the $B = 256$ plots for both BI4C and SHD-1, the first few samples do not reflect the transient behavior of the operation of interest. The Real-Time Factor for these samples is much lower than the expected values based on the other plots. This can have several causes: most likely, the tables used in the operations are not yet cached so cache misses are prolonging the execution. Another explanation could be that the OS needs some time to schedule the processes efficiently. This behavior only exists for the first measurements of the $B = 256$ data because of the batch processing nature of the measurements.

4.7.4 Single vs. Double Precision Formats

The computational tradeoff between using single (32-bit) or double (64-bit) arithmetic precision formats was examined for computing the least and most computationally intensive benchmarks on the 2020 M1 MacBook Pro. For BI4C with crossfading, single precision format was around 5% faster than its double precision counterpart for the least intensive benchmark ($B = 256$, one source), which grew to around 27% for the most intensive benchmark ($B = 1024$, 500 sources). For SHD-N with crossfading, single precision format was around 14% faster than its double precision counterpart for the least intensive benchmark ($B = 256$, one source, $N = 1$), which grew to around 48% for the most intensive benchmark ($B = 1024$, 500 sources, $N = 12$). These tradeoffs might be considered as well if computational resources are limited.

4.7.5 Relative Costs

It is clearly seen that BI4C outperforms any order of SHD for very low numbers of sources in terms of the computational cost as defined in this thesis. The tradeoffs become more complicated, however, above 10 total sources. Choosing the best HRTF interpolation algorithm in terms of complexity will come down to the computational resources available on the hardware platform of interest. For numbers of sources greater than 100, BI4C is computationally comparable to order 4 or 5 SHD, depending on block length and numerical precision. This knowledge should be combined with the results from Chapter 3 to make an informed choice about which algorithm is best suited for a platform. That said, the analysis here should not be substituted for dedicated benchmarks for specific applications, as software and hardware implementation details matter greatly.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this thesis, an overview of the performance tradeoffs of two algorithms for spatially interpolating HRTFs was presented. These algorithms are interpolation via N-th order spherical harmonic decomposition and bilinear interpolation of the four closest. The main objective was to assist those interested in building auralization engines in choosing between the two techniques. Other objectives were to provide insight into the minimum density of HRTF measurement grids necessary to build perceptually-continuous HRTF representations.

Two metrics were used for this comparison: relative quality of reconstruction according to an error function and computational cost in an online scenario. The error function was computed as a dB-scale difference of magnitude responses that are computed over a perceptual frequency axis (the Bark scale). Reconstruction was performed at grid points for SHD-N and halfway between grid points for BI4C since these locations correspond to theoretical worst-case local reconstruction quality. Reconstruction error was compared according to the spherical distribution of error as well as weighted average error. Results indicate that the preferred algorithm is a function of the measurement grid and truncation order used. However, BI4C gives better average reconstruction if only sparse grids are available but gives less consistent spherical distribution of reconstruction.

The computational cost analysis was performed by benchmarking efficient C++ implementations of BI4C and SHD-N incorporated into a virtual acoustic rendering framework using block-based audio processing. This framework assumes that at each audio cycle, a fixed number of source signals with associated query angle metadata are to be auralized via object-based binaural audio. Here, auralization refers to rendering of the direct acoustic path via filtering with HRTFs and aggregating all auralized sources into a binaural output buffer. The FIR filtering was implemented via FFT-based convolution incorporating the overlap-save method. Frequency-domain crossfading of HRTFs used in adjacent audio cycles was also implemented to create the illusion of sources moving continuously across the virtual space.

SHD-N was incorporated by quantizing the query angles to an SH coefficient lookup table grid that was sampled to promote spherical uniformity as well as fast lookups. The SH coefficients are then distributed to the source buffer samples and convolution with SH-encoded HRTFs is performed per SH channel via frequency-domain multiplication.

BI4C was incorporated by rounding query angles in four directions to find the four closest grid points, then computing bilinear weights and averaging the measured HRTFs according to these weights. Each interpolated HRTF was convolved with the source buffers via frequency-domain multiplication.

Results of the computational cost analysis indicate that BI4C can render a low number of sources faster (numSources < 5) than most orders of SHD-N. However, as the number of sources gets higher (> 5) the marginal cost of another source is higher for BI4C than SHD-N. This is because the steps required for convolution in SHD-N are independent of the number of sources, which is not true of BI4C. Still, it is difficult to draw broad conclusions about the superiority of either algorithm in terms of computational cost since the cost is a function of many variables, including block length B, SH truncation order N, arithmetic precision, hardware, and more. Moreover, the analysis given here should only be used as guidelines and should not be substituted for dedicated benchmarks in specific applications since implementation details may return differing results.

5.2 Future Work

Many performance tradeoffs of these two algorithms were not addressed in this thesis. For example, we have not quantitatively shown the cost associated with online SH coefficient computation/interpolation or the storage and retrieval of a very dense grid of HRTFs so that interpolation is not necessary.

Future work should incorporate subject-based listening tests to further validate the quality of each interpolation technique. Subjects could be asked to identify source angles in the virtual space or to what extent a source sounds like it is coming from a certain direction. Additionally, other tests could be used to validate the quality of the crossfading used in this thesis.

Other future work could involve the same performance tradeoff analysis for other HRTF interpolation techniques. These might include barycentric interpolation, principal component analysis (PCA), or machine learning techniques. Moreover, a more comprehensive overview of computation hardware could be done, including performance on dedicated DSPs, MCUs, or FPGAs, as well as comparisons of SIMD instruction sets.

Appendix A

Additional Benchmarks

Figures A.1 and A.2 show benchmark data, as developed in Chapter 4, for an early 2015 MacBook Pro. This machine contains a 2.7 GHz dual-core Intel Core i5 with 3MB of shared L3 cache and 8GB of 1866MHz LPDDR3 on-board RAM.

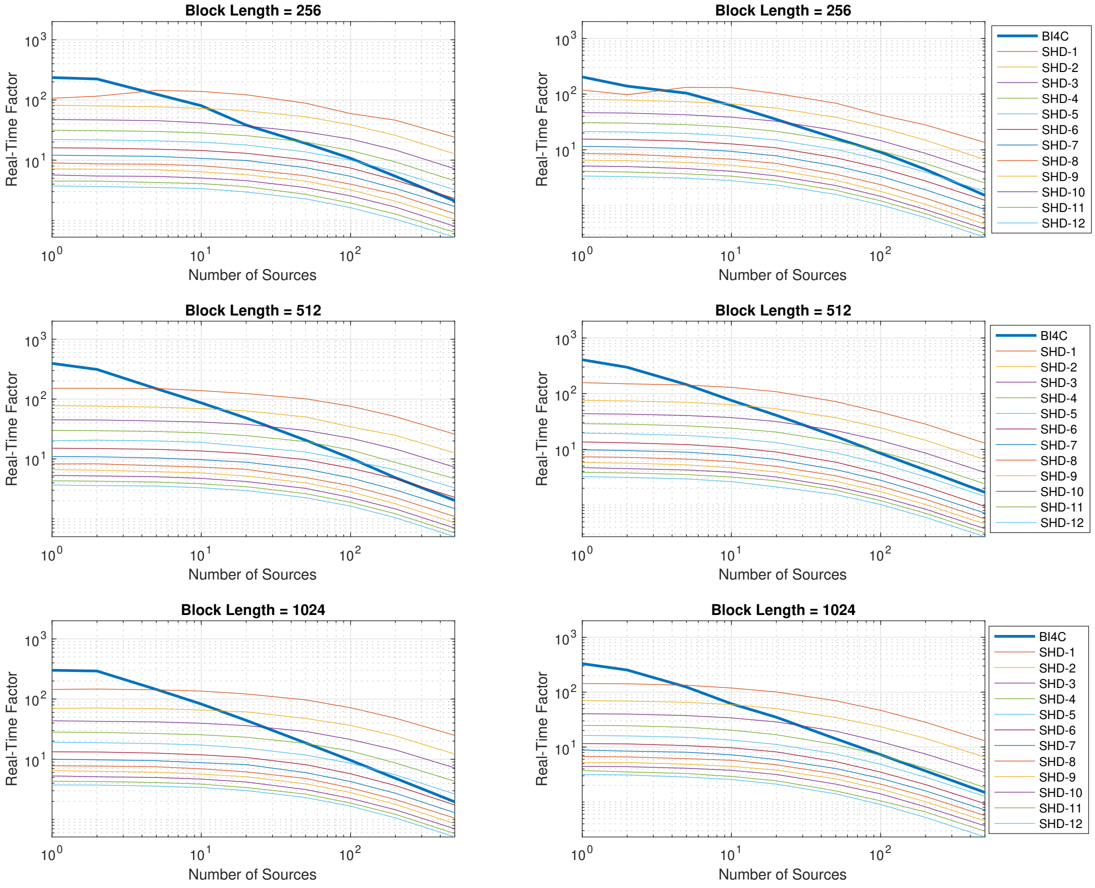

Figure A.1: Computational performance of BI4C relative to SHD-N for $N = 1, 2, \ldots, 12$ on the 2015 i5 MacBook Pro for different numbers of total sources. Both algorithms use crossfading. The left column shows computation with single-precision arithmetic while the right column is with double-precision arithmetic.

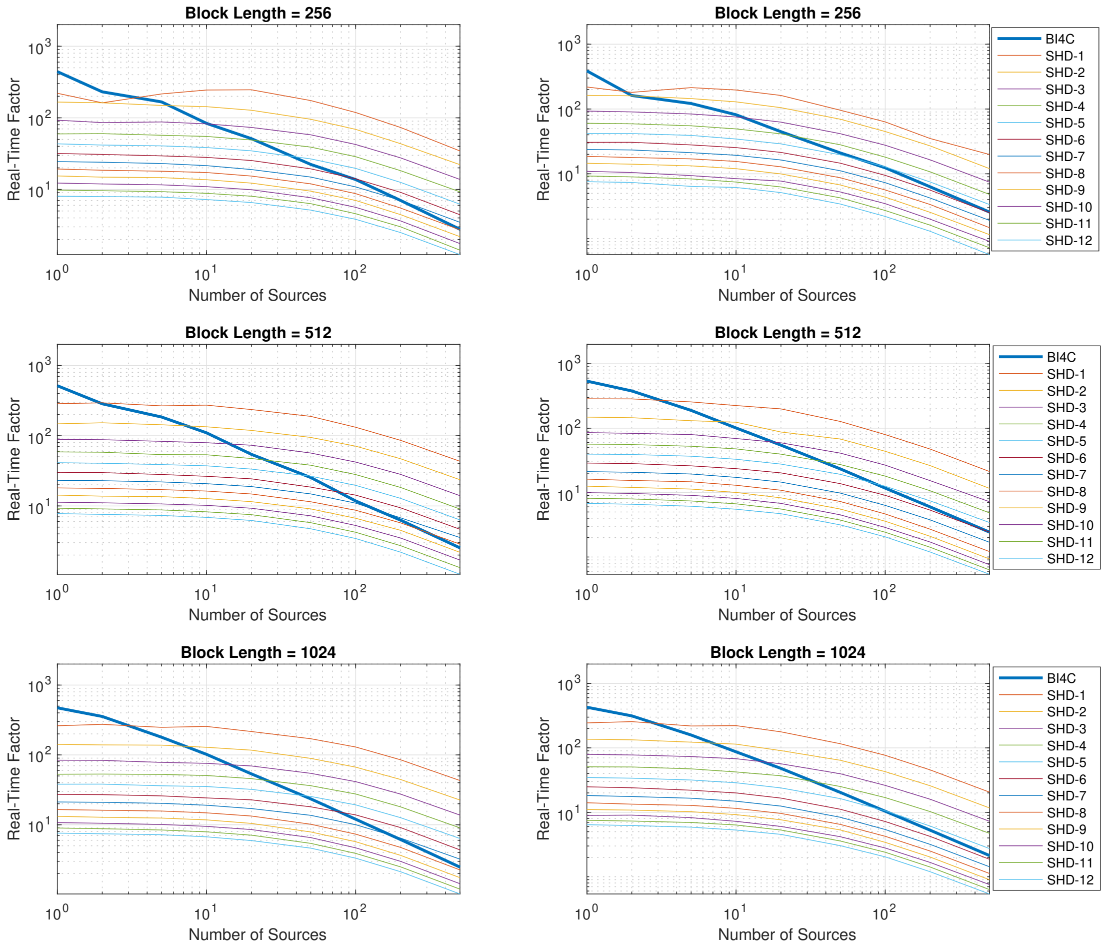

Figure A.2: Computational performance of BI4C relative to SHD-N for $N = 1, 2, \ldots, 12$ on the 2015 i5 MacBook Pro for different numbers of total sources. Neither algorithm uses crossfading. The left column shows computation with single-precision arithmetic while the right column is with double-precision arithmetic.

Bibliography

[1] B. Xie, “Spatial Hearing and Virtual Auditory Display,” in *Head-Related Transfer Function and Virtual Auditory Display*. Plantation, FL, USA: J. Ross Publishing, 2013, ch. 1, pp. 1–43.

[2] M. Gorzel, A. Allen, I. Kelly, J. Kammerl, A. Gungormusler, H. Yeh, and F. Boland, “Efficient encoding and decoding of binaural sound with resonance audio,” in *Audio Engineering Society Conference: 2019 AES International Conference on Immersive and Interactive Audio*, Audio Engineering Society, 2019.

[3] M. Cuevas-Rodríguez, L. Picinali, D. González-Toledo, C. Garre, E. de la Rubia-Cuestas, L. Molina-Tanco, and A. Reyes-Lecuona, “3D Tune-In Toolkit: An open-source library for real-time binaural spatialisation,” en, *PLOS ONE*, vol. 14, no. 3, I. Yasin, Ed., e0211899, Mar. 2019. [Online]. Available: `https://dx.plos.org/10.1371/journal.pone.0211899` (visited on 07/08/2021).

[4] Oculus VR L, “Oculus Audio SDK,” 2017. [Online]. Available: `https://developer.oculus.com/audio/`.

[5] Microsoft, “Spatial Sound in Unity,” 2017. [Online]. Available: `https://developer.microsoft.com/en-us/windows/mixed-reality/spatial_sound_in_unity`.

[6] J.-M. Batke, “The B-Format Microphone Revised,” in *Proceedings of the Ambisonics Symposium, Graz, Austria.* 2009.

[7] F. Wefers, *Partitioned convolution algorithms for real-time auralization.* Logos Verlag Berlin GmbH, 2015, vol. 20.

[8] J. Daniel, “Représentation de champs acoustiques, application à la transmission et à la reproduction de scènes sonores complexes dans un contexte multimédia,” Ph.D. dissertation, Université Paris 6, 2000.

[9] M. Morimoto and Y. Ando, “On the Simulation of Sound Localization,” *Journal of the Acoustical Society of Japan (E)*, vol. 1, no. 3, pp. 167–174, 1980.

[10] F. L. Wightman and D. J. Kistler, “Headphone Simulation of Free-Field Listening. II: Psychophysical Validation,” *The Journal of the Acoustical Society of America*, vol. 85, no. 2, pp. 868–878, 1989.

[11] W. G. Gardner, “Reverberation algorithms,” in *Applications of digital signal processing to audio and acoustics*, Springer, 2002, pp. 85–131.

[12] V. Valimaki, J. D. Parker, L. Savioja, J. O. Smith, and J. S. Abel, “Fifty years of artificial reverberation,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 5, pp. 1421–1448, 2012.

[13] L. Savioja and U. P. Svensson, “Overview of geometrical room acoustic modeling techniques,” *The Journal of the Acoustical Society of America*, vol. 138, no. 2, pp. 708–730, 2015.

[14] F. Olivieri, N. Peters, and D. Sen, “Scene-based audio and higher order ambisonics: A technology overview and application to next-generation audio, vr and 360 video,” *EBU Tech*, 2019.

[15] R. Bomhardt, M. de la Fuente Klein, and J. Fels, “A High-Resolution Head-Related Transfer Function and Three-Dimensional Ear Model Database,” in *Proceedings of Meetings on Acoustics*, 1. 2016, vol. 29, p. 050 002. [Online]. Available: `https://asa.scitation.org/doi/abs/10.1121/2.0000467`.

[16] C. Armstrong, L. Thresh, D. Murphy, and G. Kearney, “A Perceptual Evaluation of Individual and Non-Individual HRTFs: A Case Study of the SADIE II Database,” *Applied Sciences*, vol. 8, no. 11, p. 2029, Oct. 2018. [Online]. Available: `http://www.mdpi.com/2076-3417/8/11/2029` (visited on 07/08/2021).

[17] V. C. Raykar, R. Duraiswami, and B. Yegnanarayana, “Extracting the Frequencies of the Pinna Spectral Notches in Measured Head Related Impulse Responses,” *The Journal of the Acoustical Society of America*, vol. 118, no. 1, pp. 364–374, 2005. [Online]. Available: `https://doi.org/10.1121/1.1923368`.

[18] S. Mehrgardt and V. Mellert, “Transformation Characteristics of the External Human Ear,” *The Journal of the Acoustical Society of America*, vol. 61, no. 6, pp. 1567–1576, 1977. [Online]. Available: `https://doi.org/10.1121/1.381470`.

[19] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, Second Edition. Prentice-hall Englewood Cliffs, 1999.

[20] A. Kulkarni, S. Isabelle, and H. Colburn, “On the minimum-phase approximation of head-related transfer functions,” en, in *Proceedings of 1995 Workshop on Applications of Signal Processing to Audio and Accoustics*, IEEE, 1995, pp. 84–87. [Online]. Available: `http://ieeexplore.ieee.org/document/482964/` (visited on 07/08/2021).

[21] J. O. Smith, *Introduction to digital filters: with audio applications.* Julius Smith, accessed 8/16/2021, vol. 2. [Online]. Available: `https://ccrma.stanford.edu/~jos/filters/`.

[22] D. J. Kistler and F. L. Wightman, “A Model of Head-Related Transfer Functions Based on Principal Components Analysis and Minimum-Phase Reconstruction,” *The Journal of the Acoustical Society of America*, vol. 91, no. 3, pp. 1637–1647, 1992. [Online]. Available: `https://doi.org/10.1121/1.402444`.

[23] J. Nam, J. S. Abel, and J. O. Smith III, “A method for estimating interaural time difference for binaural synthesis,” in *Audio Engineering Society Convention 125*, Audio Engineering Society, 2008.

[24] Franck, Andreas, "Efficient Algorithms and Structures for Fractional Delay Filtering Based on Lagrange Interpolation," *Journal of the Audio Engineering Society*, vol. 56, no. 12, pp. 1036–1056, Dec. 2009.

[25] J.-M. Jot, V. Larcher, and O. Warusfel, "Digital Signal Processing Issues in the Context of Binaural and Transaural Stereophony," in *Audio Engineering Society Convention 98*, Audio Engineering Society, 1995.

[26] V. Välimäki, "Discrete-Time Modeling of Acoustic Tubes Using Fractional Delay Filters," Ph.D. dissertation, Helsinki University of Technology, Lab. of Acoustics and Audio, 1995.

[27] W. Martens, "Principal Component Analysis and Resynthesis of Spectral Cues to Perceived Direction," in *Proceeding of the International Computer Music Conference*. San Francisco, CA, USA, 1987, pp. 274–281.

[28] J. C. Middlebrooks and D. M. Green, "Observations on a Principal Components Analysis of Head-Related Transfer Functions," *The Journal of the Acoustical Society of America*, vol. 92, no. 1, pp. 597–599, 1992. [Online]. Available: `https://doi.org/10.1121/1.404272`.

[29] J. Chen, B. Van Veen, and K. Hecox, "A Spatial Feature Extraction and Regularization Model for the Head-Related Transfer Function," *The Journal of the Acoustical Society of America*, vol. 97, Feb. 1995.

[30] Z. Wu, F. H. Y. Chan, F. K. Lam, and J. C. K. Chan, "A time domain binaural model based on spatial feature extraction for the head-related transfer function," *The Journal of the Acoustical Society of America*, vol. 102, no. 4, pp. 2211–2218, 1997. [Online]. Available: `https://doi.org/10.1121/1.419597`.

[31] V. Larcher, O. Warusfel, J.-M. Jot, and J. Guyard, "Study and Comparison of Efficient Methods for 3-D Audio Spatialization Based on Linear Decomposition of HRTF Data," in *Audio Engineering Society Convention 108*, Audio Engineering Society, 2000.

[32] W. G. Gardner, "Reduced-Rank Modeling of Head-Related Impulse Responses using Subset Selection," in *Proceedings of the 1999 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. WASPAA'99 (Cat. No. 99TH8452)*, IEEE, 1999, pp. 175–178.

[33] E. G. Williams, *Fourier Acoustics: Sound Radiation and Nearfield Acoustical Holography*. Academic press, 1999.

[34] H. Liu, Y. Fang, and Q. Huang, "Efficient representation of head-related transfer functions with combination of spherical harmonics and spherical wavelets," *IEEE Access*, vol. 7, pp. 78 214–78 222, 2019.

[35] W. Zhang, T. Abhayapala, R. Kennedy, and R. Duraiswami, "Insights into Head-Related Transfer Function: Spatial Dimensionality and Continuous Representation," *The Journal of the Acoustical Society of America*, vol. 127, pp. 2347–57, Apr. 2010.

[36] G. F. Kuhn, "Model for the Interaural Time Differences in the Azimuthal Plane," *The Journal of the Acoustical Society of America*, vol. 62, no. 1, pp. 157–167, 1977. [Online]. Available: https://doi.org/10.1121/1.381498.

[37] Á. González, "Measurement of Areas on a Sphere Using Fibonacci and Latitude–Longitude Lattices," *Mathematical Geosciences*, vol. 42, no. 1, p. 49, Nov. 2009. [Online]. Available: https://doi.org/10.1007/s11004-009-9257-x.

[38] J. S. Brauchart and P. J. Grabner, "Distributing Many Points on Spheres: Minimal Energy and Designs," *Journal of Complexity*, vol. 31, no. 3, pp. 293–326, 2015, Oberwolfach 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0885064X15000205.

[39] R. H. Hardin and N. J. A. Sloane, "McLaren's Improved Snub Cube and Other New Spherical Designs in Three Dimensions," *Discrete & Computational Geometry*, vol. 15, no. 4, pp. 429–441, Apr. 1996. [Online]. Available: https://doi.org/10.1007/BF02711518.

[40] J. Fliege and U. Maier, "The Distribution of Points on the Sphere and Corresponding Cubature Formulae," *IMA Journal of Numerical Analysis*, vol. 19, no. 2, pp. 317–334, Apr. 1999. [Online]. Available: https://doi.org/10.1093/imanum/19.2.317.

[41] Z. Li and R. Duraiswami, "Flexible and Optimal Design of Spherical Microphone Arrays for Beamforming," *Trans. Audio, Speech and Lang. Proc.*, vol. 15, no. 2, pp. 702–714, Feb. 2007. [Online]. Available: https://doi.org/10.1109/TASL.2006.876764.

[42] V. Lebedev and A. Skorokhodov, "Quadrature Formulas of Orders 41, 47 and 53 for the Sphere," in *Russian Acad. Sci. Dokl. Math*, vol. 45, 1992, pp. 587–592.

[43] P. Lecomte, P.-A. Gauthier, C. Langrenne, A. Berry, and A. Garcia, "A Fifty-Node Lebedev Grid And Its Applications To Ambisonics," *Journal of the Audio Engineering Society*, vol. 64, pp. 868–881, Dec. 2016.

[44] J. Driscoll and D. Healy, "Computing Fourier Transforms and Convolutions on the 2-Sphere," *Advances in Applied Mathematics*, vol. 15, no. 2, pp. 202–250, 1994. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0196885884710086.

[45] W. G. Gardner and K. D. Martin, "HRTF measurements of a KEMAR," *The Journal of the Acoustical Society of America*, vol. 97, no. 6, pp. 3907–3908, 1995.

[46] D. Begault, "Implementing 3-D Sound Systems, Sources, and Signal Processing," in *3-D Sound for Virtual Reality and Multimedia*. San Diego, CA, USA: Academic Press Professional, Inc., 1994, ch. 4, pp. 132–136.

[47] F. Freeland, L. Biscainho, and P. Diniz, "Interpositional Transfer Function for 3D-Sound Generation," *Journal of the Audio Engineering Society*, vol. 52, pp. 915–930, Sep. 2004.

[48] H. Gamper, "Head-Related Transfer Function Interpolation in Azimuth, Elevation, and Distance," *The Journal of the Acoustical Society of America*, vol. 134, no. 6, EL547–EL553, 2013. [Online]. Available: https://doi.org/10.1121/1.4828983.

[49] F. P. Freeland, L. W. Biscainho, and P. S. Diniz, “Interpolation of head-related transfer functions (HRTFs): A multi-source approach,” in *2004 12th European Signal Processing Conference*, IEEE, 2004, pp. 1761–1764.

[50] J. W. Cooley and J. W. Tukey, “An Algorithm for the Machine Calculation of Complex Fourier Series,” *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

[51] J. Dongarra and F. Sullivan, “Top Ten Algorithms of the Century,” *Computing in Science and Engineering*, vol. 2, no. 1, pp. 22–23, 2000.

[52] T. G. Stockham Jr, “High-Speed Convolution and Correlation,” in *Proceedings of the April 26-28, 1966, Spring joint computer conference*, 1966, pp. 229–233.

[53] C. S. Burrus and T. Parks, “Convolution Algorithms,” *Citeseer: New York, NY, USA*, 1985.

[54] F. Wefers and M. Vorländer, “Efficient time-varying FIR filtering using crossfading implemented in the DFT domain,” in *Proceedings of the 2014 7th Medical and Physics Conference Forum Acusticum, Cracow, Poland*, 2014, pp. 7–12.

[55] E. Zwicker, “Subdivision of the audible frequency range into critical bands (Frequenzgruppen),” *The Journal of the Acoustical Society of America*, vol. 33, no. 2, pp. 248–248, 1961.

[56] F. Brinkmann and S. Weinzierl, “Comparison of head-related transfer functions pre-processing techniques for spherical harmonics decomposition,” in *Audio Engineering Society Conference: 2018 AES International Conference on Audio for Virtual and Augmented Reality*, Audio Engineering Society, 2018.

[57] C. Nachbar, F. Zotter, E. Deleflie, and A. Sontacchi, “Ambix-A Suggested Ambisonics Format,” in *Ambisonics Symposium, Lexington*, 2011, p. 11.

[58] C. S. Reddy and R. M. Hegde, “On the Conditioning of the Spherical Harmonic Matrix for Spatial Audio Applications,” en, *arXiv:1710.08633 [eess]*, Mar. 2018, arXiv: 1710.08633. [Online]. Available: `http://arxiv.org/abs/1710.08633` (visited on 07/08/2021).

[59] S. G. Johnson and M. Frigo, “A Modified Split-Radix FFT with Fewer Arithmetic Operations,” *IEEE Transactions on Signal Processing*, vol. 55, no. 1, pp. 111–119, 2006.

[60] Á. González, “Measurement of areas on a sphere using Fibonacci and latitude–longitude lattices,” *Mathematical Geosciences*, vol. 42, no. 1, pp. 49–64, 2010.

[61] M. Frigo and S. G. Johnson, “The design and implementation of FFTW3,” *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005.