UNIVERSITY OF VICTORIA
Department of Electrical and Computer Engineering

ELEC 499 – Design Project

SURROUND SOUND IMPULSE RESPONSE

Measurement with the Exponential Sine Sweep; Application in Convolution Reverb

Project Team:

Madeline Carson

Hudson Giesbrecht

Tim Perry

Project Supervisor:

Peter Driessen

June-July, 2009

Abstract

A 5-channel circular microphone array was used to measure a matrix of surround sound room impulse responses. 70 measurements were obtained in the Phillip T. Young Recital Hall: At each of 7 listener locations, the response was captured for 10 sound source positions. This source-listener matrix can be used to simulate the acoustic space from multiple perspectives.

A step-by-step approach to obtaining the multiple-measured room impulse response (MMRIR) is documented, and example measurements are presented using the exponential sine sweep (ESS) method. The ESS method has been shown to produce impulse responses with exceptional signal-to-noise ratios.

To test the results on listeners, a surround sound convolution reverb was created. By performing multi-sound-source convolutions, dry-recorded signals can be virtually placed into the measured acoustic space.

The full process from multi-channel RIR measurement to application in convolution reverb is presented. Additional applications of MMRIR measurement are addressed, and early reflection data captured by the measurements is used to graphically demonstrate spatial sound imaging.

Contents

SURROUND SOUND IMPULSE RESPONSE

Measurement with the Exponential Sine Sweep;
Application in Convolution Reverb

1. Introduction

The practice of modeling an acoustic space with its impulse response (IR) is widely used by engineers and scientists, and the results have many applications. A monophonic impulse response measurement can quite effectively capture the general behaviour of sound in a specific location within a room - but what about the directional characteristics? Stepping up to a stereophonic representation of the impulse response will reveal not only the timing and intensity of reflections, but also information about where those reflections are coming from. However, this is still just a partial representation of the acoustics in a room. Sound propagates through 3 dimensions of space, and humans perceive sound in 4 dimensions (3 dimensions of space; one dimension of time). To truly make the most of an impulse response for modeling acoustics, we would like a 3D spatial representation. The purpose of measuring an impulse response may be for acoustical analysis, or it may be for use in a *convolution reverb*, where the human hearing system is the judge. In either case, the *ideal* room impulse response (RIR) will give an accurate spatial representation of the acoustics in that room with respect to time.

This paper presents surround sound impulse response measurement with an application centered approach: capturing an acoustic environment, and virtually placing sound sources within that environment. For application in convolution reverb, which will be introduced shortly, a surround sound representation of the RIR is a practical step up from stereo. 3D spatial playback systems do exist, but do not yet enjoy the popularity of surround sound configurations.

Impulse response measurement and convolution reverb are fundamentally related. The detailed theory and implementation of each process that forms this relationship, however, have typically been kept separate. We are referring here to the process of taking a set of isolated, dry recordings of individual musicians, and turning them into a single “live recording” of an ensemble performing in a specific acoustic environment (a concert hall, for example). With surround sound we take the concept a step further: the goal is not to simply produce a faux live recording of a performance; the ultimate goal is to fool a blind-folded listener into believing that they are *attending* a live performance.

To put the above concept into perspective, we might simply put speakers on the stage of a concert hall, play back the recorded sounds, and re-record the noisy mess that results from the buzzing PA system. Instead, the approach that was implemented was a bit more involved than this, and yields a much finer result. Moreover, only one trip to the concert hall to emit obnoxious speaker sounds is

required, after which the resulting RIRs can be convolved with audio at will. The process that was undertaken can is summarized as follows:

- The first step involved researching impulse response measurement techniques, and selecting a method of RIR measurement. The exponential sine sweep (ESS) method was chosen, primarily for the enhanced dynamic range potential over other techniques.

- Using the ESS method, the second step was to obtain the multi-channel measured room impulse response (MMRIR) in different locations throughout an acoustic environment. Specifically, this paper documents a project in which a matrix of surround sound RIRs was obtained: a data set that represents various source-listener locations in a recital hall.

- The third step was to analyze these results, and to reach some conclusions about the quality and accuracy of the measurements made. Additionally, certain acoustic parameters were estimated for selected sets of data.

- The final step was to implement a *surround sound convolution reverb* by convolving dry-recorded audio signals with the measured surround sound room impulse responses.

This paper assumes that the reader has a background in mathematics and physics, and is familiar with digital signal processing fundamentals. It is also assumed that the reader has a conceptual grasp of acoustics, or a desire to learn about acoustics. Graphical representations have been used to illustrate many relationships, which may help to make this paper accessible to the casual reader. Key concepts will be introduced where applicable.

1.1. Why Measure the Impulse Response?

By capturing the impulse response, we are in essence seeking to take the audio fingerprint of a location in an acoustic environment – a model of how sound behaves at that particular location, in that particular environment. Once the impulse response in an acoustic space has been accurately measured, there are many applications that can make excellent use of this data. This includes constructing convolution reverb from the impulse response, and digitally re-constructing the acoustic environment for simulation or pro-audio applications. Another application is room correction, whereby the acoustic problems in a room are determined quantitatively by taking the impulse response, and either acoustic treatment is applied to the room (ideally), or frequency response correction is applied to the playback system (or both).

The finest digital reverbs and other digital audio effects such as amplifier simulators are based on the principle of modeling using the impulse response measured from a real world, physical

environment or object (whether that be a room, an outdoor space, a guitar amplifier cabinet, a pre-amp, or any resonant object - for example a piano or a piece of metal). In the music industry, convolution reverb is used to add a sense of spaciousness to pre-recorded sounds from a studio. In the film industry, a movie can be completed post production by giving audio conversations that were recorded in a studio an appropriate *spatial impression*. For example, voice recordings may be convolved with IRs that were measured in a forest; the resulting conversation will sound as if it took place in a forest.

A high quality set of surround sound impulse responses is also valuable data for acoustic and electroacoustic research. Psychoacoustic research can also make use of impulse response measurements and convolution reverb when studying the perceptual and cognitive elements of human hearing.

1.2. Reverberation and the Impulse Response

Clap your hands in a large room; the initial impulse of broadband noise conceptually resembles a delta function. The resulting reflections and ambience that you hear after exciting the room, is reverb. Reverberation is the indirect sound that reaches a listener from a source. From any source there is sound that reaches the listener in a direct path, as well as sound that reaches the listener indirectly through reflections in the acoustic space. This reflected sound reaches the listener later then the direct sound due to its longer travel path. It has been diffused by certain surfaces, and lost energy due to propagation through the air and absorption in the room. The reflected/diffused sound continues to interact with its surroundings, until it has been fully absorbed.

Essentially, the reverberation in a room is the characterized by the impulse response. Reverb can be separated into two main components, which can be viewed when looking at the impulse response representation of a location in room:

1. **Early Reflections** – the first reflections that we hear within about 100 ms of hearing the direct sound of the source.

2. **Late Reverberation** – the reverberant sound field after about 100 ms, until it fully decays. Late reverb is characterized by a dense texture of diffused reflections that reach our ears from many different paths. These diffused reflections are out of phase with one another, causing us to hear the comb filtering effect. We perceive this as ‘ambience’.

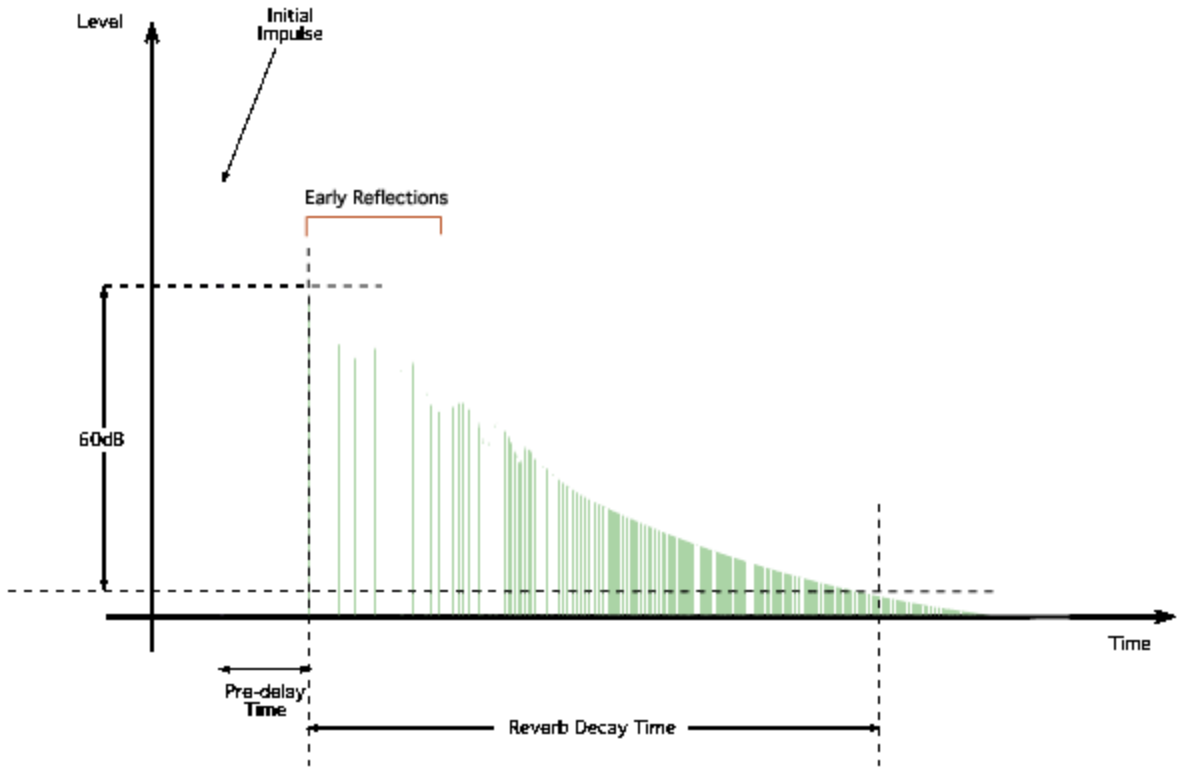


Figure 1: Typical impulse response of a room, highlighting the ITDG, early reflections, and RT60 [F1].

Two acoustic parameters that relate directly to the impulse response will be frequently discussed in this paper, particularly when we start looking at real impulse response measurement results.

- **Initial Time Delay Gap (ITDG)** – the time gap between the arrival of the direct sound that we hear, and the first early reflection. This gives us an impression of intimacy with relation to walls in a room, and helps our ears reveal our relative position within that room.

- **RT60 -** the *reverberation time, RT60,* is the time it takes for the acoustic signal to decay by 60 dB. The reverb time varies throughout the spectrum, and can be measured at discrete frequency values. For practical applications, averaging schemes are used to define the reverb time in the portion of the spectrum where human hearing is most sensitive (between about 125 Hz and 4000 Hz is where we perceive the most significant sound colouration by reverb). RT60 is defined according to the *Sabine* and *Eyring* equations as a function the volume of a room and the absorption of sound energy inside that room.

1.3. Convolution Reverb

Convolution reverb is based on the knowledge that if the impulse response of a linear time invariant (LTI) system is known, then the response of the system to any input can be calculated through the discrete time convolution sum:

$$\sum_{k=-\infty}^{+\infty} x[k]h[n-k] \qquad where \begin{cases} x[n]\ is\ the\ input\ signal \\ h[n] is\ the\ IR\ of\ the\ LTI\ system \end{cases}$$

Although an acoustic space is not a perfect LTI system, it can be considered as one for the purpose of practical impulse response measurements. A real acoustic environment can be convincingly modeled by its impulse response. In order to add reverberation to an input signal and effectively place the source into the measured acoustic space, the input signal is convolved with the impulse response of the room. The resulting convolution reverb sounds natural to a listener, as if they are listening to a sound that was recorded in the particular room where the impulse response was measured.

2. Exponential Sine Sweep Method for Impulse Response Measurement

The exponential sine sweep (ESS) method for impulse response measurement has gained widespread usage since AES-Paris in 2000. Recent advancements to the technique have been presented by Angelo Farina [1] and others throughout this decade. For IR measurement applications, the exponential sine sweep provides several distinct advantages over the linear sine sweep, as well as the MLS and IRS methods [1],[2],[3],[5]. The potential advantages of exponential sine sweeps were recognized in the early 90's; however, the adoption of the ESS technique for practical measurements was delayed due to high computational complexity and previous lack of computational power.

The ESS method uses an exponentially swept sinusoid for room excitation, and aperiodic deconvolution to extract the impulse response from the recorded room response. Unlike the linear sine sweep method, which relies on synchronous averaging of multiple sweeps to increase the IR signal to noise ratio, the ESS method uses a single, long sweep. The primary advantages of this technique over other IR measurement methods are as follows [1]:

- Better noise rejection than the MLS method, given a signal of the same length
- Near-perfect separation of non-linear effects from the desired linear response

- For systems that may have time variance, such as those involving sound propagation in air, a long sweep utilizing no synchronous averaging is useful for avoiding artefacts in the reverberant tail and high frequency phase errors.

2.1. Exponential Sine Sweep

The exponential sweep which was used for room excitation can be expressed in the continuous time domain as:

$$s(t) = \sin[\theta(t)] = \sin[\mathrm{K} \cdot (e^{-t/L} - 1)] \tag{1}$$

where

$$K = \frac{\omega_1 T}{\ln\left(\frac{\omega_1}{\omega_2}\right)} \quad , \quad L = \frac{T}{\ln\left(\frac{\omega_1}{\omega_2}\right)} \tag{2}$$

T represents the time duration of the sweep, which is a swept sinusoid that is bounded by the frequencies ω_1 and ω_2. The range of the recorded sine wave sweep will be limited by the range that can be physically be produced by the speakers used, and to a lesser extent the frequency response of the microphones (which, for good microphones is a wide range, but not necessarily a flat response).

Given a signal that varies in frequency, the energy at a specific frequency is proportional to the time duration during which the signal oscillates at that specific frequency. In other words, energy is related to the rate of change of the instantaneous frequency. Instantaneous frequency $\omega(t)$ is defined as the rate of change of $\theta(t)$, which for (1) take the form:

$$\omega(t) = \frac{d\{\theta(t)\}}{dt} = \frac{K}{L} \cdot e^{t/L} \tag{3}$$

Since the energy signal $E(t)$ is proportional to the time at a given frequency, it follows that $E(t)$ is inversely proportional to the rate of change in $\omega(t)$:

$$E(t) \propto \frac{1}{\left[\frac{d\{\omega(t)\}}{dt}\right]} = \frac{L^2}{K} \cdot e^{-t/L} \tag{4}$$

By taking the Fourier transform of (4), energy can be expressed as a function of frequency $E(\omega)$:

$$E(j\omega) \propto \frac{L^2}{K} \cdot \frac{1}{L + \omega j}$$

Defining k to be a constant of proportionality, $E(\omega)$ can be expressed as [3]:

$$E(j\omega) = \frac{kL^2}{K} \cdot \frac{1}{L + \omega j} \tag{5}$$

This leads to the important realization that with the exponential sine sweep, energy decreases as frequency increases. More specifically, if the frequency is doubled, the factor $\frac{1}{\omega}$ from (5) becomes $\frac{1}{2\omega}$,corresponding to a change of $10log_{10}(1/2) \cong -3dB$. This represents a drop of 3dB in energy for each doubling of the frequency $(-3dB/octave)$. Figure 2 shows the spectrum of the exponential sine sweep.

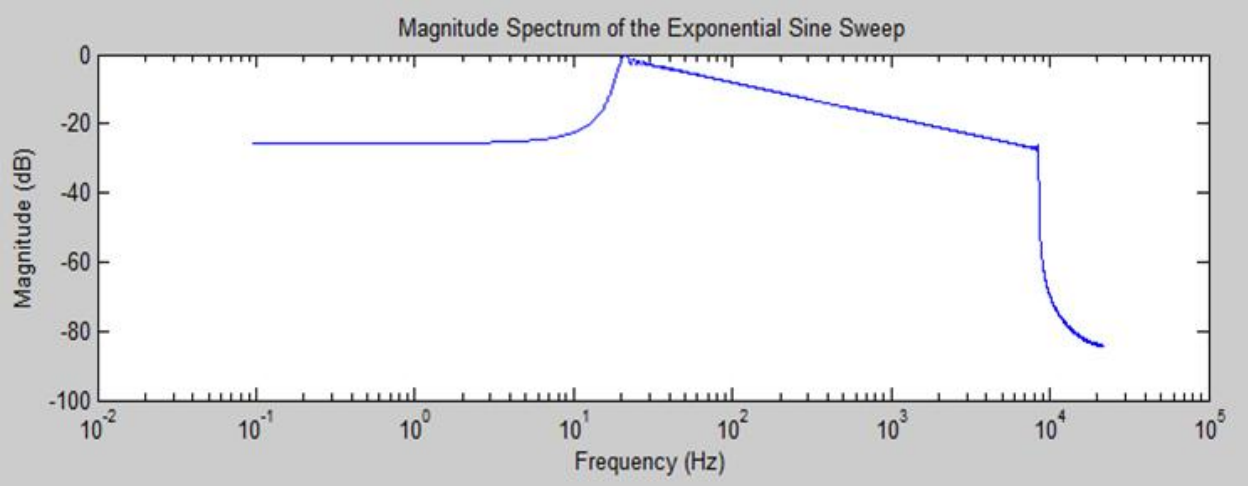


Figure 2: Magnitude response of the ESS signal plotted in Matlab (-3dB per octave slope in the energy spectrum).

2.2. Inverse Filter

Suppose that $r(t)$ denotes the room the room response to an exponential sweep, where $s(t)$ is the reference sweep defined by (1). To extract the impulse response from $r(t)$, a deconvolution is performed with the reference signal. The result of the deconvolution must achieve a flat spectrogram; however, the time-reversed ESS signal also exhibits a -3dB/octave decrease in its energy spectrum. As a result, an amplitude modulation must be applied to the spectrum of the time reversed reference signal [3]. The objective is to create an inverse filter $f(t)$ which is to be convolved with $r(t)$, yielding the impulse response $h(t)$.

$$h(t) = r(t) * f(t) \tag{6}$$

$f(t)$ is created by applying an amplitude modulation envelope of $+6dB/octave$ to the spectrum of the time reversed signal [1]. This is termed post-modulation. An alternative modulation scheme called pre-modulation was recently proposed in [3], which applies an amplitude modulation to the swept signal, allowing for less post-processing. While the prospect of pre-modulation is attractive in its simplicity, it requires special considerations that must be met prior to recording. Opting in favour of a robust system that maximizes opportunities for post-processing experimentation, post-modulation was chosen. The general form of the post-modulation function is [3]:

$$m(t) = \frac{A}{\omega(t)} = A\left[\frac{K}{L} \cdot e^{t/L}\right]^{-1} \tag{7}$$

A is an amplitude scalar for the modulation function. In the time domain the modulation envelope is $-6dB/octave$ and in the frequency domain the spectrum amplitude modulation is $+6dB/octave$, regardless of A. At time t = 0, $\omega(t) = \omega_1$ (the minimum frequency of the swept sinusoid). Opting for unity gain at ω_1: ω_1

$$1 = \frac{A}{\omega(0)} = \frac{A}{\omega_1} \quad \rightarrow \quad A = \omega_1$$

The resulting modulation function takes the form:

$$m(t) = \frac{\omega_1}{\omega(t)} \tag{8}$$

After modulating the time reversed signal with $m(t)$, we have an inverse filter $f(t)$ that has a slope of $+3dB/octave$. The computation of the inverse filter involves solving a large Toeplitz matrix (diagonal constant matrix). This matrix results because $m(t)$ is a function of the instantaneous frequency of the ESS signal, $\omega(t)$,which is in turn a function of time. Transferring this to the digital domain: $\omega(t)$is sampled, becoming the instantaneous frequency vector $\omega(nT)$, which contains a discrete frequency value for every discrete time value nT in the ESS signal. For a ten second sweep signal that is sampled at 44.1 kHz, $\omega(nT)$ is a vector of 441000 discrete frequency values. The modulation signal $m(nT)$ is inversely proportional to $\omega(nT)$. The multiplication of $m(t)$ with the time reversed ESS signal that occurs during amplitude modulation is a vector multiplication. This vector multiplication, for the above example, will result in a 441000 by 441000 matrix with constant descending diagonals from left to right. Obtaining the inverse filter directly would involve solving this matrix. As a result, a very large

number of operations and memory accesses are required to solve for $f(t)$ without using an approximation.

The example Matlab code illustrates the attempted direct computation, (the appropriate initializations have been made based on above math).

```
%=====================================================================
% Direct inverse filter computation in Matlab (insufficient memory)
%=====================================================================
T = 10;                  %(2^N)/fs;
f1=20;                   %starting frequency
f2=20000;                %ending frequency
fs=44100;

% Create exponential sine sweep
w1 = 2*pi*f1;
w2 = 2*pi*f2;
K = T*w1/log(w2/w1);
L = T/log(w2/w1);
t = linspace(0,T-1/fs,fs*T);       % time vector
s = sin(K*(exp(t/L) - 1));         % ESS signal (discrete vector form)

% Direct inverse filter computation
f_range=linspace(f1,f2,length(s)); % linear spaced increments in frequency
A = f1;
m = f1./f_range;                   % modulating function (8)
s_rev=fliplr(s);                   % time reversal of s
sinvTo = m'*s_rev;                 % vector multiplication yielding matrix
```

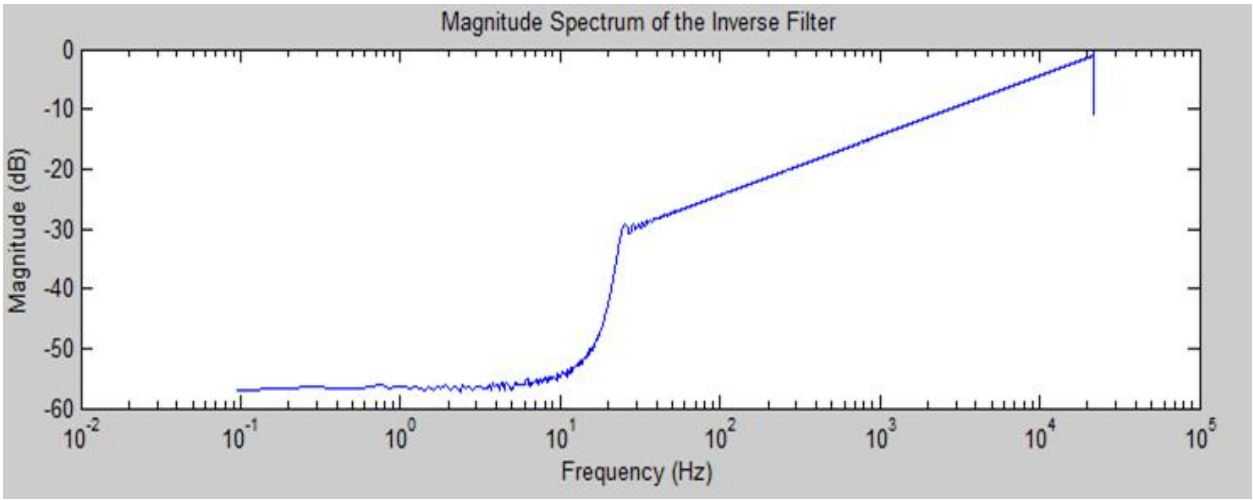


Figure 3: Magnitude response of $f(t)$, the inverse filter (+3dB per octave slope in the energy spectrum).

2.3. IR Separation from Nonlinear Response

Deconvolution is applied by convolving the recorded sweep $r(t)$ with the inverse filter $f(t)$, as in (6). However, equation (6) represents the ideal impulse response, absent from distortions. Since we are dealing with a physical system, there will also be an error/noise component that has entered the recorded signal. Major contributions to this error include non-ideal speaker response during room excitation (such as low frequency speaker distortion), and external noises occurring in the playback/recording environment. Considering these distortions, the true response that is recorded is a combination of distortion products and the linear response, $\tilde{r}(t)$. The resulting impulse response is:

$$\tilde{h}(t) = \tilde{r}(t) * f(t) \tag{9}$$

Defining ε as the combination of correlated and superimposed error that has accumulated in the system, (6) becomes:

$$\tilde{h}(t) = [r(t) * f(t)] + \varepsilon \tag{10}$$

The linear sine sweep method seeks to minimize error through synchronous averaging of multiple sweeps. However, using the ESS method of convolution with an inverse filter, synchronous averaging can be avoided. The convolution operation packs the linear response into a near-perfect impulse response, having a delay time equal to the length the reference sweep [1]. Additionally, the nonlinear response packs before the linear response in time. When the deconvolution result is plotted in the time domain, harmonic distortions are parallel to the linear response – they are oriented vertically, occurring as impulsive events at precise delay times (higher order harmonics occur earlier in time).

To demonstrate this, suppose that $r(t) = s(t)$ (the desired response is an exact copy of the reference sweep). Now, suppose $\tilde{r}(t)$ is a distorted version of the reference sweep $s(t)$. In this example, $s(t)$ has been clipped in Matlab (Appendix A), producing harmonic distortion in $\tilde{r}(t)$. Ideally, we would like to separate these nonlinearities from the linear response and obtain $r(t)$. The ideal impulse response in this example will be a delta function.

Figure 4 shows the results of the above test, which was conducted in Matlab. The nonlinear response can clearly be visualized as being separate from the linear response. Since these harmonic distortions occur before the IR, they can easily be removed by trimming the signal without affecting the linear response.

There was not sufficient memory on the computer system to perform the inverse filter computation in Matlab. Instead, an approximation of $f(t)$ was temporarily used for this test (Appendix A). For the example room impulse response measurement which will be discussed shortly, $f(t)$ was computed using the Aurora plugin, developed by Angelo Farina [1], [2].

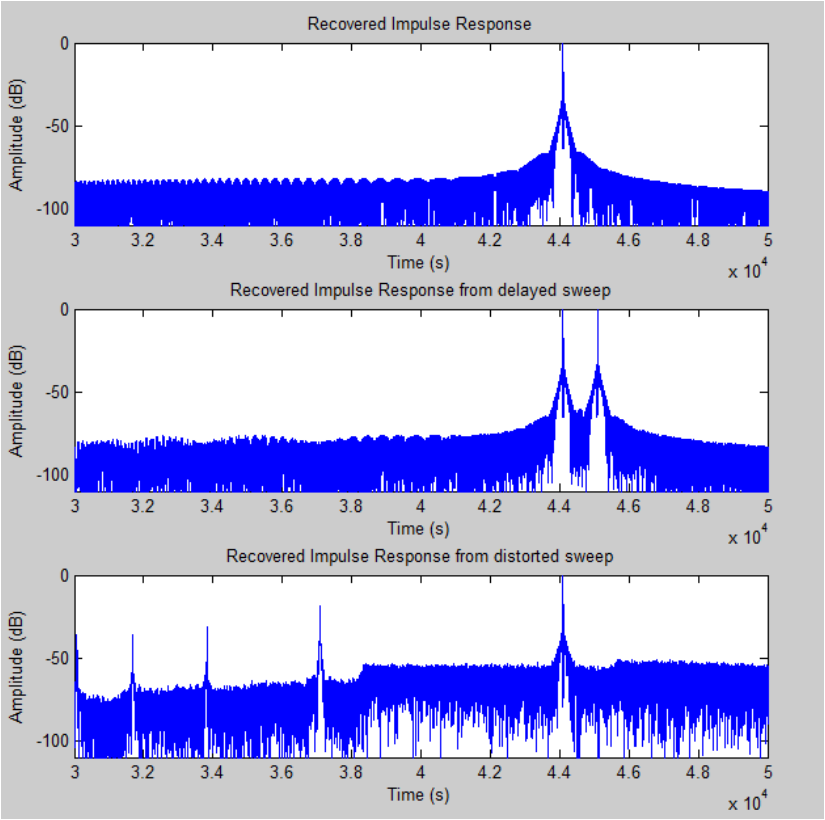


Figure 4: Verification of IR separation from nonlinear response using an inverse filter approximation (dB amplitude scale).

The odd noise floor in the above examples results from a crude inverse filter approximation in Matlab. As a result, this is purely a qualitative test: the example does illustrate the separation of harmonic distortion from the linear response. An example on a recorded test IR will be shown shortly, utilizing the correct inverse filter.

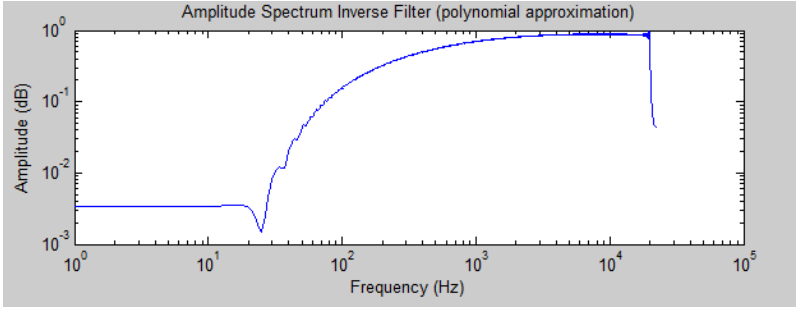


Figure 5: Crude approximation of inverse filter for above experiment.

3. Recording Setup and Technique

Recordings were made in the Philip T. Young (PTY) Recital Hall at the University of Victoria. The objective was to obtain a variety of high quality surround sound impulse responses in the hall. Key locations for the microphone array and speaker placement were chosen (7 microphone array positions in the audience, and 10 source locations on stage).

The acoustics (mainly the reverb) of the PTY hall can be adjusted by raising or lowering the banners and ceiling diffusers. Emphasis was placed on measuring the live hall (more reverb) as this data is more desirable for most IR applications. The ceiling diffusers were left in an intermediate position above the stage, and the banners were left in the second-highest position, providing a small amount of broadband absorption. At the back of the recital hall there is a poorly designed entrance corner that produces an undesirable acoustic effect called *flutter/slap echo* (rapid reflections in the high frequencies that occur between certain closely spaced walls). As a result, the absorptive banner in this corner was kept all the way down.

3.1. Routing and I/O for ESS IR Measurement

Suppose that the exponential sine sweep $s(t)$ (1) is played through a speaker to excite a room, and a measurement microphone records the room response, $r(t)$.

- $s_p(nT)$ is the exponential sweep signal that is played in the DAW (digital audio workstation).
- $s_e(t)$ is the exponential sweep that is played through the speaker, exciting the room. $s_e(t)$ is a delayed, analog version of $s_p(nT)$ that will have distortions due to imperfections in the speaker response.
- $s(nT)$ is the reference signal, which is achieved by routing $s_p(nT)$ through a delay path (the digital mixer) in order to synchronize in time with $s_e(t)$.
- $r(nT)$ is the recorded room response to $s_e(t)$ on one microphone channel

Clock synchronization between the reference signal and the recorded sweep is important to achieve a time-aligned measurement. However, the system is less dependent on tight clock synchronization that the linear sweep method, which uses synchronous averaging of multiple sweeps.

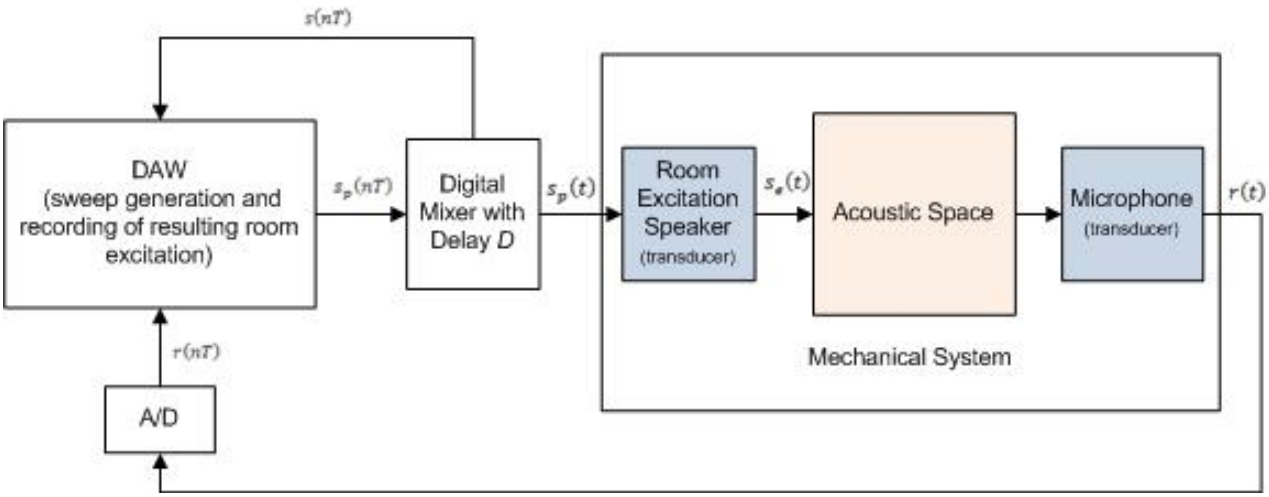


Figure 6: Signal Routing for Room Impulse Response Measurement (assuming one microphone channel). For multi-channel recording, the microphone channel represents a signal path leading from one of several microphones in an array.

3.2. Surround Sound/Spatial Microphone Array

A 1st order circular array of 5 near-coincident hypercardioids was used. Many modern multichannel applications have moved toward *Ambisonic* (also called B-format) surround/spatial recording due to the flexibility of encoding options. There is little documentation surrounding the use of circular microphone arrays for multichannel measurement.

The circular microphone array is designed to capture surround information through amplitude and timing differences between the microphones. The 5 microphones were oriented with each capsule tangential to a circle in the horizontal plane, with equal spacing between the microphones. This resulted in an angle of 72˚ between each pair of neighbouring capsules. The spacing between neighbouring capsules was chosen to be 19 cm.

Schoeps CMC 6-U small diaphragm condenser microphones were used. While not specifically designed for acoustic measurements, the Schoeps are regarded as some of the finest microphones for accurate reproduction as they exhibit a very flat frequency response. Two shotgun microphones oriented vertically (one pointing up, one pointing down) complete the microphone setup, allowing 3D spatial IR recordings. Unfortunately, constraints on the number of available ADAT channels in the digital recording setup prevented the use of these two microphones in the first two sessions. This can be remedied with an alternative scheme.

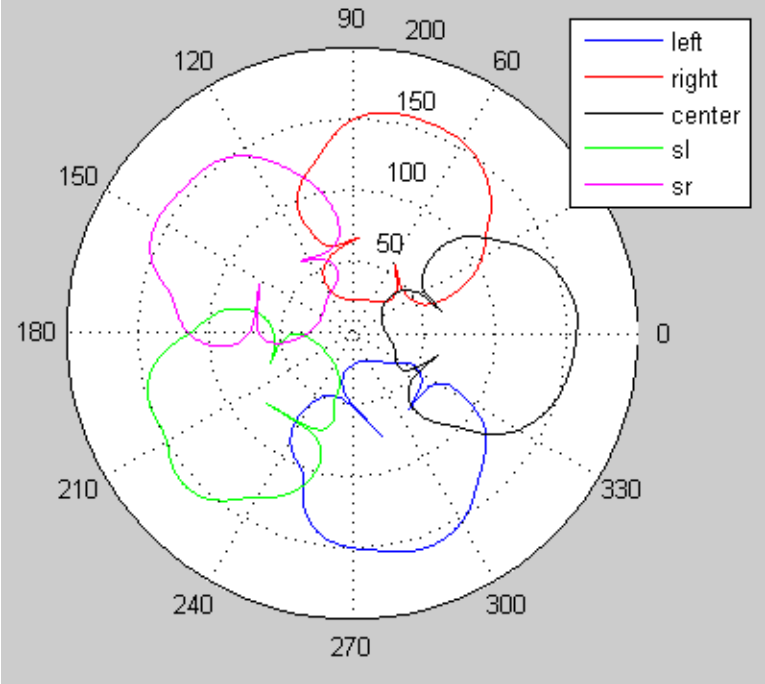


Figure 7: Polar pattern for the circular microphone array, representing the directional response characteristics in the horizontal plane of the 5 hypercardioid capsules. Measured by Yan Li [F3]. SL and SR represent the side back left and microphone, respectively, labelled according to the convention of a 5 channel surround sound setup, which uses "side" speakers as opposed to "back" speakers (discussed later).

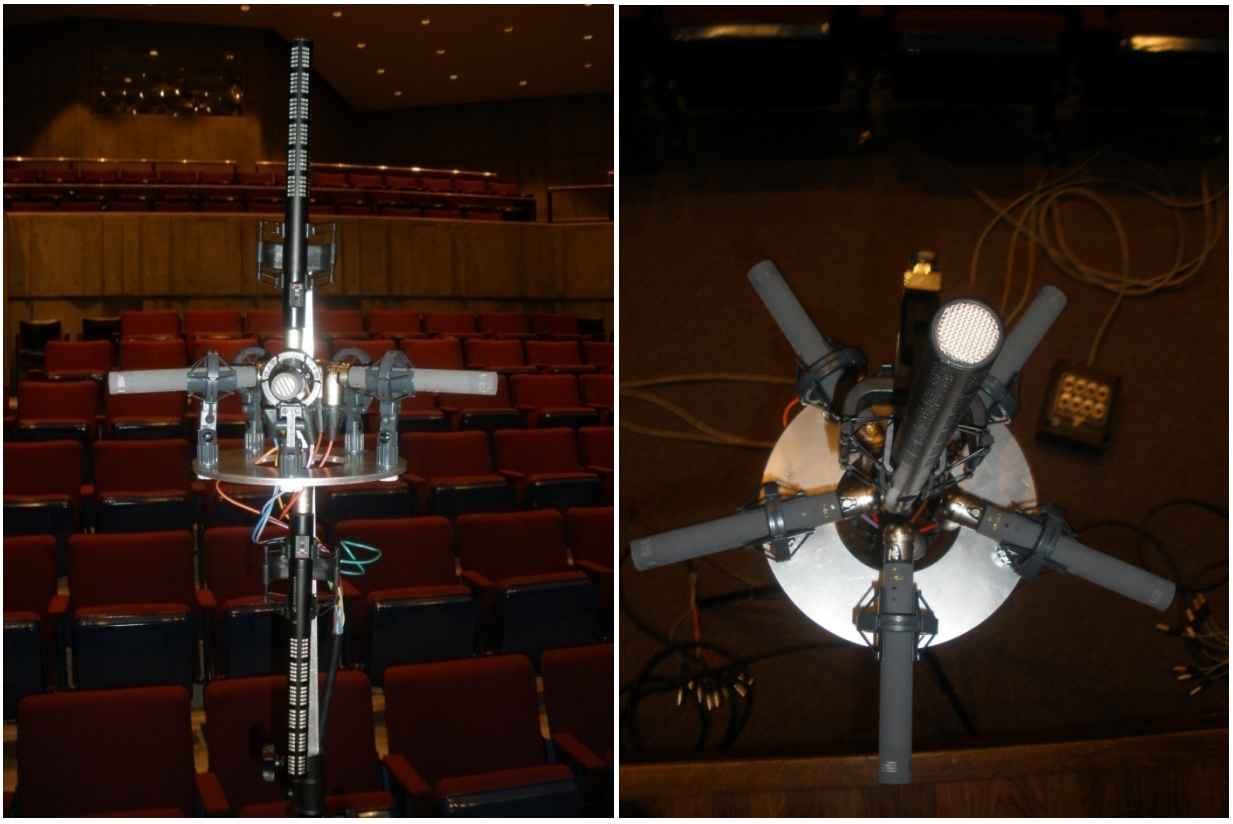


Figure 8: Front and top views of spatial microphone array.

Since hypercardioid pickup patterns are used, each microphone captures the greatest amplitude from an on-axis source (directly in front), and the least from a source directly behind it. Half way between each pair of capsules, the response of each microphone is 3dB down from its on-axis response. This translates to amplitude cues for surround sound reproduction, assuming that the speakers are positioned to compensate for the -3dB hole between pairs of microphone. Since the microphone capsules are spaced apart, sound from a source arrives earlier to capsules closer to the source, and later to capsules further from the source. The human ear picks up these timing cues upon playback, and the brain uses *interaural time differences* (ITD) between the two ears to form a preference for lateral direction. If a sound arrives at the right ear first, and 20ms later it arrives at the left ear with an equal amplitude, then the source will be perceived to the listener's left. This falls into the greater scheme of cross-correlation-based modeling of *binaural perception*, in which the brain uses dynamic/intensity cues, timing cues and frequency cues for 3D spatial localization. Additionally, the brain re-creates more subtle spatial characteristics such as ambience and listener envelopment.

There are many excellent surround sound recording techniques, including Ambisonics; however, the above method is perhaps the most intuitive as it is based on the same principles as near-coincident stereo microphone techniques. The human ear picks up amplitude and timing cues during playback, and the brain maps this into directional and spatial information through perception and cognition. Of course, humans have only two ears, not five. Therefore, surround sound perception must also rely on frequency cues; humans depend on these for localization of sounds behind us, and in the vertical plane. Naturally, to hear the results in surround sound using this method without advanced binaural modeling algorithms, the listener should be surrounded by speakers (this is not *binaural recording*, which directly allows 3D perception through headphones). The spatial localization for surround playback will depend on successful encoding of the audio for 5 channel surround sound, and the speaker locations in the listening environment. Spatial impulse response rendering (SIIR) is necessary for reproduction of 7-channel 3D impulse responses, which may be used for simulation-based research. 3D playback is possible with the correct 7-channel spatial speaker layout.

3.3. Location Map for Source and Array

The locations for RIR measurements in PTY hall were selected based on acoustics, and typical performer-audience spatial relationships. 7 locations for the microphone array were chosen in the audience region. One of these 7 locations was the hanging microphone position just in front of the stage, which is frequently used when recording performances using stereo microphone techniques. 10 speaker (source) locations were chosen on stage, representing typical performer positions. The positions near the front of the stage were chosen to be slightly back from where a live performer might be located; this was done in order to take advantage of the diffusers that are located above the stage (often referred to as “clouds”). The diffusers are designed to break up and scatter first reflections, improving the texture of the reverb. An undesirable texture, in the extreme case, might sound like an echo in a tunnel; a desirable reverb texture is typically lush with a diffuse field of reflections, giving the impression of ambience.

The source positions Qfl, Qbl, Qbr and Qfr represent a string quartet layout, respectively: first violin, second violin, viola and cello. Other logical sets of positions can be combined to represent multiple performers, which will be discussed later in application with convolution reverb. Additionally, a grand piano may be reasonable represented by combining close sets of position, such as Qbl and Qbr. This is dependent on how the dry piano was originally recorded, however, and will be addressed later.

To allow for placement of the sound source and choice of listening position (to be utilized in a convolution reverb), room excitation was performed from each of the 10 source locations, and recorded at 7 listener locations. This result is 70 distinct source-listener combinations, each represented by a 5 channel MMRIR.

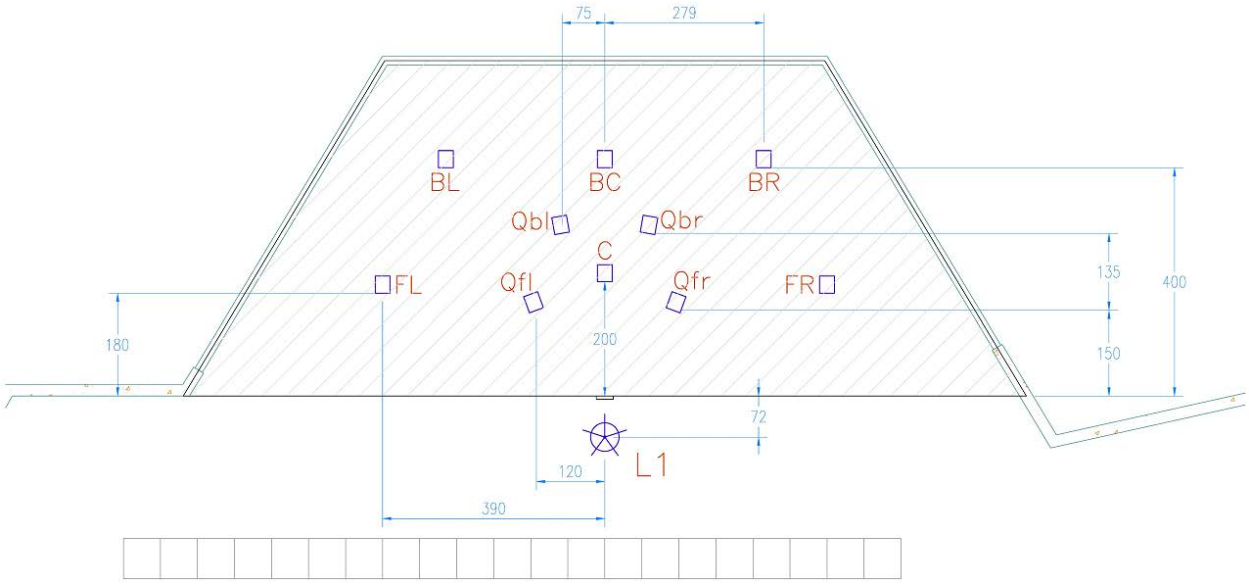


Figure 9: Stage locations for room excitation source (speaker). Microphone location L1 is also shown.

SOURCE POSITIONS (stage)	
Position	Description
C	Center (soloist)
Qfl	String Quartet - 1st Violin
Qbl	String Quartet - 2nd Violin
Qbr	String Quartet - Viola
Qfr	String Quartet - Cello
FL	Front Left
FR	Front Right
BL	Back Left
BC	Back Center
BR	Back Right

MIC ARRAY LOCATIONS (audience)		
Location	Description	Elevation Above Stage (cm)
L1	Hanging Mics Default	206
L2	3rd Row Left	256
L3	3rd Row Center	256
L4	3rd Row Right	316
L5	Sub-Balcony Center	346
L6	8th Row Left	556
L7	8th Row Center	556

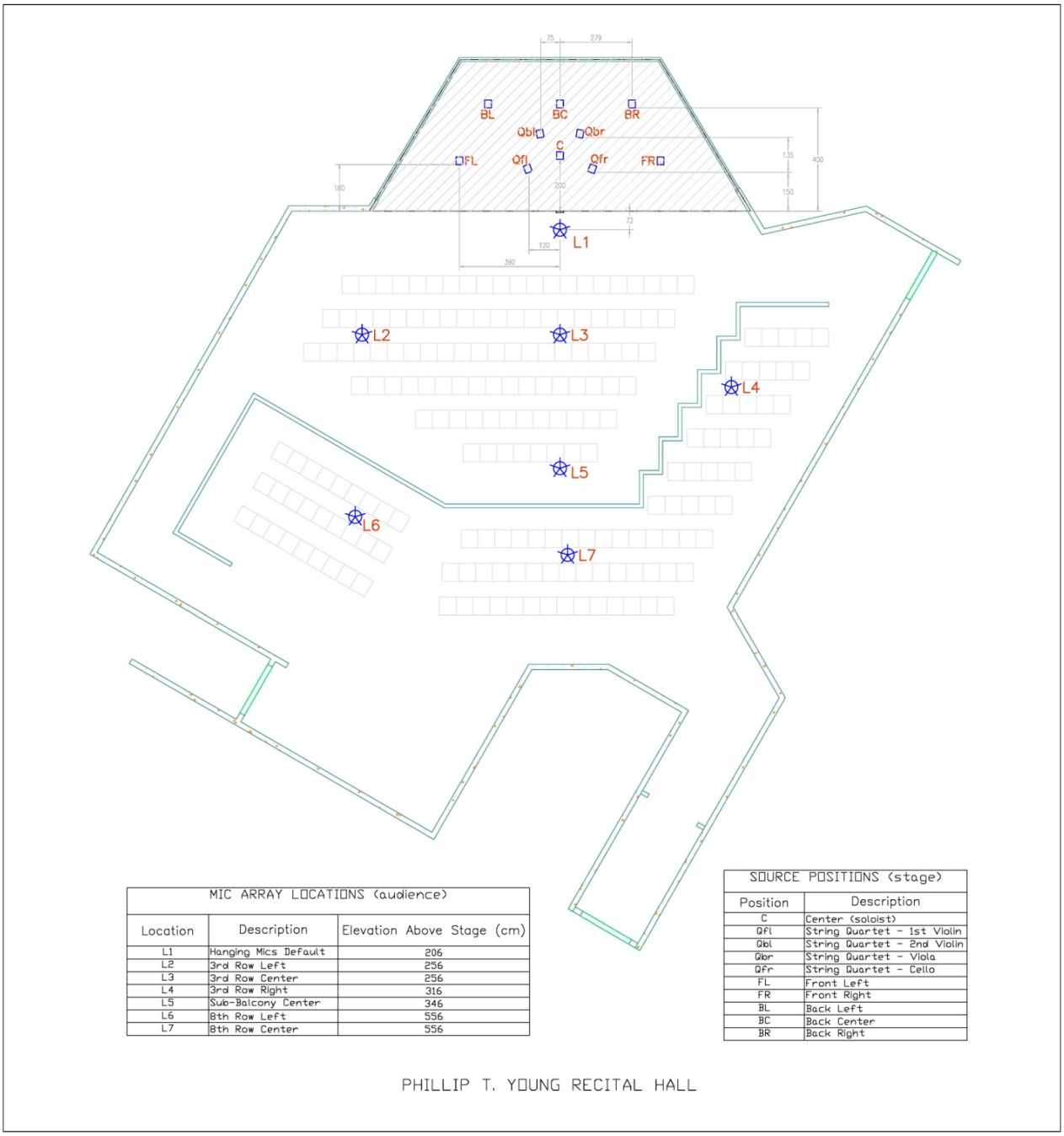


Figure 10: Source positions and microphone array locations for surround sound IR recordings in the Phillip T. Young Recital Hall.

Figure 11: Spatial microphone array (height has been lowered for setup) with room excitation source (speaker) in background.

Figure 12: Microphone array in PTY recital hall (Location L7).

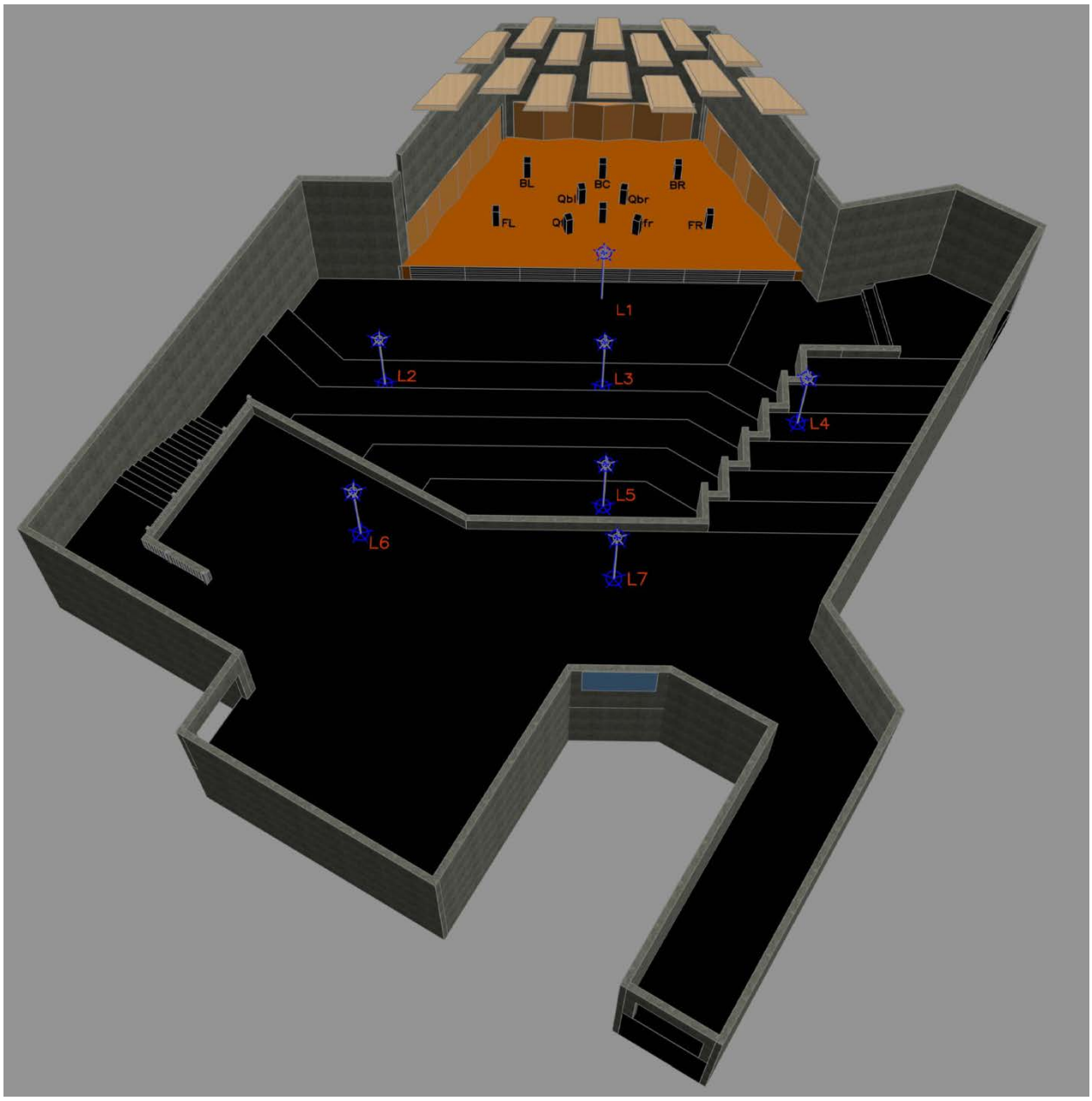


Figure 13: Perspective view of PTY Recital Hall looking toward the stage from a high vantage point. Microphone array and source locations are displayed. All available elevation data from the original architectural drawings [10] was taken into consideration during modeling.

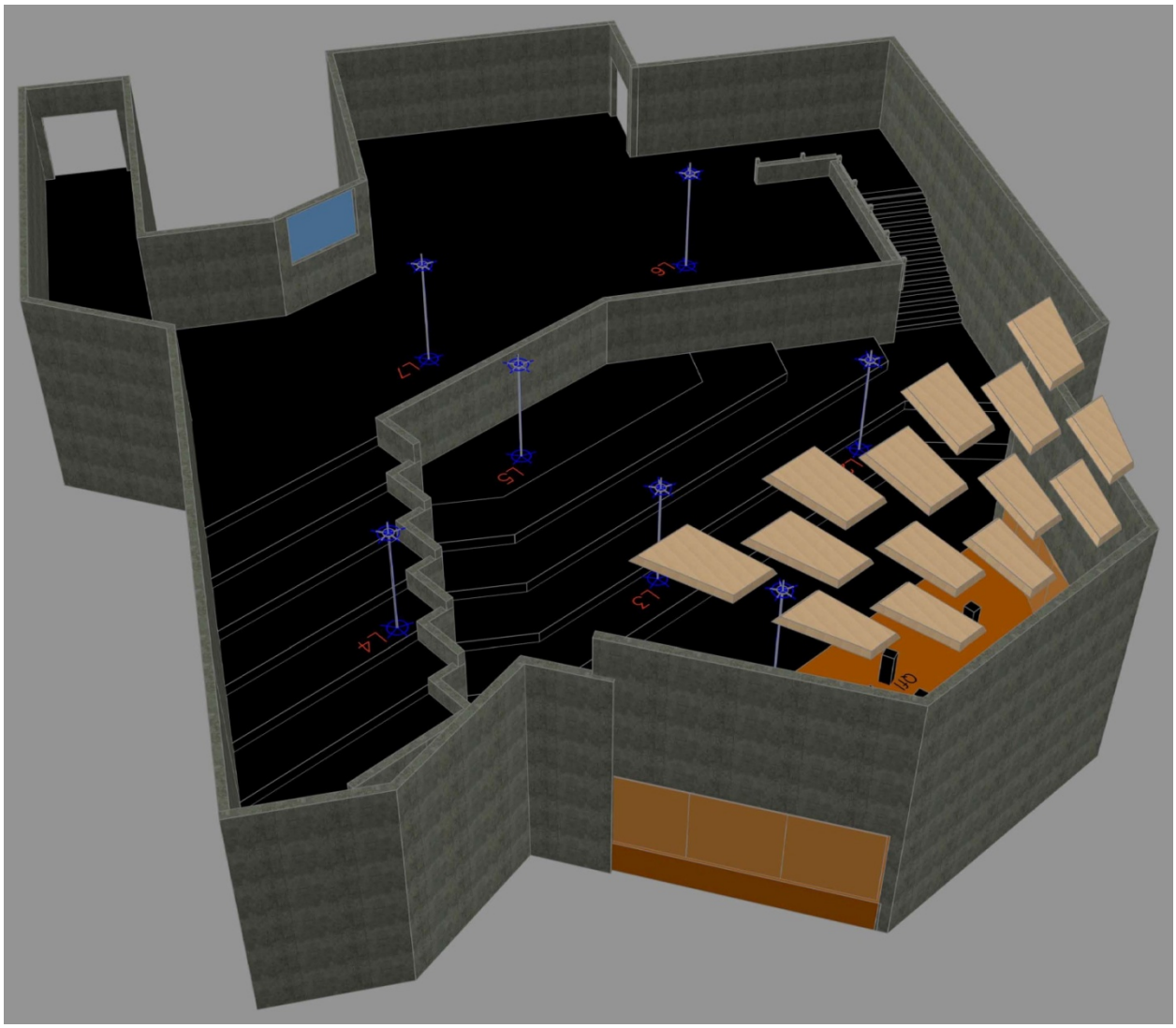


Figure 14: Perspective view of PTY Recital Hall illustrating microphone array locations on the various audience levels. All available elevation data from the original architectural drawings [10] was taken into consideration during modeling.

4. Example RIR Using the EES Method

Using the ESS method, high quality impulse responses were measured in the Philip T Young Recital Hall with the 5 channel surround sound microphone array. This section will go over the extraction of a single impulse response over one microphone channel, based on the theory discussed earlier.

4.1. Exponential Sine Sweep

The upper and lower extremities of the frequency range for the exponential sweep were chosen to be:

$$f_1 = 22\ Hz \qquad\qquad f_2 = 22\,000\ Hz$$

Ideally, this frequency range should have a span somewhat greater than the audible frequency range in humans (20 Hz to 20 000 Hz). This is because some subtle perceptual information involved in “hearing” is not actually heard by the ear, but perceived through other senses (this explains why audio played back at 96 kHz or 192 kHz seems to have greater listener envelopment and sounds “higher definition” than at 44.1 kHz). Unfortunately, the loudspeaker that was used for room excitation had a limited low end frequency response. Audible frequencies below ~36 Hz (in addition to inaudible frequencies below 20 Hz that cannot be heard, but can be felt) did not contribute to the room excitation.

Figure 15 shows the spectrum of the exponential sweep as a function of time as displayed in Adobe Audition. Figure 16 shows the spectrum of the recorded sweep as a function of time. Harmonics show up as the curved lines above the sweep tone. Clearly, harmonic distortion is occurring severely in the lower end of the speaker response, and again in the mid-frequencies. Listening back to the recording of the sweep, it can be confirmed that there is a nonlinear speaker response in the low end. Moreover, in this particular sweep recording, it sounds as if the speaker is causing excitation of higher order harmonics as it passes over the resonant frequency of the speaker stand. This is shown by the multiple lines above the fundamental in the mid frequencies. Finally, a first-order harmonic (an octave above the swept tone) appears above the fundamental for the duration of the sweep; however, it cuts off at the Nyquist frequency in the recording.

Figure 17 show the waveforms for the reference sweep and the recorded hall response, respectively. The waveform of the recorded signal is noticeably longer than the reference sweep due to reverb in the hall.

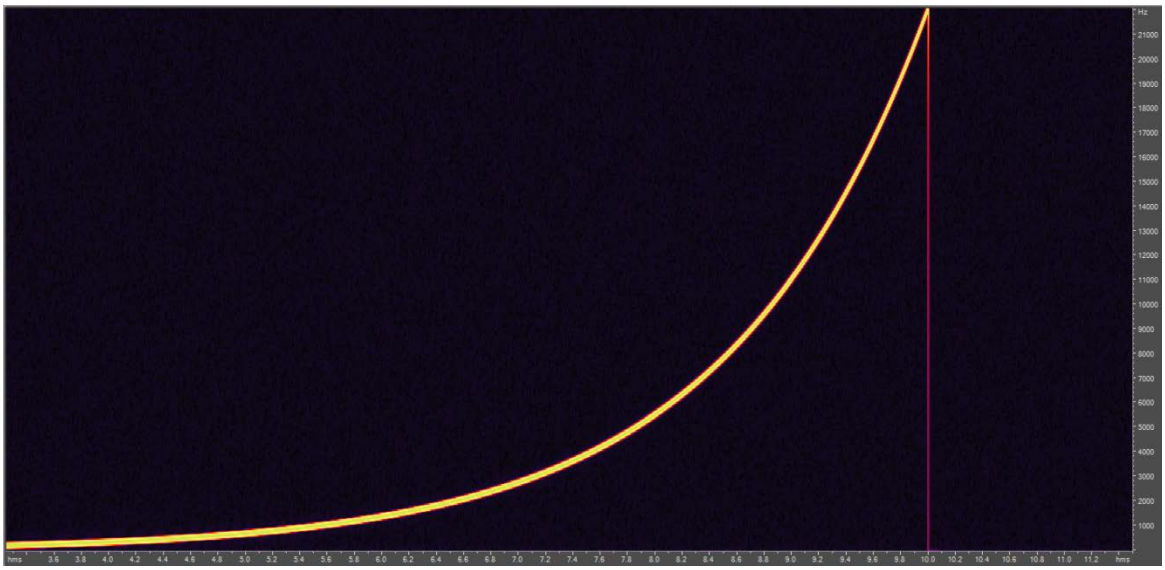


Figure 15: Colour coded amplitude spectrum of reference exponential sine sweep (frequency as a function of time). Yellow represents a signal amplitude near-unity.

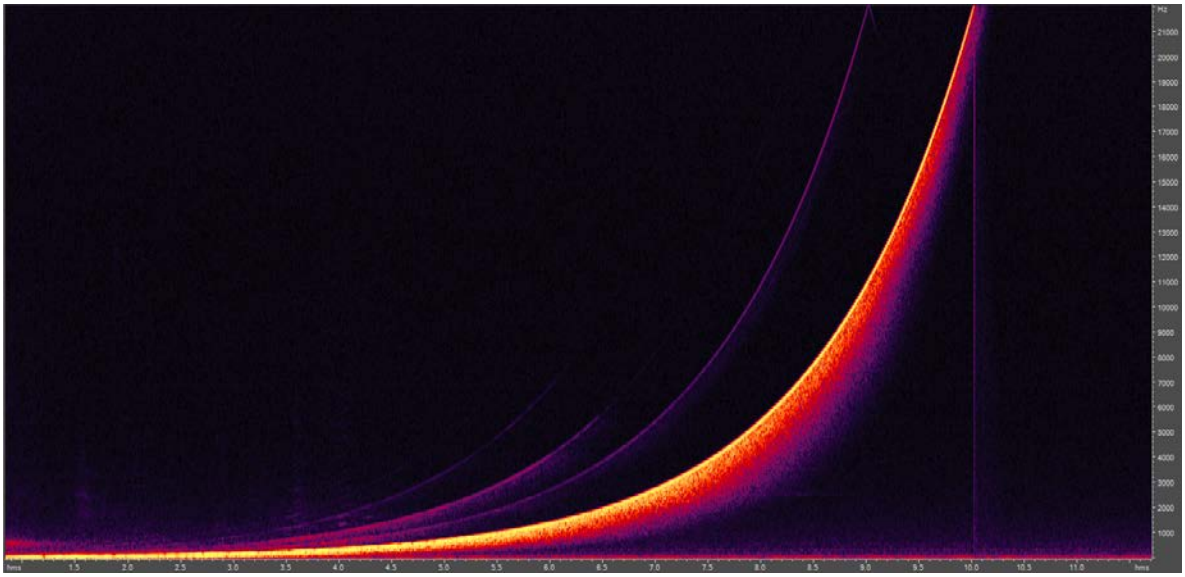


Figure 16: Colour coded amplitude spectrum of recorded ESS response in PTY Recital Hall (center microphone channel) shown with respect to time. Amplitude is indicated by relative brightness; bright yellow corresponding to an amplitude near-unity.

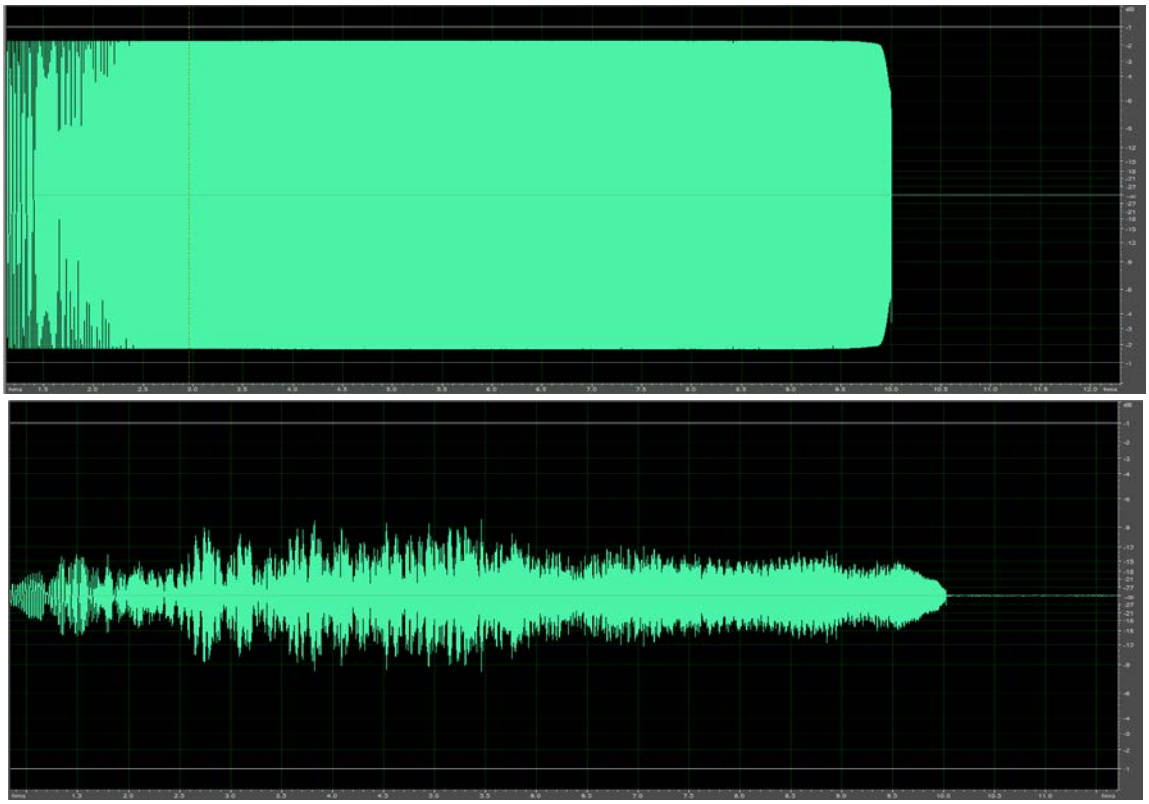


Figure 17: Waveform of reference exponential sine sweep (top); waveform of ESS response as recorded in PTY recital hall (center microphone channel). The reference sweep is 10 seconds long

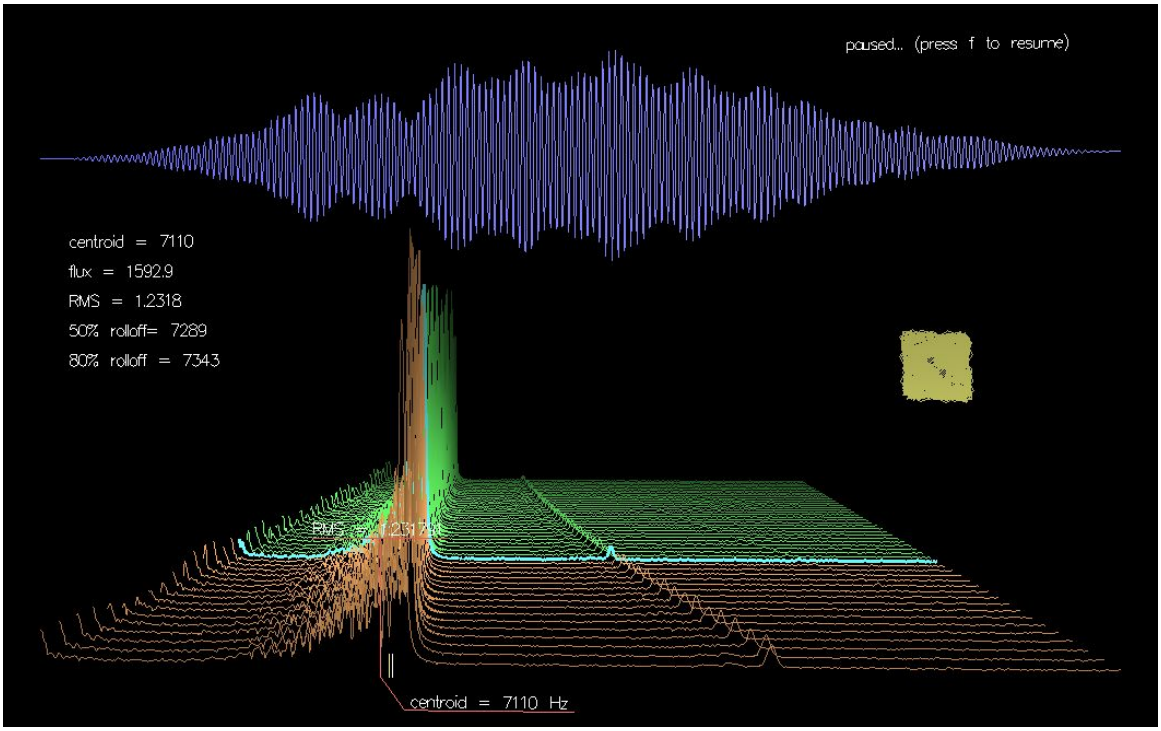


Figure 18: Real-time spectrum capture of recorded room excitation (time-frequency waterfall representation in *sndpeek* [6]). Waveform is shown on top, real-time spectrum amplitude (linear scale) is shown below; a distortion harmonic is clearly visible in the spectrum. -

4.2. Inverse Filtering Using Aurora

The Aurora plug-in Adobe Audition was created by Angelino Farina, the author of [1] and [2]. Farina has made significant contributions to the ESS technique for impulse response measurements; his Aurora suite for acoustic measurements has seen documented use in many recent scientific publications. The inverse filter $f(t)$ for this test RIR was computed in Aurora. Farina has implemented an efficient algorithm to obtain a close approximation of the inverse filter for practical RIR measurement. Using this resource, we were able to avoid solving the Toeplitz matrix that surfaces when amplitude modulating the time-reversed ESS signal.

Figure 19 compares the spectrum of the exponential sweep, the spectrum of the recorded signal, and the spectrum of the inverse filter. The $+3dB/octave$ slope of the inverse filter spectrum and the approximate $-3dB/octave$ slope of the recorded signal's spectral amplitude envelope are significant. This is designed to compensate for the energy differences at each frequency, as the sweep has a pink spectrum. Multiplying these amplitude spectrums (frequency domain multiplication of each signal's FFT is more efficient here than time domain convolution) will result in a flat spectrogram.

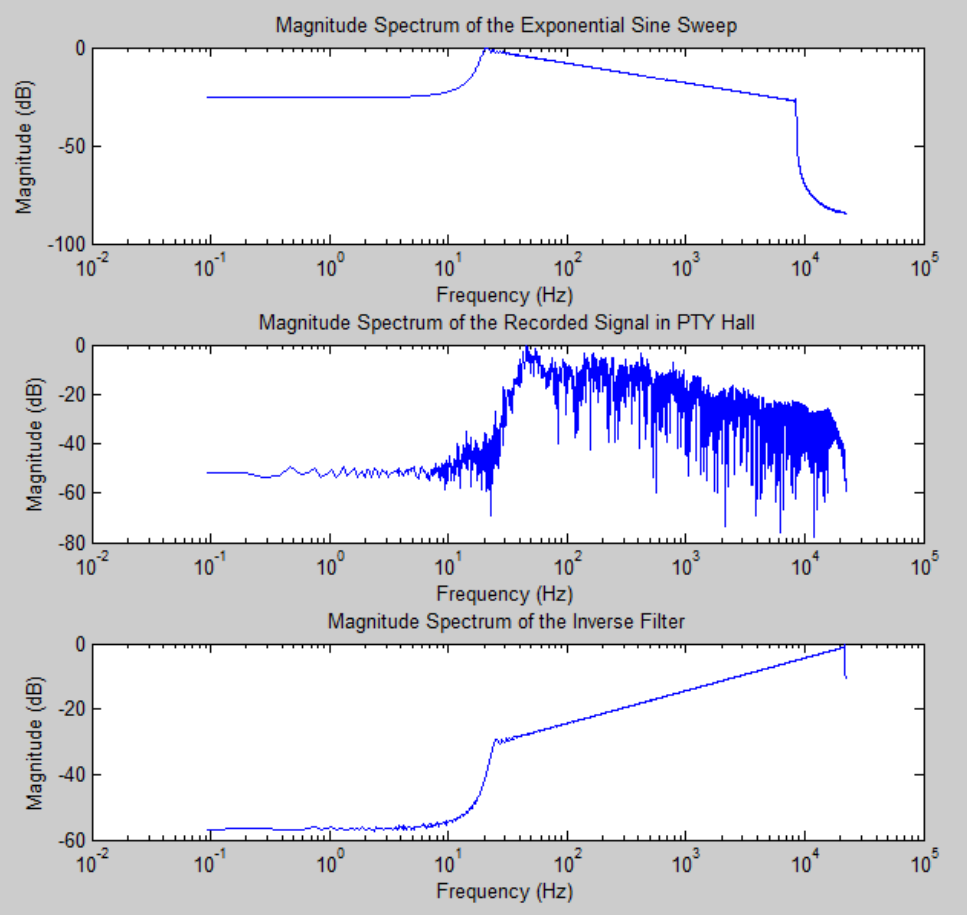


Figure 19: Magnitude response of ESS signal (top), recorded signal (center), inverse filter (bottom).

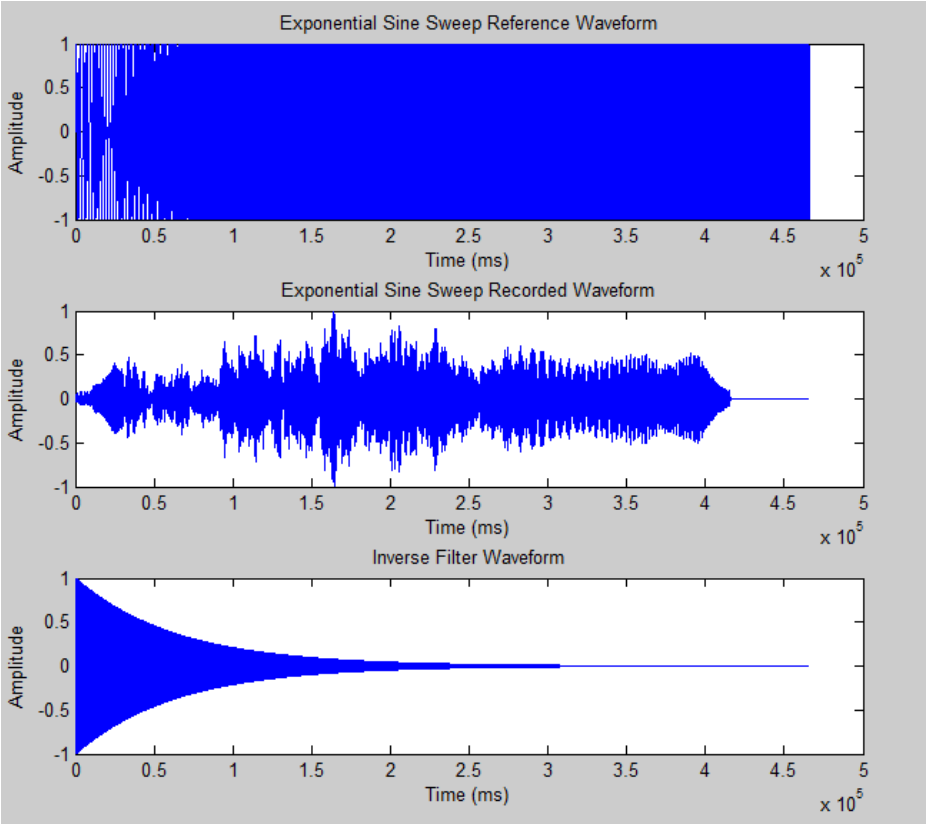


Figure 20: ESS signal (top), recorded waveform (center) and inverse filter waveform (bottom).

4.3. Impulse Response Extraction (center microphone channel)

The impulse response of the test signal was extracted in Matlab using FFT convolution (Appendix B). It was also extracted in Aurora. Clean separation of the harmonic distortion from the linear response is evident in both cases. The nonlinear response appears as a collection of vertical spikes (distortion harmonics) to the left of the linear response, therefore occurring earlier in time.

There is some ambiguity in the signal to noise ratio between the Matlab extracted IR and the Aurora IR. The IR that was extracted in Matlab indicates a noise floor with peaks in the range of -96 dB to -97 dB. This corresponds to an estimated S/N of approximately 96 dB.

The Aurora test IR shows a near-flat noise floor at about -84 dB. It appears that Aurora has applied dithering and possibly noise shaping to create a consistent noise floor directly around the IR. To utilize the IR in this case, it will require windowing the linear response after the harmonic distortions and before the noise floor begins to fluctuate rapidly. The difference in noise floor, however, is most likely due to bit rate conversions that resulting in successive ditherings of the Aurora IR. The Matlab

test IR was computed in 24 bit resolution. The Aurora IR was computed using 16 bit audio resolution; however, 32 bit processing was available in Adobe Audition, and that extra dynamic range should have been utilized. The possibility exists to increase the SNR further by conducting critical processing (inverse filtering and convolution with the inverse filter) using the Aurora plugin in Audition at 32 bit resolution.

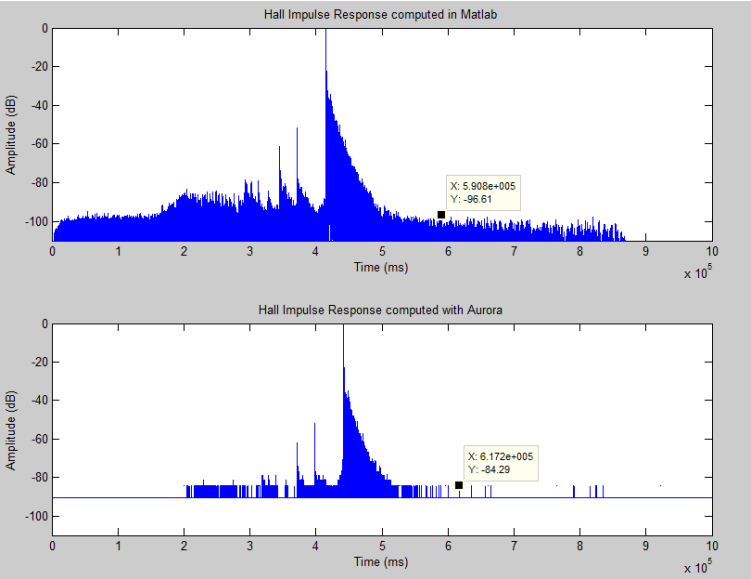


Figure 21: Extracted impulse response using Matlab (top) and Aurora (bottom).

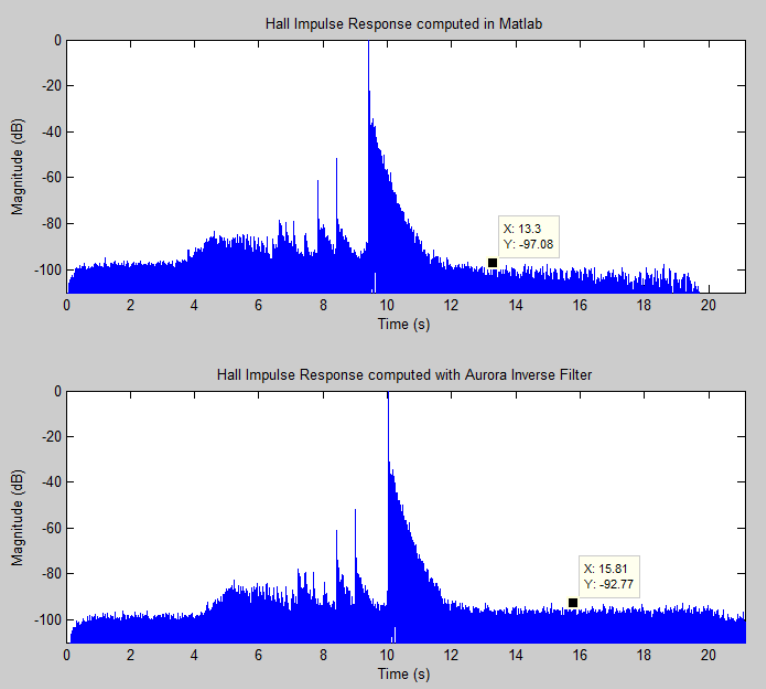


Figure 22: Extracted impulse response using Matlab (top) and Aurora (bottom). This Aurora computation used a 16-32 bit converted recorded signal convolved with a 32 bit inverse filter (calculated with a 32 bit

sweep)

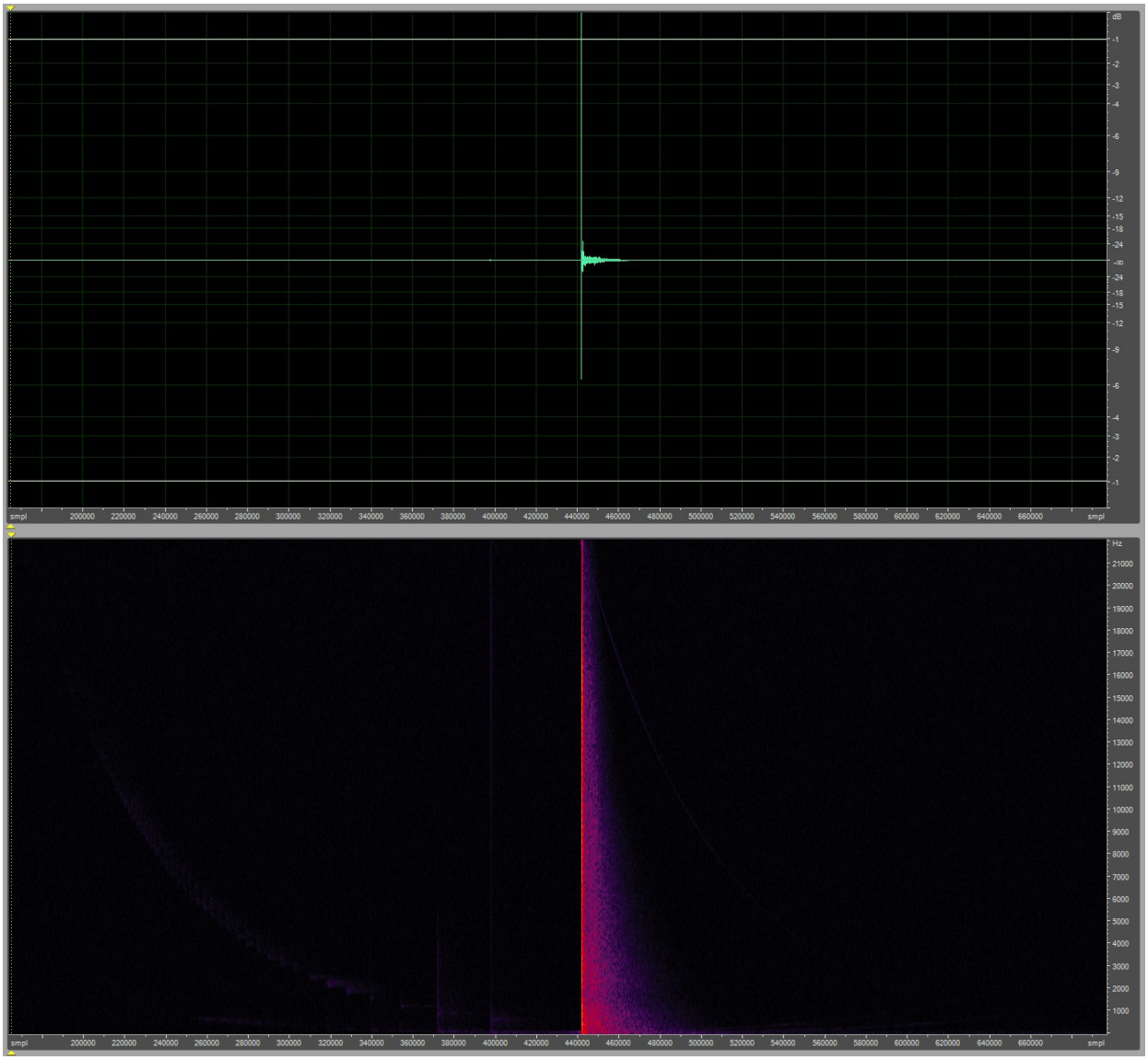


Figure 23: Test IR waveform (top) and colour coded aplitude spectrum as a function of time (bottom) as viewed in Audition.

Viewing the spectrum as a function of time in Audition, it is clear that an impulse (energy at all frequencies) occurs, followed by rapid attenuation of energy in the high end of the spectrum. As the reverb tail settles, it changes in timbre, losing richness as the mid and high frequencies die off in energy. Due to the size of the hall, resonant modes and modal decay do not have a significant impact on the IR within the spectrum that humans can hear. While the RIR can reasonably modeled as a fixed FIR filter (if we treat the acoustic space as time-invariant, which in truth it is not), the relative amplitude spectrum when viewed on a time-frequency representation does appear to change. This is because each increment in time (if we consider time to be discrete), marks a successive application of the filter to whatever signal it is filtering. For the ideal IR representation, the initial filtering is performed on a unit impulse. The next filtering is applied on the signal that follows the impulse in

time – which in the case of an acoustic space where the IR was measured, is the response to the previous filtering. The physical system in fact behaves as an IIR filter. Frequencies that have lower amplitude response values will die off in energy more quickly, as the acoustic environment is a system with feedback.

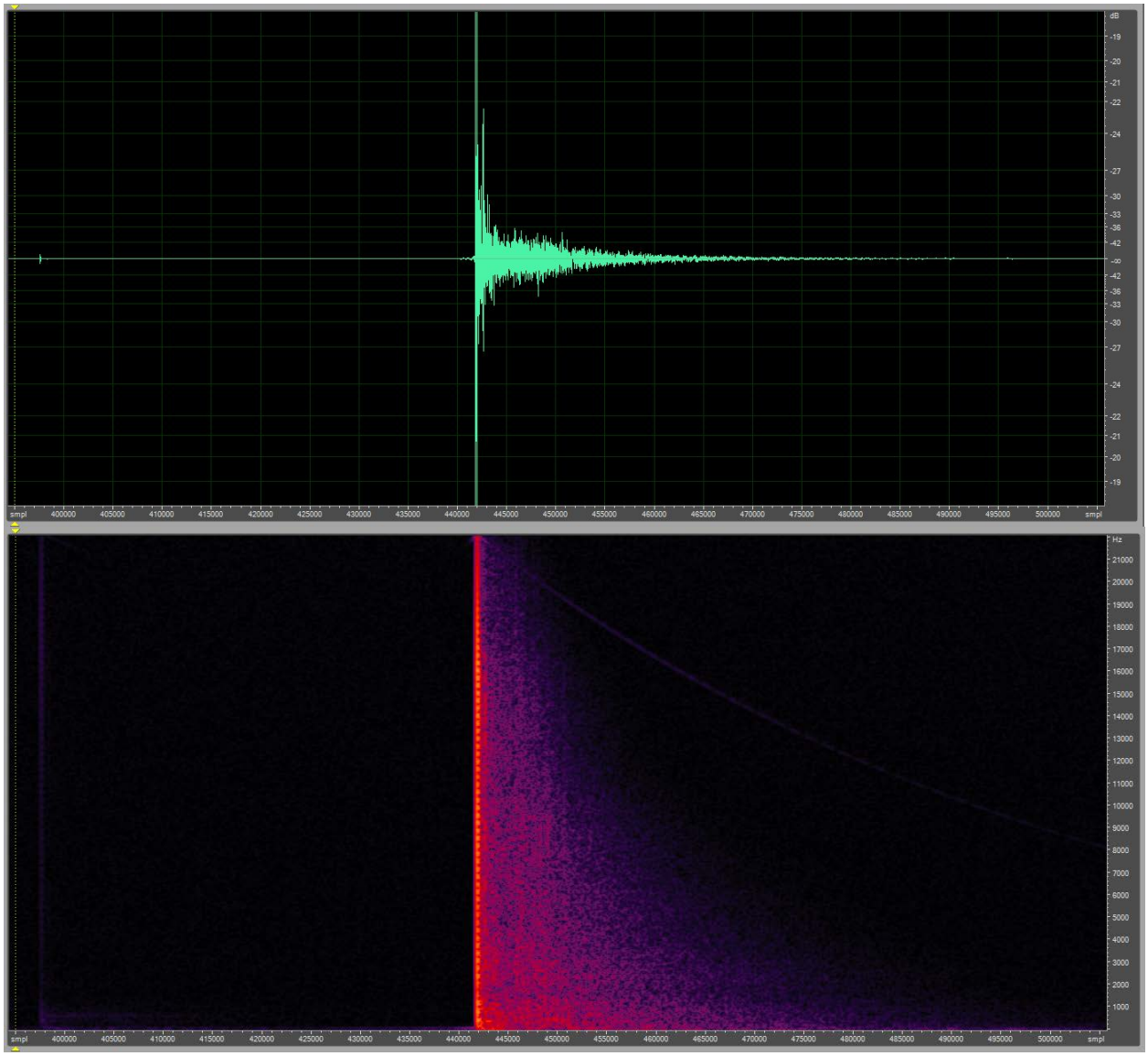


Figure 24: A closer view of Figure 23 shows a harmonic distortion peak that has been packed before the linear response in. The corresponding spectrum-time plot shows a faint vertical line (broadband energy) at the same instant in time.

4.4. SNR of Exponential vs. Linear Sine Sweep Measurements

Prior IR measurements have been conducted in the PTY Recital Hall using the linear sine sweep method. A comparison is performed below between the ESS test IR and an IR that was measured in 2004 [7]. Linear sine sweeps were used, and the IR was extracted by computing the cross-correlation of the recorded response with a synchronized reference sweep. The IR selected has the greatest energy in the lower octaves of the batch, and was recorded close to the source. A strong early reflection can be heard, and shows up in the IR plot.

Figure 25 shows that the SNR is significantly less for the IR that was measured using the linear sine sweep method, estimated to be about -56 dB. While test conditions cannot be directly compared, it is evident that the ESS method does offer a significant improvement in signal to noise ratio. This will be explored more in the next section.

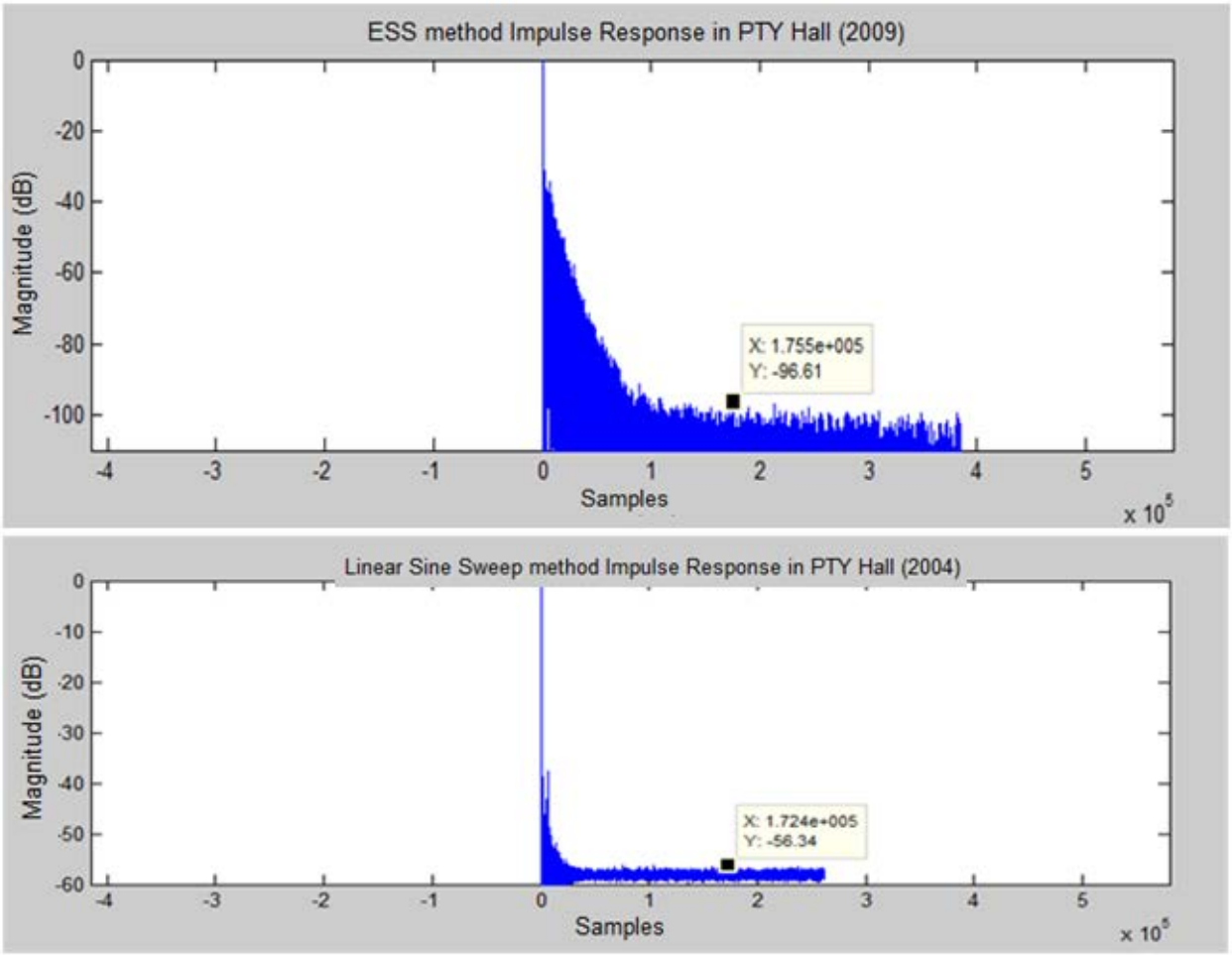


Figure 25: SNR comparison of ESS method against the linear sine sweep method if IR measurement. The linear sweep measurement was done in PTY Hall in 2004, but under different conditions.

The low end response is also better captured by using an exponential sweep as the room excitation signal. The ESS signal sweeps slowly through the low end of the spectrum, where harmonics are more tightly spaced if viewed on a linear scale. As a result, the speaker has time to respond to the excitation signal in the low end. The linear sine swept signal, on the other hand, may pass through the lower octaves as such a rate that the speaker can't physically keep up. This is further enhanced by the fact that the ESS method uses a long sweep, and the linear sine sweep method uses multiple sweeps back to back for synchronous averaging, which are typically short in duration. Listening back to the impulse responses, the ESS method IR has a low end punch that the linear sine sweep method IR lacks. This can be verified by plotting the spectrograms of both impulse responses to reveal the energy in the lower octaves. While the low frequency spectrum is expected to differ for two impulse responses under different test conditions, the lowest frequency represented in each IR (regardless of amplitude) can be used as a source of comparison.

The apparently shorter reverb tail in the linear sine sweep method IR is partly due to the reverb tail blending into the noise floor, and perhaps in part due to greater absorption coefficients in the hall, since the acoustics can be adjusted. It is interesting to note that both IRs have a clear early reflection at a very similar time, with an attenuation of about -35 dB. This suggests that while the microphone array was likely in a similar position, the speaker was in a different position relative to the array (closer, as can be heard by listening to the timbre of the IR). Zooming in closer reveals that that the initial time delay gap for the ESS measurement is smaller, and the magnitude of the first reflection is about 10 dB higher. In the ESS case, this reflection is most likely from the wooden stage banners, as the speaker was in position BC for this measurement (near the back of the stage).

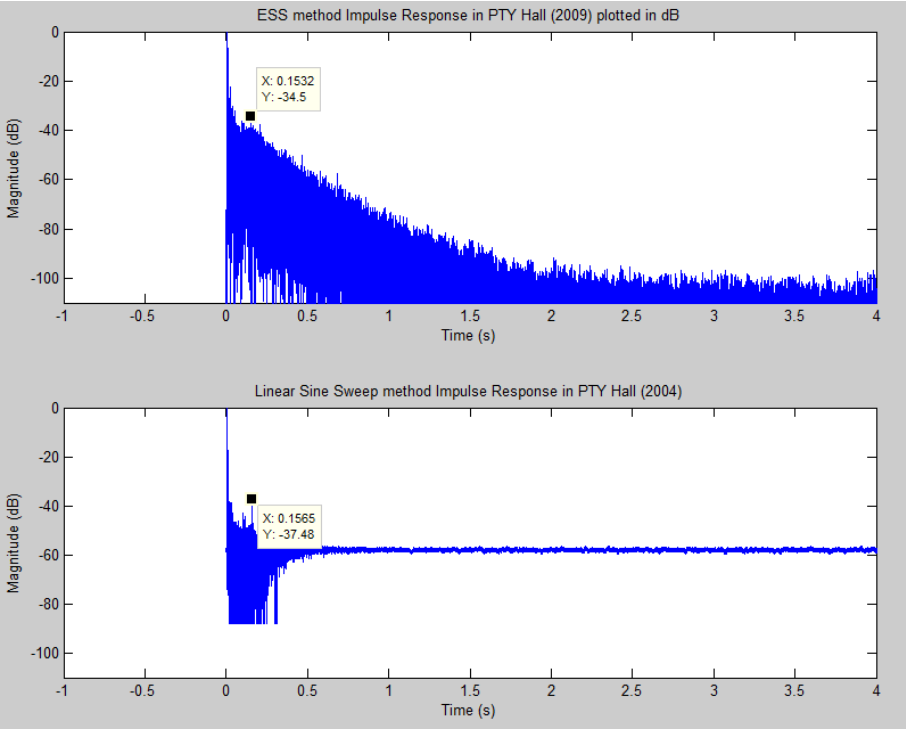


Figure 26: Comparison of early reflections with the linear sine sweep measured IR recorded in 2004.

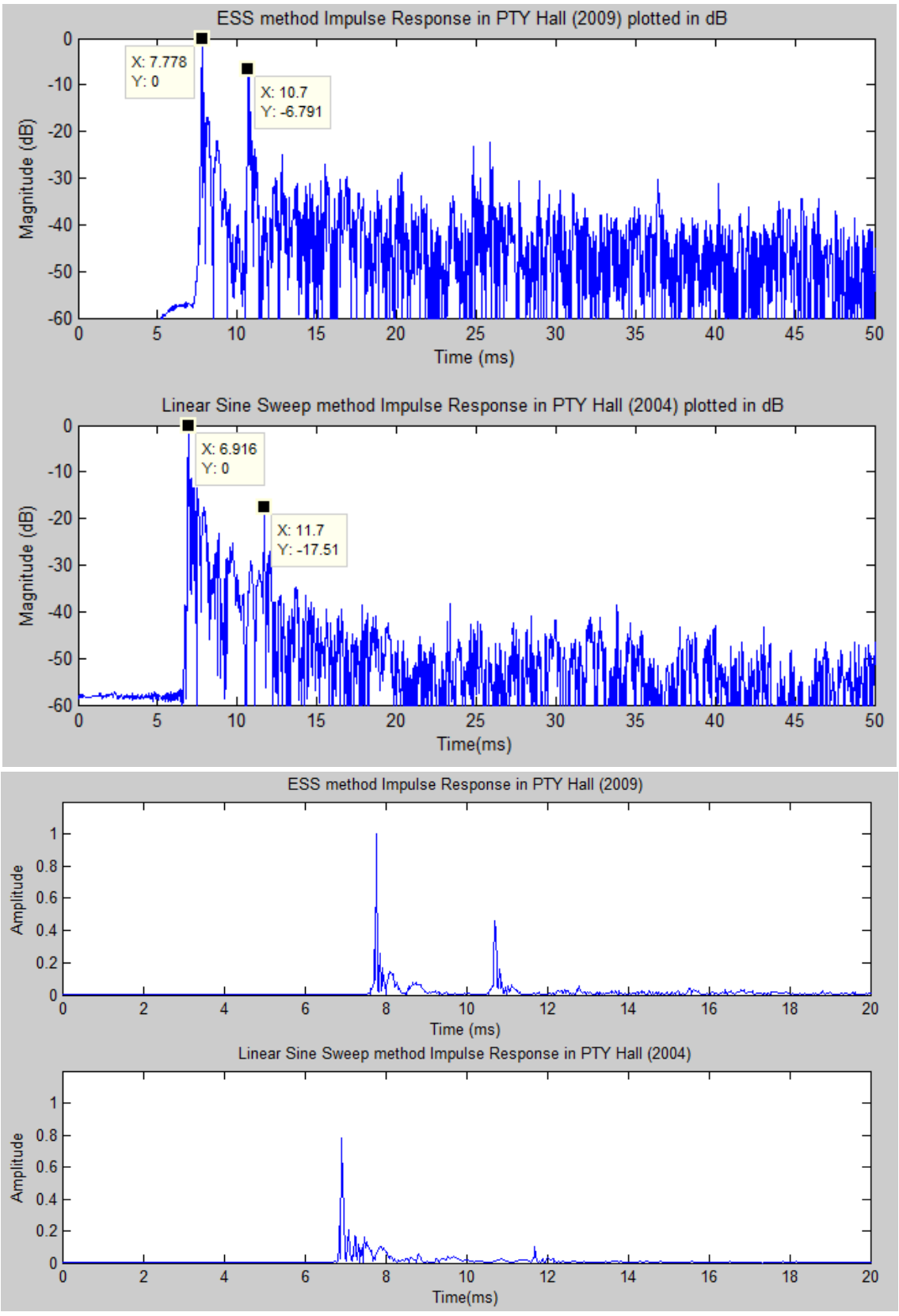


Figure 27: Comparison of early reflections with the linear sine sweep measured impulse response recorded in 2004; magnitude plotted in dB (top), IR amplitude plotted on zoomed timescale for both IRs (bottom).

5. Further Investigations of Dynamic Range

Deconvolution by convolving the excitation response $r(n)$ with the inverse filter $f(n)$ was performed in Matlab using FFT convolution (frequency domain multiplication of each signal's FFT, followed by the IFFT to bring the result back into the time domain). In order to avoid time domain aliasing, $r(n)$ and $f(n)$ were both zero padded before the operation. Several of the resulting IRs were analyzed using a phase vocoder that was specifically implemented with a function to plot the magnitude as a function of time and frequency (Appendix C).

5.1. Time-Frequency Representation and SNR Evaluation

The time-frequency representation of each impulse response (*waterfall plot*) gives further insight into the frequency response, dynamic range, broadband reflection/abortion, and reverb across the spectrum. Modal decay is less important here, as the dimensions of PTY hall place any widely spaced modal peaks below at low frequencies outside of the human hearing spectrum; additionally, the speaker did not excite frequencies below low end of the hearing spectrum in humans.

Each extracted IR was trimmed to remove the harmonic artifacts before the initial impulse, and to remove extranneous samples after the noise floor has been hit. The resulting IR length was initially designated to be 3 seconds, but was later increased to 4 seconds. Viewing the waterfall representation of the IR, it is apparent that the deconvolution operation using FFT convolution with the inverse filter has had a denoising effect on the spectrum. While the "rift valley" shaping in Figure 28 (trimmed to 3 seconds) does appear to cut off some of the high frequency response shortly after the initial impulse, it is in general sympathetic with the reverb tail across the spectrum. The reverb tail hits the noise floor sooner at high frequencies than at low frequencies. As discussed earlier, the timbre changes with time due to the variation in decay rate for energy accross the spectrum.

The apparent surface discontinuity of the time-frequency represenation is a natural result of the deconvolution operation. The cutoff frequency corresponds to the instantaneous frequency of the inverse filter, with the time axis normalized to the IR length (so, a time compressed inverse filter that retains the original pitch, from 22 kHz to 20 Hz). When time domain convolution is performed with the inverse filter (shown later) we get a similar effect; however, the rolloff is slightly smoother.

The above observation suggests that the denoising trend accross the spectrum is advantagous in increasing the dynamic range as time progresses. The effect is lowpass filtering with a sharp rolloff and a cutoff frequency that changes with time – resembing a form of dynamic lowpass filtering. Ideally, this cutoff frequency will arrive at a specific frequency at the instant in time

when that energy at that frequency hits the noise floor. At this istant, the noise floor will be cut out at that frequency, boosting the overall SNR of the signal averaged over the spectrum. Ideally the rolloff will be more gradual. For IRs with a lower dynamic range and a clearly audible noise floor, the sudden cutoff will percieved by a listener.

The above also suggests that the SNR findings from preliminary tests may not provide an accurate repesentation of dynamic range. According to the time domain plots of a number of measured IRs, the noise floor after the reverb tail has settled is below -105 dB. This represents the noise floor at that instant, and not the dynamic range during the early stages of the impulse response. For a fair estimation of dynamic range, an IR without this shaping artifact should be viewed.

Comparing the waterfall plots of the measured IRs with the sample linear sine swept IR measurement from 2004, we find further possible results in favour of a long sweep with a logarithmic change in frequency: the ESS signal has a more consistent broadband response at the instant of the impulse than the linear sine sweep. However, a closer look reveals that variations in the ESS computed IR spectrum occur at consistent, closely spaced frequency increments in the example of Figure 28. This trend was not seen in most examples, but in this case there does appear to be a harmonic relationship between the peaks. A possible cause is clipping that went un-noticed during a *wavwrite* function call in Matlab. The IRs have since been re-computed with a longer length and a more careful normalization scheme (see Section 6), which has remedied the problem.

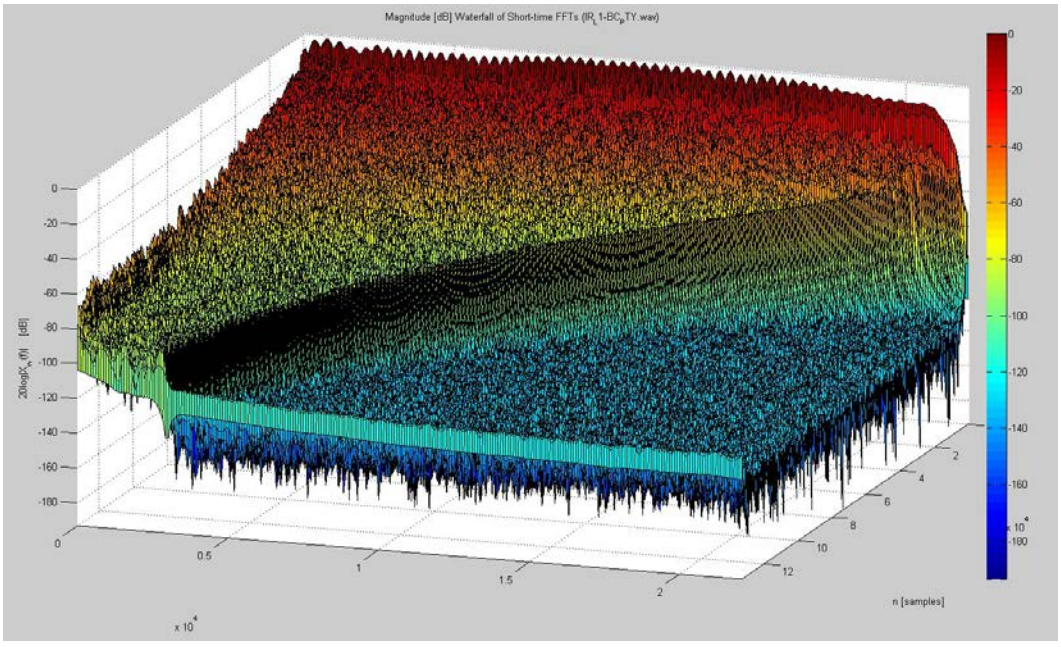


Figure 28: Time-frequency (waterfall) representation of center microphone channel IR, Array-Source location L1-CB. Time axis is directed out of the page, frequency axis is across the page.

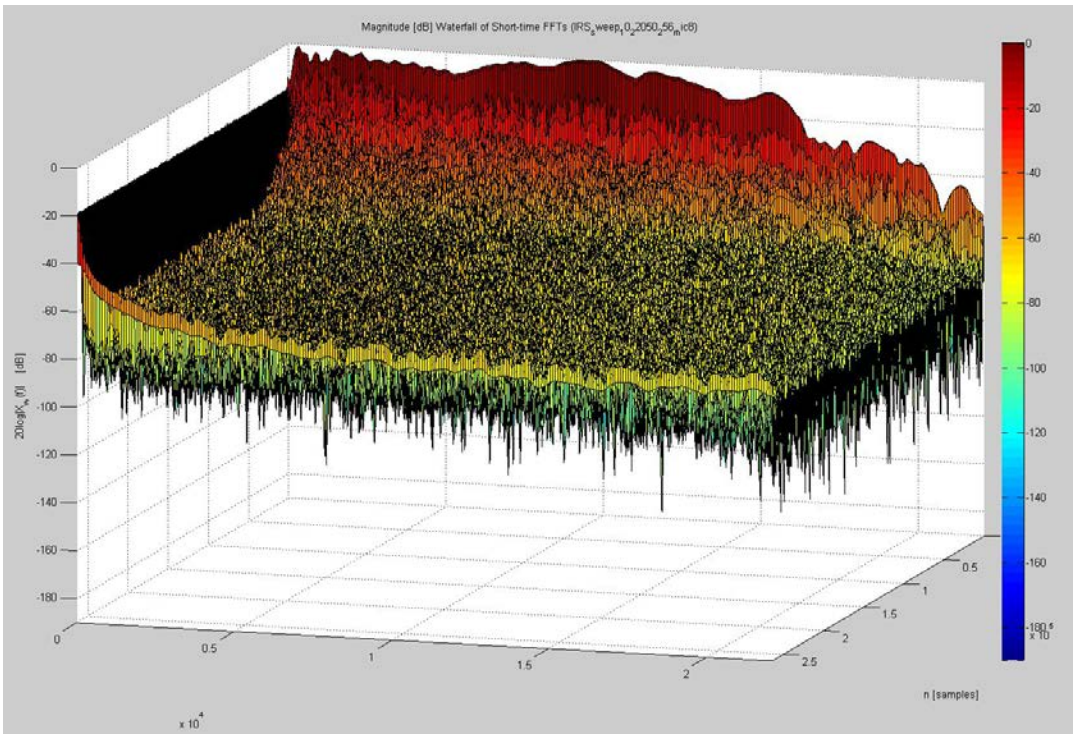


Figure 29: Time-frequency (waterfall) representation of an IR measured in PTY Recital Hall in 2004 using the linear sine sweep method [7]. Time axis is directed out of the page, frequency axis is across the page.

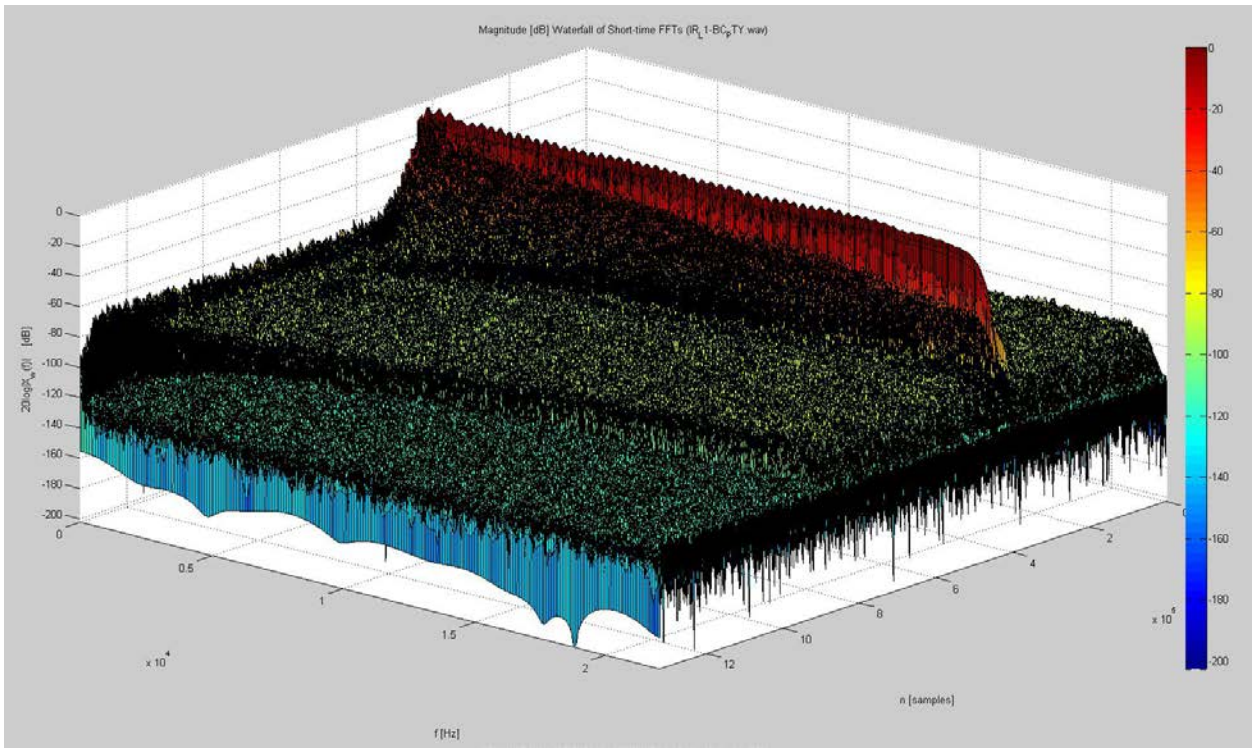


Figure 30: Time-frequency representation for center microphone IR of Source-Array configuration L1-BC, computed using time domain deconvolution. The colour bar indicates a noise floor below -80dB across the entire audible spectrum.

Viewing an un-trimmed IR waterfall plot where time domain convolution with the inverse filter has been used instead of FFT convolution (Figure 30), the noise floor across the spectrum consistently points to a dynamic range of over 80dB. Several additional IRs that were computed in the time-domain were plotted, all having the same trend.

5.2. Comparison with *Aurora*

To confirm the dynamic range, several additional IRs were computed with the *Aurora* [7] plug-in for Audacity. The IR for the center microphone of Source-Array configuration L1-FR is shown in Figure 31. This was computed in 32 bit coding space, and re-quantized to 24 bit. An incredible dynamic range of over 95 dB is displayed, with a reasonably consistent noise that adds validity to the measurement.

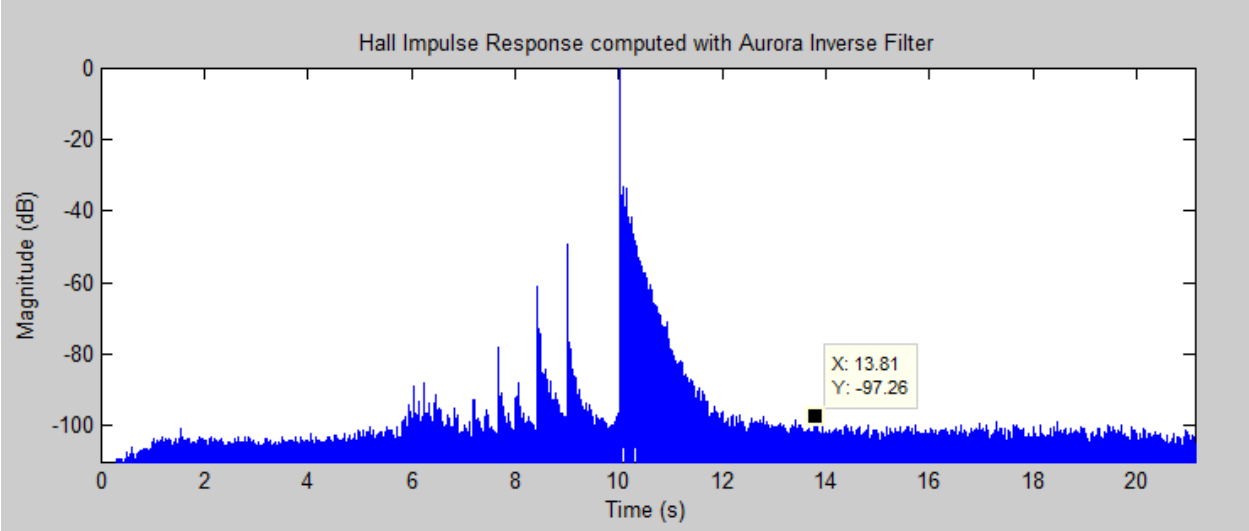


Figure 31: Un-trimmed IR for Source-Array configuration L1-FR as computed in Aurora.

Figure 32 shows impulse responses computed in Matlab and Aurora for the same microphone-Array-Source configuration. Comparing these two IRs, we can confirm that indeed, the additional denoising effect resulting from FFT convolution has resulted in an exaggerated dynamic range when viewed purely in the time domain. However, the Aurora-computed IR does still exhibit an impressive dynamic range of over 90 dB.

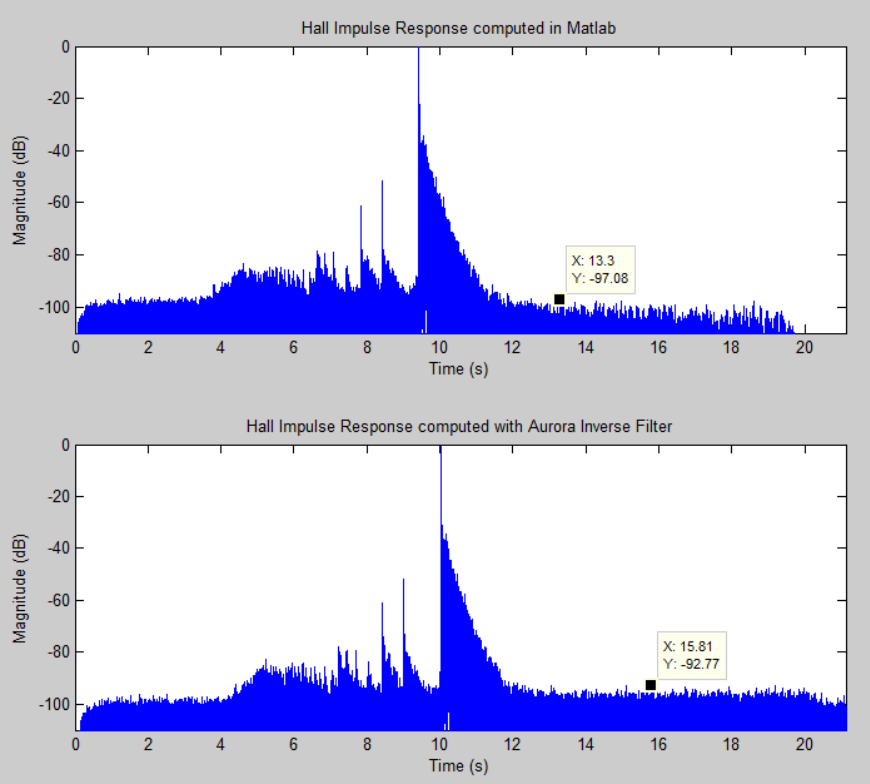


Figure 32: IR Computed with FFT convolution in Matlab (top); Un-shaped IR computed in Aurora (bottom).

The findings are consistent with [2] and [5], which demonstrate the superior SNR of the ESS measured impulse response over other methods. The examples in the above figures are significantly above the 80 dB lower dynamic range threshold indicated by Farina in [2]; however, the testing conditions cannot be compared directly, and different equipment was used. Additionally, most of the above plots were produced based on excitation locations that were reasonably close to the source. In such configurations, a higher sound intensity was received than in more distant Array-Source configurations. The next section will introduce measurements taken throughout PTY hall, with all active microphone channels.

5.3. Dynamic Range of 0 dBu Reference

Naturally, the highest SNR is expected to be found in the location where the highest sound intensity was received. This corresponds to the center microphone channel recording at array-source configuration L1-C. To give a fair estimate of the dynamic range, the IR was computed with acyclic convolution in the time domain (Figure 33), which was orders of magnitude slower than FFT convolution due to the length of the recorded excitation signal and inverse filter (both 10 seconds). The same IR computed with FFT convolution (Section 6, Figure 37) suggests a dynamic range of over 105 dB, but we have already discovered that the shaping of the noise floor lends itself to generous SNR claims.

With time domain acyclic convolution, the dynamic range is still impressive. If trimmed to size 4 seconds after the impulse (in addition to trimming the harmonic distortion components that occur before the IR), the dynamic range is just over 100dB. One peak sticks out from the noise floor, occurring just past the 14 second mark on the timescale of Figure 33. Given this data, a fair estimate for the maximum SNR out of the batch of IRs measured is 100dB. This is similar to the noise floor level the instant before the impulse response occurs.

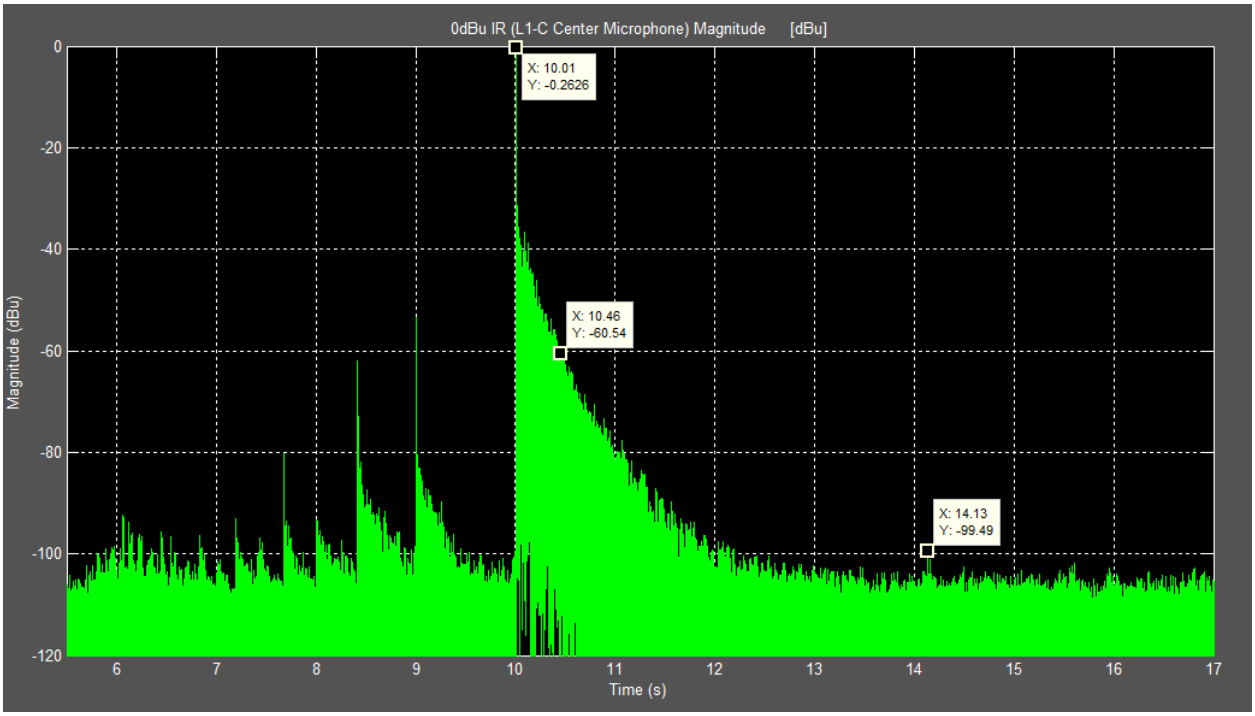


Figure 33: 0 dBu reference IR computed with acyclic time domain convolution to ensure minimal noise floor shaping. The dynamic range is at least 100dB. If trimmed between 10 and 14 seconds on the above timescale, the dynamic range can be considered to be slightly over 100dB. The -60dB point in this plot does not correspond with the reverb time of PTY Hall.

If we try to estimate the reverberation time *RT60* for PTY Hall by looking at Figure 33, the value will be incorrect. This is because the center microphone channel (hypercardioid) of Array-Source relation L1-C is picking up a response almost exclusively from the stage. RT 60 estimations of PTY hall will be made at distances greater than the *critical distance*, the distance from source to receiver where the direct sound is equal in magnitude to the magnitude of the reverberant field in the room. Without additional information, Figure 33 may have us believe that RT60 is 0.45 seconds, which is much too short for a Recital Hall. Of course, the reverb time varies across the spectrum, and a waterfall plot is helpful for conceptually viewing that trend. To estimate RT60, however, locations further away from the source will first be checked see if they meet the critical distance requirement.

The waterfall plot of the location L1-C center microphone channel impulse response, henceforth called the *0 dBu IR* (from which all others have been normalized as discussed in Section 6), are shown below for both acyclic time domain convolution and FFT convolution extraction methods. The noise floor indicated for both cases is close to -100dB across most of the audible spectrum.

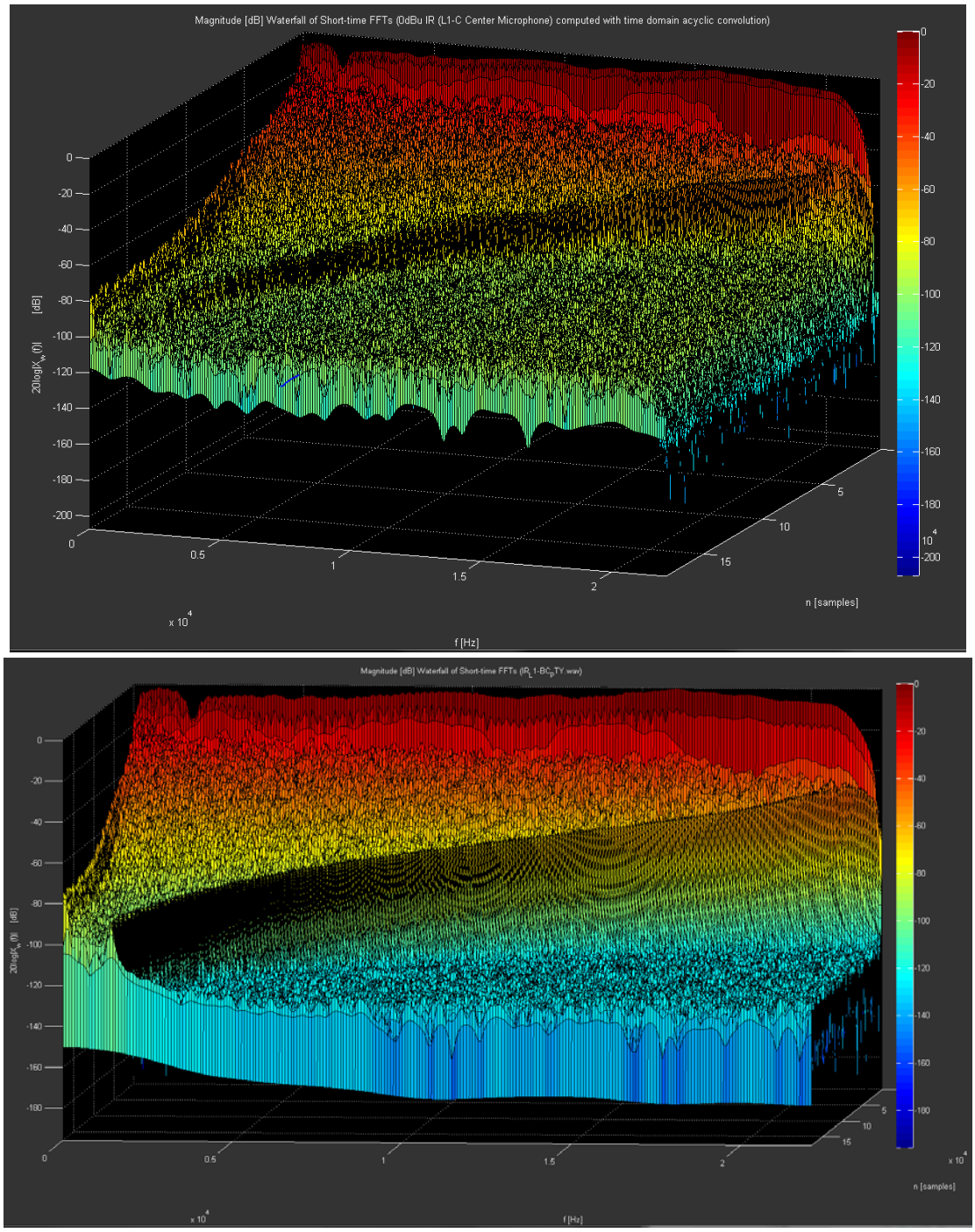


Figure 34: Waterfall plot of 0 dBu ref. IR as computed with time domain convolution (top) and FFT convolution (bottom), trimmed to 4 seconds long in both cases.

5.4. Further Extensions of the SNR

The SNR can be extended further by using 32 bit floating point processing for all stages of the IR extraction. Minimal bit rate conversions should be used to avoiding quantization noise and successive dithering operations, which will raise the noise floor. The noise floor can be reduced further still by up sampling and noise shaping in a larger coding space (such as 32 bit, 192 kHz). The IR can then be down sampled and re-quantized to 24 bit, 44.1 kHz. The result will be a premium quality impulse response that can be used for pro audio applications or acoustic research.

6. Surround Sound IR Results

All impulse responses were normalized to a common reference . The reference peak amplitude was taken to be the maxim absolute amplitude of the floating point (un-normalized) IR for array-source location LC-1. The center microphone channel in this location recorded the highest amplitude signal during room excitation.This reference was normalized to be to be 0.2626 dB below unity (Figure 33), where unity is 0 dBu. Normalizing all IRs to a common 0 dBu peak reference was necessary for two reasons:

1. to facilitate surround sound amplitude panning between the microphones.
2. to preserve the difference in sound intensity differences recieved at each listening location, and from each source location.

Timing differences between the IR channels were also preserved. For analyis, a closely synchronized data set may be desired when studying the intitial time delay gap (ITDG), early reflections, and performing comparisons between Array-Source locations. The timing difference between microphones in the array is of course essential for imaging, which will be discussed as it related to playback in the next sections.

By relating timing differences with intensity differences between various locations, the set of IRs becomes a more powerfull tool for anylizing an acoustic space than a set of usnynchonized IRs that lack a common level reference. The data set as a whole, for example, may be usefull for comparing attenuation caused by absorbtive surfaces in the space versus attenuation resulting from propigation through air over various distances. The early reflection data set as a whole may be used to reconstruct a 2D representaion of the acoustic space (or a 3D representation, if spatial RIR measurements had been made).

Analyis of the extracted impulse responses for each microphone in the surround sound array confirms the impressive dynamic range that was achieved, particularily in locations close to the source. Using the ESS method, a remarkable dynamic range of over 80dB was reported by Angelo

Farina in [2]. While test conditions can not be directly compared, the SNR displayed in Figure 2 points to a dynamic range that is significantly greater than 80dB. However, before jumping to conclusions about the dynamic range, the shaping effect on the noise floor from resulting from the FFT convolution must be taken into consideration. The maximum SNR was computed in the above section to be 100dB. In general, the actual dynamic range is 5dB-6dB less than that suggested by the plots below, using the 3 second mark as a point of reference.

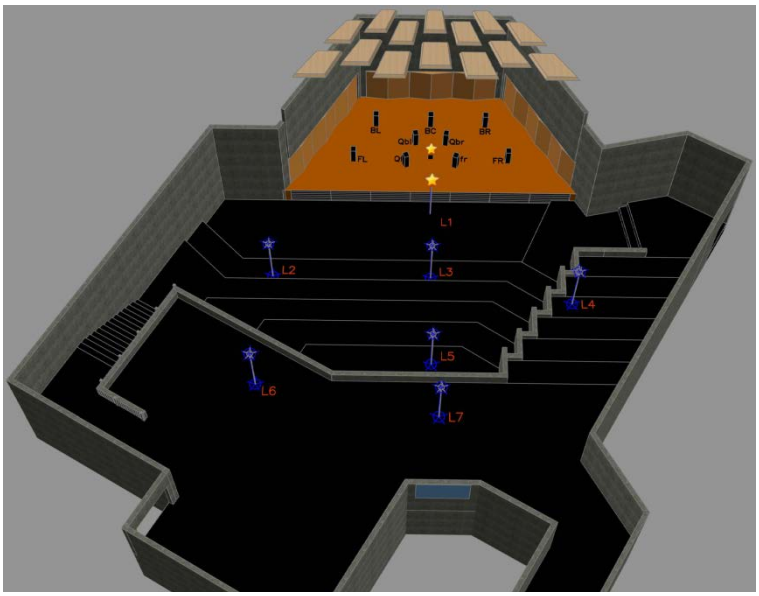


Figure 35: Array-Source configuration L1-C.

The figures below compare the impulse response for each microphone channel under various Array-Source position relationships. The entire data set contains 350 mono channel IRs (70 Array-Source relations times 5 microphone channels), corresponding to 70 surround sound IRs. Several example results will be presented here.

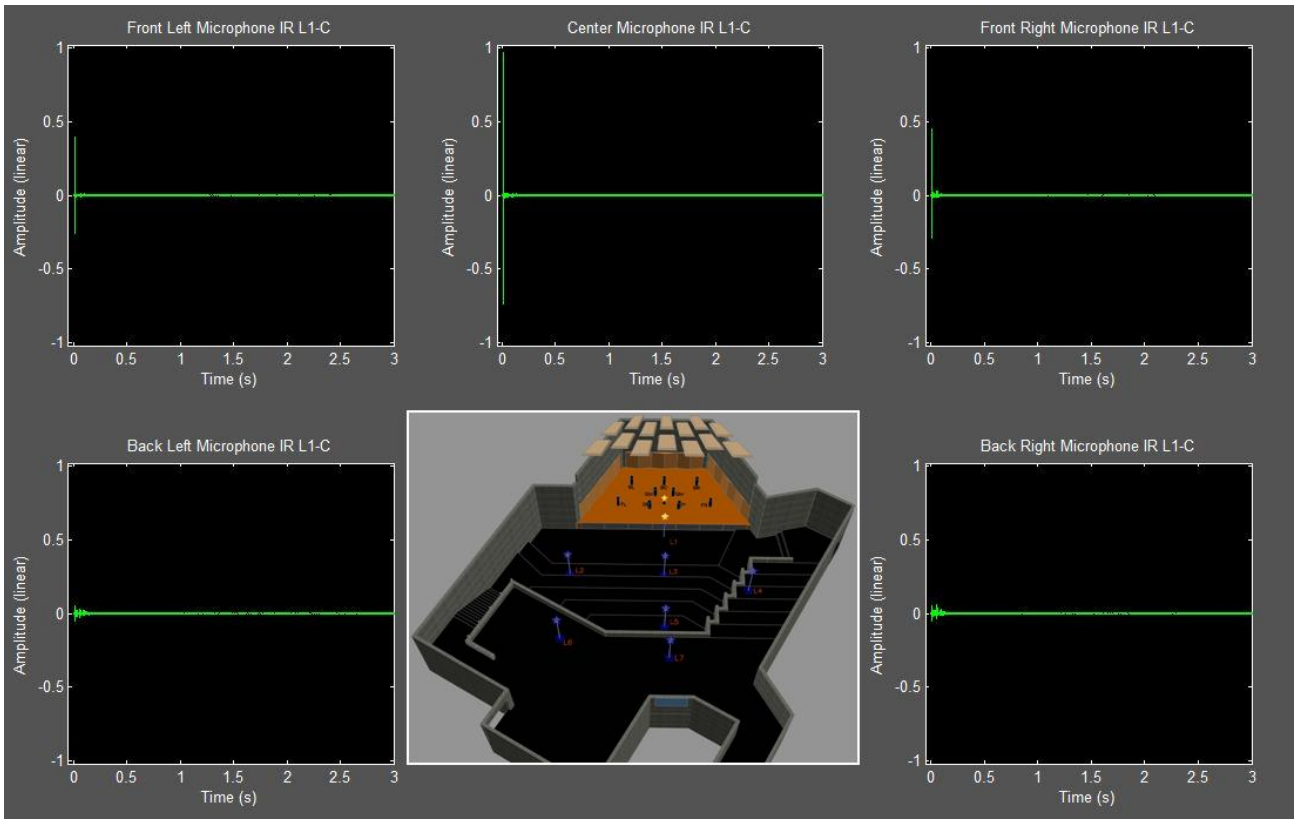


Figure 36: IR amplitude for each microphone channel in surround array. Array-Source relation is L1-C.

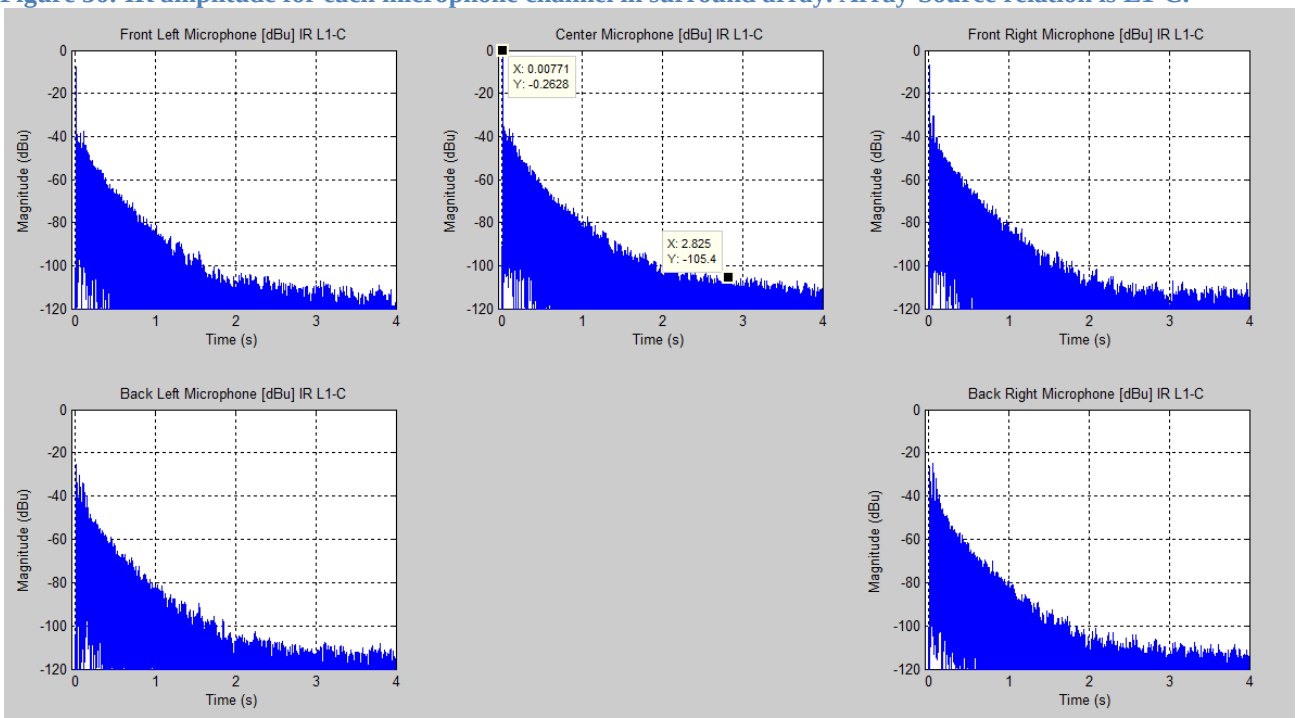


Figure 37: IR for each microphone channel in surround array plotted in dBu. Array-Source relation is L1-C. While the noise floor is below -105 dB for the center microphone channel at close to 3 seconds (FFT convolution was used), the maximum fair estimation for dynamic range at this location is 100dB, as discussed in the previous section.

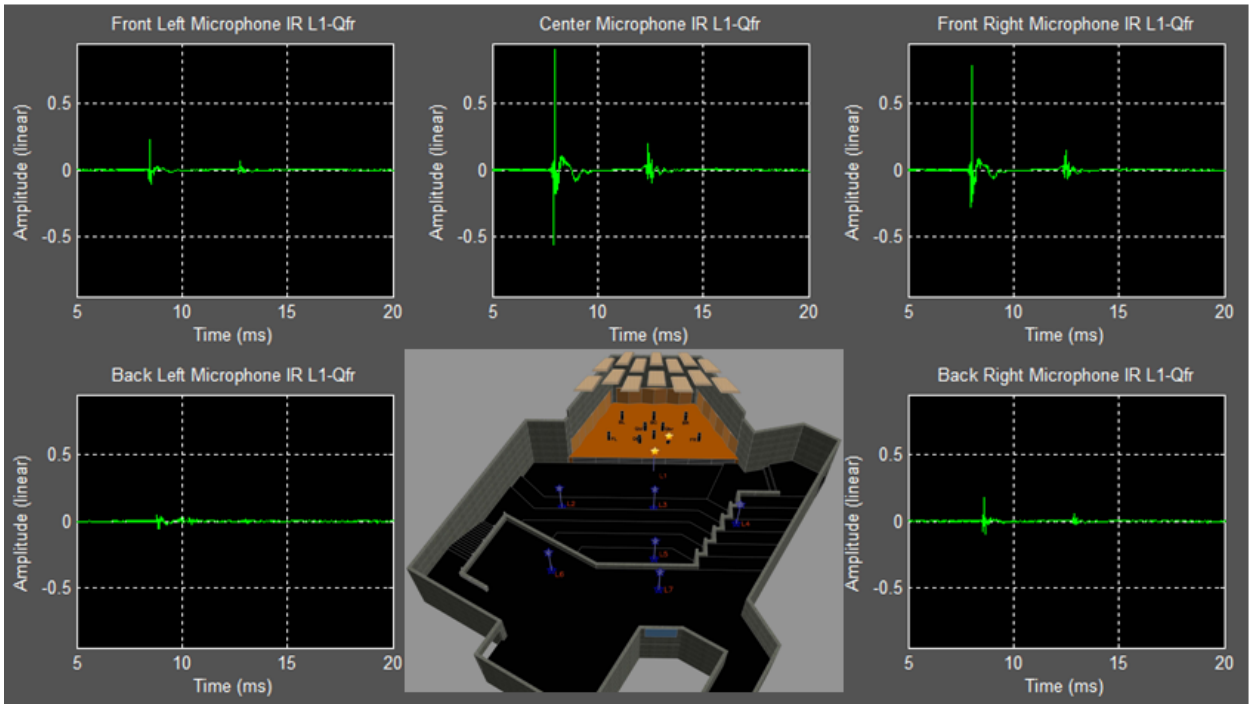


Figure 38: Surround IR channels (zoomed timescale) plotted for Array-Source relation L1-Qfr. Timing and amplitude differences are visible between the channels.
-First reflections are shown; *ITDG* = 5.24 ms for center microphone and 6.38 ms for front right microphone.
-*First wavefront* arrives at front right microphone channel 0.57 ms before front left microphone channel.

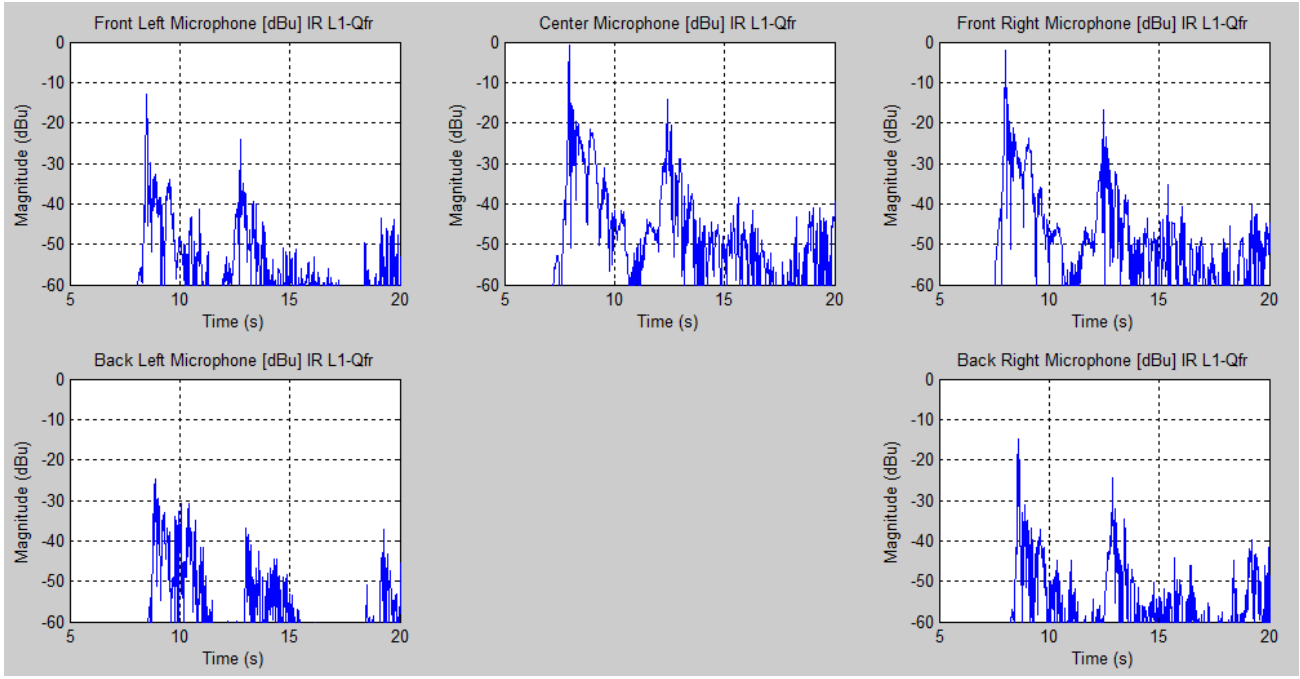


Figure 39: Surround IR channels (zoomed timescale) plotted in dBu for Array-Source relation L1-Qfr. Timing and amplitude differences are visible between the channels.
-First reflection is 13.42 dB below the direct source magnitude for the center channel.

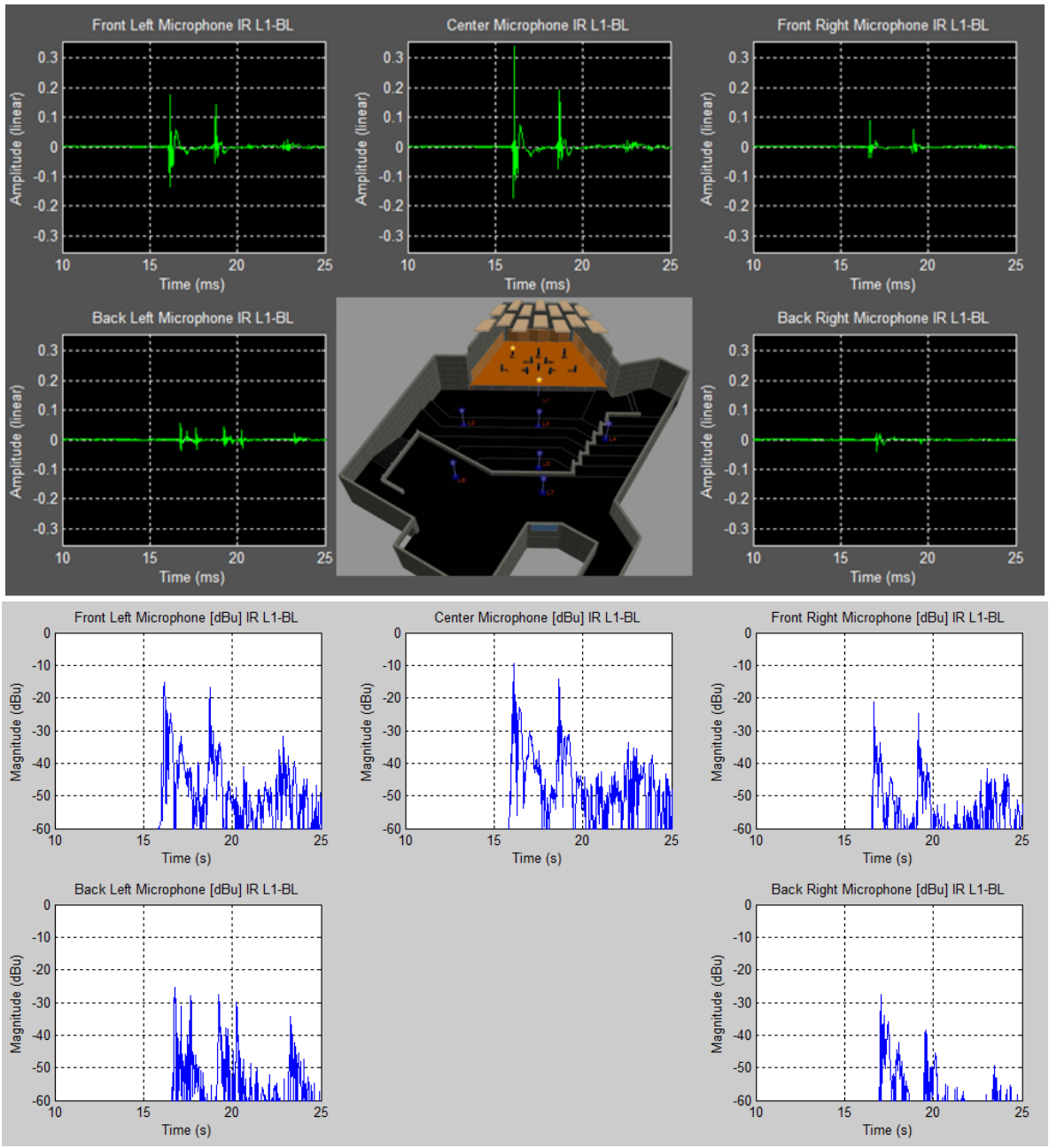


Figure 40: Surround IR channels (zoomed timescale) for Array-Source relation L1-BL, plotted on a linear amplitude scale (top) and in dBu (bottom).
-*ITDG* = 2.56 ms for center microphone channel.
-First reflection is 5.24 dB below the direct source magnitude for the center channel.
-*First wavefront* arrives at front left microphone channel 0.48 ms before front right microphone channel.

Several relationships can easily be seen between the two locations presented in the preceding images. Due to the relative proximity to the stage walls, we see a shorter ITDG for the BL source than for the Qfr source when the microphone array is in location L1. Additionally, we see a drop of just under 10 dB peak magnitude in the center microphone position by moving the source from position Qfr to postion BL. These plots also indicate that the relative timing between stage positions has been preserved.

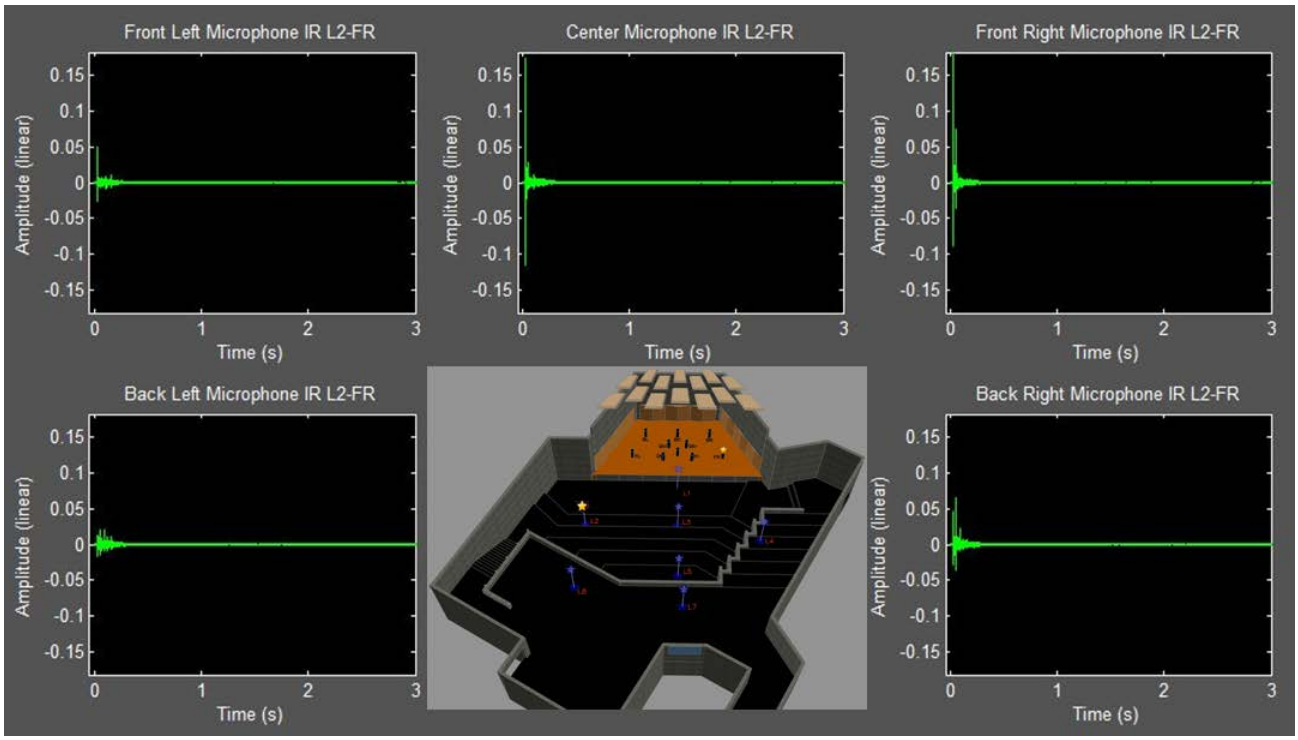


Figure 41: Surround IR channels plotted for Array-Source relation L2-FR.

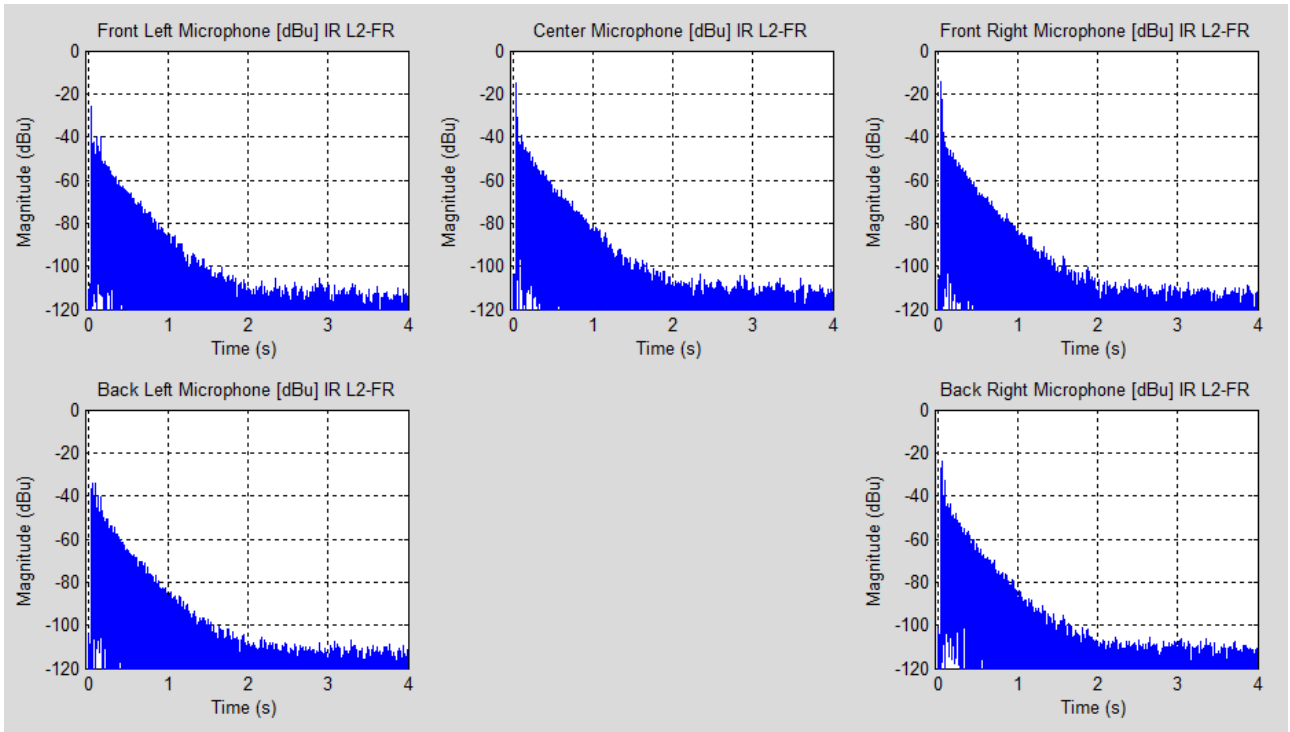


Figure 42: Surround IR channels plotted in dBu for Array-Source relation L2-FR.
–RT60 estimation is about ~ 0.75 seconds.

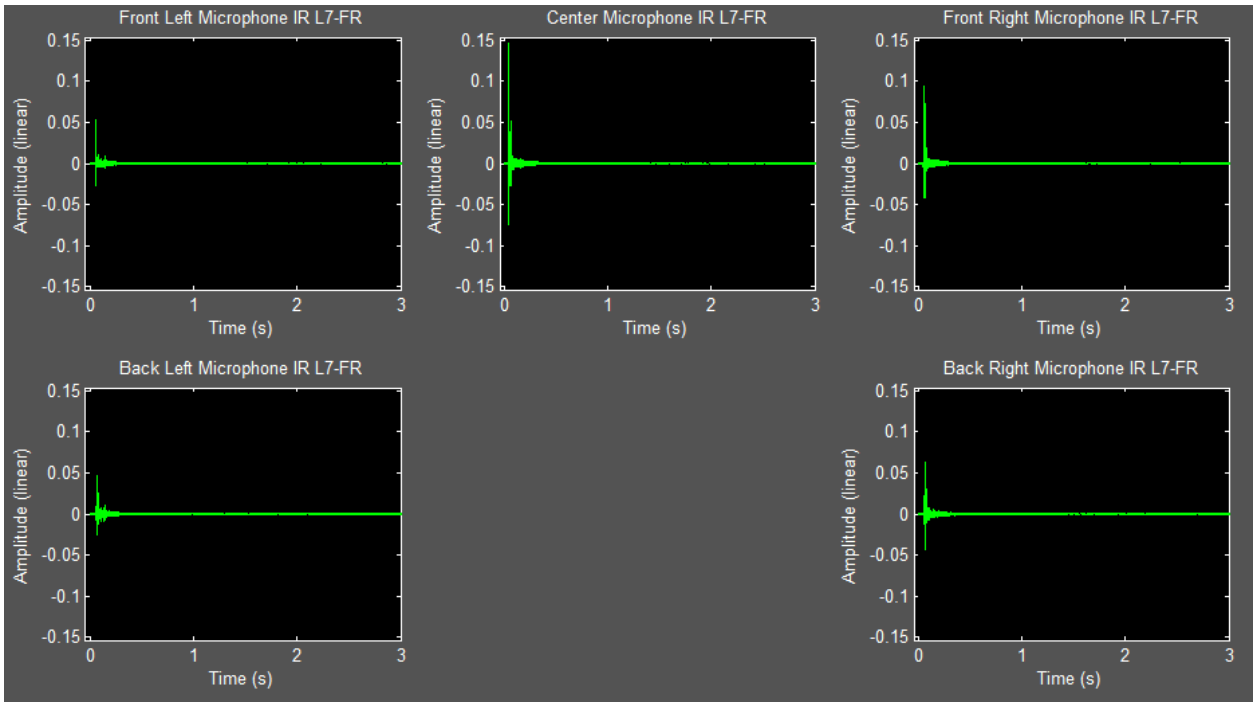


Figure 43: Surround IR channels plotted for Array-Source relation L7-FR.

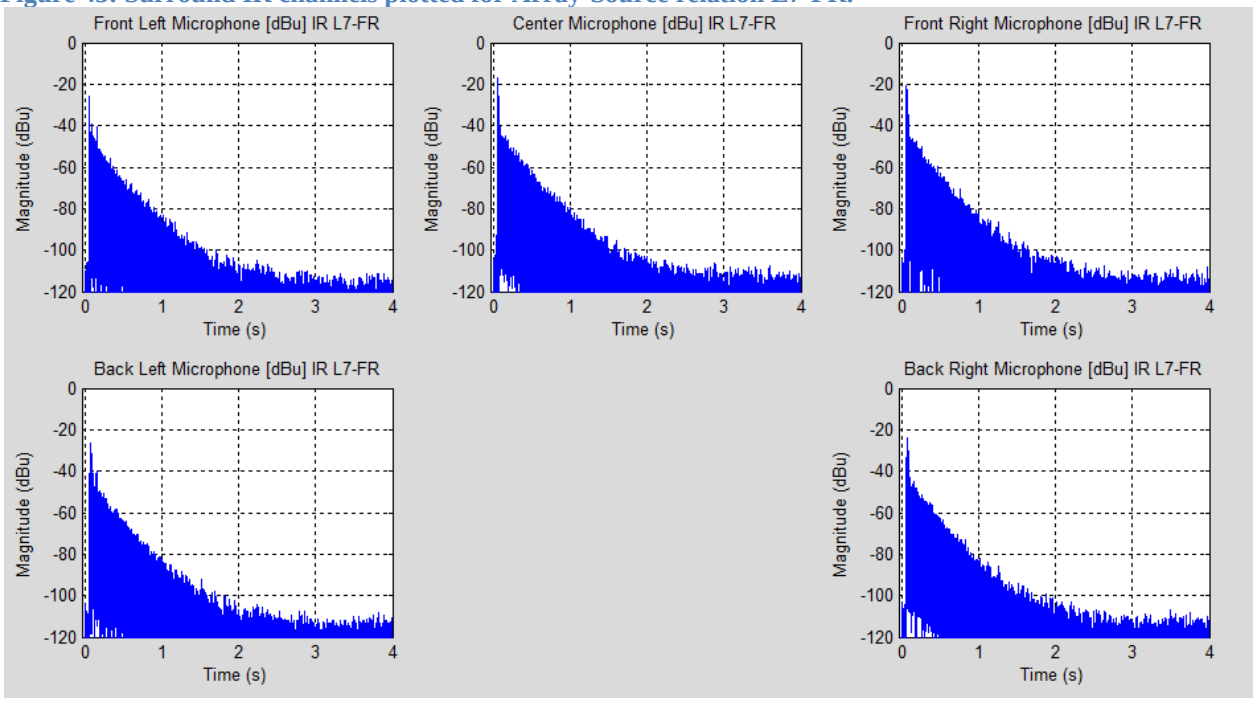


Figure 44: Surround IR channels plotted in dBu for Array-Source relation L7-FR. –RT60 estimation is ~ 0.75 seconds.

For locations L2 and L7, an RT60 estimation based purely on the time domain impulse response yields practically the same result, about 0.75 seconds. This suggests that for a source at stage position FR, we are outside of the critical distance at when listening from location L7, and at a distance greater than or equal to the critical distance when listening from L2.

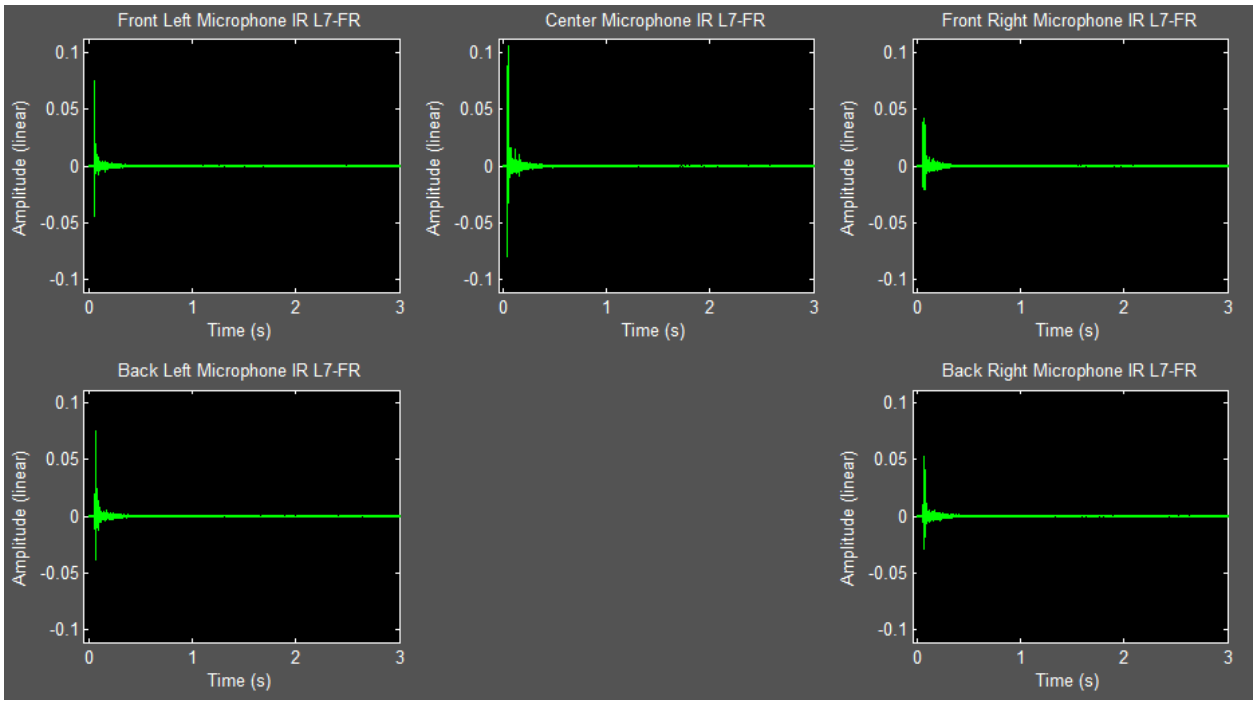


Figure 45: Surround IR channels plotted for Array-Source relation L7-FL.

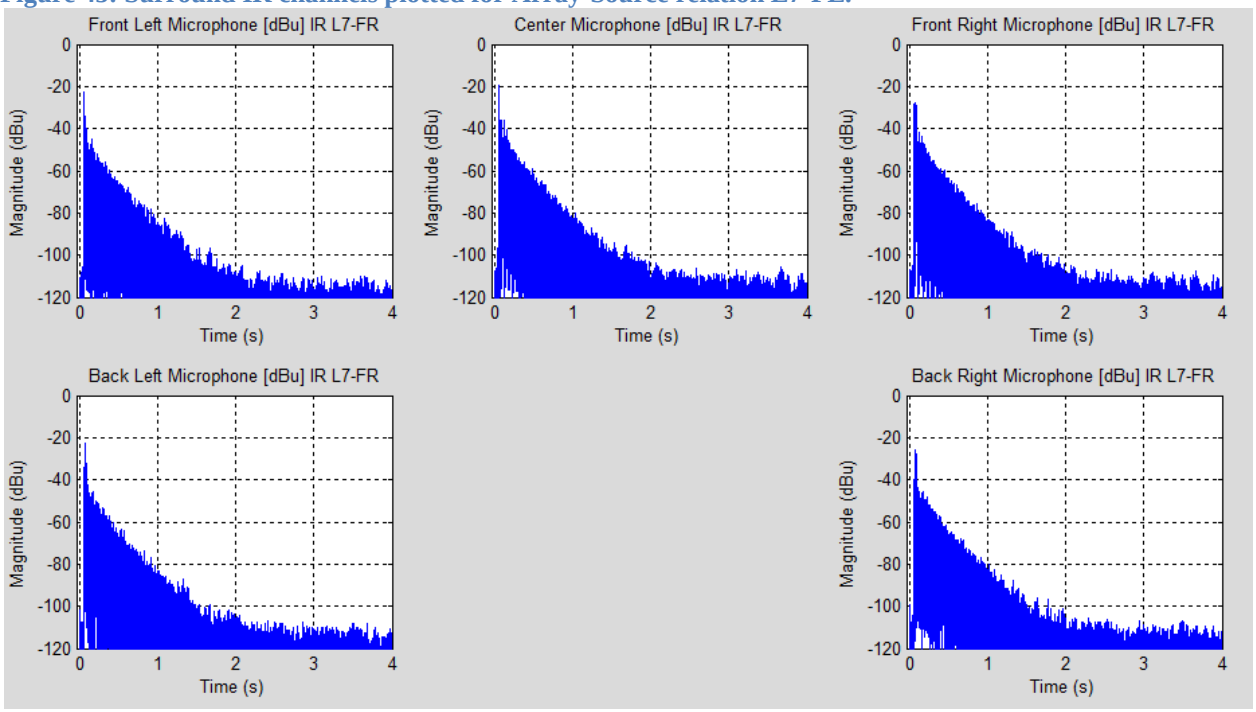


Figure 46: Surround IR channels plotted for Array-Source relation L7-FL.

Listening to the stereo rendered impulse responses from L3 and L4 were originally quite shocking, as they were incorrect. Initial attempts were made to explain the bizarre imaging in position L4 in terms of an offensive first reflection off of the concrete side wall, arriving on axis with the front right hypercardiod microphone. However, taking a closer look at the apparent timing and amplitude

differences in Figure 47, this is simply not possible. Despite the source being located at an azimuth angle to the left of the microphone array normal (taken to be the longitudinal axis of the center microphone), the front right microphone capsule, according to Figure 47, receives the greatest sound intensity. Apparently, this trend occurs regardless of stage location. While heavy acoustic insulation treatment on the left sidewall and a highly directed sound source reflecting off of the right side wall could possibly produce such bizarre results, the timing confirms that there is indeed an error. It is not possible for a sound located to the left of the array to reach the right size of the array earlier when traveling through open air. Zooming in on the plotted IRs for location L4 revealed that the front right microphone capsule apparently received the first wavefront before the front left microphone under all conditions. This is wrong.

Clearly, the error is that the microphone array was misaligned in locations L3 and L4 to the wrong center microphone reference. This was a simple mistake during the initial stages of the experiment that caused significant confusion until the IRs were normalized to a common 0dBu reference and plotted in a surround configuration. To remedy the problem, all IRs for location L3 were circular shifted by one position clockwise, and for L4, counter clockwise (in other words, each mono IR was re-named). While surround sound playback would have easily revealed the rotated position, all stereo audio processing for these locations (using the front 3 microphone positions) was disorienting to listen to due to the incorrect and unexpected channel mapping. Surround sound and stereo rendering has been re-done to correct this problem.

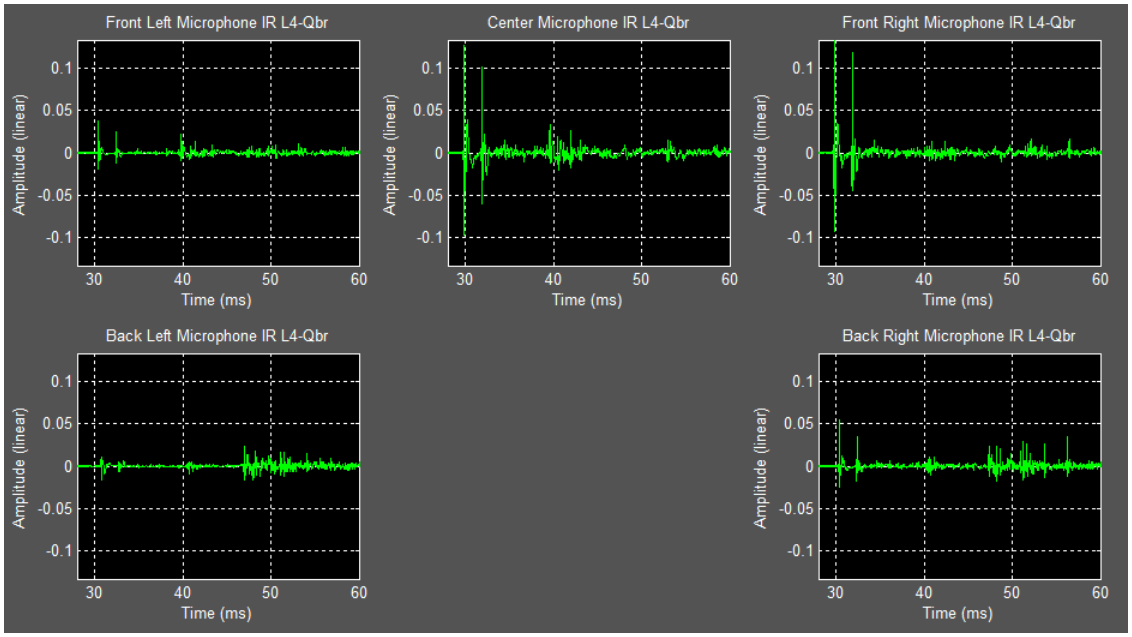


Figure 47: Incorrect surround IR channel configuration (zoomed timescale) for Array-Source relation L4-Qbr, plotted on a linear amplitude scale (top) and in dBu (bottom). The microphone array was out of alignment by one microphone channel. The large reflection at about 57 ms in the plots labelled "Back Right Microphone" is from the sidewall of the hall, and is one of several clues that reveal this microphone's true identity: the *Front Right Microphone*.

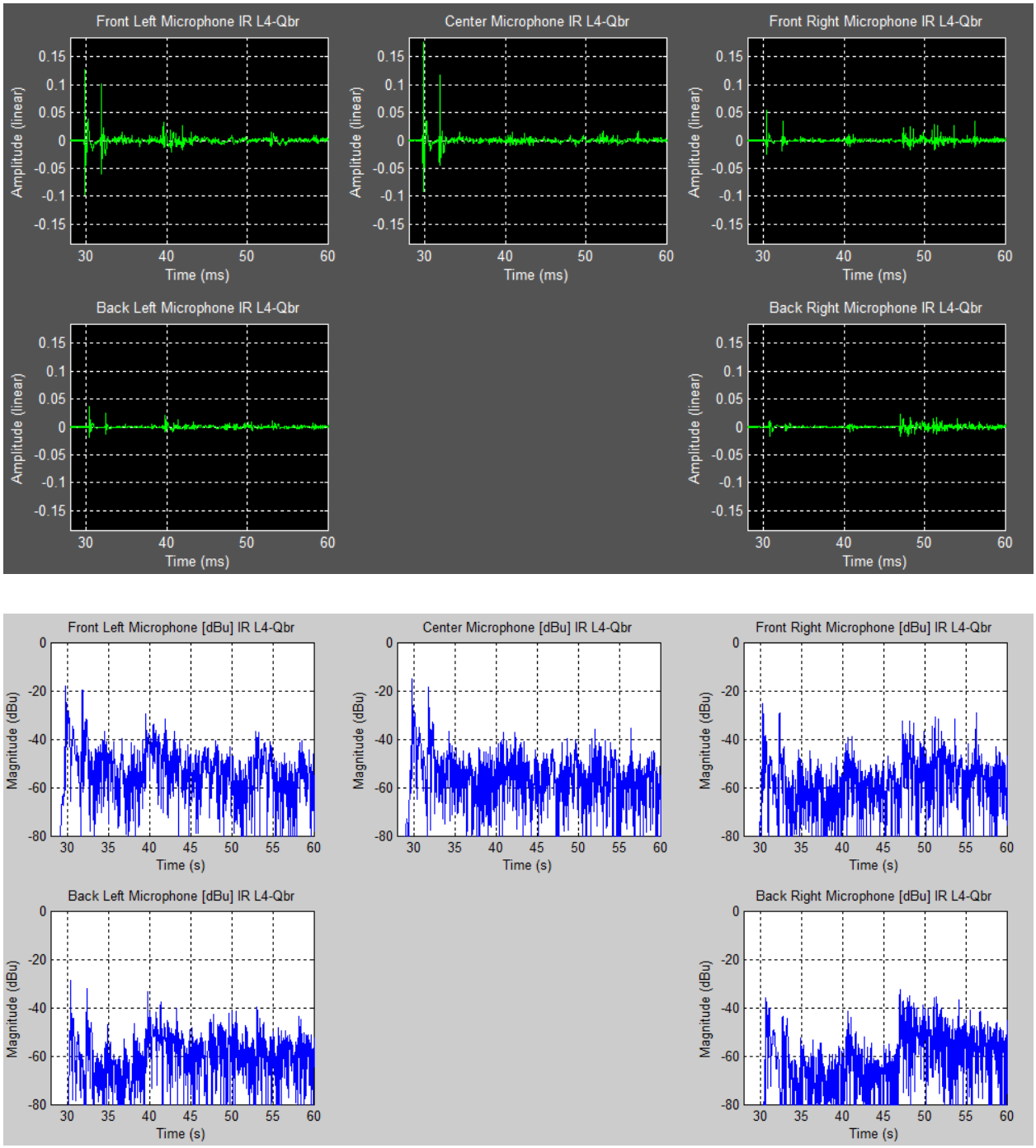


Figure 48: Corrected channel mapping (zoomed timescale) for Array-Source relation L4-Qbr, plotted on a linear amplitude scale (top) and in dBu (bottom).

6.1. Reverb across the Spectrum

Viewing the time-frequency representation of an IR, it is easy to visualize the reverberation at each frequency. This is a reasonable technique for viewing reverberation data quantitatively. We can see that energy decays more quickly in the high frequencies, and that the reverb tail is characterized primarily by the response in the lower frequencies.

We are most interested in the reverberation in the lower end of the spectrum, where humans perceive the greatest effects of sound colouration provided by reverb (between about 125 Hz and 4 KHz). To visualize this data as it applies to human hearing, a logarithmic frequency scale (Figure 50) is most appropriate when viewing the time-frequency representation.

Figure 50 also reveals a discrepancy between the IR measurement and the true room response below about 20Hz: since this frequency range was not excited, it is not represented by the IR. However, the noise floor is higher in the very low end of the spectrum. Luckily, human hearing is relatively insensitive in this frequency range. A slight increase in the noise floor below 40 Hz will not have a pronounced effect on the dynamic range that we perceive. In music production, highpass filtering is typically performed to remove most of the low end energy below about 30-40 Hz. This is to improve the clarity of the signal. To compensate for the loss in low end energy, equalization is applied to emphasize selected low frequencies that contribute less to a muddy sounding signal (*muddy* describes a signal lacking clarity).

There are techniques for measuring reverberation time that produce more accurate results than the graphical methods used in the above examples. Since this paper focuses on IR measurement, acoustic parameters are being estimated directly from the impulse response representations. However, an extension to this project will involve obtaining more accurate measurements of parameters such as RT60. There are a number of strategic approaches that might be employed, such as measuring the *early decay time (EDT)* or using an algorithm to calculate the “optimal” RT60 decay time at one-third octave band increments, according to the *Topt* method. A full map of RT60 trends across the spectrum in each microphone array location may be achieved in this way.

Additionally, the frequency response can be mapped throughout the PTY recital hall. This simply involves taking the Fourier Transform of each measured RIR. The waterfall plots represent the spectrum as is changes with time (successive FFT frames), and as a result provide a great visual representation - most of the above data is contained within the time-frequency representation.

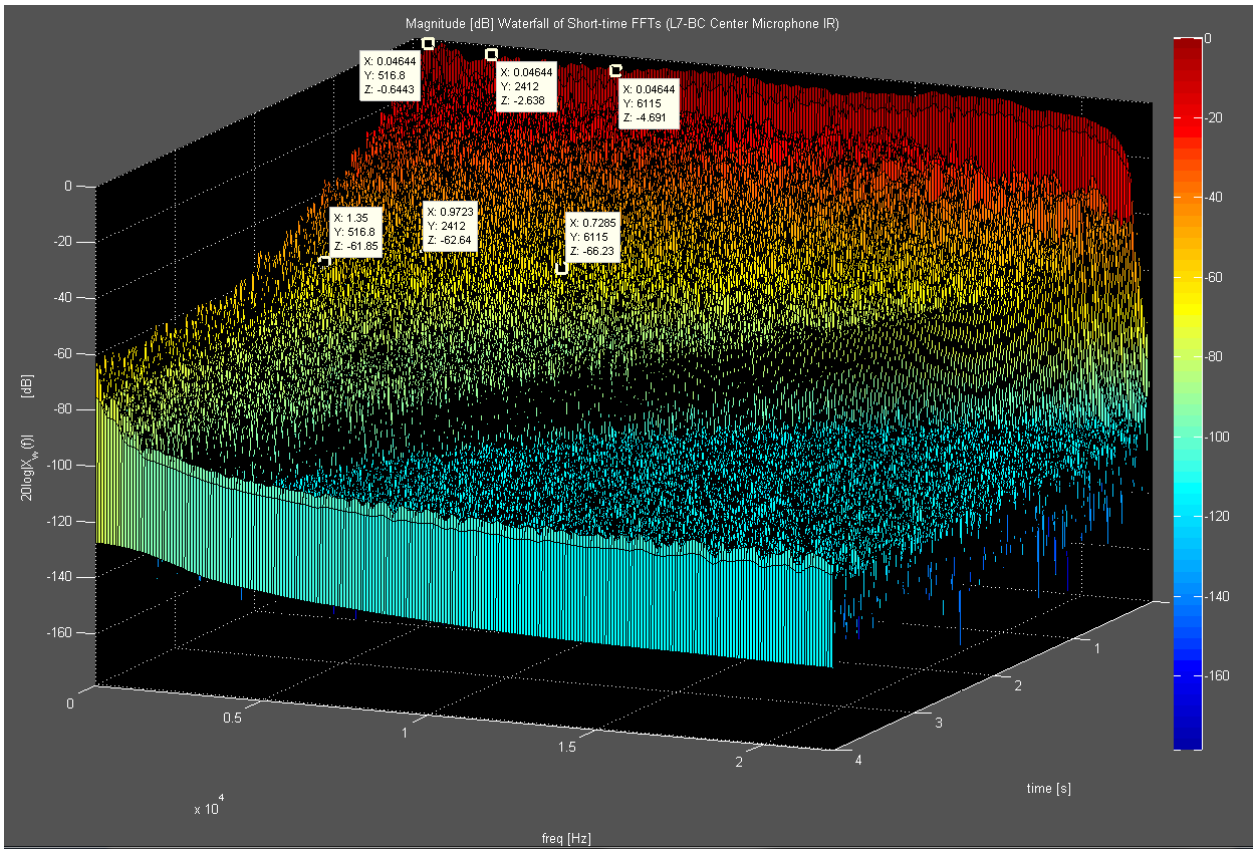


Figure 49: RT60 estimations at 3 frequencies as measured on the center microphone waterfall plot of MMRIR L7-BC.

$$RT60 \sim \begin{cases} 1.3 \text{ seconds at } 516\text{Hz} \\ 0.9 \text{ seconds at } 2412\ Hz \\ 0.7 \text{ seconds } 6115\ Hz \end{cases}$$

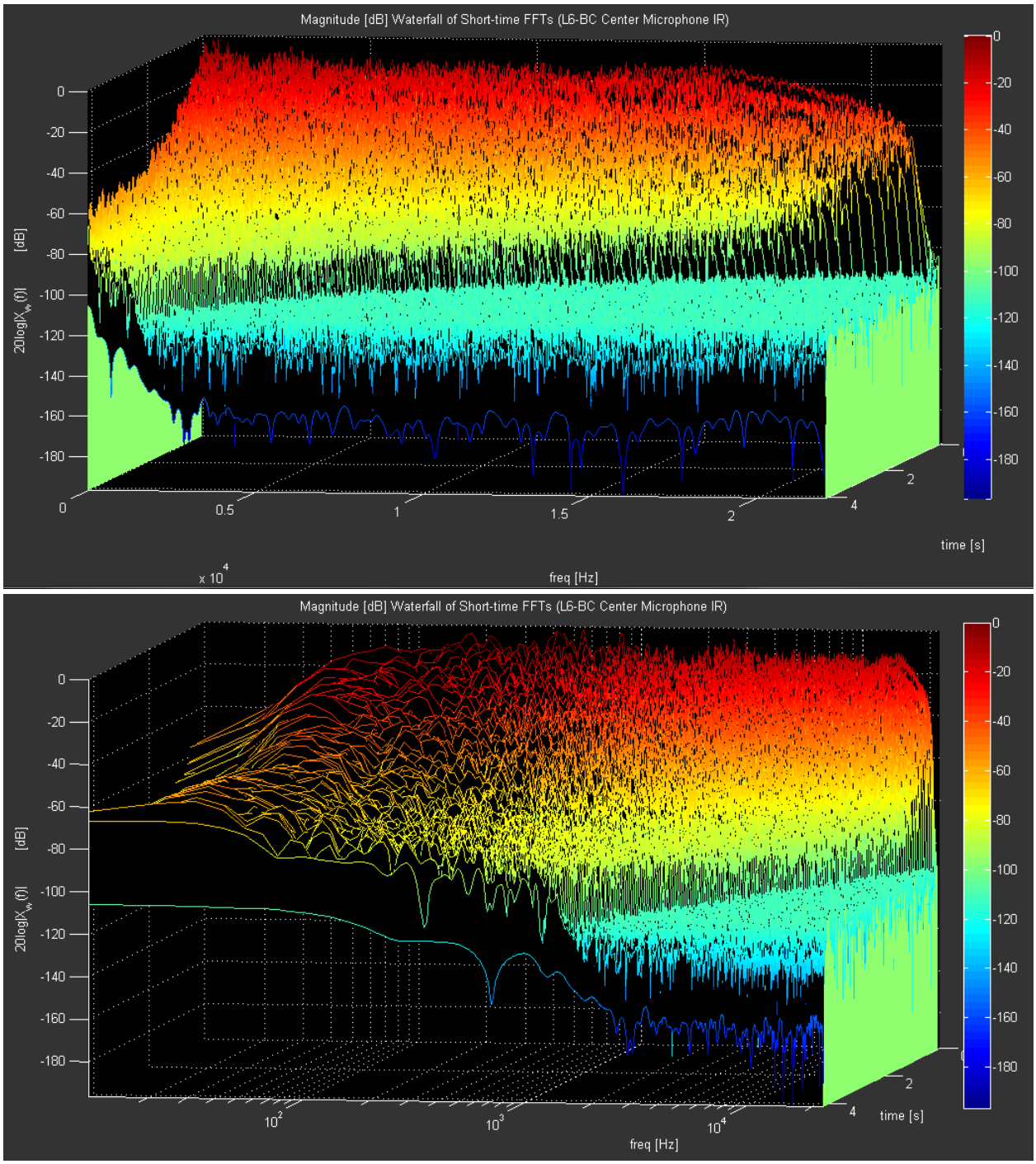


Figure 50: Waterfall representation of center microphone IR for MMRIR L6-BC on a linear frequency scale (top) and a logarithmic frequency scale (bottom).

7. Convolution Reverb Processing

Convolution reverb is achieved by convolving an audio signal with a measured IR. The desired effect is for the processed audio to sound as if it was played in the room where the impulse response was measured. The process of taking a dry sound recorded sound and convolving it with the IR of a room will be outlined below. Later, multi-channel and multi source convolution reverb processing will be introduced.

7.1. FFT Convolution of Source with Measured Impulse Response

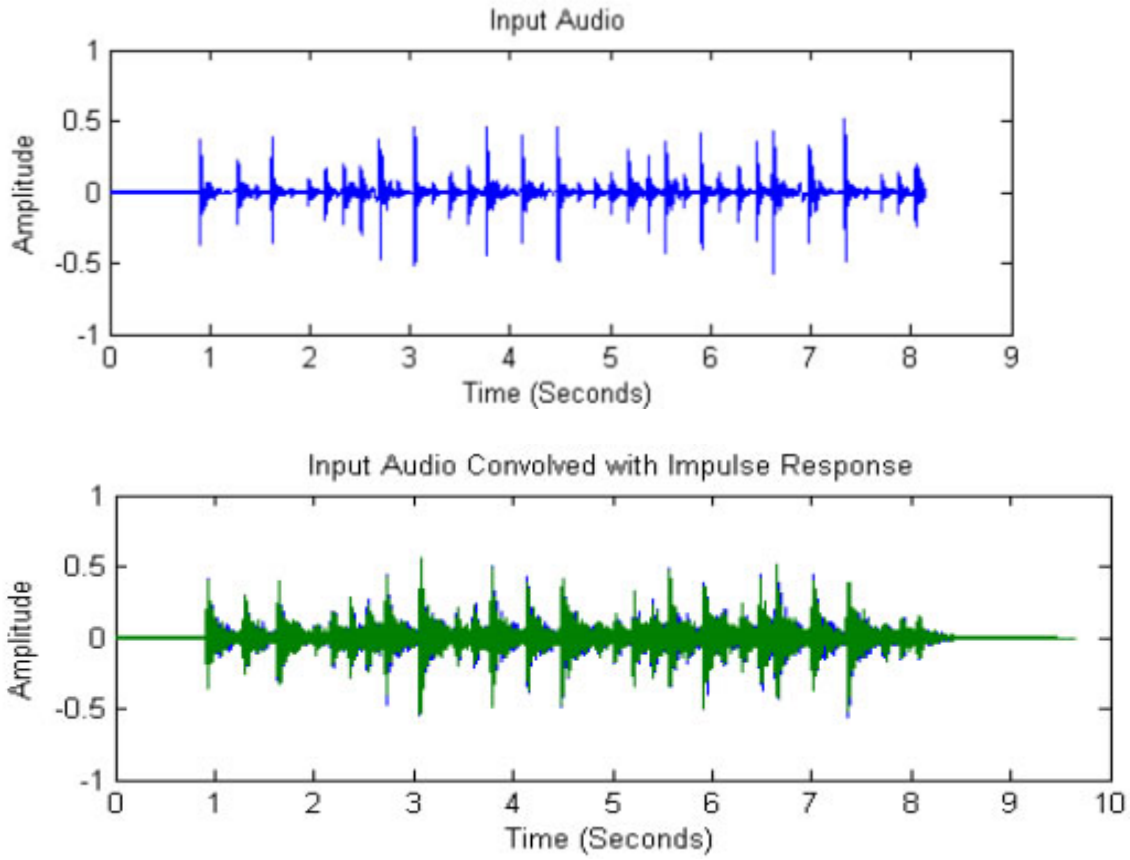


Figure 51: Example dry input audio (top) and convolved output audio (bottom) waveforms.

Audio signals are typically of sufficient length to render direct time domain convolution impractical. Computation time becomes ridiculous with longer signals, and real-time implementation is not possible. An alternative is, *High Speed Convolution* using windowed Fast Fourier Transforms and frequency domain multiplication. FFT algorithms have a computational complexity of $O[nlog(n)]$ versus $O[n^2]$ for the direct application of the discrete-time Fourier transform. The maximum efficiency is achieved if n is a power of 2.

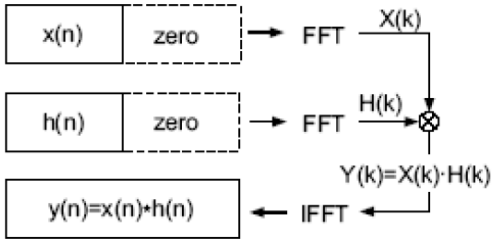


Figure 52: High Speed/Fast Convolution [11]

The result that we want to obtain can be computed by calculating the product of RIR's FFT and the input signal's FFT, and taking the inverse FFT of the result:

$$y(n) = IFFT\{FFT\{h(n)\} \bullet FFT\{x(n)\}\} \qquad (11)$$

However, we can typically do this faster by first chopping up the longer signal. FFT convolution with rectangular windowing and overlap-add was implemented for the convolution of dry source $x(n)$ with a room IR, *h(n)*. For the time being, *h(n)* represents the measured IR of a single microphone channel, which will be referred to as the filter. The input signal was segmented into blocks. Convolution was performed in the frequency domain, convolving successive blocks of the input signal with *h(n)*. Taking *Nx* to be the length of *x(n)* and *Nh = N +1* to be the length of *h(n)*, The procedure was as follows [11]:

- The filter *h(n)* was zero padded to a length *NFFT = 2N = 2(Nh-1).* Next, the zero padded filter's FFT was taken, yielding *H(k).*
- The dry signal *x(n)* was segmented into blocks $x_i(n)$, of length *N*. Each of these blocks was zero padded to a length *2N = NFFT.*
- The FFT of each zero padded frame was taken, yielding $X_i(k) \quad with\ k = 0, 1, 2, \dots 2N - 1.$
- Frequency domain multiplication was applied:

$$Y_i(k) = X_i(k)H(k)$$

- The IFFT was taken of each $Y_i(k)$.
- The convolution results were overlap added in the time domain, yielding the output signal *y(n).* The length of the output signal is *Nx + Nh – 1.*

An example Matlab script for performing the above convolution can be found in Appendix D.

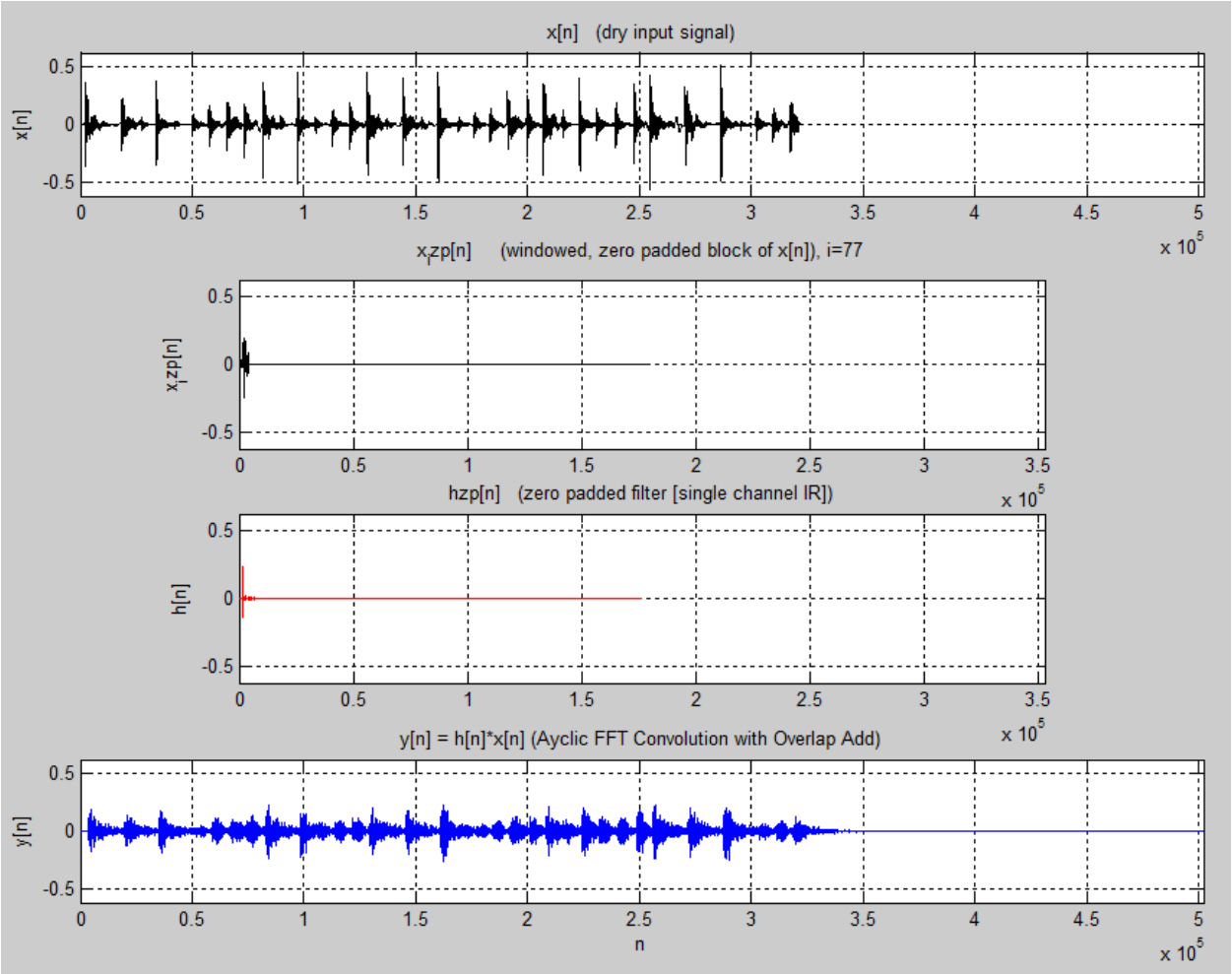


Figure 53: Fast convolution of a dry input signal with the center microphone channel for MMRIR L5-C. The time domain plots are as follows: dry source (top), last windowed block of dry source (2nd down), zero padded IR (3rd down), output (bottom). The input source audio is a Cajun percussion clip.

By implementing acyclic convolution using cyclic convolution with zero-padding, the FFT can be used to perform a cyclic convolution when its length N is a power of 2 [12]. Using zero padding in the time domain results in more samples and tighter spacing in the frequency domain (like a higher sampling rate will achieve). The resulting number of nonzero output samples will be at most $Nx + Nh - 1$.

The motivation to embed and acyclic convolution into a zero-padded cyclic convolution (*High Speed or Fast Convolution*) is that a cyclic convolution is very efficient to compute. However, if insufficient zeros are added with this method, the same problem as encountered with cyclic convolution will occur: some convolution terms will wrap around and add back onto earlier terms, causing time domain aliasing.

The above method of FFT convolution yields the same result as a linear convolution, but is much more computationally efficient for long signals (and hence faster). Specifically, for filter kernel lengths over about 50 to 80 samples, depending on the hardware, FFT convolution is faster than standard convolution. In practice, precision of a convolution result (assuming a correct convolution without aliasing) is determined by the speed of the calculation. This is due to round-off error in the computation, which is proportional to computation time [13].

An alternative method of FFT convolution was initially used, which employed raised-cosine windows of length $WLen = NFFT = 4096$ samples and a hop size of $R = WLen/4$ for windowing the input signal and overlap-adding the convolved segments. This method was significantly less efficient; it involved segmenting the IR in addition to the input signal, since the IR length is much longer than 4096 samples. While the FFT size was smaller, many more FFTs were required than with the above high speed convolution scheme. Additionally, it resulted in a lower frequency resolution.

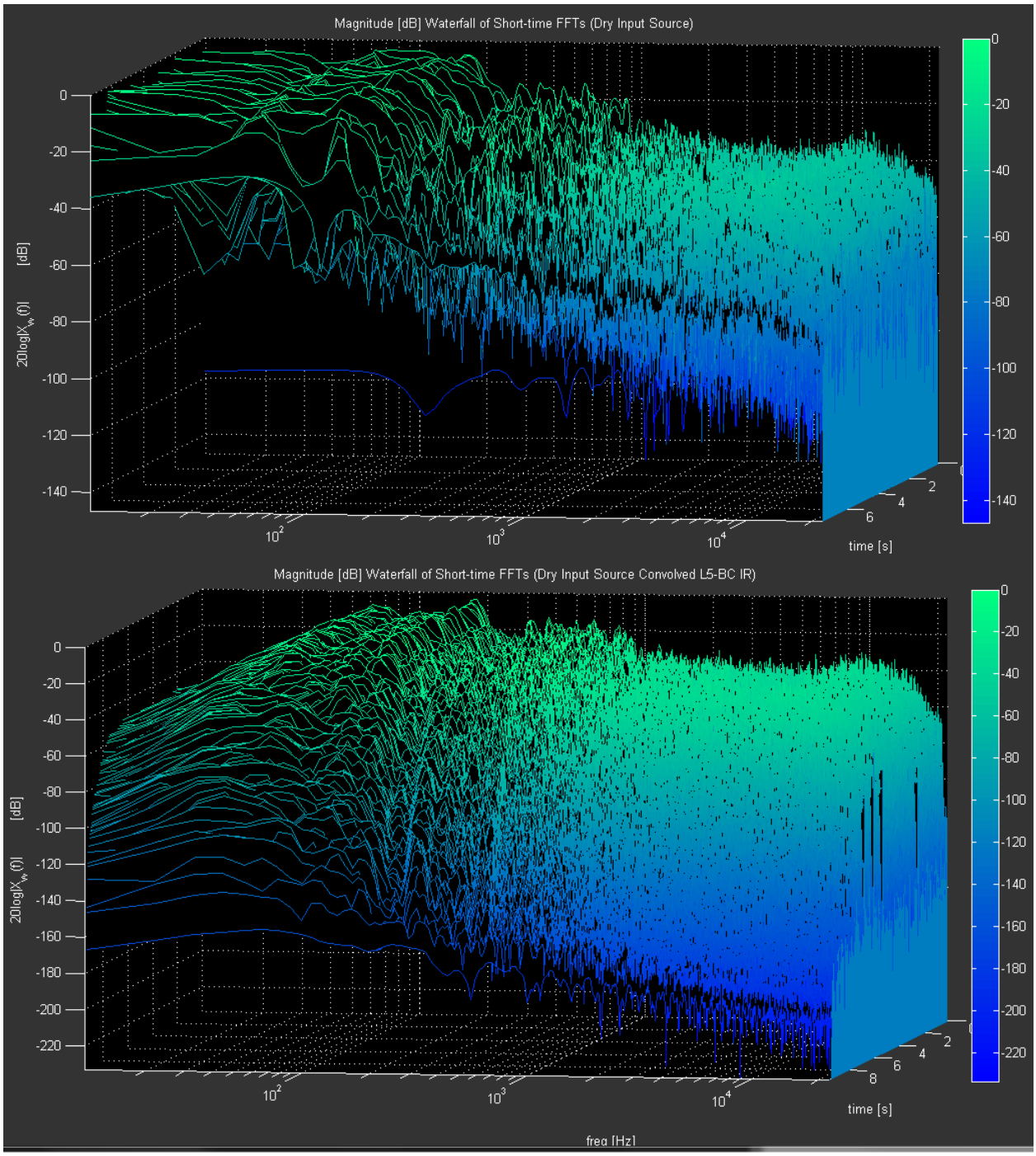


Figure 54: Magnitude waterfalls of dry input audio (top), and source convolved with L5-BC center microphone IR (bottom). The input source audio is a Cajun percussion clip.

Comparing the dry versus wet waterfall plots of sources convolved with the room IR, several interesting observations can be made. We see that the timbre has changed, and that the decay time for a signal has increased due to the convolution reverb. However, it's interesting to note that while the convolved percussion audio sounds like it has a deeper low end, the energy spectrum below about 30 Hz is in fact greatly reduced from the original signal. This is because when the IR was measured, the room excitation in the low end of the spectrum was limited by the speaker low end response. Also, since the sweep started at 20 Hz, there was no room excitation below that. We don't hear frequencies below about 20 Hz (we only feel them), so the resulting filtering of these low frequencies is hardly noticeable under most playback conditions. The frequency range where we perceive low end energy is typically between about 60 and 120 Hz, and the convolution reverb has accentuated this frequency range.

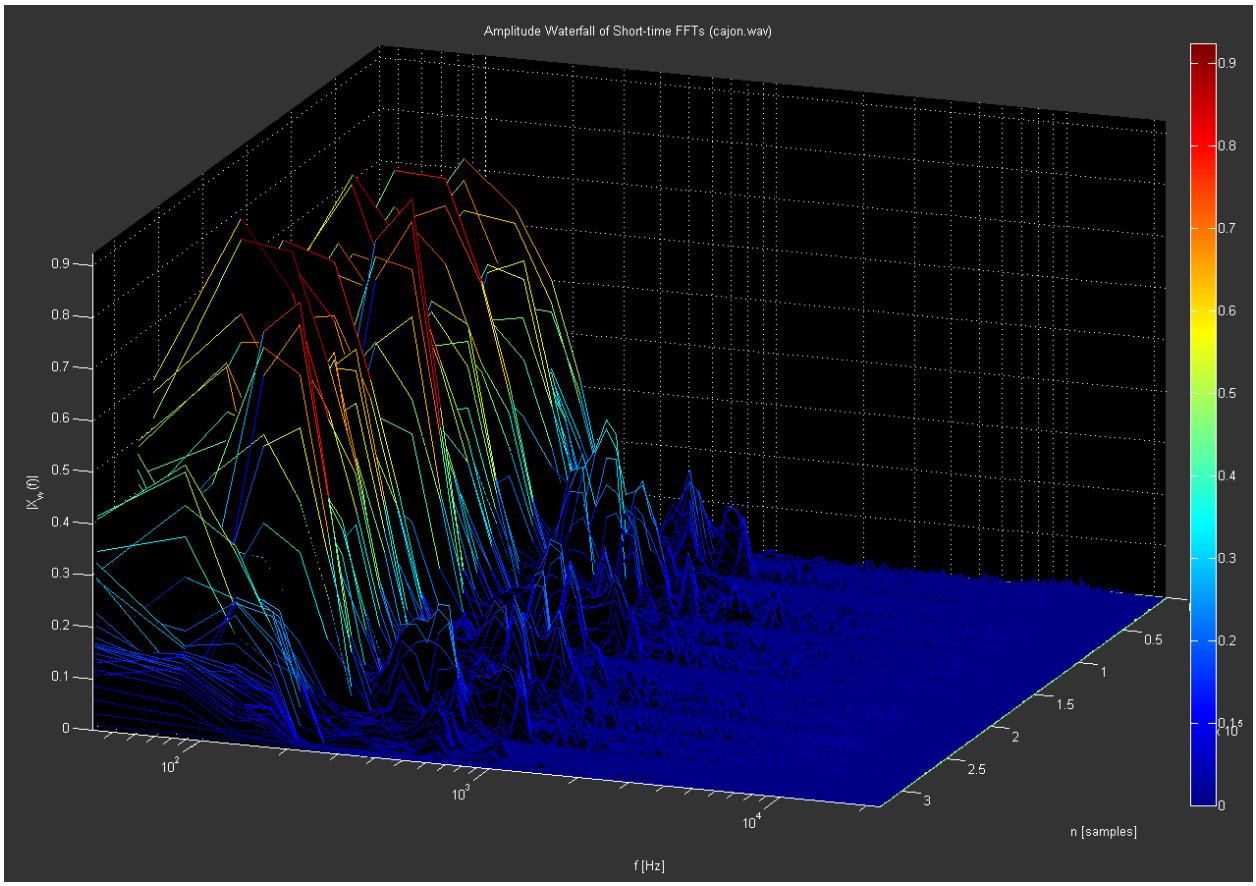


Figure 55: Cajun percussion dry amplitude waterfall as analyzed with a phase vocoder; FFT size is 1024 samples.

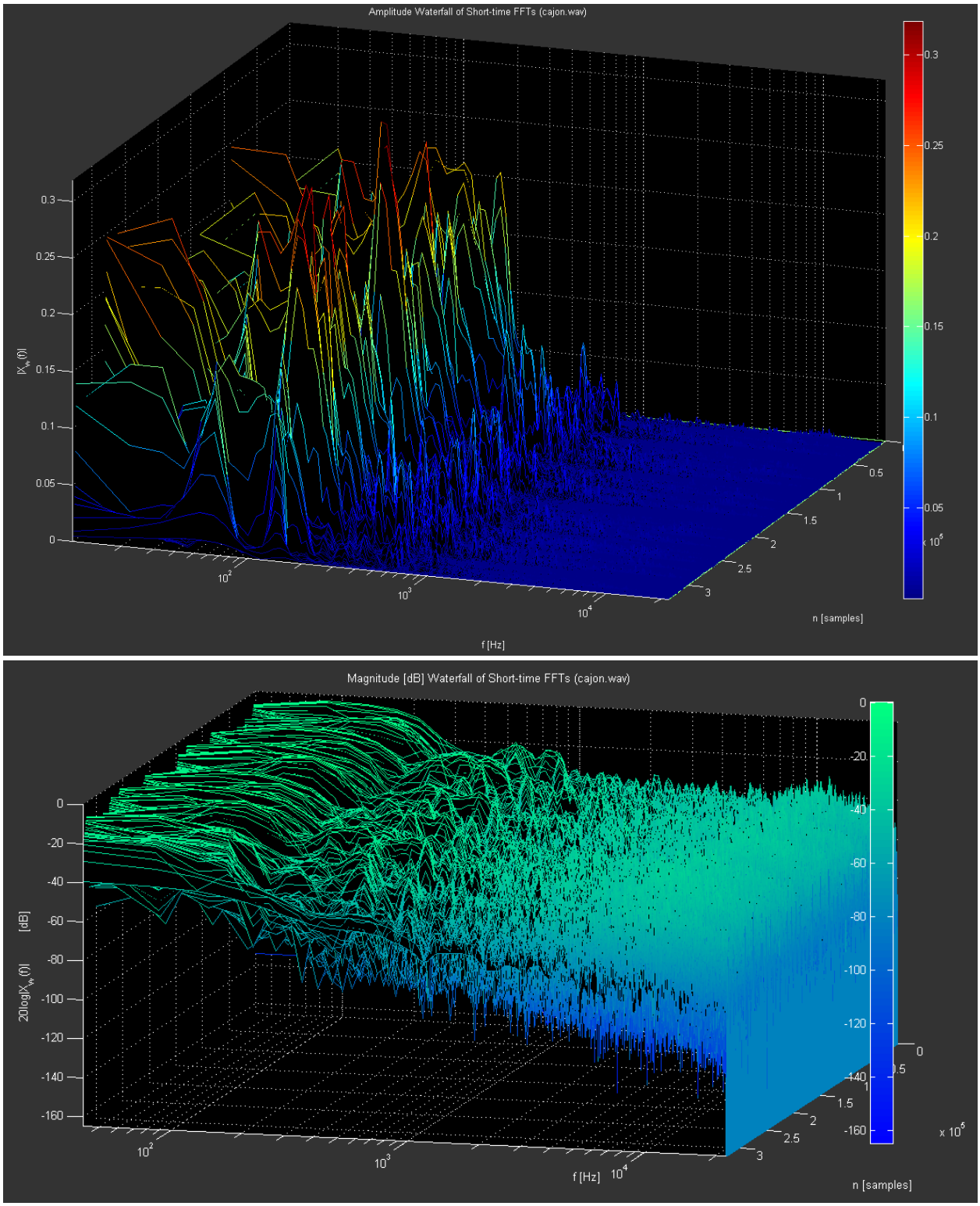


Figure 56: Cajon percussion dry amplitude (top) and magnitude (bottom) waterfalls.

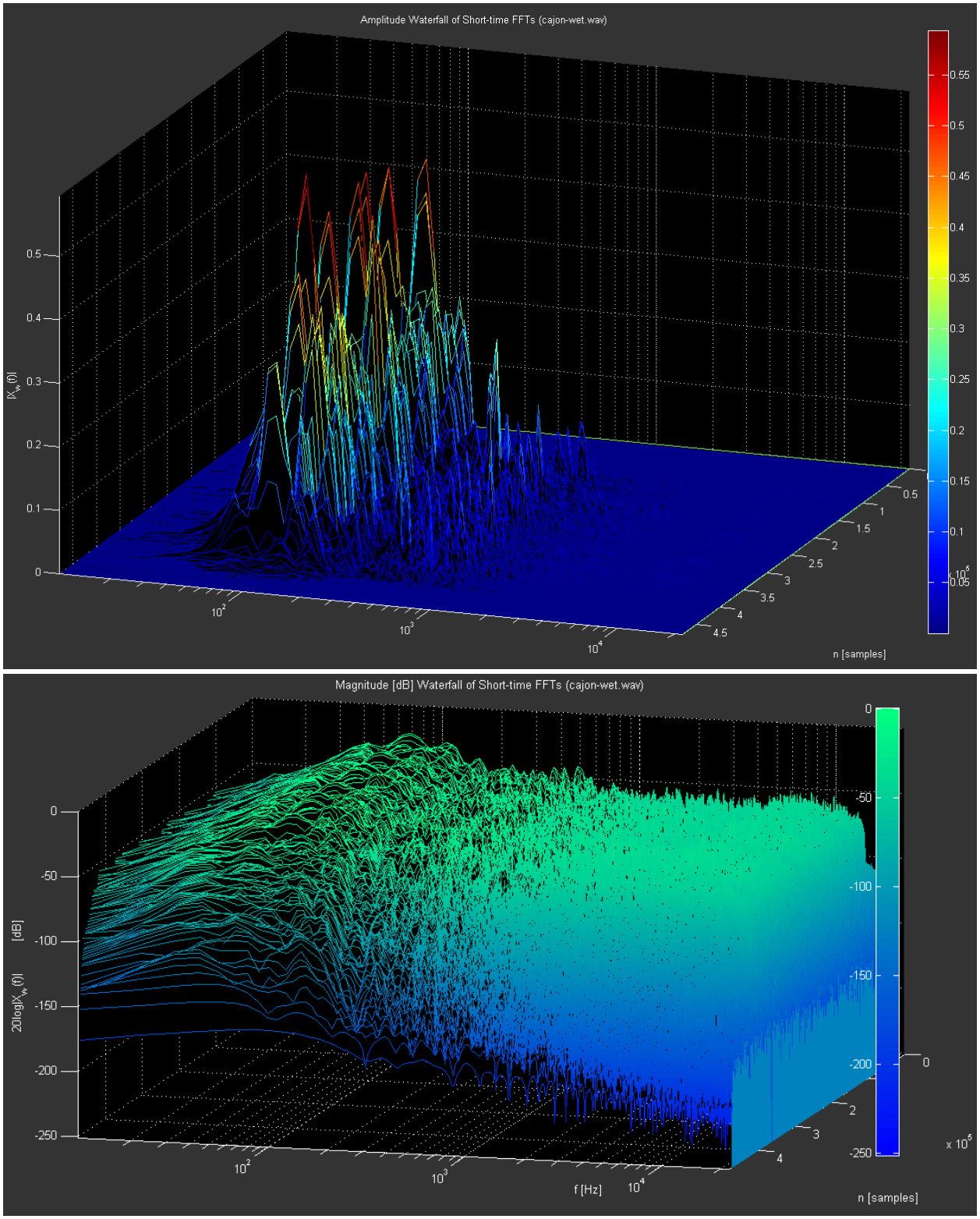


Figure 57: Cajon percussion convolved with L1-BC center microphone IR (waterfall amplitude and magnitude representations). FFT size is 4096.

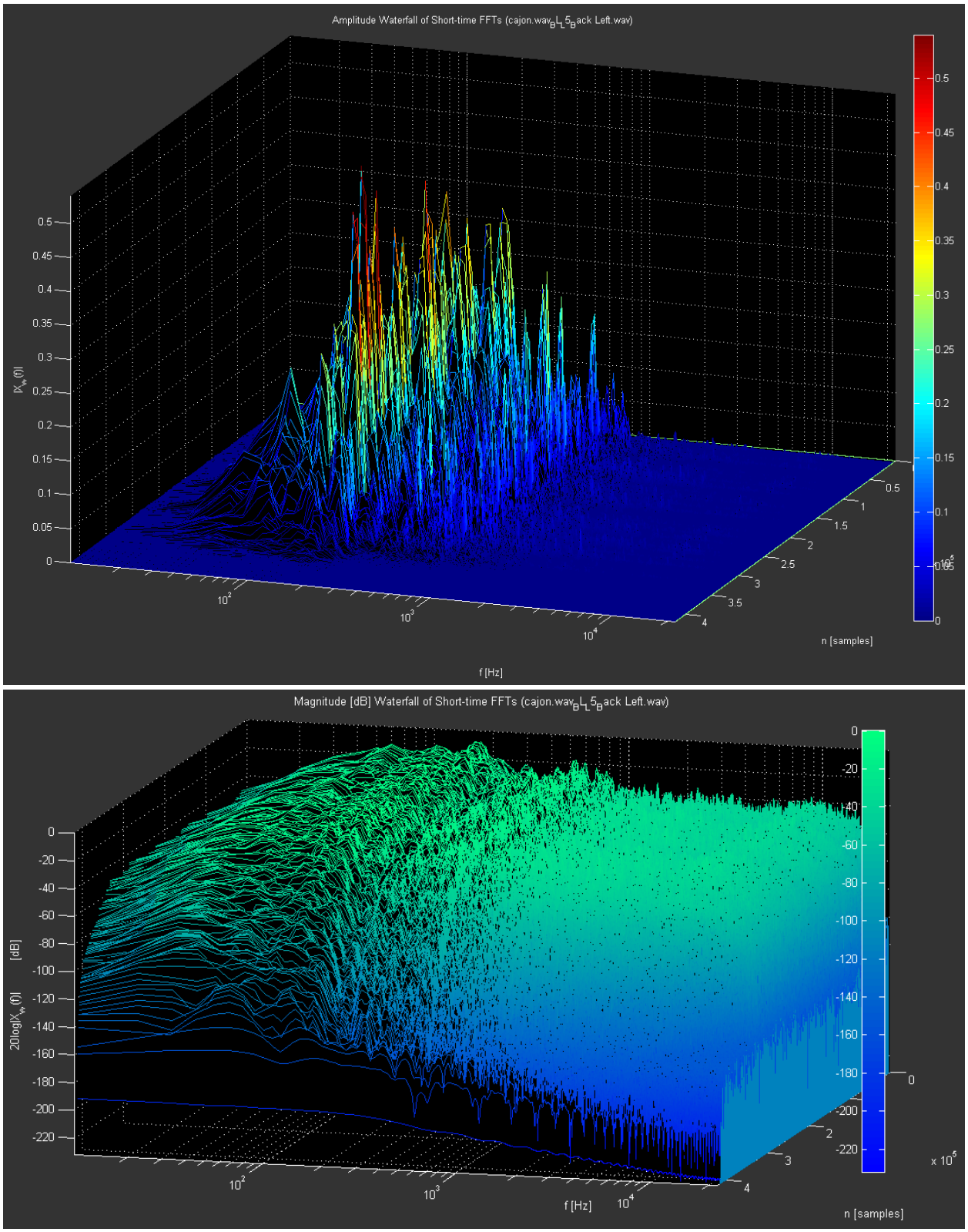


Figure 58: Cajon percussion convolved with L5-BC center microphone IR (waterfall amplitude and magnitude representations). FFT size is 4096.

7.2. Surround Sound Convolution Reverb

For a reverberant sound field to be well replicated and reproduced over loudspeakers, multi channel playback is important. Lateral sound imaging relies on the fact that we have two ears. Additionally, we are able to distinguish sounds behind us, above us, and below us using frequency based *pinna cues.* The pinna is the external part of our ear; it acts as a filter, having a varying frequency response that depends on the direction of incident sound. Pinna cues fill in the spatial imaging deficiencies of intensity and timing cues, allowing for surround sound and 3D audio perception. Spatial impression, however, is very much defined by the timing differences of diffused reflections as they reach our two ears, slightly out of phase with one another. Even if we ignore the need for spatial imaging, convolution reverb begs to be reproduced over multiple channels – this is to help our ears recreate the subtle comb filtering effect that we perceive as 'ambience'. Playing back a source panned to center with stereo reverb versus a source with monophonic reverb, one can confirm that monophonic reverb sounds 'one dimensional'. More channels lead to enhanced spaciousness, and if implemented correctly, more natural sounding reproduction of an ambient acoustic signal. For our purposes, that acoustic signal is a dry recording convolved with the 5 channel surround sound IR of an acoustic environment.

In order to create surround sound convolution reverb, the dry input audio is convolved with each IR from the surround microphone array. High speed convolution was implemented, as described in the previous section. Here, we extend that FFT convolution scheme to surround sound; for the sake of clarity, however, a general convolution notation will be used. $h(n)$ and $y(n)$ from above will be extended to their multi-channel versions**,** $\boldsymbol{h(n)}$ and $\boldsymbol{y(n)}$, where:

$$\boldsymbol{h(n)} = [h_c(n) \quad h_l(n) \quad h_r(n) \quad h_{bl}(n) \quad h_{br}(n)] \tag{12}$$

$$\boldsymbol{y(n)} = [y_c(n) \quad y_l(n) \quad y_r(n) \quad y_{bl}(n) \quad y_{br}(n)]$$

$\boldsymbol{h(n)}$ is a vector representation of the center, front left, front right, back left, and back right microphone channels, and $\boldsymbol{y(n)}$ is a vector representation of the center, front left, front right, back left, and back right output channels (for playback or channel rendering, depending on the playback scheme).

Each channel is convolved with the input using the block-by-block FFT convolution scheme, such that the following convolutions are performed:

$$y_c(n) = h_c(n) * x(n)$$
$$y_l(n) = h_l(n) * x(n)$$
$$y_r(n) = h_r(n) * x(n)$$
$$y_{bl}(n) = h_{bl}(n) * x(n)$$
$$y_{br}(n) = h_{br}(n) * x(n)$$

It’s important that the peak amplitude and timing differences of the original room excitation recording are preserved, and referenced for scaling the output. These will transfer to the intensity and timing cues that are required for surround sound playback. Since the IRs were normalized using a common 0dBu reference, each output channel can simply be scaled to the peak level of the corresponding microphone’s IR.

The following images illustrate the surround sound convolution reverb process. We take a dry, mono signal and virtually place it as a distant source in a reverberant, surround sound acoustic environment.

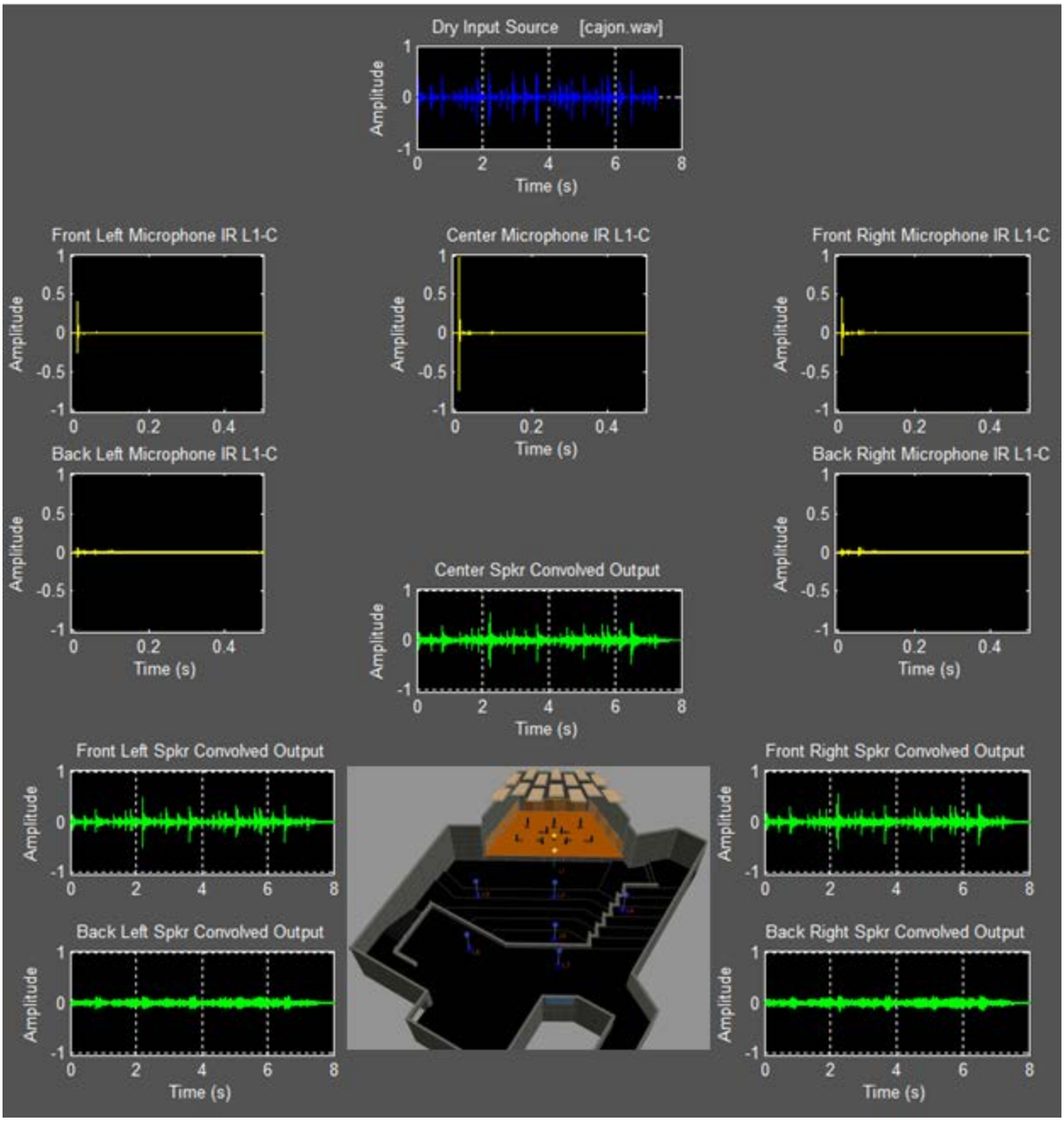


Figure 59: Percussive input source convolved with surround sound array IRs; Array-Source configuration L1-C. Convolved audio has been scaled such that the highest amplitude channel is normalized to the peak input level.

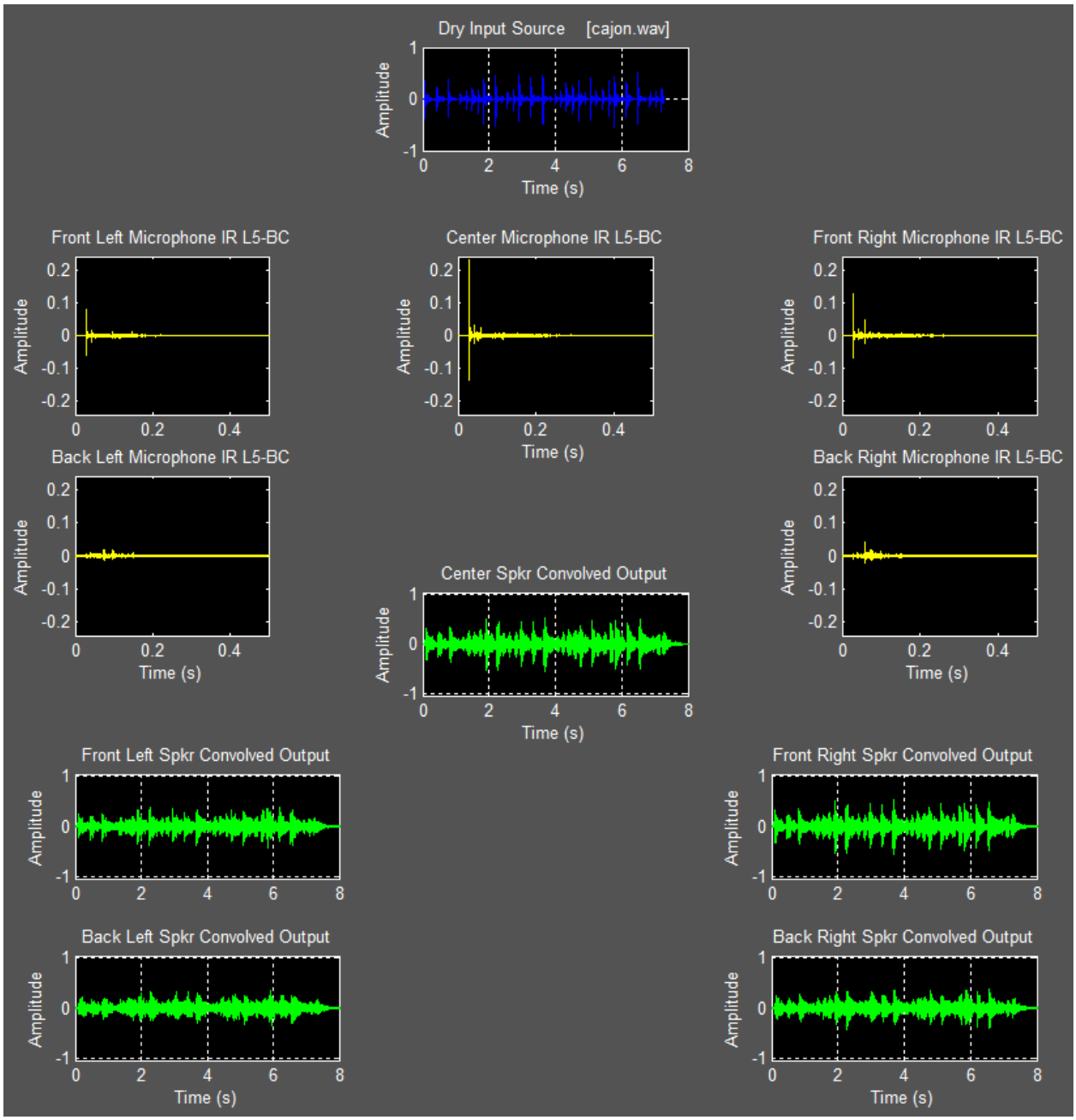


Figure 60: Percussive input source convolved with surround sound array IRs; Array-Source configuration L5-BC. Convolved audio has been scaled such that the highest amplitude channel is normalized to the peak input level.

Comparing Figure 59 and Figure 60 we can observe that as the microphone array and source are moved further apart, the convolution with the reverberant hall smoothes out the transients of the signal, resulting in less clarity, but more ambience. At locations further from the source, the difference in levels between the various microphone channels is smaller. One we get out from the critical distance, the direct source sound is more difficult to image, as the sound intensity of the reverberant field matches that of the direct source.

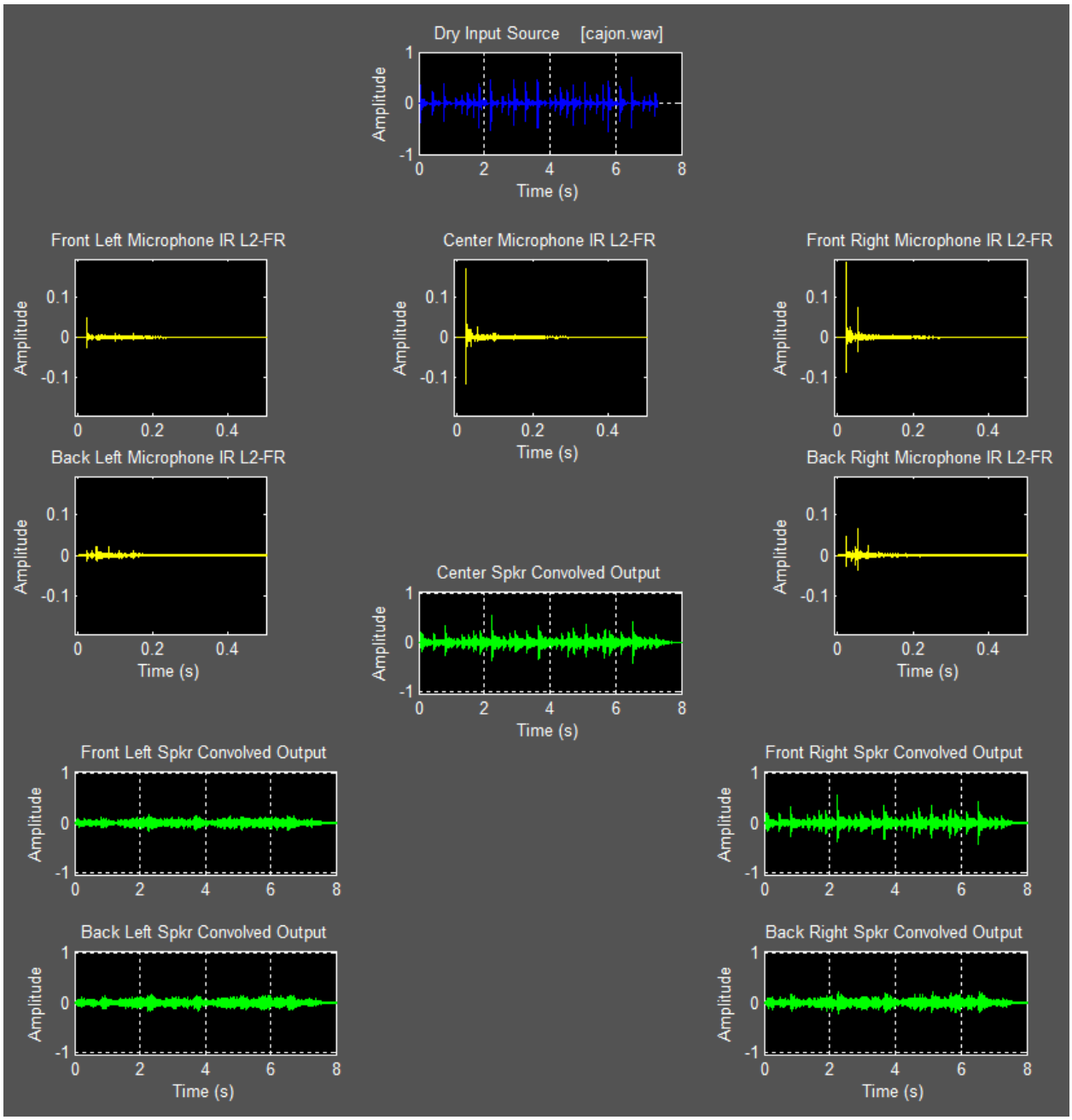


Figure 61: Percussive input source convolved with surround sound array IRs; Array-Source configuration is L2-FR. Convolved audio has been scaled such that the highest amplitude channel is normalized to the peak input level.

8. Multichannel Playback Rendering and Spatial Imaging

For multichannel playback rendering, it is critical that all channels of the convolved signal are normalized to a common amplitude reference. The original amplitude and timing differences recorded during the IR measurement stage must preserved for reproduction. For the circular microphone array, amplitude panning relies on the relative sound intensity picked up by each microphone. The listener depends of the interaural intensity difference (IID) provided by amplitude panning for part of surround sound perception, with interaural timing difference (ITD) and pinna cues being the other main components for spatial imaging of sound. Additionally, realistic depth/distance perception between the different Array-Source locations depends in part on the relative sound intensity recieved at each location. This becomes important when directly comparing different Array-Source configurations during playback.

8.1. Stereo Rendering

Stereo rendering with the circular microphone array requires the use of the 3 forward-facing microphone channels, and involves a small amount if additional processing. The center microphone must be used to aviod the massive “hole in the middle” effect that will result from only using the front left and front right microphones. Stereo amplitude panning between two hypercardioid capsuls that are angled 148˚ apart will essentially bounce left or right in the stereo field; this will be the effect if only the two forementioned microphones are used.

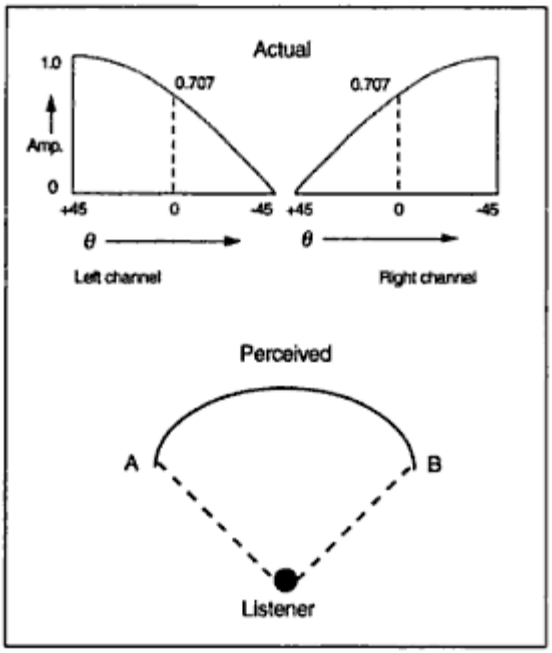


Figure 62: Constant-power panning is required to preserve the sound intensity in the center and facilitate proper stereo imaging [11].

The center microphone must be factored in for proper imaging. A weighting that corresponds to central panning is added to both the left and right speaker channels, such that constant intensity is preserved. Assigning the desired intensity I in the center to be 1, constant power panning is used to find the weight of the center channel that must be added to both the left and right channels:

$$I = 1 = \sqrt{0.5 + 0.5} = \sqrt{0.707^2 + 0.707^2} \tag{13}$$

Therefore, a gain factor of 0.707 (-3dB) is the desired weighting. Let $Cmic$, $FLmic$, and $FRmic$ represent the center, front left, and front right microphone signals, respectively. The desired signal can now be found for the left (L) and right (R) speaker channels of a stereo playback configuration:

$$\begin{aligned} L &= FLmic + 0.707Cmic \\ R &= FRmic + 0.707Cmic \end{aligned} \tag{14}$$

Due to the timing differences betwen the center microphne channel and the left and right channels, the above summation is prone to comb filtering. The near-coincident hypercardiod channels that are being summed together are not in perfect phase with one another; as a result, this system is not perfectly stereo compatable (just as near-coincident stereo recordings are not truely mono compatable). A fixed timing adjustments made to $0.707Cmic$ will not fix the problem, as the timing difference with respect to both $FLmic$ and $FRmic$ will vary with recorded source location.

The *Decca Tree* stereo method, however, uses a similar encoding scheme with great success. The ill-effects of out of phase signal summing should also be more apparent with the Decca Tree, as the basic microphone technique relies purely on timing differences between the microphones for imaging (3 omnidirectional microphones are the classic configuration). With the Decca tree, the comb filtering turns out to be innoffensive, and is percieved as added ambience. Additionally, imaging turns out to be very good for the Decca tree.

We can expect an altered spatial impression when switiching from 3 channel stereo (which requires no rendering) to two channel stereo. However, this is typically the case when moving between multi channel formats.

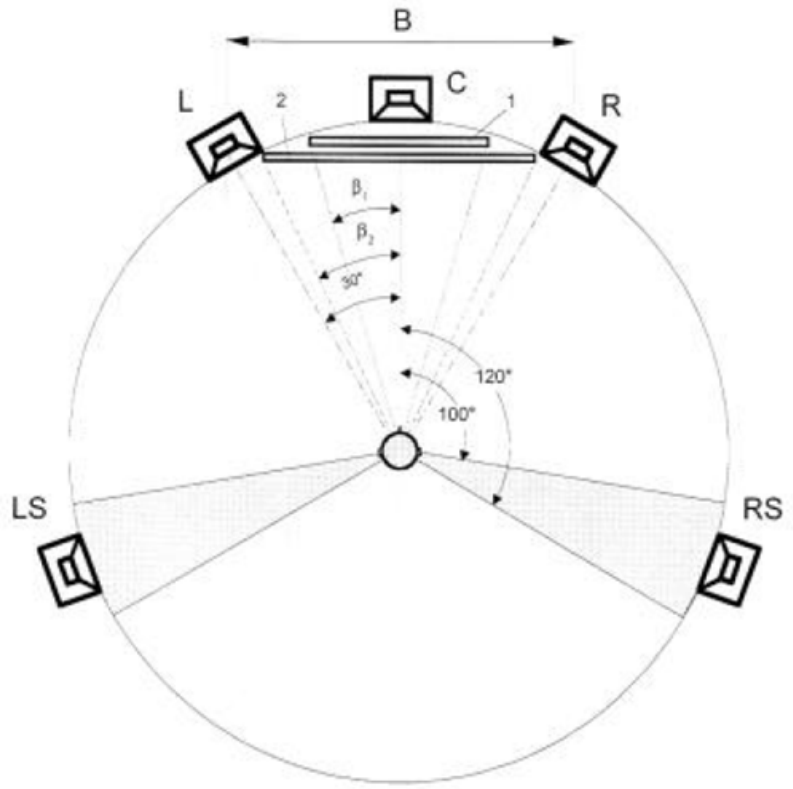
8.2. Playback and Imaging in 5.0 Surround

Complex Spatial Impulse Response rendering is not necessary for 5-channel playback, thanks to the intuitive nature of the microphone array used. For playback, each microphone channel is sent to a separate speaker of a 5 channel surround sound playback configuration (Figure 63). The recommended standard for 5 channel surround playback is documented in [14] and [15].

Unlike binaural playback schemes, we have not encoded the pinna cues required for spatial sound localization. 5-channel surround sound playback over speakers reproduces the intensity and timing differences as picked up by the microphones during IR measurement. Each microphone is directly mapped to one speaker. By placing the speakers in the correct configuration (around the listener), the human ears can recreate lateral imaging with IID and ITD. Additionally, the pinna cues are naturally perceived as a result of speaker placement. Human cognition gives these frequency cues directional meaning, and stereo imaging is expanded to surround sound. A further expansion to 3D imaging can be achieved if the full 7 channel spatial microphone array is utilized for measurement, and the results are mapped to 7 correctly oriented speakers. The two additional speakers would be oriented above and below the listener; similar playback configurations are outlined in [15].

It should be noted that the playback scheme of Figure 63 dues not accurately map the timing and intensity differences of the circular microphone array. The circular surround sound array has equal spacing between each microphone. Alternative speaker configurations are likely to produce superior imaging when using the basic system of mapping each microphone to one speaker. For practical use, however, it is desirable to map to a playback standard that is compatible with other recordings (the standard surround sound configuration of Figure 63 is switchable to the stereo standard).

Imaging with a standard 5 channel reference monitor configuration is typically very good between the front 2 speakers, where stereo reproduction takes place. It is, however, difficult to create stable phantom images outside of this range. This is due in part to the large angle between front L/R and back L/R (labelled LS/RS in Figure 63), and in part to a deficiency in our natural imaging ability as sources move far left and right (we have to rely more on pinna cues, which are less precise to our hearing system that timing and intensity cues). For our purposes with convolution reverb, however, we focus on imaging in front of the listener. The surround sound element is primarily there to add a greater impression of spaciousness, as previously discussed. For extending accurate imaging into surround sound, the speaker configuration should be changed, and ideally additional speakers would be added (for example, a total of 7 speaker channels for surround sound imaging, and additional channels for 3D spatial imaging if desired).



Screen 1: Listening distance = $3H$ ($2\beta_1 = 33°$)
Screen 2: Listening distance = $2H$ ($2\beta_2 = 48°$)
H: Screen height
B: Loudspeaker basis width

Acoustical Center	*Angle*	*Height*	*Tilt*
C	0°	1.2m*	0°*
L, R	±30°	1.2m	0°
LS, RS	±100–120°	≥1.2m	≤ 15°

Figure 63: 5-channel surround sound speaker layout [F2]. Dimensions marked with * are dependent on the height can be taken as specified, assuming no visual display screen is present. For studio monitoring applications, the 120• specification is typically preferred for the rear speakers.

8.3. Spatial Imaging from IR measurements

The spatial imaging that is reproduced with a multichannel playback scheme is dependent on the accuracy of the microphone technique used to record intensity and timing differences between the channels. With the circular hypercardiod array, the potential for accurate imaging is very good. However, the original room excitation when recording the IRs must be considered. Did the excitation source excite the room such that it emitted sound with similar directional characteristics to an instrument/performer located in that same position? Also, did the recorded IRs capture the frequency response in the room at frequencies where the timbre of various musical instruments, including the human voice, is defined?

Locations L3, L5, and L7 exhibit the most natural imaging with sufficient reverb to give an ambient spatial impression. Since the speaker that was originally used for room excitation is not omnidirectional at high frequencies, imaging in L1 will likely reproduce a string quartet inaccurately. This is because real string instruments are more or less unidirectional. The voice of a soprano vocalist, however, may be reproduced with more typical imaging. Those who have recorded vocalists with near-coincident cardioids and spaced pair stereo microphone techniques may have noticed wandering imaging on high notes, particularly if the microphone pair was sufficiently close to the vocalist. This can happen when the voice is being projected in a certain direction, and the vocalist turns her head. In such cases, imaging is affected not only by the location of the performer in relation to the microphones, but also by the direction in which the sound is emitted. To relate this to our IR measurements: in speaker locations close to the microphone array, imaging in the mid to high frequencies is affected by the direction in which the speaker was facing during room excitation.

9. Multi-Source Convolution and Interactive Listening

For interactive listening where a listener is virtually placed in the PTY hall, the original amplitude encoding with respect to the 0 dBu reference must be preserved. Also, if multiple sources are being virtually placed in the acoustic space, timing differences between different Array-Source configurations need to be preserved. Listening to a virtual sound from location L1, the listerner should recieve a greater sound intensity than if they are listening from location L7 – attenuation of sound energy must be presereved. The listener should also hear a sound that is played from source position C earlier than a sound that is played from source position BC, due to the finite speed of sound.

9.1. Virtual Placement of Phantom Sources

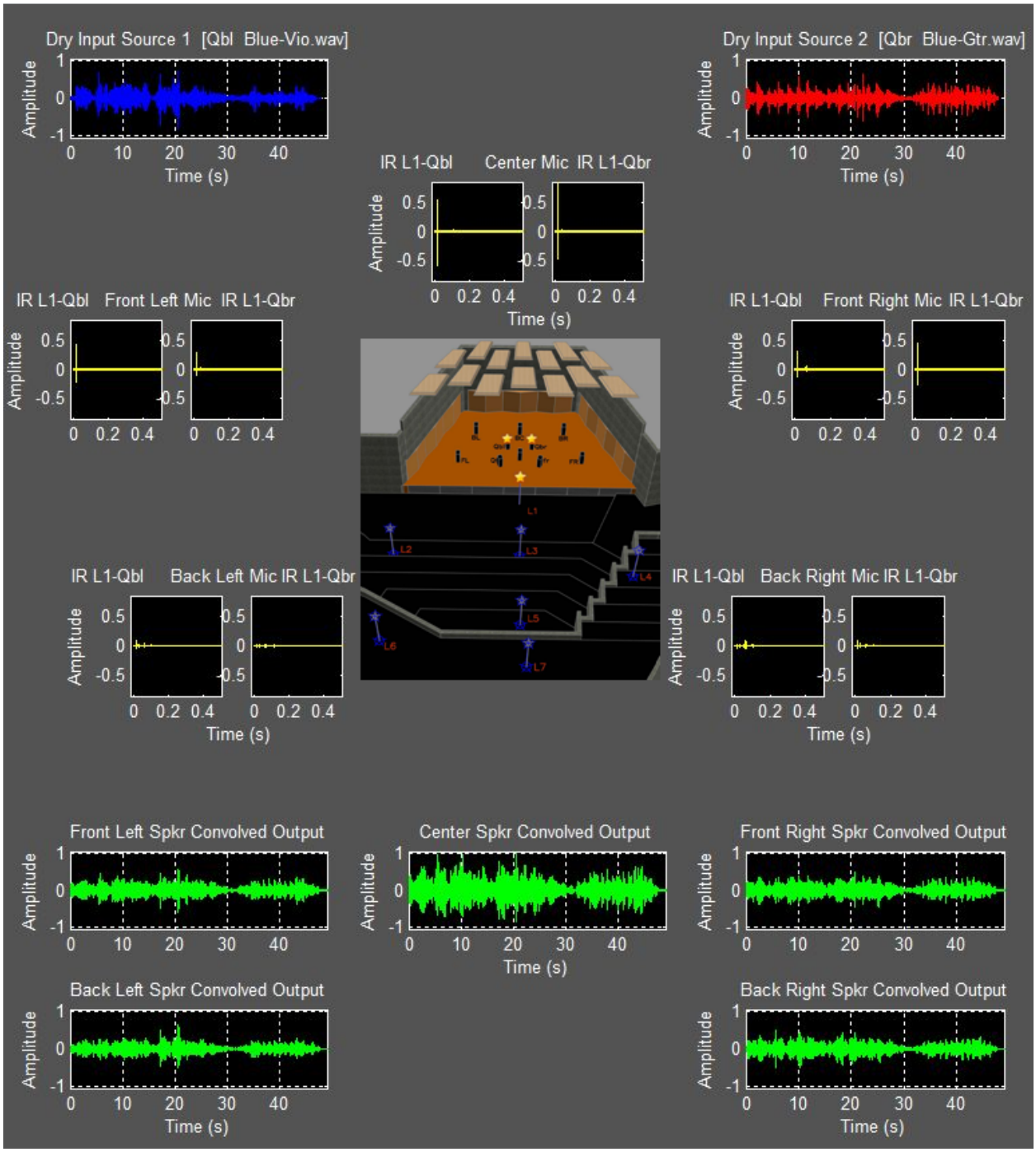
Through superposition, the effect of multiple sources at multiple stage locations can be created. Phantom sound sources can be virtually relocated to any defined stage position, changing the relative locations of the recorded performers (or whatever the sound sources may be) in the soundscape.

Multi-source convolution reverb is accomplished by using a different MMRIR for convolution with each dry signal. Specifically, the source positions from which room excitation occurred during measurement should be different. The microphone array position should be kept the same, as this is the listener's point of perspective. To combine the multiple convolutions into a single surround sound representation, the results of each convolution are added together for each microphone channel. For 2 sources $x_1(n)$ and $x_2(n)$ being convolved with the MMRIRs $\boldsymbol{h_1(n)}$ and $\boldsymbol{h_2(n)}$, respectively, the output channels can be represented as:

$$\begin{aligned} y_c(n) &= h_{1c}(n) * x_1(n) + h_{2c}(n) * x_2(n) \\ y_l(n) &= h_{1l}(n) * x_1(n) + h_{2l}(n) * x_2(n) \\ y_r(n) &= h_{1r}(n) * x_1(n) + h_{2r}(n) * x_2(n) \\ y_{bl}(n) &= h_{1bl}(n) * x_1(n) + h_{2bl}(n) * x_2(n) \\ y_{br}(n) &= h_{1br}(n) * x_1(n) + h_{2br}(n) * x_2(n) \end{aligned} \tag{15}$$

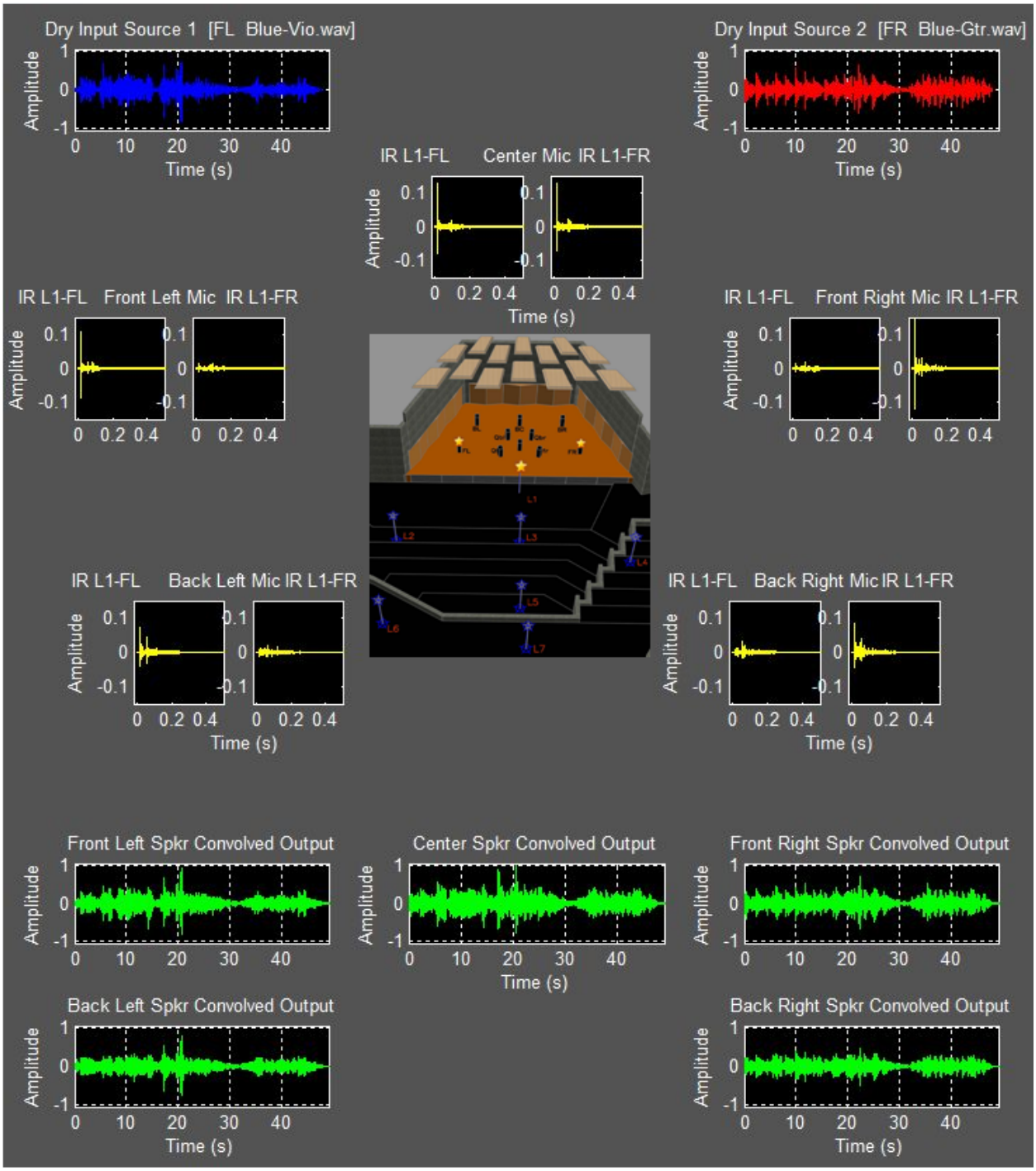
In order to fully preserve depth perception between a close source and far source, each convolved signal should be normalized to its specific IR level (which was normalized with respect to the 0dBu IR) before it is combined with other signals. Renormalization should be conducted after the superposition has occurred, as the combined amplitudes may otherwise cause clipping.

The following figures visually demonstrate the multi-source convolution concept for the case of two sources. The recordings were made in the PTY recital hall. For the first three figures, the input sources are close microphone recordings of a violin-guitar duo, Megan and Adrian Verdejo. In addition, the performance was captured with two stereo microphone techniques: a spaced pair, and an ORTF coincident cardioid configuration in the hanging microphone position. The performers were originally located in similar positions to where their close microphone recordings have been mapped on the virtual stage.



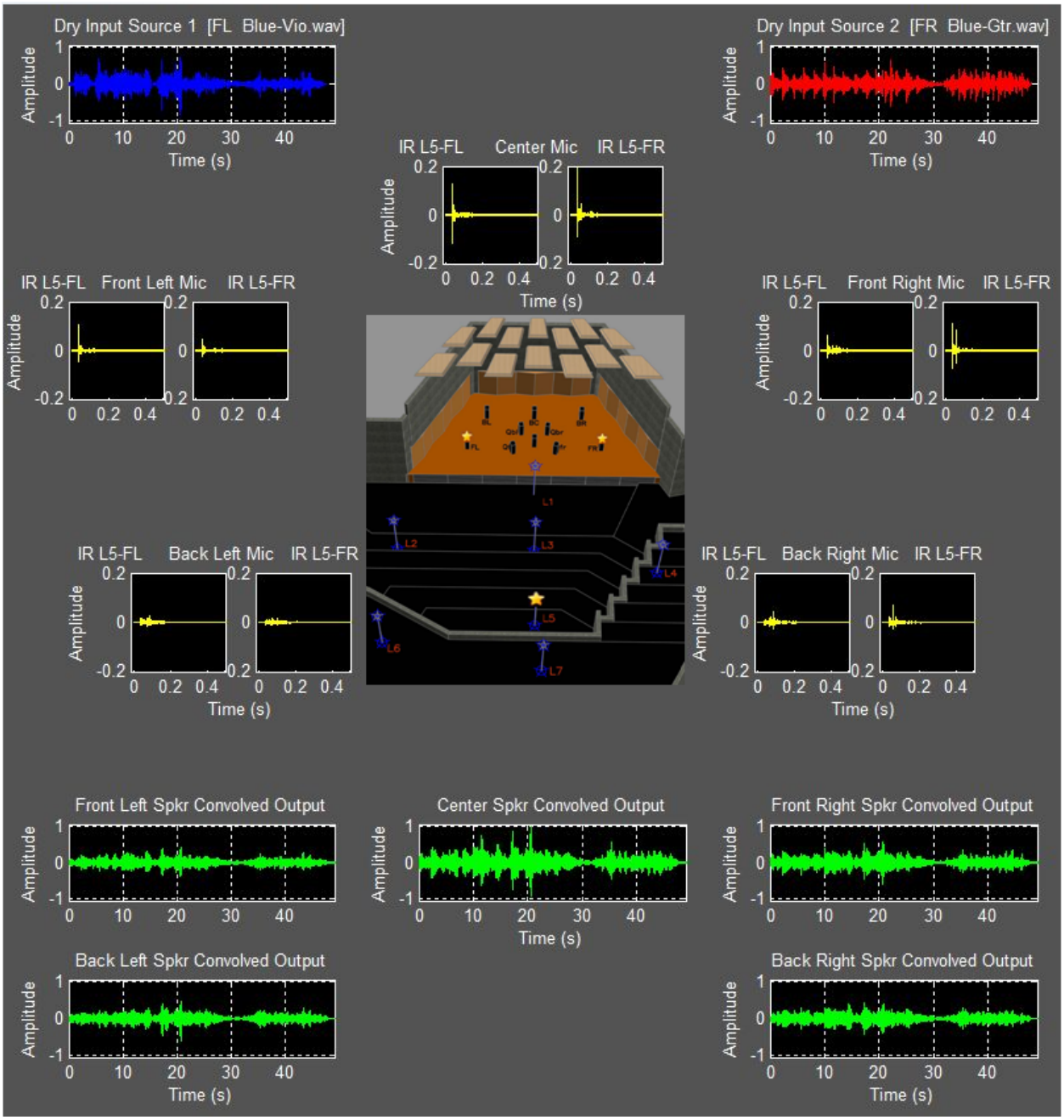
L1-Qbl RIR Peak Channel Levels	L1- Qbr RIR Peak Channel Levels	Output Peak Channel Levels (norm)
Cmic_Max = -4.7255dBu	Cmic_Max = -1.8927dBu	Cspkr_Max = -0.17548dBu
FLmic_Max = -7.6029dBu	FLmic_Max = -11.1816dBu	FLspkr_Max = -4.4967dBu
FRmic_Max = -9.9345dBu	FRmic_Max = -7.2039dBu	FRspkr_Max = -5.9534dBu
BLmic_Max = -21.4557dBu	BLmic_Max = -29.7164dBu	BLspkr_Max = -4.0606dBu
BRmic_Max = -21.4092dBu	BRmic_Max = -21.9011dBu	BRspkr_Max = -6.0948dBu

Figure 64: Dual-source convolution using MMRIR L1-Qbl for the left source and MMRIR L1-Qbr for the right source. Output audio has been normalized for playback, not for comparison between array locations.



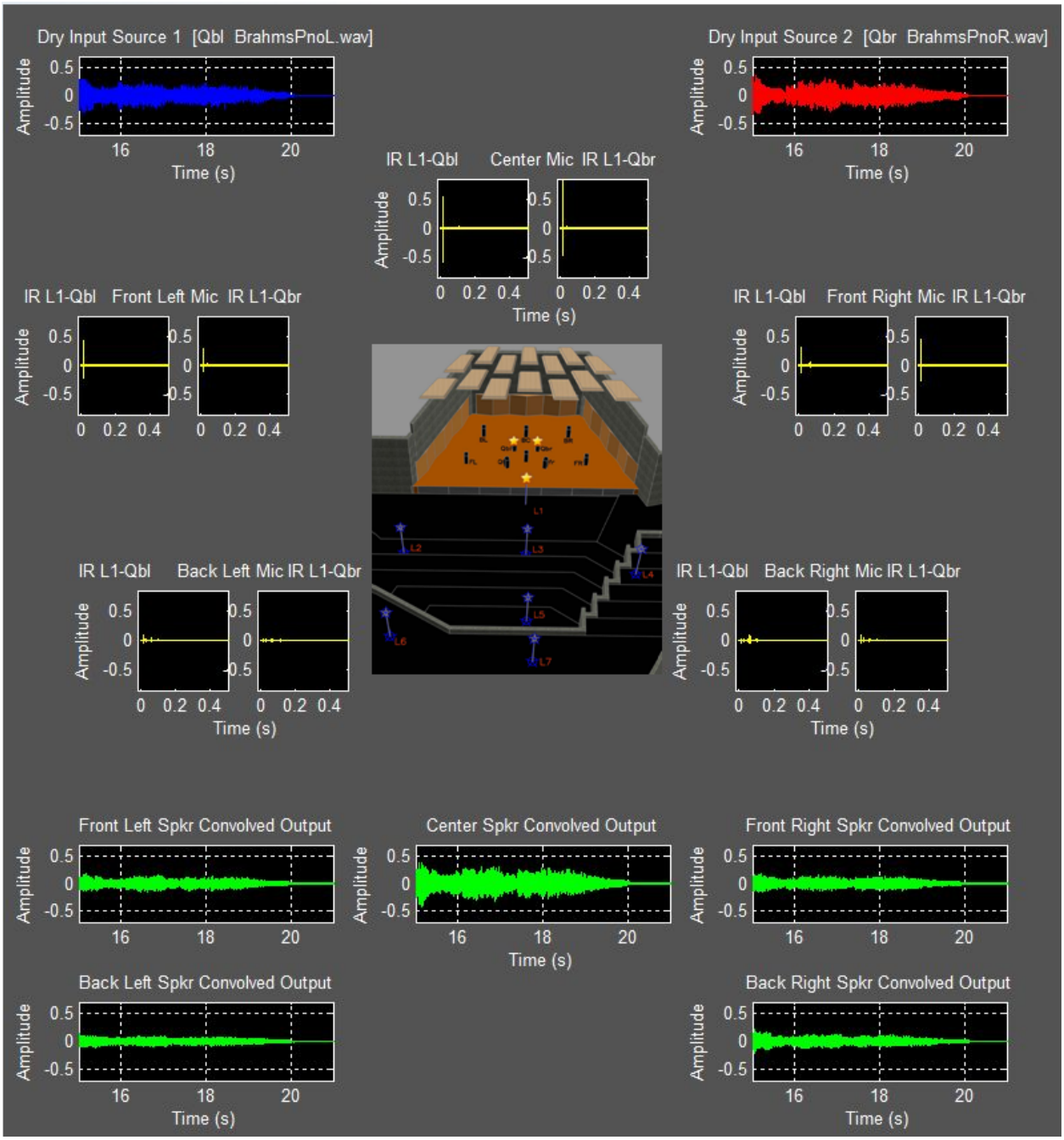
L1-FL RIR Peak Channel Levels	L1-FR RIR Peak Channel Levels	Output Peak Channel Levels (norm)
Cmic_Max = -17.9974dBu	Cmic_Max = -17.9576dBu	Cspkr_Max = -1.0212dBu
FLmic_Max = -19.2317dBu	FLmic_Max = -35.9941dBu	FLspkr_Max = -0.66088dBu
FRmic_Max = -35.1147dBu	FRmic_Max = -16.8907dBu	FRspkr_Max = -2.8778dBu
BLmic_Max = -23.0106dBu	BLmic_Max = -34.2021dBu	BLspkr_Max = -2.115dBu
BRmic_Max = -30.1391dBu	BRmic_Max = -21.4777dBu	BRspkr_Max = -6.0891dBu

Figure 65: Dual-source convolution using MMRIR L1-FL for the left source and MMRIR L1-FR for the right source. Output audio has been normalized for playback, not for comparison between various array locations.



L5-FL RIR Peak Channel Levels	L5-FR RIR Peak Channel Levels	Output Peak Channel Levels (norm)
Cmic_Max = -17.6309dBu	Cmic_Max = -14.2496dBu	Cspkr_Max = -0.16387dBu
FLmic_Max = -19.6129dBu	FLmic_Max = -26.2284dBu	FLspkr_Max = -7.5191dBu
FRmic_Max = -23.7918dBu	FRmic_Max = -18.8325dBu	FRspkr_Max = -4.4497dBu
BLmic_Max = -30.5566dBu	BLmic_Max = -34.5645dBu	BLspkr_Max = -4.4505dBu
BRmic_Max = -27.484dBu	BRmic_Max = -22.9188dBu	BRspkr_Max = -7.2521dBu

Figure 66: Dual-source convolution using MMRIR L5-FL for the left source and MMRIR L5-FR for the right source. Output audio has been normalized for playback, not for comparison between various array locations.



L1-Qbl RIR Peak Channel Levels	L1- Qbr RIR Peak Channel Levels	Output Peak Channel Levels (norm)
Cmic_Max = -4.7255dBu	Cmic_Max = -1.8927dBu	Cspkr_Max = -5.3159dBu
FLmic_Max = -7.6029dBu	FLmic_Max = -11.1816dBu	FLspkr_Max = -10.2824dBu
FRmic_Max = -9.9345dBu	FRmic_Max = -7.2039dBu	FRspkr_Max = -8.5197dBu
BLmic_Max = -21.4557dBu	BLmic_Max = -29.7164dBu	BLspkr_Max = -13.4599dBu
BRmic_Max = -21.4092dBu	BRmic_Max = -21.9011dBu	BRspkr_Max = -11.4778dBu

Figure 67: Dual-source convolution using MMRIR L1-Qbl for the left source and MMRIR L1-Qbr for the right source. Output audio has been normalized for playback, not for comparison between array locations.

If a grand piano was recorded using two or more close microphones (with as little bleed between microphones as possible), is can potentially be represented as a single wide source. For example, the MMRIR for L1-Qbl and L1-Qbr was convolved with the left and right oriented close microphone recordings, and the results were superimposed (Figure 67). Since the piano is hardly a point source, it will not be ideally represented by two point sources (particularly if we are imaging from a close location such as L1). However, it will provide a more realistic listener's point of perspective when compared with the dry recording.

9.2. GUI and Graphical Feedback

A graphical user interface was created so that various MMRIR convolutions with dry audio signals can be compared with one another. The user can select a listening location and place sources at the desired stage position, according to the 3D representation of PTY recital hall.

The visual feedback provided by a 3D graphical representation of the listener-source relationship was helpful for troubleshooting, particularly with regards to imaging. If a convolved audio source does not sound as if it is coming from the supposed mapped location, we know that something is wrong. This did happen during testing, and the next step in troubleshooting involved plotting the RIR and convolved audio magnitude levels on a high resolution timescale, and referring to the measured positions on a graphical representation. With this data, we were able to find the problem, and then systematically look for the cause (typically a glitch in one of the Matlab scripts, but experimental errors that occurred during measurement were also found, and corrected). Without these graphical representations, most of the post-processing problems that were encountered would have been less intuitive to solve. Additionally, communicating specific information regarding various MMRIRs would have been difficult.

Lots of IR data was checked graphically. However, with a large data set, plotting and visually analyzing every outcome is not practical. This is audio after all, and our hearing has a very tuned sense of what sounds right. If the picture matches the sound, and the normalized levels are correct, then we have a reasonable working result. If not, then we move beyond the GUI abstraction and dive deeper into the problem area.

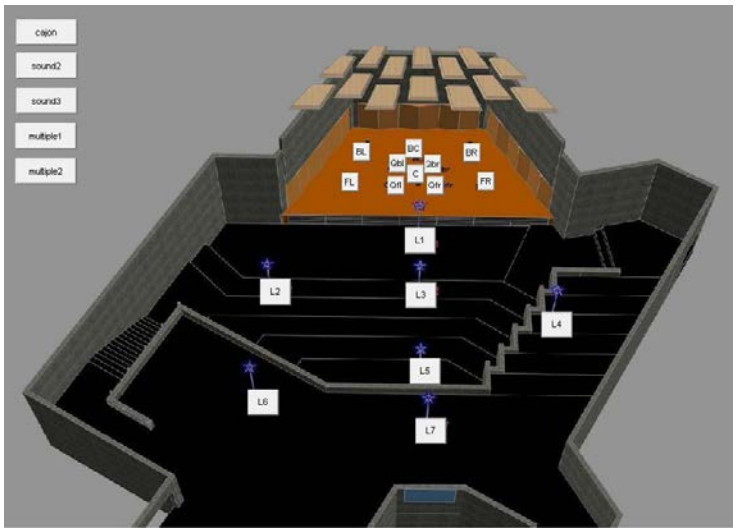


Figure 68: Basic GUI implementation for testing convolution reverb results. Push buttons are used to select listener location and source positions.

A further development of the GUI into an applet could make a very effective web presentation relating to acoustics. In addition to audio playback and a 3D representation of listener and source in the hall, additional displays could be added. For example, the corresponding IRs, peak level or intensity information, acoustic parameters, and waterfall representation could be combined to create a tool for conceptually learning the principles of acoustics.

10. Additional Applications

Data was obtained that reveals meaningful information about the ITDG, the behaviour of the first reflections, and RT60. This information is potentially very useful for recording engineers when selecting microphone and source placements. Additionally, the dataset can be used to reveal problematic areas with undesirable acoustic parameters or offensive first reflections, and acoustic treatment can be applied. Acoustical engineers will likely find surround sound IR data valuable when designing potential acoustic spaces that have similarities to existing acoustic spaces. While modern engineering relies heavily on high quality simulations, physical measurements frequently reveal behaviour that modeling does not predict. By analyzing real world data, it may be possible to avoid certain design decisions that can lead to unexpected acoustical phenomenon. Similarly, IR measurements can be extended to stress analysis in fields such as structural engineering. Since resonance can lead to structural failure, a suitable frequency response for each stressed component is important for structural integrity.

Timing and amplitude difference between the microphones are revealed with the surround sound IR plots. This data can be used to evaluate the performance of the microphone configuration used with regards to imaging (through intensity and timing differences). Moreover, combined with the acoustic parameters data relating to the first reflections, the information gathered from the

surround microphone array can be used for analyzing source-listener relationships in the acoustic space. By adding the two vertical channels to the setup, 3D spatial imaging can be studied.

11. Conclusions

High quality surround sound impulse response measurements were made using the exponential sine sweep (ESS) method. Key locations in the Philip T. Young Recital Hall were chosen for placement of the excitation speaker and the circular microphone array during room excitation recordings. The result is 70 multiple-measured room impulse responses.

Selected measured IRs were analyzed, revealing a maximum SNR of about 100 dB in locations that received the highest sound intensity during playback, and a dynamic range consistently over 80 dB for the batch. These SNR finding are exceptionally high. The results support the findings in [1], [2], [3], and [5], which present evidence that the ESS method is superior to traditional methods of IR measurement in terms of dynamic range potential. Dynamic range results were compared with a RIR that was previously measured in PTY recital hall, for which synchronous averaging of linear sine sweeps was used. Here, the ESS results point to a significant dynamic range improvement; however, original testing conditions are not known in detail. Verifying this finding will require a side-by-side comparison of the two sine sweep techniques, as performed in [1] and [5]. This is beyond the scope of the project documented here, which focuses on the process of surround sound RIR measurement and the application of MMRIRs in convolution reverb.

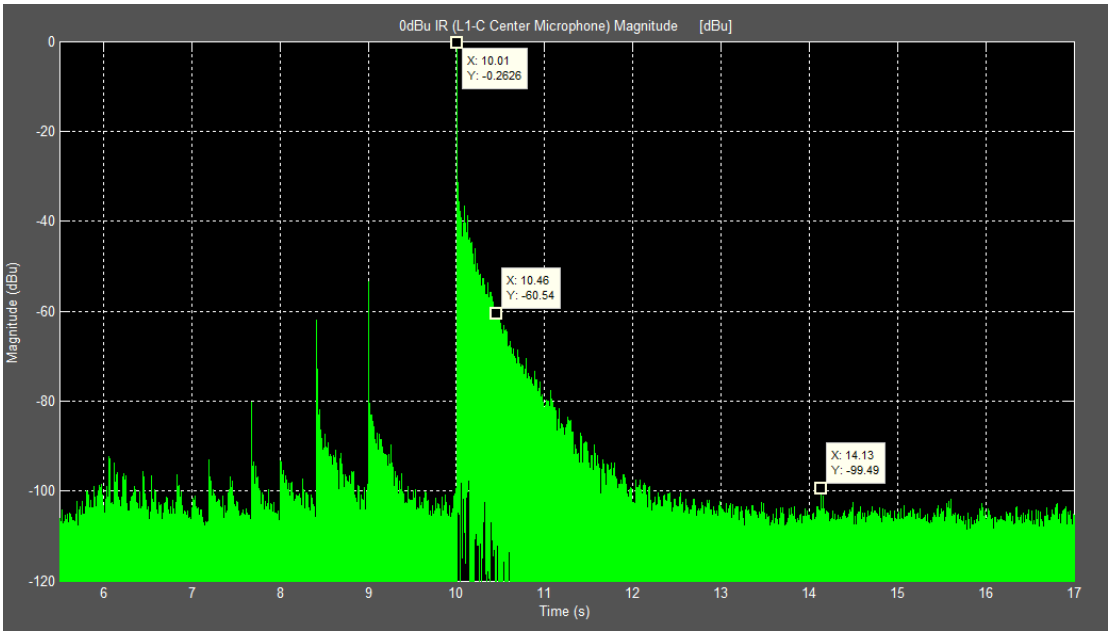


Figure 69: 0 dBu reference IR computed with acyclic time domain convolution. The dynamic range is at least 100dB. If trimmed between 10 and 14 seconds on the above timescale, the dynamic range can be considered to be slightly over 100dB. The -60dB point in this plot does not correspond with the reverb time of PTY Hall, as this measurement was taken close to the excitation source.

MMRIRs were analyzed both in terms of their individual channels, and as a combined entity for surround sound imaging. A detailed acoustical analysis involving an algorithmic extraction of acoustic parameters was not performed, due to time constraints. Several RT60 and ITDG estimations were made for selected measurements. These estimations were conducted by analyzing IR measurements using time-frequency representations and multi-channel time domain representations. Graphical measurement techniques were used for the estimation of these parameters; they were also used for troubleshooting software problems while developing Matlab scripts, and as aids for correcting experimental errors involving microphone misalignment.

Surround sound convolution reverb was implemented using two variations of FFT convolution. Switching from cyclic FFT convolution with raised cosine windows, cyclic shift, and overlap add with a fixed window size to *high speed convolution* using rectangular windowed zero padded segments, a speedup of orders of magnitude was achieved (for a 3 second IR convolved with a 7 second audio clip, computation time was cut by a factor of over 500).

Listening back to convolved audio revealed several problem locations where the imaging did not match a 3D representation of the source-microphone array locations in PTY Hall. Inconsistencies between the IRs and the resulting convolved audio were typically due to software bugs that caused relative amplitude scaling problems between IRs, and were later corrected based on graphical feedback.

Multi-source surround sound convolution reverb with user-configurable listening positions and source locations was created. With this system, the acoustics of PTY Recital Hall as a whole may be explored from multiple perspectives. Additional graphical feedback may be built into the existing GUI, which will provide an abundance of qualitative information about the acoustic space. A further development could include the display of measured acoustic parameters, and might be useful for selecting potential microphone placement for recordings, or evaluating the differences between performer locations on stage.

Surround sound listening tests in a controlled playback environment must be performed before concrete conclusions regarding phantom source imaging can be reached. These will be conducted in the very near future.

3D spatial IR measurements will be performed in the future using the 5 channel circular microphone array, in addition to the 2 vertically oriented shotgun microphones. The results of these measurements will be useful for spatial localization of sound using early reflection data and ray tracing. Additionally, 3D audio rendering for possible playback over a custom speaker configuration could be performed. A quality 3D MMRIR data set may be used to realize 3D convolution reverb, or 3D acoustic simulations.

12. References

CITED REFERENCES

[1] Farina, A., “Advancements in impulse response measurements by sine sweeps”, *Presented at the 122ed AES Convention, Vienna, Austria*, May 2007.

[2] Farina, A., “Simultaneous measurement of impulse response and distortion with a swept sine technique”, *Presented at the 108th AES Convention, Paris, France,* 2000.

[3] Meng, Q.; Sen, D.; Wang, S.; Hayes, L., "Impulse response measurement with sine sweeps and amplitude modulation schemes," *Signal Processing and Communication Systems, 2008. ICSPCS 2008. 2nd International Conference on* , vol., no., pp.1-5, 15-17 Dec. 2008.

[4] Norcross, S.; Soulodre, G; Lavoie, M., “Further Investigations of Inverse Filtering”, *Presented at the 115th AES Convention, New York, New York*, 2003.

[5] Stan, G.; Embrechts, J.J.; Archambeau, D., –“Comparison of Different Impulse Response Measurement Techniques”, *JAES* Vol. 50, No. 4, p.249, 2002 April.

[6] Driessen, P., Behrins, C., [Impulse response measurements], Philip T. Young Recital Hall, University of Victoria, Canada, 2004.

[7] Wang, G.; Perry, C.; Misra, C.; Tzanetakis, G., *Sndpeek: real-time audio visualization,*[Software], Princeton University, 2007.

[8] Farina, A., *Aurora 4.0 Plug-ins for Adobe Audition*., [Software], Italy: LAE Group, 2007.

[9] Hodgson, J. (Jan. 1973), “Recital Hall Floor Plan & Elevations, Recital Hall Seating, Recital Hall Banners and Clouds, Wing of the Maclaurin Building, University of Victoria” Drawing no. 41-47 (Rev: as built), Ref no. A155-A161, *Architectural Drawings.*

[10] Udo Zölzer, *DAFX*., (John Wiley & Sons, 2002., West Sussex)

[11] Roads, C., *The Computer Music Tutorial*, (MIT Press, 1996)

[12] Smith, W. (2007), *The Scientist and Engineer’s Guide to Digitial Signal Processing*, [On-line], Available at FTP: http://www.dspguide.com/pdfbook.htm [2009, July 27].

[13] Smith, J. (2009, Mar.), *Spectral Audio Signal Processing*, [On-line], Available at FTP: http://ccrma.stanford.edu/~jos/sasp/ [2009, July 27].

[14] AES Technical Council, Document AESTD1001.1.01-10, “Multichannel surround sound systems and operations”, 2008. [2009, July 26], Available at FTP: http://www.aes.org/technical/documents/AESTD1001.pdf

[15] Huber, D.; Runstein, R.; Modern Recording Techniques, 6 ed., Oxford: Focal Press, 2005.

GENERAL REFERENCES

Müller, S.; Massarani, P., – “Transfer-Function Measurement with Sweeps”, *JAES* Vol. 49, Number 6 pp. 443 (2001).

Everest, F., The Master Handbook of Acoustics, 4 ed. , New York: McGraw-Hill, 2001.

Beranek, L., " Concert Hall Acoustics—2008*," J. Audio Eng. Soc., vol. 56, no. 7/8, pp. 532-544, 2008 July/August.

CITED FIGURES [On-line]

[F1] Sound on Sound, [Online Image], 2006 Available at FTP: http://www.soundonsound.com/sos/may00/articles/reverb.htm 2006 [2009 July 25].

[F2] AES Technical Council, Document AESTD1001.1.01-10, “Multichannel surround sound systems and operations”, [Online Image], 2008. Available at FTP: http://www.aes.org/technical/documents/AESTD1001.pdf [2009, July 26].

[F3] Li, Y., “Frequency Sweep Method of Analysisng Room Acoustics and Microphone Response”, [Online Image], 2006, Availabe at FTP: http://www.ece.uvic.ca/~yli/upmix/hall/ir_mic.html [2009, July 28].

APPENDICES: SELECTED MATLAB EXAMPLES

Appendix A - Test IR MATLAB Scripts

```
%Elec 499 Summer 2009 Hudson Giesbrecht        Last Rev: Tim Perry 09-07-06
%
% This script runs a test to verify the Exponential Swept Sine
% method for recorded room impulse measurements % as outlined by
% Angelo Farina in his paper "Simultaneous Measurement of Impulse Response
% and Distortion with a Swept Sine Technique"
%
% This script verifies the method by filtering the input with a single tap
% delay. It is also verified that any harmonic distortion artifacts are
% packed behind the impulse reponse where they can be windowed out
%
%The inverse filter used is an approximation

clear all;
close all;

T = 1;%(2^N)/fs;
f1=20; %starting frequency
f2=20000; %ending frequency
fs=44100;

% Create the swept sine tone
w1 = 2*pi*f1;
w2 = 2*pi*f2;
K = T*w1/log(w2/w1); %parameter for creating the sweep
L = T/log(w2/w1); %%parameter for creating the sweep
t = linspace(0,T-1/fs,fs*T); %time vector
s = sin(K*(exp(t/L) - 1)); %sweep vector
LT=length(s);


%Create Inverse Filter: Ideally is the reverse of the sweep with a slope of
%20dB per decade (6dB per octave) to compensate for the difference in
%energy at low and high frequencies
%
%Our inverse filter is missing amplitude scaling and does not have a
%perfect 20dB per decade slope

x=linspace(f1,f2,length(s)); %This inverse filter is not perfect
slope=fliplr(x).^3;          % but is sufficient to prove the method works
sinv=slope.*fliplr(s); %Create the inverse filter
sinv=sinv./max(sinv); %normalize the inverse filter
SINV=fft(sinv);


%=====================================================================
% Different approach to inverse filter (based on eq (7) report)
%=====================================================================
% f_range=linspace(f1,f2,length(s)); %This inverse filter is not perfect
% A = f1;
% m = f1./f_range;      % modulating function (8)
% s_rev=fliplr(s);     %time reveresal of s
% sinv2 = m'*s_rev;
```

```
% sinv2=sinv2./max(sinv); %normalize the inverse filter
% SINV2=fft(sinv2);


figure(1)
NFFT=LT;
S = fft(s);
f = fs/2*linspace(0,1,NFFT/2); %create plotting vector
subplot(2,1,1), loglog(f,abs(S(1:NFFT/2)/max(abs(S)))) % Plot single-sided amplitude
spectrum of the sweep
title('Amplitude Spectrum of the Exponential Sine Sweep')
ylabel('Amplitude (dB)')
xlabel('Frequency (Hz)')
subplot(2,1,2), loglog(f,abs(SINV(1:NFFT/2))/max(abs(SINV))) % Plot single-sided
amplitude spectrum of inverse filter
title('Amplitude Spectrum of the Inverse Filter')
ylabel('Amplitude (dB)')
xlabel('Frequency (Hz)')


figure(2)
subplot(2,1,1), plot(s)
title('Exponential Sine Sweep')
ylabel('Amplitude')
xlabel('Frequency (Hz)')
subplot(2,1,2), plot(sinv)
title('Exponential Sine Sweep Inverse Filter')
ylabel('Amplitude')
xlabel('Frequency (Hz)')
SINV=fft(sinv);

%convolve sweep and inverse filter using Part b of Elec 484 Ass 5
x1=sinv;
x2=s;
x1l=[x1 zeros(1,length(x1)-1)]; %zero pad the input vectors to avoid circular
convolution
x2l=[x2 zeros(1,length(x2)-1)];

X1L=fft(x1l);
X2L=fft(x2l);
YL=X1L.*X2L; %frequency domain multiplication performs time domain convolution
imp=ifft(YL);
imp=imp./max(imp);

%convolve filtered sweep and inverse filter using Part b of Elec 484 Ass 5
delfilt=[1 zeros(1,1000) 1]; %delay filter
sdel=conv(delfilt, s);
x1=sinv;
x2=sdel;
x1l=[x1 zeros(1,length(x2)-1+length(x2)-length(x1))]; %zero pad the input vectors to
avoid circular convolution
x2l=[x2 zeros(1,length(x2)-1)];

X1L=fft(x1l);
X2L=fft(x2l);
YL=X1L.*X2L; %frequency domain multiplication performs time domain convolution
impdel=ifft(YL);
impdel=impdel./max(impdel);

%convolve clipped (distorted) sweep and inverse filter using Part b of Elec 484 Ass 5
%This demonstrates the advantage of using the inverse filter, any
%distortion artifacts are placed behind the impulse and can be windowed out
wavwrite(s*1.9,fs,'clipped_sweep.wav');
```

```
sclip=wavread('clipped_sweep.wav');
x1=sinv;
x2=sclip';
x1l=[x1 zeros(1,length(x1)-1)]; %zero pad the input vectors to avoid circular
convolution
x2l=[x2 zeros(1,length(x2)-1)];

X1L=fft(x1l);
X2L=fft(x2l);
YL=X1L.*X2L; %frequency domain multiplication performs time domain convolution
impclip=ifft(YL);
impclip=impclip./max(impclip);

% IR plots on semi-logarithmic scale
figure(5)
subplot(3,1,1), semilogy(imp/max(imp))
title('Recovered Impulse Response')
ylabel('Amplitude (log scale)')
xlabel('Time (s)')
AXIS([3e4 5e4 1e-5 1])
subplot(3,1,2), semilogy(impdel/max(impdel))
title('Recovered Impulse Response from delayed sweep')
ylabel('Amplitude (log scale)')
xlabel('Time (s)')
AXIS([3e4 5e4 1e-5 1])
subplot(3,1,3), semilogy(impclip/max(impclip))
title('Recovered Impulse Response from distorted sweep')
ylabel('Amplitude (log scale)')
xlabel('Time (s)')
AXIS([3e4 5e4 10e-8 1])
wavwrite(imp*0.9999,fs,'test_impulse.wav');
wavwrite(s*0.9999,fs,'t.wav');

% IR plots on dB scale
figure(6)
subplot(3,1,1)
plot(20*log10(abs(imp/max(imp))))
title('Recovered Impulse Response')
ylabel('Amplitude (dB)')
xlabel('Time (s)')
AXIS([3e4 5e4 -110 0])
subplot(3,1,2)
plot(20*log10(abs(impdel/max(impdel))));
title('Recovered Impulse Response from delayed sweep')
ylabel('Amplitude (dB)')
xlabel('Time (s)')
AXIS([3e4 5e4 -110 0])
subplot(3,1,3)
plot(20*log10(abs(impclip/max(impclip))));
title('Recovered Impulse Response from distorted sweep')
ylabel('Amplitude (dB)')
xlabel('Time (s)')
AXIS([3e4 5e4 -110 0])
wavwrite(imp*0.9999,fs,'test_impulse.wav');
wavwrite(s*0.9999,fs,'t.wav');
```

```
%Elec 499 Summer 2009 Hudson Giesbrecht       Last Rev: Tim Perry 09-07-06
%
%
%Room Impulse Measurements by Exponential Sine Sweep
%Method as outlined by Angelo Farina in his paper
%"Simultaneous Measurement of Impulse Response and Distortion %  with a Swept Sine
Technique"
%The paper can be found at his website http://pcfarina.eng.unipr.it/
%The measurement sweep and inverse filter were generated using his Adobe
%Audition plugin Aurora http://www.aurora-plugins.com/
%
%Running this matlab script will output imp.m which is the impulse response
%recovered from PTY_Ceneter_95_trim.wav which is a recorded sine sweep
%through in the Phillip T Young Hall
%
%The wave files read in below must be in the same directory as this script

clear all;
close all;
s=wavread('10 sec single'); %The sweep file
s=s';
[s_hall,fs,nbits]=wavread('PTY_Center_95_trim'); %The sweep file recorded through the
hall
s_hall=s_hall';
sinv=wavread('10 sec inverse mono.wav'); %Inverse sweep from Aurora
sinv=sinv';
s_hall=s_hall(1:length(sinv)); %Trim the response to the length of the inverse filter
s=s(1:length(sinv));
aur_imp=wavread('PTY_Center_95-Impulse.wav'); %Hall impulse response calculated in
Aurora

s_hall=s_hall/max(s_hall);%Normalize
SINV=fft(sinv);
S_HALL = fft(s_hall);
S=fft(s);


figure(1)
NFFT=length(s_hall);

f = fs/2*linspace(0,1,NFFT/2); %create plotting vector
subplot(3,1,1), loglog(f,abs(S(1:NFFT/2)/max(abs(S)))) % Plot  amplitude spectrum of
the sweep
title('Amplitude Spectrum of the Exponential Sine Sweep')
ylabel('Amplitude (dB)')
xlabel('Frequency (Hz)')
subplot(3,1,2), loglog(f,abs(S_HALL(1:NFFT/2))/max(abs(S_HALL))) % Plot amplitude
spectrum of the hall response
title('Amplitude Spectrum of the Recorded Signal in PTY Hall')
ylabel('Amplitude (dB)')
xlabel('Frequency (Hz)')

subplot(3,1,3), loglog(f,abs(SINV(1:NFFT/2))/max(abs(SINV))) % Plot amplitude spectrum
of the inverse filter
title('Amplitude Spectrum of the Inverse Filter')
ylabel('Amplitude (dB)')
xlabel('Frequency (Hz)')


figure(2)
subplot(3,1,1), plot(s) %Plot the time domain sweep
title('Exponential Sine Sweep Reference Waveform')
ylabel('Amplitude')
```

```
xlabel('Time (ms)')
subplot(3,1,2), plot(s_hall) %Plot the time domain hall response
title('Exponential Sine Sweep Recorded Waveform')
ylabel('Amplitude')
xlabel('Time (ms)')
subplot(3,1,3), plot(sinv)% Plot the time domain inverse filter
title('Inverse Filter Waveform')
ylabel('Amplitude')
xlabel('Time (ms)')

%---------------convolve sweep and inverse filter------------
x1=sinv;
x2=s_hall;
x1l=[x1 zeros(1,length(x1)-1)]; %zero pad the input vectors to avoid circular
convolution
x2l=[x2 zeros(1,length(x2)-1)];

X1L=fft(x1l);
X2L=fft(x2l);
YL=X1L.*X2L; %frequency domain multiplication performs time domain convolution
imp=real(ifft(YL));
imp=imp/max(imp);%Normalize



%-----------------IR plots----------------

IR_abs = abs(imp);
IR_abs = IR_abs/max(IR_abs);            %normalize

aurIR_abs = abs(aur_imp);
aurIR_abs = aurIR_abs/max(aurIR_abs);   %normalize

figure(6)
subplot(2,1,1)
plot(20*log10(IR_abs)); %Plot the calculated impulse response
axis([0 10e5 -110 0])
title('Hall Impulse Response computed in Matlab')
ylabel('Amplitude (dB)')
xlabel('Time (ms)')
subplot(2,1,2)
plot(20*log10(aurIR_abs)); %Plot the impulse response calculated in Aurora
axis([0 10e5 -110 0])
title('Hall Impulse Response computed with Aurora')
ylabel('Amplitude (dB)')
xlabel('Time (ms)')


wavwrite(imp,fs,nbits,'imp.wav')
```

Appendix B - RIR Extraction MATLAB

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Elec 499 Summer 2009 Hudson Giesbrecht      Last Rev: Tim Perry 2009-07-29
%
%Extracts the impulse response from the recorded sweep by convolving with
%an inverse filter
%the inverse filter is 24 bit
%the recorded sweeps are 24 bit
%should we dither before the wavwrite
% extracts a directory worth of impulses (batch processing)
%sweeps hit the noise floor after 10.35 seconds

clear all;
close all;

%=========================================================================
%----NORMALIZE ALL IRs TO IR OF HIGHEST AMPLITUDE SWEEP RECORDIN-----
%       (compute with dBu_IR.m)

dBu_sweep = wavread('L1-C_PTY_Center_74-02.wav');
dBu_PEAK = dBu_IR(dBu_sweep);             % get reference amplitude for 0dB


%=========================================================================
% Process all IRs with respect to dBu_ref value
%=========================================================================

Tk=4; %How many seconds of the impulse response to keep
[sinv, fs, nbits]=wavread('auroraInverseFilt_10s-24bit.wav'); %Inverse sweep
from Aurora
sinv=sinv';
SINV=fft(sinv);

files = dir('Recorded Sweeps\*PTY*.wav'); %Read in list of recorded sweeps in
the directory
size(files)
mkdir(pwd,'IR') %Create a subdirectory of the current matlab directory to
store the IR's

x1=sinv; % set variable x1 to the inverse filter
x1l=[x1 zeros(1,length(x1)-1)]; %zero pad to avoid circular convolution
X1L=fft(x1l);

numIRs = numel(files)

tic
for i = 1 : numel(files)
    name=files(i).name; %select the current file name from the array
    dirname=fullfile(pwd,'Recorded Sweeps' ,name);
    [x2,fs,nbits]=wavread(dirname); % Read in current recorded sweep into
variable x2
    x2=x2';
    x2=x2(1:length(sinv)); %Set the length of the recorded sweep to the
inverse filter
```

```
    x2l=[x2 zeros(1,length(x2)-1)]; %zero pad the to avoid circular
convolution
    X2L=fft(x2l);
    YL=X1L.*X2L; %frequency domain multiplication performs time domain
convolution
    imp=real(ifft(YL)); %The impulse is recovered from the convolution

    %imp=0.98*imp*(max(abs(x2))/max(abs(dBu_ref)));
    IRfloat = imp;
    IRfloat_max = max(abs(imp));
    imp=0.98*imp./dBu_PEAK;  %Normalize the IR with respect to the IR of
highest amplitude sweep in batch

    %Trim the impulse to size, removes the harmonic artifacts before the
    %impulse as well as extranneous samples after
    imp=imp(floor(length(imp)/2):ceil(length(imp)/2+Tk*fs)); %Trim the IR to
the length specified

    %Prepare a name for the wavfile of the recovered IR
    wavname=num2str(files(i).name);
    wavname=['IR ' wavname];
    dirwavename=fullfile(pwd,'IR',wavname);

    wavwrite(0.99*imp,fs,nbits,dirwavename); %Write the recovered IR

    disp(i);

    %Plot the IR to determine the noise floor
    t=(1:length(imp))/fs;
    plot(t,20*log10(abs(imp)))
end
toc
```

Appendix C - Time-Frequency Analysis MATLAB

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% waterfall_Plot.m                                          Author: Tim Perry
% Elec484: DAFX                                                      V00213455
% Final Project, Phase 1                                          2009-07-08
%
% Function to perform time-frequency representation plots (waterfall plots)
% from the phase and moduli data provided by a phase vocoder.
% Plots the following:
%          - Amplitude & Phase Waterfalls
%          - Magnitude Waterfall (dB scale)
%          - Phase vs. Time plot of Freq Bins near max amplitude
%              (in the case of a single tone sinusoid, will be centered
%              around pitch)
%
% FIG = WATERFALL_PLOT(Moduli,Phases,nPlotIndices,fs,WLen,Ra,Rs,N,nameTAG)
%
%           Moduli = amplitude matrix
%           Phases = phase matrix (input or output phases)
%           nPlotIndices =  sample indexes for FFT frame plots
%           fs = sampling frequency
%           WLen = analysis and synthesis window size
%           Ra = analysis hop size
%           Rs = synthesis hop size
%           N = length of time axis in samples (signal length, typically)
%           nameTAG = 'String-For-Naming-Plots'
%
%           P = plot handle
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [waterAmpFIG waterPhaseFIG] =
waterfall(Moduli,Phases,nPlotIndices,fs,WLen,Ra,Rs,N,plotTAG)

figNumStart = get(0,'CurrentFigure');
%clear figure(figNumStart);
figNum = figNumStart + 1;

waterAmpFIG = figure('Colormap',jet(128),'Name','Waterfall Amplitude Plot', ...
    'Position',[10,20,1800,950]);
clear figure(figNum);
figNum = figNum + 1;

waterPhaseFIG = figure('Colormap',jet(128),'Name','Waterfall Phase Plot', ...
    'Position',[20,20,1800,950]);
clear figure(figNum);
figNum = figNum + 1;


%=============================================================
% Time-Frequency Plots
%=============================================================

%ModuliIn_norm = 2*ModuliIn/(WLen/2); %Amp spectrum in quantity peak
Amp_norm = Moduli/Ra;                 % Amp spectrum in quantity peak
Mag_dB = 20*log10(Moduli/max(max(Moduli)));

f0 = fs/WLen;                         % frequency resolution
numBins = WLen;                       % # of frequency bins
kBins = linspace(0,fs/2,numBins/2 + 1); % lin spaced freq bins up to Nyquist
```

```
[n, k] = meshgrid(nPlotIndices,kBins);    % rectangular domain



set(0,'CurrentFigure',waterAmpFIG)
%=================================================================
%-----------------Amplitude Waterfall--------------
%=================================================================
hold on
%C = del2(n,k,AmpIn_norm(1:(numBins/2+1), :));           %colour mapping
waterAmp = surf(n,k,Amp_norm(1:(numBins/2 + 1), :));  %plot input amps
%waterAmp = waterfall(n,k,AmpIn_norm(1:(numBins/2+1), :),'.');
%waterAmp = stem3(n,k,AmpIn_norm(1:(numBins/2+1), :),'.');
%colormap winter
colorbar
axis([0,N,0,fs/2,0,max(max(Amp_norm))])
title(['Amplitude Waterfall of Short-time FFTs (', plotTAG, ')']);
grid on
ylabel('f [Hz]')
xlabel('n [samples]')
zlabel('|X_w(f)|')
hold off

view(viewmtx(-55,25,25));              % set viewpoint


%=================================================================
%-----------------Phase Waterfall------------------
%=================================================================
set(0,'CurrentFigure',waterPhaseFIG)
hold on
waterPhase = surf(n,k,Phases(1:(numBins/2 + 1), :));  %plot  phases
colormap('jet')
colorbar
axis([0,N,0,fs/2,-3.2,3.2])
title(['Phase Waterfall of Short-time FFTs (', plotTAG, ')']);
grid on
ylabel('f [Hz]')
xlabel('n [samples]')
zlabel('Arg{X_w_(f)}  [rad]')
hold off


view(viewmtx(-55,75,25));              % set viewpoint



%=================================================================
% Magnitude Waterfall (dB)
%=================================================================
figure('Name','Magnitude Waterfall [dB]','Position',[50,100,1700,800]);
clear figure(figNum);
figNum = figNum + 1;

hold on
waterMag = surf(n,k,Mag_dB(1:(numBins/2 + 1), :));  %plot mag
colormap('jet')
colorbar
axis([0,N,0,fs/2,1.2*min(min(Mag_dB)),0])
title(['Magnitude [dB] Waterfall of Short-time FFTs (', plotTAG, ')']);
grid on
```

```
ylabel('f [Hz]')
xlabel('n [samples]')
zlabel('20log|X_w(f)|        [dB]')
hold off

view(viewmtx(-55,25,25));              % set viewpoint



%================================================================
%----------Phase vs. Time (@ Freqs Bins near max amplitude)-------
%================================================================

%------find freq bins where max amplitude occures-----
[maxAmps, bins_max]=max(Moduli, [ ], 1);         % get indices @ max
center_bin = median(bins_max) - 1;               % bin w/ most peaks
bins_around = 5;                      % # bins above and below to include

%--------define bins to plot (close to center_bin)-------
if (center_bin <= bins_around)                        % freq bins above f
    k_closeBins = [center_bin:center_bin+bins_around - 1];
elseif (center_bin >= numBins/2 - bins_around)        % freq bins below f
    k_closeBins = [center_bin-5:center_bin];
else                                                  % freq bins surrounding f
    k_closeBins = [center_bin-bins_around:center_bin+bins_around - 1];
end


[n, k_close] = meshgrid(nPlotIndices,k_closeBins*f0);  % rect domain

figure('Name','Phase vs. Time','Position',[800,50,850,450]);
clear figure(figNum);
figNum = figNum + 1;


hold on
waterfall(n,k_close,Phases(k_closeBins + 1, :));
%stem3(n,k_close,PhasesIn(k_closeBins, :),'.');
colorbar
axis([0,N,min(min(k_close)),max(max(k_close)),-3.2,3.2])
title(['Phase vs Time for Freq Bins Near Fundamental (', plotTAG, ')']);
grid on
ylabel('f [Hz]')
xlabel('n [samples]')
zlabel('Arg{X_w_(f)}  [rad]')
hold off

view(114,77)                              % set viewpoint (azimuth, elevation)
%view([20*N, 2*fs,400*max(max(Moduli))])    % set viewpoint
```

```
%=========================================================================
% SpeedConvOLA.m                                          Author: Tim Perry
% Elec499: MMRIR and Conv Reverb                                  V00213455
%                                                                2009-06-14
%
% High Speed Convolution
% Convolves two sound files using Acyclic (aperiodic) FFT convolution in
% frequency domain.Zero padding, rectangular windows overlap-add are used.
% Zero padded acyclic convolutions are imbedded in cyclic convolutions to
% optimize computation speed, but prevent time domain aliasing caused by
% cyclic convolution.
%=========================================================================
clear all;
close all;

%-----------------------------------------------------------
% Inputs (2 mono audio files, ex: room IR and dry recording)
%-----------------------------------------------------------
wavfile1 = 'dBu_IR L1-C_PTY_Center_74.wav'
wavfile2 = 'cajon.wav';
[h fs nbits] = wavread(wavfile1);   % h(n) is the filter
[x fs nbits] = wavread(wavfile2);   % x(n) is the audio to be filtered

Nx = length(x)                      % original length of audio files
Nh = length(h)

%----------------check input files------------------------------
% If col vectors, change to row vectors for matrix operations
if size(h,1)>1,
  h=h';
end
if size(x,1)>1,
  x=x';
end

%-------------------------------------------------------------------------
% Parameters for Acyclic conv with rectangular windows and overlap-add
%-------------------------------------------------------------------------
N = Nx;                                % Set Nx to be the signal length
L = Nh;                                % Set Nh to be the filter length
M = 4096;

    %--------Option 1 FFT Size----------
%NFFT = 2^nextpow2(M + Nh + - 1);

    %------------Option 2 FFT-----------
NFFT = Nh+M;                              % size of FFT
M = NFFT-Nh+1                             % optimized window length

Ra = M;                           % hop size for acyclic with rectangular win
Nframes = 1+floor((N-M)/Ra);              % # of full blocks to convolve
% NsigOut = 2^nextpow2(N + NFFT)          % for Acyclic, NsigOut is smallest
                                          % power of 2 that is  >= |Nx+Nh-1|
```

```
tic
%HHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH
%================================================================
% Acyclic Convolution (Frequency Domain) with zero padding
%================================================================
hzp = [h zeros(1,NFFT-Nh)];         % pad filter kernal h to FFT size
H = fft(hzp);                       % frequency response of filter kernal

y = zeros(1,N + NFFT);                    % vector for output signal

%---apply FFT convolution with overlap add----
for m = 0:(Nframes-1)
    index = m*Ra+1:min(m*Ra+M,N);         % mth frame indices
    xm = x(index);                        % rectangular windowed mth frame
    xmzp = [xm zeros(1,NFFT-length(xm))]; % zero padding

        %----Frequency domain processing----------
    Xm = fft(xmzp);         % FFT of padded mth frame
    Ym = Xm.* H;            % freq domain multiplication = time domain conv

        %----Synthesis (overlap-add)-----------
    ym = real(ifft(Ym));                % return results to time domain
    indexOut = m*Ra+1:(m*Ra + NFFT);
    y(indexOut) = y(indexOut) + ym;     % apply overlap adding
end

%HHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH
toc


%-------------------------------------------------------
% Output Audio File
%-------------------------------------------------------
y = y/max(abs(y));  %normalize output mag to prevent wavwrite  clipping
y = y*0.98;             % bring level down further

file_out=strcat('TestSpeedConvOA-',wavfile1(1:8),'.wav');  % name output file
wavwrite(y, fs, nbits, file_out);                   % write output audio file
```

Appendix E - Surround Sound Convolution Reverb MATLAB

```
function [yl yr yc ybl ybr inputmax]= cverb_surround(input,stage_loc,aud_loc,write)

% %-------------------------------------------------------------------------
% Convolution Reverb
% Hudson Giesbrecht Summer 2009
%
% Implements convolution reverb based on impulse responses measure in the
% Univeristy of Victoria Philip T. Young recital hall.
%
% input - name of the wave files to be processed. input(n) will be
% processed with stage_loc(n)
% hop_size - size of the analysis hop, window is 4x hop_size
% stage_loc - location of the sound to be processed on the stage options
% are (FL, BL, BC, BR, FR, Qfl, Qbl, Qbr, Qfr, and C)
%
% aud loc - listening location. Options are(L1, L2, L3, L4, L5, L6,and L7)
% write - 1 for write a wave file, 0 for no wave file
%-------------------------------------------------------------------------

files = dir('IR\*PTY*.wav'); %Read in list of recorded sweeps in the directory

%Find left center mic IR
for i=1:length(files)
    IR_list=num2str(files(i).name);
    stageloc=strfind(IR_list, stage_loc);
    audloc=strfind(IR_list, aud_loc);
    micpos=strfind(IR_list, 'Lcenter');
    if length(stageloc)==1 && length(audloc) == 1 && length(micpos)==1
        IRL=num2str(files(i).name);
    end
end

%Find right center mic IR
for i=1:length(files)
    IR_list=num2str(files(i).name);
    stageloc=strfind(IR_list, stage_loc);
    audloc=strfind(IR_list, aud_loc);
    micpos=strfind(IR_list, 'Rcenter');
    if length(stageloc)==1 && length(audloc) == 1 && length(micpos)==1
        IRR=num2str(files(i).name);
    end
end

%Find center mic IR
for i=1:length(files)
    IR_list=num2str(files(i).name);
    stageloc=strfind(IR_list, stage_loc);
    audloc=strfind(IR_list, aud_loc);
    micpos=strfind(IR_list, 'center');
    if length(stageloc)==1 && length(audloc) == 1 && length(micpos)==1
        IRC=num2str(files(i).name);
    end
end

%Find back left mic IR
for i=1:length(files)
    IR_list=num2str(files(i).name);
    stageloc=strfind(IR_list, stage_loc);
    audloc=strfind(IR_list, aud_loc);
    micpos=strfind(IR_list, 'Lback');
```

```
    if length(stageloc)==1 && length(audloc) == 1 && length(micpos)==1
        IRBL=num2str(files(i).name);
    end
end

%Find back right mic IR
for i=1:length(files)
    IR_list=num2str(files(i).name);
    stageloc=strfind(IR_list, stage_loc);
    audloc=strfind(IR_list, aud_loc);
    micpos=strfind(IR_list, 'Rback');
    if length(stageloc)==1 && length(audloc) == 1 && length(micpos)==1
                IRBR=num2str(files(i).name);
    end
end

dirIRL=fullfile(pwd,'IR' ,IRL);
dirIRR=fullfile(pwd,'IR' ,IRR);
dirIRC=fullfile(pwd,'IR' ,IRC);
dirIRBL=fullfile(pwd,'IR' ,IRBL);
dirIRBR=fullfile(pwd,'IR' ,IRBR);

[irl, fs, nbits]=wavread(dirIRL); %impulse for front left channel
[irr, fs, nbits]=wavread(dirIRR); %impulse for front right channel
[irc, fs, nbits]=wavread(dirIRC); %impulse for center channel
[irbl, fs, nbits]=wavread(dirIRBL); %impulse for back left channel
[irbr, fs, nbits]=wavread(dirIRBR); %impulse for back right channel

ir_length=length(irl);
WLen=2*ir_length; %window size as per Zolzer DAFX page 265

% Read in audio to be processed
dry=wavread(input);
input_size=size(dry);
x1l=[dry; zeros(WLen-mod(input_size,ir_length),1)]; %lengthen input vector to next
multiple of ir_length
inputmax=max(abs(dry));


IRL=fft([irl; zeros(WLen-ir_length,1)]); %pad the IR with zeros and take the fft
IRR=fft([irr; zeros(WLen-ir_length,1)]);
IRC=fft([irc; zeros(WLen-ir_length,1)]);
IRBL=fft([irbl; zeros(WLen-ir_length,1)]);
IRBR=fft([irbr; zeros(WLen-ir_length,1)]);

yl=zeros(ceil(length(x1l))+ir_length,1);  %allocate left output vector
yr=zeros(ceil(length(x1l))+ir_length,1);  %allocate right output vector
yc=zeros(ceil(length(x1l))+ir_length,1);  %allocate center output vector
ybl=zeros(ceil(length(x1l))+ir_length,1);  %allocate back left output vector
ybr=zeros(ceil(length(x1l))+ir_length,1);  %allocate back right output vector

% Process Audio
tic
pin = 0;
pend=length(x1l)-ir_length;
while pin < pend
    %common to all channels
    ch=[x1l(pin+1:pin+ir_length); zeros(ir_length,1)]; %Each chunk is the size of the
window
    CH=fft(ch);

    %left channel audio
    CONV=CH.*IRL; %convolve the input signal with the front left impulse response
```

```
    conv=real(ifft(CONV));
    yl(pin+1:pin+WLen)=yl(pin+1:pin+WLen)+conv; % overlap and add

    %right channel audio
    CONV=CH.*IRR; %convolve the input signal with the right impulse response
    conv=real(ifft(CONV));
    yr(pin+1:pin+WLen)=yr(pin+1:pin+WLen)+conv; % overlap and add

    %center mic audio
    CONV=CH.*IRC; %convolve the input signal with the right impulse response
    conv=real(ifft(CONV));
    yc(pin+1:pin+WLen)=yc(pin+1:pin+WLen)+conv; % overlap and add

    %back left mic audio
    CONV=CH.*IRBL; %convolve the input signal with the right impulse response
    conv=real(ifft(CONV));
    ybl(pin+1:pin+WLen)=ybl(pin+1:pin+WLen)+conv; % overlap and add

    %back right mic audio
    CONV=CH.*IRBR; %convolve the input signal with the right impulse response
    conv=real(ifft(CONV));
    ybr(pin+1:pin+WLen)=ybr(pin+1:pin+WLen)+conv; % overlap and add

    pin=pin+ir_length;
end

toc

yl=yl(1:input_size+WLen); %remove extraneous samples
yr=yr(1:input_size+WLen);
yc=yc(1:input_size+WLen);
ybl=ybl(1:input_size+WLen);
ybr=ybr(1:input_size+WLen);

% Scale and write the wave files if write=1
if write ==1

%Rescalling
maxamp=[max(abs(yl)) max(abs(yr)) max(abs(yc)) max(abs(ybl)) max(abs(ybr))];
MAX_AMP=max(maxamp);

if MAX_AMP==maxamp(1)
        norm_value=maxamp(1);
        display('left is biggest')

   elseif MAX_AMP==maxamp(2)
        norm_value=maxamp(2);
        display('right is biggest')

   elseif MAX_AMP==maxamp(3)
        norm_value=maxamp(3);
        display('center is biggest')

   elseif MAX_AMP==maxamp(4)
        norm_value=maxamp(4);
        display('back left is biggest')

   else
        norm_value=maxamp(5);
        display('back right is biggest')

        yl=0.98*yl*(max(abs(dry))/norm_value);
        yr=0.98*yr*(max(abs(dry))/norm_value);
```

```
        yc=0.98*yl*(max(abs(dry))/norm_value);
        ybl=0.98*ybl*(max(abs(dry))/norm_value);
        ybr=0.98*ybr*(max(abs(dry))/norm_value);

end

    mkdir(pwd,'Processed Audio')
    locations=struct('name',{'Front Left','Front Right','Center','Back Left','Back
Right'});
    output=struct('channel',{yl,yr,yc,ybl,ybr});

    for i=1:5
        wavename=strcat(input,'_',stage_loc,'_',aud_loc,'_',locations(i).name,'.wav');
        dirwavename=fullfile(pwd, 'Processed Audio' ,wavename);
        wavwrite(output(i).channel,fs,nbits,dirwavename)
   end
end
```

Appendix F - Multi-Source Convolution Reverb MATLAB

```
function []= dualcverb_surround(aud_loc, source1, stage_loc1, source2, stage_loc2)

%-----------------------------------------------------------------------
% Outputs 5 files for surround sound playback
% Use to process two sound sources playing at any two stage locations
% simultaneously
%-----------------------------------------------------------------------

[x, fs, nbits]=wavread(source1);
write=0;
[yl yr yc ybl ybr inputmax]=cverb_surround(source1,stage_loc1,aud_loc,write);
[yl2 yr2 yc2 ybl2 ybr2 inputmax2]=cverb_surround(source2,stage_loc2,aud_loc,write);

if length(yl)>length(yl2)
    yl=yl(1:length(yl2),:);
    yr=yr(1:length(yr2),:);
    yc=yc(1:length(yc2),:);
    ybl=ybl(1:length(ybl2),:);
    ybr=ybr(1:length(ybr2),:);
end

if length(yl2)>length(yl)
    yl2=yl2(1:length(yl),:);
    yr2=yr2(1:length(yr),:);
    yc2=yc2(1:length(yc),:);
    ybl2=ybl2(1:length(ybl),:);
    ybr2=ybr2(1:length(ybr),:);
end

yl=yl+yl2;
yr=yr+yr2;
yc=yc+yc2;
ybl=ybl+ybl2;
ybr=ybr+ybr2;


if inputmax > inputmax2
    drymax=inputmax;
else
    drymax=inputmax2;
end

if drymax >= 1
    display('time to rescale')
%Rescalling
maxamp=[max(abs(yl)) max(abs(yr)) max(abs(yc)) max(abs(ybl)) max(abs(ybr))];
MAX_AMP=max(maxamp);

    if MAX_AMP==maxamp(1)
        norm_value=maxamp(1);
        yl=0.95*yl*(drymax/norm_value);
        yr=0.95*yr*(drymax/norm_value);
        yc=0.95*yc*(drymax/norm_value);
        ybl=0.95*ybl*(drymax/norm_value);
        ybr=0.95*ybr*(drymax/norm_value);
        display('dual left is biggest')

    elseif MAX_AMP==maxamp(2)
        norm_value=maxamp(2);
        yl=0.95*yl*(drymax/norm_value);
```

```
        yr=0.95*yr*(drymax/norm_value);
        yc=0.95*yc*(drymax/norm_value);
        ybl=0.95*ybl*(drymax/norm_value);
        ybr=0.95*ybr*(drymax/norm_value);
        display('dual right is biggest')

        elseif MAX_AMP==maxamp(3)
        norm_value=maxamp(3);
        yl=0.95*yl*(drymax/norm_value);
        yr=0.95*yr*(drymax/norm_value);
        yc=0.95*yc*(drymax/norm_value);
        ybl=0.95*ybl*(drymax/norm_value);
        ybr=0.95*ybr*(drymax/norm_value);
        display('dual center is biggest')


        elseif MAX_AMP==maxamp(4)
        norm_value=maxamp(4);
        yl=0.95*yl*(drymax/norm_value);
        yr=0.95*yr*(drymax/norm_value);
        yc=0.95*yc*(drymax/norm_value);
        ybl=0.95*ybl*(drymax/norm_value);
        ybr=0.95*ybr*(drymax/norm_value);
        display('dual back left is biggest')

     else
        norm_value=maxamp(5);
        yl=0.95*yl*(drymax/norm_value);
        yr=0.95*yr*(drymax/norm_value);
        yc=0.95*yc*(drymax/norm_value);
        ybl=0.95*ybl*(drymax/norm_value);
        ybr=0.95*ybr*(drymax/norm_value);
        display('dual back right is biggest')
    end

end


%Write the wave files
    mkdir(pwd,'Processed Audio')
    locations=struct('name',{'Front Left','Front Right','Center','Back Left','Back
Right'});
    output=struct('channel',{yl,yr,yc,ybl,ybr});

        for i=1:5

wavename=strcat(source1,'_',stage_loc1,'_',source2,'_',stage_loc2,'_',aud_loc,'_',loca
tions(i).name,'.wav');
        dirwavename=fullfile(pwd, 'Processed Audio' ,wavename);
        wavwrite(output(i).channel,fs,nbits,dirwavename)
        end







function []= tricverb_stereo(aud_loc, source1, stage_loc1, source2,
stage_loc2,source3,stage_loc3)

%Use to process two sound sources playing at any two stage locations
```

```
%simultaneously
% hop_size=1024;
% aud_loc='L1';
% source1='Canon3-Vio 10sec';
% stage_loc1='Qfl';
% source2='Canon3-Gtr 10sec';
% stage_loc2='Qfr';

[x, fs, nbits]=wavread(source1);
write=0;
[output inputmax]=cverb_stereo(source1,stage_loc1,aud_loc,write);
[output2 inputmax2]=cverb_stereo(source2,stage_loc2,aud_loc,write);
[output3 inputmax3]=cverb_stereo(source3,stage_loc3,aud_loc,write);

%Extend Clips for Adding
out_length=[length(output) length(output2) length(output3)];
max_length=max(out_length);

if max_length==out_length(1)
%     display('case 1')
    output2=[output2; zeros(out_length(1)-out_length(2),2)];
    output3=[output3; zeros(out_length(1)-out_length(3),2)];

elseif max_length==out_length(2)
%     display('case 2')
    output=[output; zeros(out_length(2)-out_length(1),2)];
    output3=[output3; zeros(out_length(2)-out_length(3),2)];

else
%     display('case 3')
    output=[output; zeros(out_length(3)-out_length(1),2)];
    output2=[output2; zeros(out_length(3)-out_length(2),2)];
end

output=output+output2+output3;

drymax=max([inputmax inputmax2 inputmax3]);
maxamp=[max(abs(output)) max(abs(output2)) max(abs(output3))];
MAX_AMP=max(maxamp);

if MAX_AMP==maxamp(1)
        norm_value=maxamp(1);
        %display('output1 is biggest')

elseif MAX_AMP==maxamp(2)
        norm_value=maxamp(2);
        %display('output2 is biggest')

else
        norm_value=maxamp(3);
        %display('output3 is biggest')

end
output=0.98*output*(drymax/norm_value);


%Write the wave files
mkdir(pwd,'Processed Audio')
wavename=strcat(source1,'_',stage_loc1,'_',source2,'_',stage_loc2,'_',source3,'_',stag
e_loc3,'_',aud_loc,'.wav');
dirwavename=fullfile(pwd, 'Processed Audio' ,wavename);
wavwrite(output,fs,nbits,dirwavename)
```