MULTIDIMENSIONAL SYNTHESIZED CONVOLUTION REVERB FOR AMBISONIC PLAYBACK

Olivia Canavan, Sam Schachter, Kyle Tworek

University of Rochester, Department of Electrical and Computer Engineering

ABSTRACT

Creating multidimensional reverbs using measured impulse responses is challenging and unrealistic for most users, as it requires an ambisonic microphone array. For this reason we have designed a method for calculating a synthesized impulse response based on user input, and we have used it to create a 3-dimensional reverberation effect. This method has produced realistic-sounding output for a 25.2 speaker array.

Index Terms— Ambisonics, reverberation, vector base amplitude panning, impulse response synthesis

1. INTRODUCTION

Historically, impulse responses have been measured using a microphone and sound source, or variations of this method [3]. These methods provide one with a location-specific acoustical experience that can not be easily manipulated unless several impulse response measurements are taken. It follows that a more flexible means of 3D impulse response reverberation is desired, one that will eliminate the need for measuring impulse responses or expensive ambisonic microphone arrays. This paper introduces a method for synthesizing reverberation and rendering it to multi-channel playback systems.

2. BACKGROUND INFORMATION

2.1 Convolution Reverb

Although we are implementing our reverberation in a multidimensional case, it is useful to first examine methods used in simpler conditions. Accordingly, mono reverb will be discussed first. For a single channel audio source, reverberation can be achieved by measuring an impulse response, or the signal defined by the following equation:

$$\delta(n) \equiv \{1,\ n = 0\ \&\ 0\ n \neq 0\}$$

Equation 1

It is typically measured using a balloon or sudden sound source. The measured signal, $\delta(n)$, provides one with information about reverberation decay time and thus sonic characteristics of the room [4]. When this data is convolved with an input signal, the resulting sound is "placed" in the space of the measured impulse response. Mathematically, this is described by :

$$y(t) = x(t) * h(t) = \sum_{k=-\infty}^{\infty} x(k)h(t-k)$$

Equation 2

where $x(t)$ is the input signal and $h(t)$ is the measured impulse response. The output, $y(t)$, sounds like the original signal but is reverberated in the space defined by the impulse response *h(t)*. In frequency, convolution is defined with multiplication, or:

$$Y(j\omega) = X(j\omega) * H(j\omega)$$

Equation 3

where $X(j\omega)$ is the Fourier Transform of the input signal and $H(j\omega)$ is the Fourier Transform of the impulse response.

2.2 Vector Base Amplitude Panning

Another fundamental processing technique used extensively in this project is Vector Base Amplitude Panning (VBAP), a method of positioning sounds in two-dimensional space as introduced by Ville Pulkki in his 1997 paper in the AES Journal [5]. Traditional stereo panning allows placement of a "virtual source" at any point on a straight imaginary line drawn between two loudspeakers by scaling the relative magnitude between signals sent to each channel. Pulkki extends this concept to two-dimensional space, allowing a virtual source to be placed anywhere within the triangle formed by three noncollinear loudspeakers [5]. Using this method, it is possible to place a virtual source at any given point on a spherical array of loudspeakers, even if this point falls on a space between loudspeakers.

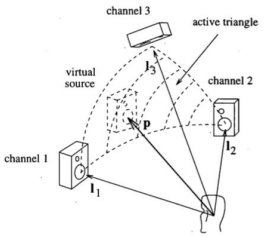

Figure 1 : Visualization of a virtual source and the loudspeakers used to construct it with VBAP. [5]

Figure 1 shows an example of a situation where a virtual source p is constructed using the loudspeakers l_1, l_2, and l_3. If p falls within the active triangle formed by l_1, l_2, and l_3 — as it does in this case — we can use Equation 4 below to calculate the vector $\mathbf{g} = [g_1 \ g_2 \ g_3]$, where g_n is the gain scaling factor corresponding to speaker l_n. [5].

$$\mathbf{g} = \mathbf{p}^T\mathbf{L}_{123}^{-1} \text{ , where}$$
$$\mathbf{p} = \text{unit column vector to } p$$
$$\mathbf{L}_{123} = [\mathbf{l}_1 \ \mathbf{l}_2 \ \mathbf{l}_3]^T ; \mathbf{l}_n = \text{unit column vector to } l_n$$

Equation 4 [5].

In this project, a 3-D impulse response is generated, which contains impulse data corresponding to an extremely high number of unique locations on the unit sphere. The playback system used for this project utilizes only 25 speakers, arranged roughly in the shape of a sphere with the bottom "cap" missing. Using VBAP, the three nearest noncollinear speakers to each unique impulse location are employed to construct a virtual source at the impulse location, allowing a nearly complete spherical soundfield for playback using only 25 speakers. As the lowest speakers in the system are only 30 degrees below the horizontal, virtual sources that would fall on the bottom cap of the sphere below this latitude are excluded from playback at this time.

3. SYNTHESIS OF 3D IMPULSE RESPONSE

3.1 Graphical User Interface (GUI)

One of the goals of this project was to allow the user of our program to have full control over the length of the reverberation and the location of the source sound in the ambisonic environment. To achieve this, a GUI was implemented that allow users the ability to specify four parameters: room size, room reflectivity, and source and receiver location. When the user runs the program in MATLAB, they are prompted to enter the length, width, and height of the room in meters (Figure 2). Once they choose these three parameters, the user is then asked to choose between several types of room (Figure 2). These room types correspond to different reflection coefficients, which determine what percentage of the impulse is reflected by each surface of the room.

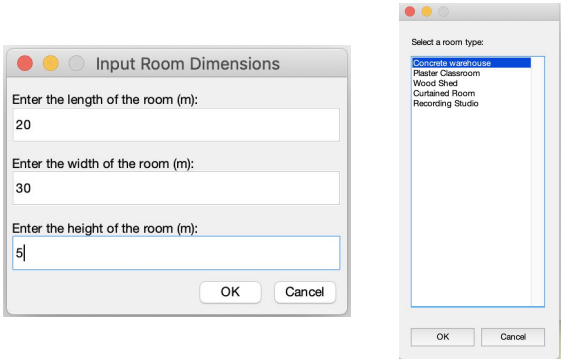

Figure 2 : GUI asking user to select room size and type.

Once the room is specified, the user is prompted to select where in the room they would like the source to be located, and where they would like the speaker to be located (Figure 3). Once they select the location of the source, a marker appears on the plot to indicate where they chose, so that they may choose the receiver location accordingly. The user first chooses the location relative to the width and length of the room, then relative to the height, as MATLAB does not allow for 3-dimensional inputs.

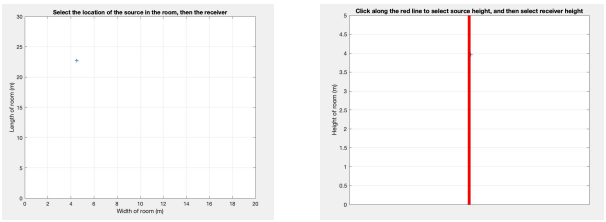

Figure 3 : GUI asking user to select source and receiver location in the generated room.

3.2 Synthesis of Impulse Response

A multidimensional impulse response is much like a typical impulse response, but with the inclusion of directional information. The output of a multidirectional impulse response includes the magnitude of each impulse, the time at which it is registered by the receiver, and the direction from which it arrives. Usually, to measure a multidimensional impulse response, an ambisonic microphone array is used. Our program allows the user to synthesize such a response without expensive equipment and with the added benefit of simulating an environment with zero background noise, a condition which is impossible to meet in nature [4].

The multidimensional impulse response is calculated using the method of images. This method greatly simplifies the calculation of the angle and magnitude data for each reflection by simulating images of the generated room. A simplified 2-dimensional example of this is shown in Figure 4 below.

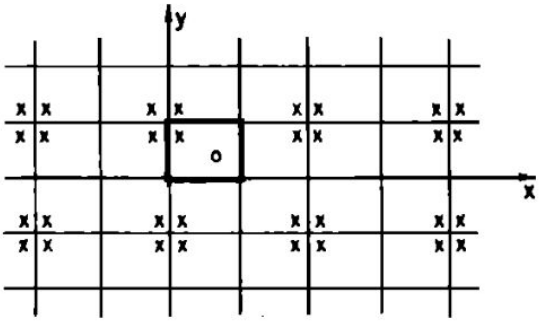

Figure 4 : 2-dimensional representation of image space surrounding the original room, represented by bolded black lines [2].

Every rectangle surrounding the bolded rectangle is an image of the original room, with the source, marked by an x, flipped accordingly. An array of images surrounding the original room is subsequently created by adding images until the desired number of reflections are represented. The program calculates the vector distance between the receiver and each source image, taking into account the number of 'walls' it passes through before reaching the receiver. This represents the number of reflections needed for the source image to reach the receiver in the original room. The magnitude of each impulse is calculated using:

$$D_{mag} = \sqrt{x^2 + y^2 + z^2}$$

Equation 5

where x, y, and z represent the vector distances between the source and receiver. Then, to account for the type of room chosen by the user, each impulse is scaled according to the reflectivity of the room and the number of reflections using:

$$D_{imp} = R^n / D^2_{mag}$$

Equation 6

Where R is the reflectivity constant of the selected room ($0 < R < 1$) and n is the number of reflections that the impulse needed to reach the receiver. The time of arrival for each reflection is then calculated by dividing the magnitude by the speed of sound in air (343 m/s). When this data is compiled and sorted by magnitude, the resultant output is the multidirectional impulse response, with directions specified in terms of vector distances.

Using the room size, reflectivity, and source and receiver locations acquired in the GUI, we calculate the multidimensional impulse response for the specified room [1]. The result of this calculation is illustrated in Figure 5 below. Early reflections are plotted using warmer colors (orange and yellow), while late reflections are plotted using cooler colors (blue and purple). The length of the line represents the magnitude of the reflection vector.

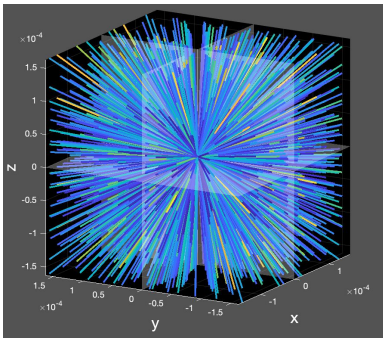

Figure 5 : 3D impulse response as generated by our program.

4. VECTOR BASE AMPLITUDE PANNING

The raw 3D impulse response generated in the previous stage of the project contains position data for each individual 'spike' across the time domain of the response, as visualized in Figure 5 above. Vector Base Amplitude Panning is used to ultimately process this data into a set of N individual impulse responses, where N is the number of speakers in the playback system (in our case, 25). A script was developed to iterate through each unit impulse of the multidimensional response, and populate the correct channel IRs with the panned signal using VBAP.

To start, we make the assumption that all speakers and virtual sources are on the surface of the unit sphere, and calculate the spherical coordinates for each speaker in the playback system. With the speaker positioning data established, each unit impulse of the multidimensional impulse response goes through the following processing.

First, the spherical-coordinate location of the given unit impulse (virtual source) is calculated from vector position data provided by the 3-D impulse calculation. The straight-line distances between the virtual source and each speaker are calculated, and the speakers corresponding to the three shortest distances are selected. Next, it must be verified that the selected speakers are not collinear. If they are found to be collinear (if all three share the same value of theta or phi, shown collectively as "Case 1" in Figure 6

below), the script interrupts and moves to the next unit impulse. If the speakers are verified to be noncollinear, it is assumed that the virtual source falls within the active triangle. In actuality, due to suboptimal placement of the speakers in our particular playback system, there exists a case where the virtual source falls outside the active triangle of its three nearest noncollinear speakers. This situation is illustrated as "Case 2" in Figure 6 below. At this time, the issue remains unaddressed, as it does not cause any perceivable artifacts. At this point, the VBAP process described in detail in Section 2.2 above is invoked to calculate gain vector **g** corresponding to the channel gains for each of the three selected speakers. Using the timing and magnitude information corresponding to this unit impulse from the original 3D impulse response data, the individual channel IRs of the three selected speakers are each updated with an impulse of the originally calculated magnitude scaled by the relevant value from **g**.

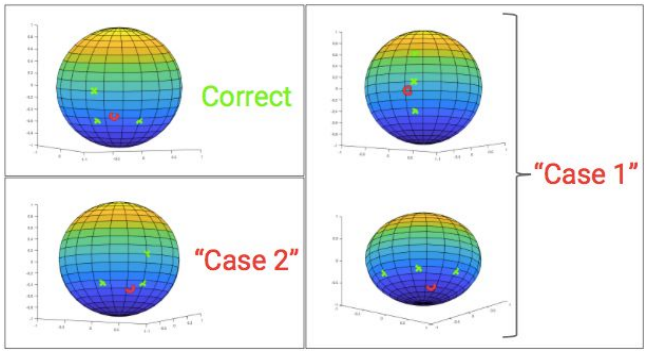

Figure 6 : Example cases of a virtual source "O" and the three nearest speakers

Once this process has been completed across the entire time domain of the generated 3D impulse response, the individual channel IRs are complete, and ready for the convolution and output stages.

5. CONVOLUTION REVERB

As previously stated, convolution is a mathematical operation that produces an output shaped by two input signals, X and H. In the context of this project, we convolved our synthetic impulse response and input signal (any stereo .wav or .mp3 file of choice), yielding the reverberated signal.

5.1. Convolution Method

In an effort to minimize our project runtime, an overlap-add (OLA) convolution was implemented. This approach is regarded as the most efficient implementation of convolution for lengthy signals [6]. In this project the built in MATLAB function, fftfilt(), was used to complete this process. Fftfilt() carries out the following, $y(t) = ifft(fft(H, nfft) * fft(X, nfft)$, where nfft is the length of the *fft*. OLA works more efficiently because it breaks the Fourier Transform into smaller blocks that are the length of nfft. By processing it in smaller frames, MATLAB can run the convolution on each frame faster than it can the entire signal. It then sums the convolved frames and produces the fully transformed output. At this step, the processed audio is ready for playback.

6. AMBISONIC PLAYBACK

For mono or stereo audio playback, it is usually sufficient to use the *soundsc()* function in MATLAB. However, this playback method does not work for a higher number of channels. As a result, the MATLAB audioDeviceWriter object is needed. This writes audio directly to the computer's sound card and therefore has no limits in terms of channel number. This object is highly customizable, allowing one to specify parameters such as device, sample rate, bit depth, and channel mapping. The audioDeviceWriter generates multichannel waveforms, as shown below.

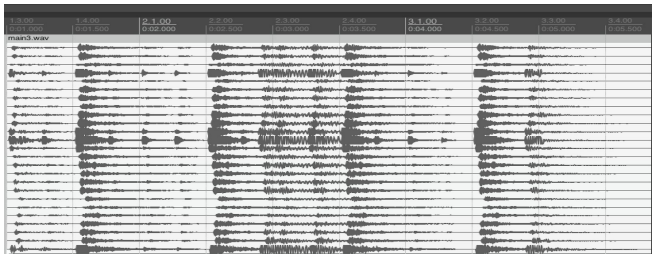

Figure 7 : 25 channel output in Reaper

7. RESULTS

The process for achieving authentic-sounding reverberation is a success. The approach yields realistic output as directionality and appropriate gain levels are preserved. Our GUI allows the user to specify their own room conditions, therefore personalizing the experience as desired. The implementation of VBAP is also a success because it adeptly processes and sorts the synthetic impulse responses. Implementing the OLA method for convolution enabled us to perform the processing at a more rapid rate, which is ideal for real use. This working implementation of artificial reverberation will be useful for acousticians for modeling purposes.

8. REFERENCES

[1] Bocko, M (2018) Imp_Resp_w_Angle_3.m source code (Version 1.0) [Source code].

[2] B. Allen, J. (1976). Image method for efficiently simulating small-room acoustics. Acoustical Society of America Journal. 60. 9-. 10.1121/1.2003643

[3] Stan Guy-Bart, Embrechts Jean-Jacques, Archambeau Dominique. (2002). Comparison of different impulse response measurement techniques. Audio Engineering Society Journal. 50.

[4] G. Defrance, L. Daudet, and J.-D. Polack, "Finding the onset of a room impulse response: Straightforward?," *The Journal of the Acoustical Society of America*, vol. 124, no. 4, 2008.

[5] Pulkki, V. (1997). Virtual Sound Source Positioning Using Vector Base Amplitude Panning. J. Audio Eng. Soc., Vol. 45, No. 6, 1997 June.

[6] Smith, J.O. *Spectral Audio Signal Processing,* http://ccrma.stanford.edu/~jos/sasp/, online book, 2011 edition.