Real-time phase analysis utility for improving microphone placement in multi-microphone multi-source scenarios

Christian Steinmetz

Clemson University

Holcombe Department of Electrical and Computer Engineering

Riggs Hall, Clemson, SC 29634

Submitted for consideration in the IEEE Region 3 Student Paper Competition

Student Branch Counselor:

William J. Reid

Real-time phase analysis utility for improving microphone placement in multi-microphone multi-source scenarios

Abstract

When utilizing multiple microphones to record or reproduce a source, the relative distance from the source to each microphone introduces a time delay between the signals. In a stereo configuration, this delay affects how the source is localized in the stereo field, and when summed can introduce comb filtering. It is common to utilize two microphones placed above a drum kit (known as overheads) where it is convention to maintain phase coherence between the kick and snare drums to avoid undesirable localization and frequency distortion from comb filtering. In order to simplify the microphone placement process, a real-time Audio Unit/Virtual Studio Technology (AU/VST) plugin was developed that employs the Generalized Cross Correlation with Phase Transform (GCC-PHAT) to provide the audio engineer with feedback about the relative time difference of arrival (TDOA) of sound sources to the overhead microphones. The plugin was tested under real world conditions and shown to provide accurate position offset estimation within one centimeter. This utility provides straightforward information about microphone placement where previous methods require additional training and intuition, and enables the audio engineer to make an informed decision about the overhead placement before beginning to record or reinforce a performance as these relative delays cannot be corrected afterwards.

1 Introduction

It is common, both in live and studio settings, for audio engineers to employ multiple microphones in order to capture a single instrument or an entire ensemble [1]. In both cases, the relative phase coherence of the signals reaching the microphones has a number of important implications in regards to the resultant audio quality.

In the case of a single instrument being recorded or reproduced, differences in the distance from the sound source to each microphone will result in a time delay between the sound reaching each microphone. When these signals are summed to mono, the time delay results in a comb filtering effect that produces the so-called comb shaped frequency response on the summed output [2]. Engineers sometimes refer to this quality of sound as being “phasey”, and is nearly always an undesirable effect.

It is also important to note that the relative time delay present in such situations also has the effect of modifying the localization of the sources within the stereo image due to the precedence effect [3], and therefore is a concern in situations where the resultant signals will not be summed, and used in a stereo configuration.

The potential for comb filtering exists whenever recording or reinforcing a source with multiple microphones, and therefore the common practice of utilizing two microphones as overheads to capture an acoustic drum kit introduces the opportunity for comb filtering and the incorrect localization of sound sources in the stereo field. It is important to note that since the traditional acoustic drum kit is composed of multiple sound sources: cymbals, snare drum, kick drum, and tom-toms, all of which are placed around the drummer, it is impossible, and undesirable, to obtain phase coherence between all of the sound sources simultaneously. Instead, convention dictates that it is desirable to place the snare drum and kick drum in the center of the stereo image, which has the effect of ensuring coherence when summing as well [4].

In general, engineers are trained to identify potential phase coherence issues by carefully listening to the summed signals and actively inverting the polarity of various microphones [5]. While this technique has been shown to be effective when employed by well-trained engineers, it can still leave uncertainty about microphone placement and pose a significant challenge for amateur engineers and those who lack a high fidelity monitoring system.

In [5], techniques to check the relative phase coherence using a correlation meter or oscilloscope are outlined, yet these method provide little insight about the physical positioning of the microphones in relation to the resulting phase coherence, making the iterative process tedious and largely unguided.

The goal of this project was to provide the audio engineer with concrete and descriptive information about the relative phase coherence of sound sources from multiple microphones by supplying sample accurate delay estimation for single sources, as well as the associated path length differences. This was achieved using the Generalized Cross Correlation with Phase Transform (GCC-PHAT) [6], an established method for calculating the time delay of arrival (TDOA) between signals, which has been shown to provide robust delay estimation for acoustic sources.

The GCC-PHAT has been utilized in past research [7] to reduce comb filtering on arbitrary musical sources by correcting the time delay present between microphones in real-time. Across a number of tests, this implementation of the GCC-PHAT was shown to be highly successful. Although, such an approach does not allow for the correction of phase coherence issues that arise in the reproduction of the aforementioned drum kit. Due to the differing locations of the kick and snare drums, adjusting the relative time delay between the overhead microphones, in order to correct for phase incoherence between a single source, introduces additional incoherence in relation to the other source. Therefore,

the only solution in such a situation is not one that has the ability to adjust the relative time delays, but one that enables the audio engineer to place the microphones so both the kick and snare drums achieve phase coherence without post processing. This paper will focus on the development of a system that enables the audio engineer to achieve satisfactory microphone placement.

2 Theory

The GCC-PHAT functions by estimating what is known as the time difference of arrival (TDOA) for sources, as this is the only value that can be computed when the input is two signals, one that is delayed by some arbitrary time in relation to the other. In other words, the system does not allow for the estimate of the absolute distance from the source to the input transducer in space, simply the relative time delay between the two. The Generalized Cross Correlation (GCC) for two signals $\mathrm{x}_1[n]$ and $\mathrm{x}_2[n]$, is defined as follows,

$$\psi_G[k] = X_1^*[k] \cdot X_2[k] \tag{1}$$

in the frequency domain and then transformed to the time domain as

$$\psi_G[n] = \mathcal{F}^{-1}\{\psi_G[k]\} \tag{2}$$

where k is the current frequency bin between 0 and $N-1$, $*$ represents the complex conjugate, $\mathrm{X}_1[k]$ and $\mathrm{X}_2[k]$ are $\mathrm{x}_1[n]$ and $\mathrm{x}_2[n]$ transformed to the frequency domain, and $\mathcal{F}^{-1}$ represents the Inverse Fourier Transform. The aforementioned time difference of arrival, τ, is calculated by finding the position of the maximum output of the above function, the position at which the GCC produces the highest cross-correlation value.

Uncorrelated noise from reverberation and other undesirable sources has been shown to negatively impact the delay estimation of the GCC, and extensive research has covered a number of possible transforms to the GCC in order to help increase robustness with regards to uncorrelated noise [6] [8] [9] [10]. One of the more popular of these transforms is the Phase Transform introduced in [6]. The Phase Transform normalizes the GCC with respect to the magnitude of the inputs, and therefore examines the phase content when determining the estimated delay. This method has been shown to help improve performance in reverberant environments. The GCC-PHAT can be written as

$$\psi_P[k] = \frac{X_1^*[k] \cdot X_2[k]}{|X_1^*[k] \cdot X_2[k]|} \tag{3}$$

in the frequency domain and

$$\psi_P[n] = \mathcal{F}^{-1}\{\psi_P[k]\} \tag{4}$$

in the time domain. As before, the time delay is estimated by finding the position of the maximum output of the time domain function, which relates to the sample delay where the greatest cross-correlation value is achieved.

The GCC-PHAT calculates the relative time delay between the input signals by calculating the difference in phase between the inputs in the frequency domain, and then transforming the result back into the time domain to make the estimation. This is an effective method for estimating the time delay since the phase coherence of the two signals contains information about the time delay when the two signals emanate from the same source, i.e. are correlated to some degree.

For example, Figure 1 shows two signals in the time domain from the left and right overhead microphones when a snare drum was struck. From visual inspection we can see that there is a delay of about 50 samples (about 1.1 ms at 44.1 kHz sampling rate) present between the two microphones.

In Figure 2 the GCC-PHAT has been performed on the two input signals from Figure 1. The location of the maximum of this output gives the estimated time delay of arrival between the signals. There is a clear peak that occurs at a lag of 44 samples indicating the delay between the two signals is 44 samples. This agrees with our inspection of the time domain signal.

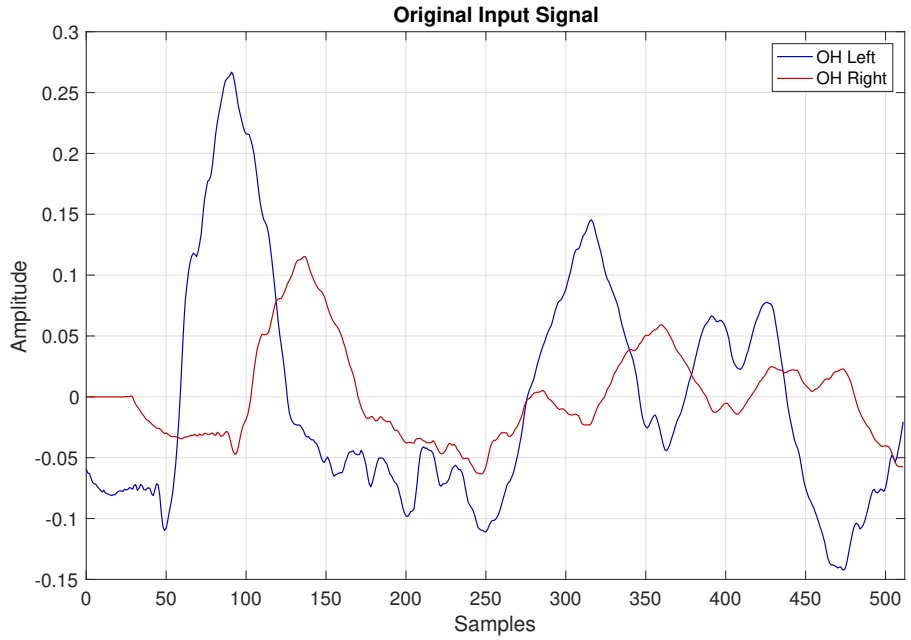

Figure 1: Signals from the left and right overhead microphones placed over the drum kit when the snare drum was struck. Notice the signal from the right overhead is delay.

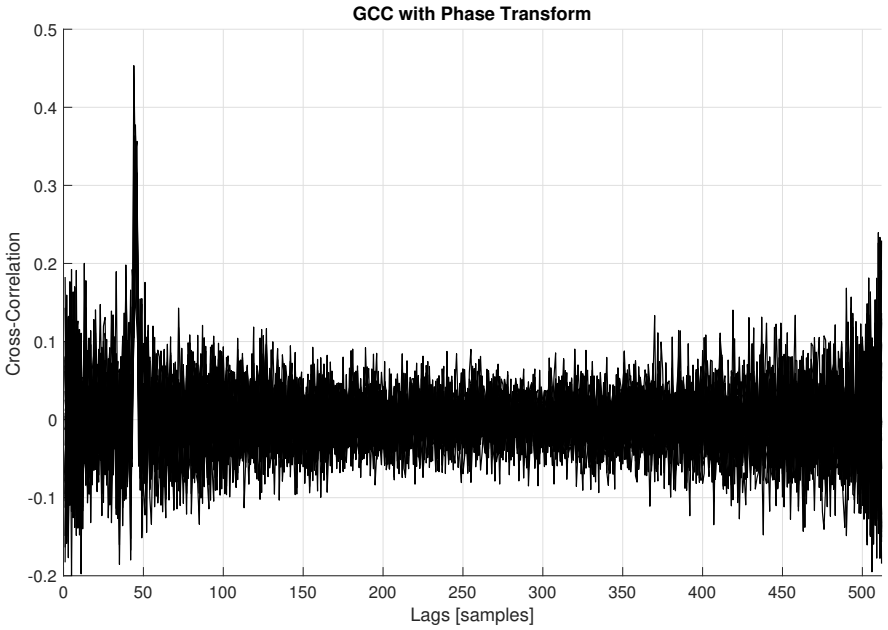

Figure 2: GCC-PHAT output from the left and right overhead microphones when the snare drum was struck.

When implementing the GCC-PHAT, and as is common when implementing the Discrete Fourier Transform (DFT), the input signal is broken into overlapping frames, and each frame has a windowing function applied to it so as to decrease spectral leakage when performing the DFT [11]. In this case the Hann, Blackman-Harris, and Hamming windows were used as results from [7] show high performance on percussive signals.

The Hann window is defined as follows

$$w(n) = \frac{1}{2}\left(1 - cos\left(\frac{2\pi n}{N-1}\right)\right) \tag{5}$$

The current input frame is windowed by multiplying it with a Hann window of the same length. The window serves to smooth out the sample values both at the beginning and end of the frame, as non-zero values at the ends of frame in the time domain can cause spectral leakage in the frequency domain upon performing the DFT. Different windowing functions impart different characteristics on the resultant output when transformed for this reason, and therefore multiple windowing functions were included to provide the user with a choice when using the utility in different environments.

In addition to the windowing applied to the signal, there was a set hop size for the frame analysis. This control dictates how many samples the analysis moves along the input after the previous frame

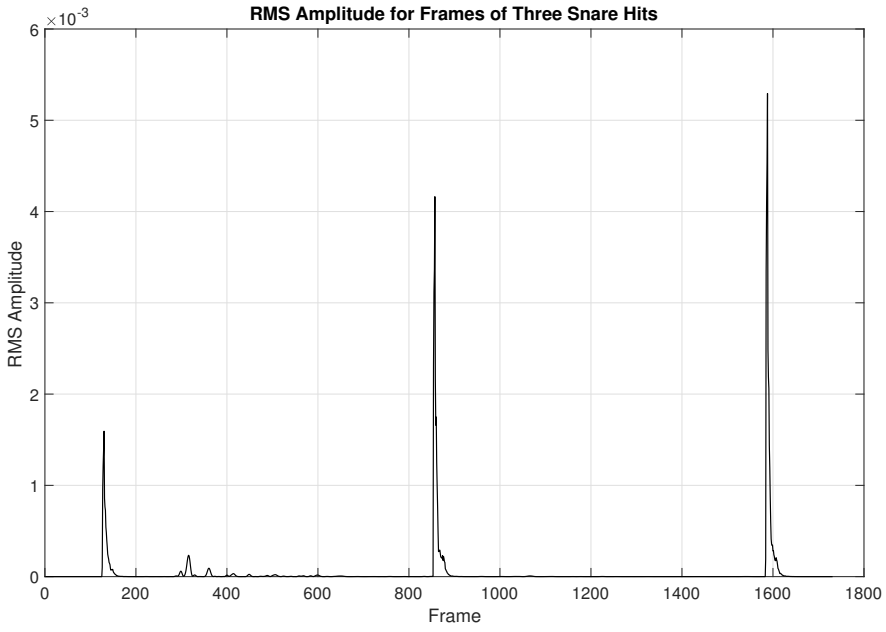

Figure 3: Calculated RMS amplitude for individual analysis frames of the left channel for three consecutive snare drum hits

analysis. This method allows for the entirety of the input signal to be analyzed more evenly since the window function has the effect of distorting the amplitude of the input at the ends of the frame [12].

Another key part of this work involves the variable analysis threshold. Many times input signals contain a significant number of samples with near zero values at the noise floor of the system. Due to the fact that these samples contain little to no information about the time delay of the source in question, they have a negative impact on the delay estimation. In order to increase the robustness of the GCC-PHAT, the RMS amplitude of each frame is measured and frames that are below the user set threshold are discarded. The RMS amplitude for a collection of frames from three successive snare drum hits are shown in Figure 3. Note that the majority of the frames have a very low RMS amplitude, and therefore can be discarded in the analysis.

3 Implementation

The development process began with a model of the GCC-PHAT operating in MATLAB to first experiment with the GCC-PHAT implementation, windowing functions, and noise removal. First, experimentation was performed comparing the GCC with and without the Phase Transform, which can be seen in Figure 4 and Figure 5. The addition of the Phase Transform has a significant effect on sharpening the peak in the GCC, therefore improving the delay estimation.

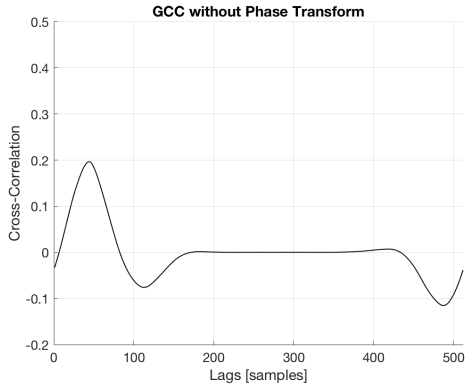

Figure 4: GCC without Phase Transform from single frame of snare drum hit

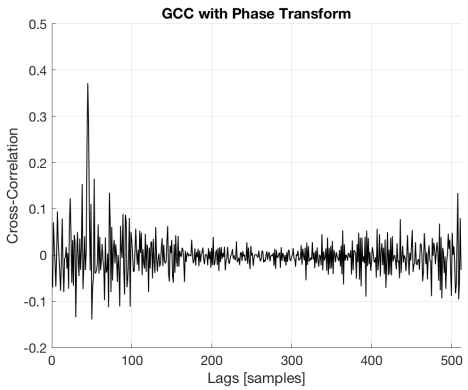

Figure 5: GCC with Phase Transform from single frame of snare drum hit

The exclusion of frames containing low amplitude signals was shown to improve the delay estimation performance as well. It also greatly improved the analysis time by significantly reducing the total number of frames analyzed from the input signal.

Highpass and lowpass filtering was performed on recorded snare drum samples in an effort to improve the delay estimation by removing extraneous noise from the input signal. All attempts resulted in equivalent delay estimation or, in cases when the cutoff frequency of the filters began to reduce the bandwidth of the input more significantly, a degraded delay estimation accuracy. These results agree with the findings of [13], wherein it was determined that band-limiting the input to the GCC-PHAT resulted in decreased delay estimation, especially when using a rectangular window.

With these preliminary findings, development of the real-time C++ plugin began. The JUCE Framework was utilized as it enabled a more straightforward approach to creating AU/VST plugins in C++ to be used in common Digital Audio Workstations (DAWs) [14]. The graphical user interface (GUI) for the plugin was developed using the JUCE framework, as it provided a number of features geared towards developing such applications.

The sample delay estimation is featured prominently in the window of the GUI (shown in Figure 7) as this is the main feedback provided to the user via the GCC-PHAT analysis. Using this estimation, a number of other metrics are calculated and provided to the user.

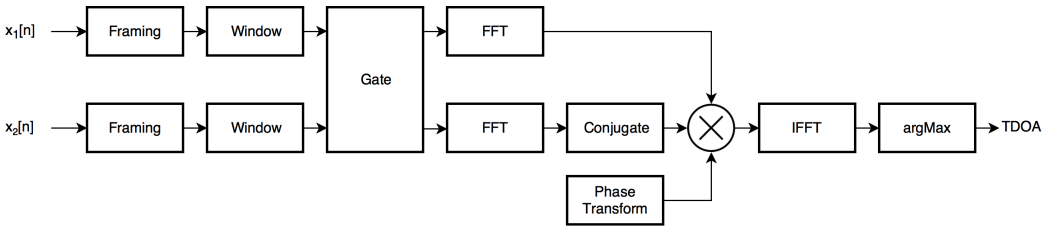

Figure 6: Block diagram for the delay estimation algorithm.

The user is also given a number of parameters that can be adjusted to modify the GCC-PHAT analysis, and potentially provide superior delay estimation performance with different incoming audio signals. These controls include the hop size divisor, the analysis frame size, the window type, and the analysis threshold. The implementation of these parameters will be explained in the following paragraphs and a block diagram for the full delay estimation algorithm employed is shown in Figure 6.

The core GCC-PHAT analysis is centered around a circular analysis buffer of 4096 samples. This buffer is filled by each incoming audio frame from the host system. When the buffer has been completely filled the analysis begins. The number of total frames that will be analyzed is calculated based on the user set frame size and hop size. Next, the appropriate samples for each frame are loaded into a separate buffer and the GCC-PHAT analysis is performed on each frame iteratively after the user selected windowing function has been applied. This only occurs if the RMS amplitude of both channels is greater than the user set threshold, otherwise the frame is discarded, the next frame is loaded, and no delay estimation is returned for the previous frame.

The GCC-PHAT returns the position of the maximum correlation, which relates to the delay estimation. As noted in [13], the sample delay is assumed to be within the first half of the frame and this knowledge is used to identify which channel of the input is delayed in relation to the other. This information is then displayed on the GUI. This has the effect of requiring that the frame size be twice as great as the desired sample delay to be measured. For the default frame size of 1024 samples this means that sample delays up to 512 samples or 11.6 ms can be measured at 44.1 kHz sampling rate.

The output of these individual analyses are stored in an array. The mode of this array is found and is said to be the sample delay for the current analysis period. Once the sample delay has been determined, a number of other metrics are calculated to be displayed on the GUI along with the sample delay.

Latency is calculated by

$$T = \frac{\tau}{F_s} \cdot 1000\ ms \tag{6}$$

where T is the latency measured in milliseconds, F_s is the sampling rate, and τ is the sample delay estimation.

Path Length Offset is calculated by

$$\Delta = \frac{T \cdot c}{10}\ cm \tag{7}$$

where Δ is the path length offset in centimeters, T is the latency in milliseconds, and c is the speed of sound in meters per second [15], as determined by (8),

$$c = 331.3\sqrt{1 + \frac{\theta}{273.15}}\ m/s \tag{8}$$

and θ is the ambient temperature measured in degrees Celsius.

The accuracy metric is determined by counting the number of frames that resulted in a sample delay of $\pm$ 1 sample from the delay estimation, and then by dividing this by the total number of frames analyzed. Therefore, the greater number of frames that returned the same sample delay as the given estimation, the greater the certainty the provided estimation is accurate.

Finally, a delay correction feature was included that utilizes a separate circular buffer to allow for one channel of the incoming audio stream to be delayed by the current delay estimation. This is not meant to be used as a production correction tool, but instead as way for the engineer to evaluate timbral or localization effects of the current microphone placement for a source.

The current implementation is intended to be used as a tool for audio engineers when placing overhead microphones for a drum kit. By placing the plugin on a stereo track where the left channel is the signal from one overhead microphone, and the right channel is the signal from the other overhead microphone, the plugin will provide sample accurate delay estimation for the source being tested.

For example, by striking the snare drum, assuming the amplitude is greater than the set threshold, the delay between the two microphones will be calculated along with the associated path length difference. Using this information the audio engineer can adjust the microphone placement to achieve better phase coherence. This process can be completed for the kick drum as well until a placement that results in minimal phase incoherence is found. A demonstration of this process and the resulting performance of the plugin are presented in the following section.

The code for this plugin is open source and available on GitHub [1]. Using the JUCE Framework AU/VST plugins can be built from the source for a number of different platforms with support for many DAWs.

4 Results

To test the performance of the real-time plugin, a test methodology that reflected the real world use of the plugin was designed. Previous testing of the GCC-PHAT [7] [13] employed a wide array of theoretical signals and pseudo-realistic signals from various recordings played over loudspeakers.

[1] `http://bit.ly/2ts0xmy`

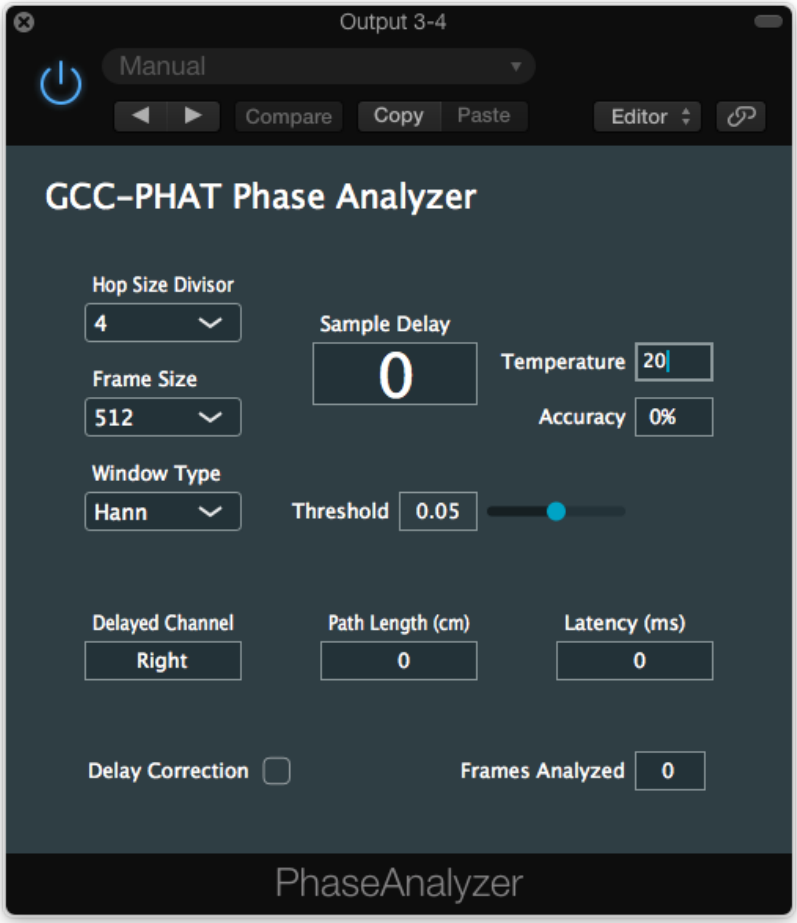

Figure 7: Graphical User Interface shown in Logic X, built with the JUCE framework

Since the goal of this project was to develop a system to be used by audio engineers on real sources, the tests included here were performed by striking real snare and kick drums with overhead microphones placed above the drums in a recording studio setting. The output of the plugin GUI was observed while striking the drums, as the audio engineer would.

To perform the tests, each drum was set with two overhead microphones placed above the drum at an equal height. One microphone was then moved upwards from this position in 1 cm intervals up to and offset of 30 cm. This setup can be seen in Figure 8 with an offset of 1 cm between the two microphones. At each interval the drum was struck three times and the sample delay and path length difference that appeared after each strike was recorded. This process was repeated for the kick drum. The temperature of the room was recorded and entered into the plugin in order to achieve the best possible path length difference estimation.

During these tests the AU build of the plugin was run in Logic Pro X at 24 bit depth and 44.1 kHz sampling rate. For this set of tests the user parameters were held constant with the hop size divisor

Figure 8: Experimental setup showing two KM184s placed above the snare drum with a 1 cm offset

set to 4, the analysis frame size set to 1024 samples, with the Hann window type, and a threshold of 0.05. The measured temperature in the room was 20.2 degrees Celsius.

The distances returned by the delay estimation for each offset distance were averaged and then the actual offset values were subtracted from these estimations to produce residual values. These residuals were then plotted over the offset distance for the snare drum in Figure 9 and for the kick drum in Figure 10. For both sources, the estimated path length distance was within 1 cm of the actual path length offset. This was true over the 30 cm range tested, which covers the expected offset difference present in most overhead microphone placement scenarios.

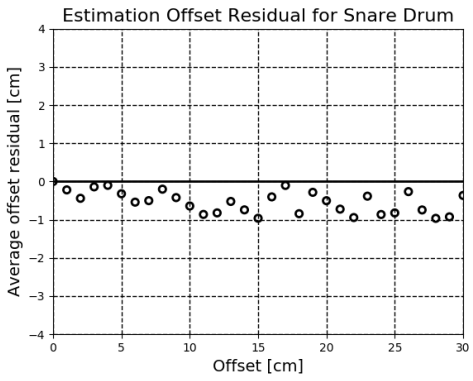

Figure 9: Average residual values over three trials for the snare drum delay estimation

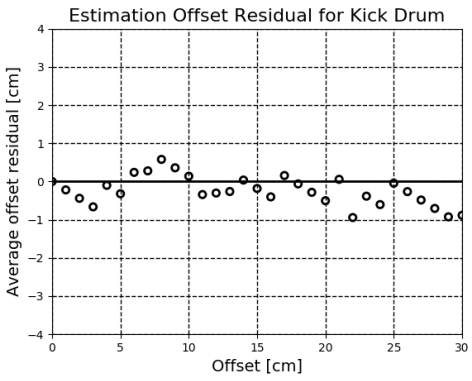

Figure 10: Average residual values over three trials for the kick drum delay estimation

5 Future Work

In this work a functioning real-time plugin was developed to provide audio engineers with a utility to simplify the process of achieving satisfactory placement of overhead microphones. The current plugin employs a fairly advanced user interface and would certainly be further simplified to allow engineers to more easily utilize the software. In addition there is interest in extending the current plugin to further simplify the microphone placement process via other means. We are interested in exploring possible augmented reality (AR) systems that would be able to convey proper placement locations to the user via visuals placed onto the audio engineer's physical space. This could be achieved by using a mobile application that incorporates an augmented reality framework, a smartphone camera, and a network connection between the plugin and the mobile application to more effectively convey positional information. This would make it significantly easier for proper microphone placement to be achieved.

6 Conclusion

The developed real-time plugin is successful in conveying sample accurate delay estimation for a pair of overhead microphones when recording or reproducing a drum kit. By providing the audio engineer with detailed information derived from the sample delay estimation, they can gain insight about the phase coherence for the kick and snare with current microphone placements, and can also to use this information to improve microphone placement. This plugin reduces the demand placed on the audio engineer to detect phase incoherence via auditory inspection, expedites the microphone setup process, and ultimately gives the audio engineer an extra level of assurance in regards to microphone placement before recording or reproduction begins.

References

[1] G. Sigismondi, R. Walker, and T. Vear, "Microphone techniques for recording," *Shure Incorporated*, 2014.

[2] J. O. Smith, *Physical Audio Signal Processing*. `http://ccrma.stanford.edu/~jos/-pasp/`, accessed 2018, online book, 2010 edition.

[3] H. Wallach, E. B. Newman, and M. R. Rosenzweig, "The precedence effect in sound localization (tutorial reprint)," *J. Audio Eng. Soc*, vol. 21, no. 10, pp. 817–826, 1973.

[4] G. Massenberg. In studio with george massenburg - ep. 1 : miking the drums. Audiofanzine. [Online]. Available: https://www.youtube.com/watch?v=OZOVZQgXl9k

[5] B. Owsinski, *The Recording Engineer's Handbook*. Artistpro, 2004.

[6] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 4, pp. 320–327, Aug 1976.

[7] N. Jillings, A. Clifford, and J. D. Reiss, "Performance optimization of gcc-phat for delay and polarity correction under real world conditions," in *Audio Engineering Society Convention 134*, May 2013.

[8] G. C. Carter, A. H. Nuttall, and P. G. Cable, "The smoothed coherence transform," *Proceedings of the IEEE*, vol. 61, no. 10, pp. 1497–1498, Oct 1973.

[9] M. S. Brandstein and H. F. Silverman, "A robust method for speech signal time-delay estimation in reverberant rooms," in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, Apr 1997, pp. 375–378 vol.1.

[10] D. Hertz, "Time delay estimation by combining efficient algorithms and generalized cross-correlation methods," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 1, pp. 1–7, Feb 1986.

[11] F. J. Harris, "On the use of windows for harmonic analysis with the discrete fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, Jan 1978.

[12] C. Roads, *Microsound*. The MIT Press, 2002.

[13] A. Clifford and J. Reiss, "Using delay estimation to reduce comb filtering of arbitrary musical sources," *J. Audio Eng. Soc*, vol. 61, pp. 917–927, 11 2013.

[14] "Juce cross platform c++ framework." [Online]. Available: https://juce.com/

[15] D. Howard, D. M. Howard, and J. Angus, *Acoustics and Psychoacoustics*. Focal Press, 2009.