Audio Engineering Society

Conference Paper

Presented at the International Conference on Audio for Virtual
and Augmented Reality, 2018 August 20–22, Redmond, WA, USA

Wave Acoustics in a Mixed Reality Shell

Keith W. Godin, Ryan Rohrer, John Snyder, and Nikunj Raghuvanshi

Microsoft Corp., 1 Microsoft Way, Redmond, WA, USA

Correspondence should be addressed to Keith Godin (kegodin@microsoft.com)

ABSTRACT

We demonstrate the first integration of wave acoustics in a virtual reality operating system. The Windows mixed reality shell hosts third-party applications inside a 3D virtual home, propagating sound from these applications throughout the environment to provide a natural user interface. Rather than applying manually-designed reverberation volumes or ray-traced geometric acoustics, we use wave acoustics which robustly captures cues like diffracted occlusion and reverberation propagating through portals while reducing the design and maintenance burden. We describe our rendering implementation, materials-based design techniques, reverberation tuning, dynamic range management, and temporal smoothing that ensure a natural listening experience across unpredictable audio content and user motion.

1 Introduction

In the development of personal computers, a command-line operating system (OS) first gave users access to one application at a time. Graphical shells were then invented, supporting simultaneous access to multiple applications. Graphical 2D shells were often based on a ‘desktop’ metaphor meant to supply users with a mental model for system behavior. Early VR systems, on the other hand, transport the user out of the shell to various self-contained VR experiences one at a time.

A simple extension to VR of the 2D desktop metaphor is the 3D ‘house’. Just as the desktop leveraged users’ real-world experiences with windows, documents and folders to intuitively access multiple data sources and applications, the house (or other environment) naturally extends this to 3D. Executing this design principle demands audio-visual realism, motivating our integration of a wave-based acoustics system into a VR shell.

Windows Mixed Reality lets the user interactively place multiple rectangular application windows as slates on the walls of a virtual house, shown in Figure 1. Rather than rendering the audio generated by these applications directly to the device speakers or headphones, the operating system inserts a spatial audio rendering pipeline into the output stream that applies head-related transfer functions (HRTFs) and environmental effects. These effects are a 6D function of application location (3D) and listener location (3D).

Acoustics modeling for games commonly employs a bank of reverberation filters that are selected as the listener moves between different parts of the scene [1], [2]. Such systems require significant work to specify listener volumes, select appropriate filter presets, and design distance-attenuation curves based on geometric heuristics. The work is typically done by hand and must be maintained as the environment is revised. The approach also fails to capture true 6D acoustic variation, instead swapping between piece-wise constant volumes based solely on listener location and ignoring dependence on source location.

Figure 1. 'Cliff House'

These limitations motivate physically-based approaches.

One example of a physically-based approach is geometric acoustics, which traces rays of sound in an infinite-frequency approximation to the wave equation [3]. Though recent commercial [4] and academic [5], [6] systems capture some propagation effects, accurate modeling of diffracted occlusion, scattering and reverberation within CPU budgets for VR remains a research challenge. Geometrically complex scenes are especially difficult, requiring tracing an enormous number of paths, with insufficient path tracing causing audio artifacts from spatially inconsistent results [7].

To achieve audio realism efficiently and minimize maintenance costs for a complex and fast-evolving product, we employ the wave-based system Triton [8], which has recently found practical application in games [9]. Triton solves the wave equation directly, modeling diffracted occlusion around corners and at doorways. All propagation paths are included (though not explicitly traced) so that overall sound level, reverberation level, and tail-length vary smoothly as the user moves between rooms or between indoors and outdoors. This enables, for example, the scenario where movie sounds bleed quietly but naturally from the theater doorway, ensuring user awareness of application activity but avoiding unrealistic interference with other applications.

The rest of this paper describes our approach and findings. Section 2 summarizes Triton. Section 3 outlines the system architecture and rendering, which extends the Windows Sonic for Headphones system [10]. Section 4 describes our materials-based design approach, and shows how it automatically creates distinct listening spaces while speeding prototyping and reducing the maintenance burden. Section 5 shows how non-linear temporal smoothing can account for the sudden level changes due to the common VR locomotion technique of teleportation. Section 6 describes the development and tuning of custom impulse responses which achieve a neutral sound appropriate for a wide range of content. Section 7 presents a novel technique for dynamic range management which accounts for the varied source level of application sounds while preventing clipping and preserving distance and occlusion cues. Section 8 describes how we accounted for the scene dynamics limitations imposed by the pre-computation stage inherent in Triton's design. Finally, Section 9 summarizes our findings and speculates on the future of acoustic simulation as part of the operating system.

2 Background

Triton performs wave simulation directly on visual scene geometry, eliminating the need for a designer to generate and maintain simplified scene geometry for acoustics. Wave simulation is performed in a pre-processing stage, with the results stored in a scene-specific data file shipped with the product. Impulse responses, each having tens of thousands of dimensions, depending on sampling rate and tail length, are difficult to smoothly interpolate, so the wave-simulated impulse responses are encoded to a low-dimensional perceptual representation we refer to as 'acoustic parameters'. These are four dimensional, closely related to well-known objective room acoustic parameters [10], and consist of 1) contribution of scene geometry to first-arrival

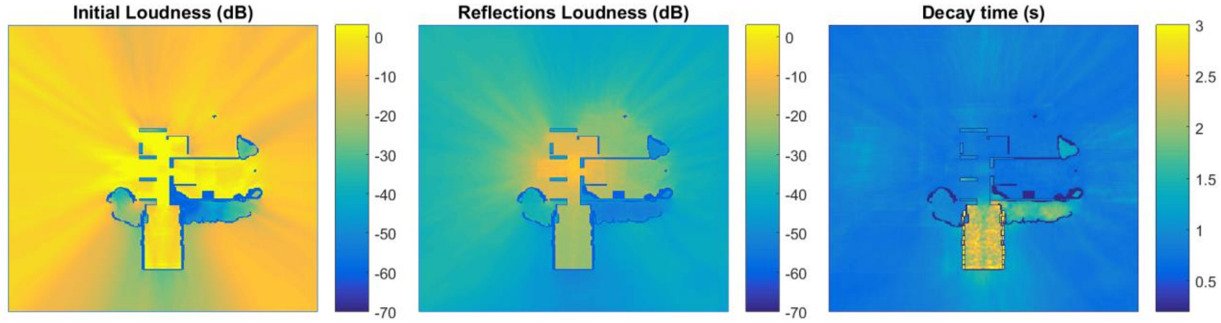

Figure 2. Perceptual parameter fields on the "Cliff House" scene (horizontal 2D slice of 3D field).

loudness, i.e. first-arrival power relative to the $\frac{1}{r^2}$ decay in distance r, 2) total reflections power, 3) early (reflections) decay time and 4) late reverberation decay time (RT_{60}).

Figure 2 shows an example for a single listener position in the simulated environment. The resulting parameter fields are spatially smooth, compressible, and easy to interpolate. Further, this low-dimensional representation allows impulse responses to be reconstructed at runtime from a small, finite set of prototype filters, providing designer control over perceptually important but spatially invariant impulse response characteristics. This reconstruction method also results in an efficient rendering by performing convolution once for each prototype filter on mixed inputs, rather than once per sources as for other auralization systems.

3 System architecture

The architecture of the Mixed Reality shell consists of components, such as a physics engine and an audio engine, that are also found in gaming applications. However, in contrast to the self-contained design of a virtual reality game, the MR shell hosts 3rd-party applications, which complicates the architecture by adding additional security and stability requirements. As the host to these applications, the operating system provides the acoustics simulation and audio rendering that make up the virtual world these applications share, while the applications supply the audio streams, which the user then perceives as emitted by the virtual objects representing the windows within which these applications render.

Audio from all applications hosted in the shell is routed from the public Windows audio APIs (e.g. WASAPI [12]) into the shared system HRTF renderer embedded inside the Windows audio engine, the same renderer instance used by the Windows Sonic for Headphones feature exposed by the ISpatialAudioClient (ISAC) API [11]. Multi-channel streams are downmixed to mono for rendering as a single point-source at the center of the application window. The renderer also incorporates the distance-based dynamic range management algorithm described in Section 6, and the filter impulse responses described in Section 5. This design ensures that the simulation is applied to applications without deliberate intervention by the application developer.

We used the filtering technique described in [8] to implement the dynamic RT_{60} effect. In this filter architecture (Figure 3), filters of different RT_{60}s are each set up as an effects send bus, and an algorithm specifies the send levels for each source into each filter to achieve the desired perceptual RT_{60} for that source. Figure 4 shows an impulse response and its transfer function reconstructed in this manner. The convolution cost is independent of the number of sources and depends only on the perceptual accuracy and desired RT_{60} range while the total cost is linear in the number of sources, with the per-source contribution being the mix cost and an input FFT. Further cost reduction comes from integrating environment modelling and HRTF processing, which

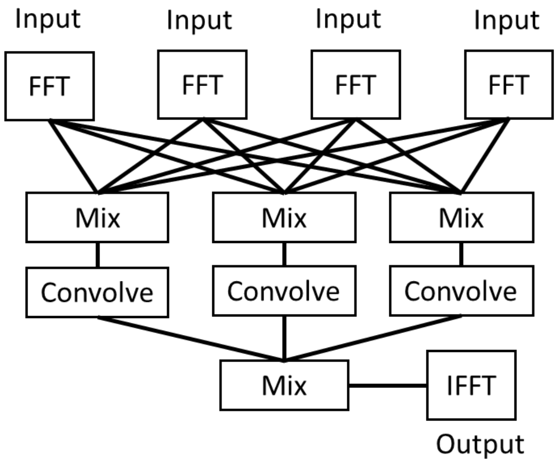

Figure 3. Signal flow diagram for dynamic RT_{60} filtering.

allows these processing paths to share FFTs and IFFTs.

4 Materials-based design

The acoustic design process used with a wave-based solver entails design in terms of the acoustic absorption properties of materials in the scene, replacing the more traditional design workflow involving reverberation volumes and attenuation curves. Materials properties are a more stable design surface, in the sense that the solver automatically handles the acoustic impact of changes in the environment layout, such as room volume changes, or the addition or removal of portals. Default materials properties can also seed prototype environments with usable initial acoustic properties, speeding the prototyping process. The wave propagation result also provides realistic attenuation and reverberation for the combinatorial design problem of sounds propagating from each space in the environment to each of the other spaces in the environment. Therefore, the use of a wave-based solver does not eliminate the task of acoustics design, but rather transforms the design process into one based on materials selection.

As a practical matter, Triton reads the same physically-based rendering (PBR) materials properties of scene objects used by the lighting system, and a lookup table is used to map materials names to acoustic absorption coefficients. Acoustics design then amounts to developing this lookup table to achieve the desired acoustic dynamics. Then, as in traditional sound design techniques, the rendered sound is a combination of the acoustics design and the underlying filters chosen for rendering, as described in Section 5, 'Filter tuning.'

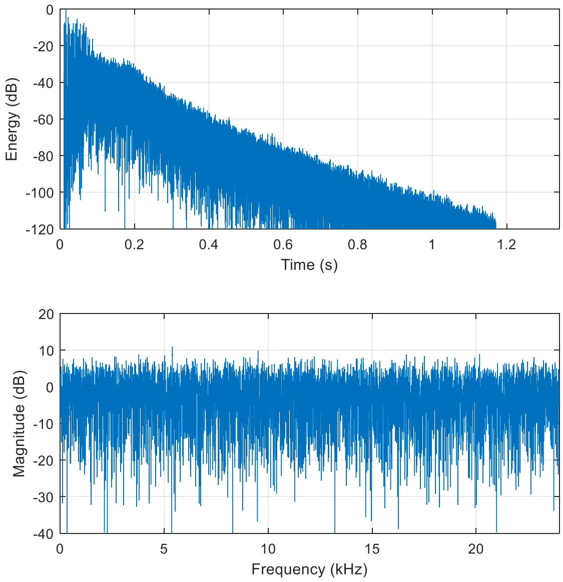

Figure 4. Impulse response an transfer function magnitude synthesized with EDT_{60} = 750ms, RT_{60} = 750ms.

We proceeded with the initial design pass intending to make the Mixed Reality shell sound as realistic as possible. Materials were selected that most accurately reflected the substances the space was 'built' with. While the initial result for the outdoor and some indoor spaces, such as the theater room, was pleasant to listen to and met the needs of our use-cases, some main-floor rooms proved too reverberant. This is because the simulation is run on an empty 'house' of typical hard materials (wood floors and gypsum wallboard walls) that is to be later filled by the user with furniture and application windows. Anyone who has stood in an empty apartment has heard this kind of realistic but 'unnatural' reverberation sound.

To remedy this, non-realistic absorption materials were applied to the walls and floors of certain main-floor spaces in the house to account for the effects of additional objects. We began by considering likely use cases of the house, including likely additional objects and total occupancy of objects placed in the space, and estimated the total surface area of walls, floor, and ceiling that would be covered by such objects. The absorption coefficient was then the weighted average of the absorption coefficients for these estimated objects and the unoccluded portions of the walls, floor, and ceiling, weighted by their estimated surface area. This resulted in a more pleasant reverberation level for these spaces, and one more in line with expectations for apartment or house-type spaces, while preserving the advantages of spatially smooth and accurate source propagation inherent to the wave-based solver.

5 Temporal transitions

Windows Mixed Reality supports 6 degrees-of-freedom (6DOF) tracking and users can walk about the Mixed Reality shell, but like most VR experiences the primary method of movement is teleportation. Because teleportation allows users to move large distances instantaneously, large acoustic changes can occur in the span of a single audio processing frame (21.3ms, which is 1024 samples at 48kHz sampling rate), despite the spatial smoothness of acoustic parameters inherent in Triton's design. These large changes can be unpleasant in the kinds of media consumption and productivity scenarios that are the primary use-cases of the shell. Therefore, nonlinear dampening was applied to the non-directional acoustic parameters to spread large changes over time.

The dampening operation is to cap the maximum change that can occur in each non-directional acoustic parameter on each frame. While non-linear and stateful, the effect is simple to model mentally, making tuning of the cap level a straightforward exercise. In this damping operation, a large jump in one of these parameters takes many audio frames, but small adjustments such as head movement or a user walking through the space are instantaneous. A linear smoothing operation such as a low-pass filter, on the other hand, would slow both large changes and small changes.

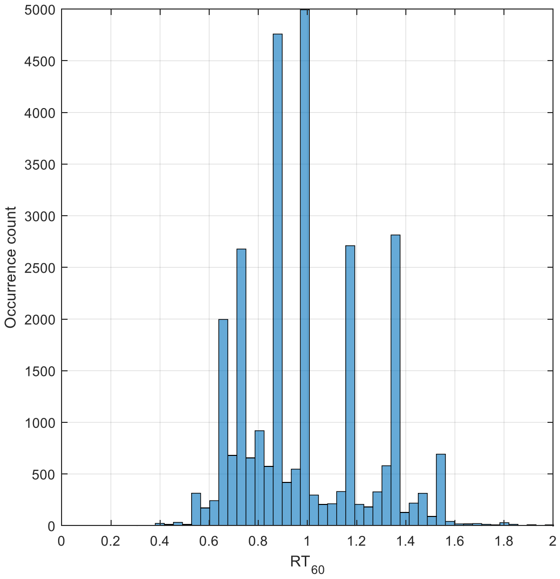

Figure 3. RT_{60} distribution in the Cliff House from 28,680 samples.

Smoothing is not applied to the arrival direction. In our experience, users were not bothered by sudden jumps in arrival direction due to teleportation. Instead, in our informal listening tests, any deviation from low-latency arrival direction updates in any scenario was immediately apparent and undesired.

6 Filter tuning

Audio from applications hosted in the Mixed Reality shell consists of a wide variety of pre-recorded and dynamic 3rd-party content, some for entertainment purposes and others for productivity, informational, or educational purposes. As such, the shell sound design experience demands a content-neutral approach to develop the right reverberation sound for these diverse categories, which is especially difficult considering the divergent needs of movies, podcast content (speech), and ambient sounds like bird and ocean sounds. As described in Section 3, materials-based design partly addresses these needs by creating separate spaces in the environment tuned for these

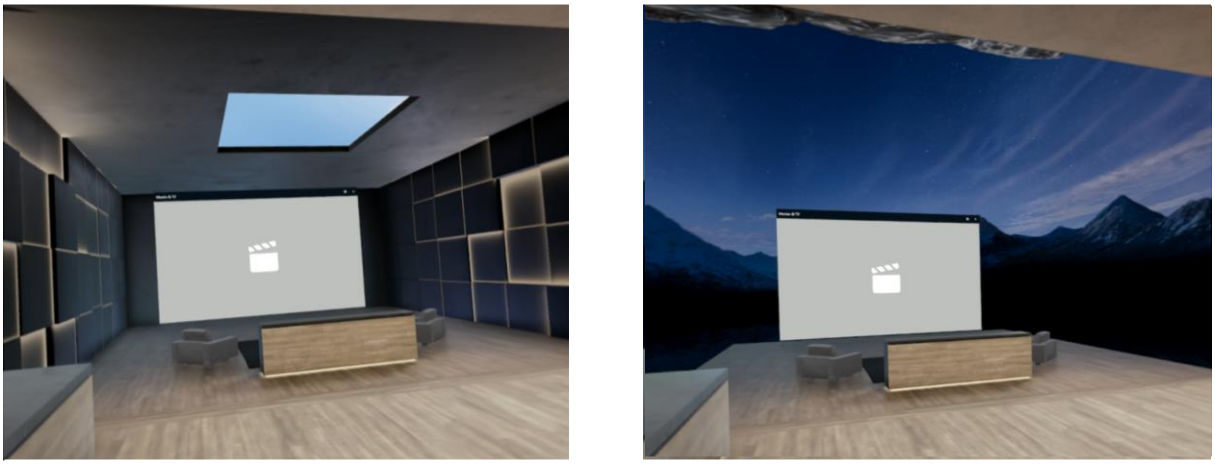

Figure 4. Cliff House theater shown with walls closed and open

different purposes. The overall sound, however, depends also on the impulse responses used for the dynamic RT_{60} system. To ensure the greatest possible control over the sonic experience, the impulse responses were custom designed in MATLAB according to the canonical filter method described in [8]. Also, because the prototype filters are fixed instead of generated at runtime, various perceptually important but spatially invariant characteristics can be subject to the aesthetic determination of a designer.

The synthesis technique for the prototype filters has two parameters forming the aesthetic control surface, they are 1) the diffuse energy fraction and 2) the initial diffuse power. The diffuse energy fraction is the fraction of the energy in the early reflections filter (comprising the first 200ms of each prototype filter) supplied by the filtered white noise that forms the diffuse part of the response, with the remaining fraction supplied by specular reflections. The diffuse portion of the response exponentially increases from the initial level specified by the 'initial diffuse power' parameter.

Taken together, these parameters influence the well-known 'mixing time' parameter of room impulse responses that denotes the transition period between specular early reflections and diffuse late reverberation. The shell environment consists primarily of indoor spaces where the mixing time is expected to be short; further, a higher diffuse energy fraction was found to be more pleasant in general for the informally evaluated media consumption scenarios. We set the diffuse energy fraction to 75%. The effects can be seen in the filter time domain plot in Figure 3.

A histogram of RT_{60} values was used to select the set of RT_{60} values for the canonical filters to balance accuracy in interpolated perceptual RT_{60} with filter cost (Figure 5). The histogram was generated by finding the acoustic parameters for the pairwise cross of a set of probe points evenly sampling the region of possible player locations. The RT_{60}s selected for the three canonical filters were the 95% minimum, 95% maximum, and mode of the RT_{60} histogram in the scene, namely 500ms, 1.6s, and 1s. The histogram shows some peaks due to quantization of the RT60 values inside the encoder.

7 Distance-based dynamic range management

The 3rd party applications hosted by the shell are written to an API set originally designed for desktop and tablet use cases, and so the audio output levels of these are mastered according to that expectation. Therefore, we assume that the audio output level from each application is the intended 'comfortable' or 'usable' listening level, and that this therefore should be the listening level associated with a comfortable viewing distance from the application window. However, at distances other than this expected viewing distance, realism of the simulation implies a distance-based gain and attenuation of the application output level. Attenuation may be safely applied, but the audio output level might be so high any gain would cause clipping. Or, there may be some amount of gain that could be applied to near distances. To wring as much realistic dynamics from the experience as possible, we devised a distance-based dynamic range management scheme that estimates the headroom available in an audio stream and uses the available headroom to apply distance-based gain when the source is nearer than the predefined comfortable listening distance.

The algorithm proceeds in two stages, first using a finite time window to estimate signal headroom, and then by computing the exponential gain coefficient that would spread that headroom over the region between the comfortable viewing distance and a minimum viewing distance. To compute a linear source gain α at each 21.3ms frame k, for input signal $s[n]$, frame length N, source distance d, comfortable listening distance d_l, minimum listening distance d_m, maximum headroom G, gain history buffer length T, and gain g_g of the source relative to free-field decay as estimated by Triton:

$$g_s(k) = \max_{i\in[kN,kN+N-1]} s[i] \tag{1}$$

$$g_k = \max\left(\frac{G}{\max_{i\in[0,T-1]} g_s(k-i)}, 1\right) \tag{2}$$

$$c_k = \begin{cases} 1, & d \geq d_l \\ \min\left(\frac{\log g_k - \log g_g}{\log d_l - \log d_m}\right), & d < d_l \end{cases} \tag{3}$$

$$\alpha_k = \begin{cases} 1, & d = 0 \\ g_g\left(\frac{d_l}{d}\right)^{c_k}, & d > 0 \end{cases} \tag{4}$$

8 Scene dynamics

Each acoustic bake in Triton is for a fixed scene, with fixed materials, portals, and occluders. There are three general approaches to scene dynamics when using Triton for acoustic modelling. One option is to simply ignore scene dynamics by baking the scene in a 'minimal occluder' configuration, with all portals open and other dynamic occluders disabled. Another is to use a separate bake for each possible state of the scene, which is possible if there are very few states. A third option is to use a simple occlusion test to overlay additional attenuation when dynamic occluders such as doors are closed.

The Mixed Reality shell uses a combination of the first two techniques. User-placed content such as furniture and application windows don't participate in the acoustic simulation. There is one dynamic occluder, the ceiling and walls of a basement theater room (Figure 5). Two separate bakes are used, one for the 'open' state and one for the 'closed' state. The wall opening and closing process is also a motivation for the temporal smoothing.

9 Conclusions

We demonstrated the first integration of wave-based acoustics in a virtual reality operating system interface. Our system modelled the propagation of hosted applications' sound to provide the user with a natural interaction model. This use-case could have applied acoustics techniques like geometric acoustics or manually-designed reverberation volumes and attenuation curves, but the realism of a wave solver provides a more intuitive interaction model, and reduces the sound design and maintenance burden. The solver automatically captures cues like diffracted occlusion and reverberation propagation through portals. We described our rendering implementation, materials-based design techniques, impulse response tuning, dynamic range management, and temporal smoothing for virtual-reality locomotion.

Future work could include incorporating additional methods such as dynamic diffusion filters or directional reverberation or reflections, propagation of object radiation patterns, and frequency-dependent filtering.

10 Acknowledgements

The authors would like to acknowledge the contributions of the software development team that made this project possible: Noel Cross, Mike Chemistruck, Ashu Tatake, and Hakon Strande.

11 References

[1] VisiSonics Corp., "Technology - RealSpace3D Audio," [Online]. Available: https://realspace3daudio.com/technology/. [Accessed 1st March 2018].

[2] Google Inc., "Overview | Resonance Audio," [Online]. Available: https://developers.google.com/resonance-audio/develop/overview. [Accessed 1st March 2018].

[3] L. Savioja and U. P. Svensson, "Overview of geometrical room acoustic modeling techniques," *The Journal of the Acoustical Society of America,* vol. 138, pp. 708-730, 01 8 2015.

[4] Valve Corp., "Steam Audio," [Online]. Available: https://valvesoftware.github.io/steam-audio/index.html. [Accessed 1st March 2018].

[5] C. Schissler, R. Mehra and D. Manocha, "High-order Diffraction and Diffuse Reflections for Interactive Sound Propagation in Large Environments," *ACM Trans. Graph.,* vol. 33, 7 2014.

[6] D. Schroder, Physically Based Real-Time Auralization of Interactive Virtual Environments, Logos Verlag, 2011.

[7] C. Cao, Z. Ren, C. Schissler, D. Manocha and K. Zhou, "Interactive Sound Propagation with Bidirectional Path Tracing," *ACM Transactions on Graphics (SIGGRAPH Asia 2016),* 2016.

[8] N. Raghuvanshi and J. Snyder, "Parametric Wave Field Coding for Precomputed Sound Propagation," *ACM Trans. Graph.,* vol. 33, 7 2014.

[9] N. Raghuvanshi, J. Tennant and J. Snyder, "Triton: Practical pre-computed sound propagation for games and virtual reality," *The Journal of the Acoustical Society of America,* vol. 141, pp. 3455-3455, 2017.

[10] ISO 3382-1:2009, "Acoustics - Measurement of room acoustic parameters - Part 1: Performance spaces," *International Organization for Standardization.*

[11] Microsoft Corp., "ISpatialAudioClient interface," [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/mt779259(v=vs.85).aspx. [Accessed 1st March 2018].

[12] Microsoft Corp., "IAudioClient Interface," [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/dd370865(v=vs.85).aspx. [Accessed 1st March 2018].