Audio Engineering Society

Conference Paper

Presented at the AES International Conference on Audio for Virtual and Augmented Reality, 2020 August 17–19, Online

This Convention paper was selected based on a submitted abstract and 750-word precis that have been peer reviewed by at least two qualified anonymous reviewers. The complete manuscript was not peer reviewed. This convention paper has been reproduced from the author's advance manuscript without editing, corrections, or consideration by the Review Board. The AES takes no responsibility for the contents. This paper is available in the AES E-Library, http://www.aes.org/e-lib. All rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.

Realistic Audio AI: Spatial Sound Modelling to provide NPCs with Sound Perception

B. Cowan[1], B.Kapralos[1], and K.Collins[3]

[1] *Ontario Tech University, 2000 Simcoe St. N. Oshawa, ON L1G0C5*

[2] *University of Waterloo, 200 University Ave W. Waterloo, ON N2L3G1*

Correspondence should be addressed to B.Cowan (brent.cowan@uoit.ca)

ABSTRACT

Despite advancements in AI for games in recent years, non-player characters (NPCs) still do not perceive the world in a realistic manner. NPC's sense of hearing has been limited or ignored. Building on our previous work that saw the development of GrAF, a graph-based spatial sound framework capable of modelling the propagation of sound through complex three dimensional virtual environments in real-time, here we apply this method to NPCs, providing them with the ability to "perceive" sounds in a more realistic manner, ultimately leading to more realistic NPC behavior.

1 Introduction

Gamers are generally dissatisfied with the quality of game AI, and prefer human-controlled opponents [1][2][3]. While it is generally true that visuals/graphics sell games, poor AI is something gamers routinely complain about, and this in turn can negatively impact reviews and sales. Perhaps the most common complaint regarding game AI is the inability of non-player characters (NPCs) to behave in a believable manner [3].

Part of the reason for the failure of NPCs to behave realistically is their lack of perception of the virtual world. Game programmers often create NPCs that prioritize computational efficiency over realistic behaviours. Many games feature some form of combat as their core game-play mechanic, and the NPCs may take on the role of enemies attempting to converge on the player's position. It is not unreasonable to suggest that an NPC's most common behaviour is to seek out (and often attack) the player-character. This seeking behaviour is usually accomplished primarily through visual perception, simulated by testing for line-of-sight between the NPC and the player's character. Line-of-sight can be calculated by casting a ray through the scene starting from the NPC's head, pointing toward the player character. The NPC's sense of hearing is often ignored, or simply distance based without taking the environment into account. In many games, the NPCs behave at times as if they are deaf, unable to hear loud noises such as gunfire in an adjacent room. This type of NPC only becomes aware of the player when a direct line-of-sight is established. Other games provide the NPCs with a rudimentary sense of hearing that is often distance-based, that is, a straight-line distance between the sound and the listener. The NPC becomes aware of the player if the player makes a sound within a specified radius of the NPC. Without sound propagation modelling, occluding structures such as walls, ceilings, and floors are often ignored, which can, at times, lead to NPCs detecting the player under impossible circumstances.

The ability to model sound propagation for dozens of NPCs was not possible in the past. However, recent

spatial sound modelling techniques developed for games are capable of modelling sound propagation through complex 3D environments in real-time [4][5]. While the technology was developed with the goal of rendering realistic spatial sound for human listeners (the player), it is possible to employ the same concepts to provide a sense of hearing for NPCs too.

We present here an overview of a GrAF, a realistic, graph-based spatial audio rendering system which has been adapted for NPC AI. We first introduce the GrAF and then describe how we can use such a tool to expand the possibilities in AI audio behaviour. Finally, we conclude with a summary and some next steps in the development of this concept.

2 Background: GrAF

Existing spatial sound APIs typically approximate reverberation and occlusion based on the listener's immediate surroundings, and do not trace the path of sound propagation (e.g., Resonance Audio [6]. For an overview of the current state of the art see [7]). Without this simulation of sound propagation, it is not possible to mimic realistic human sound localization errors, or natural environmental occlusion caused by objects or other characters in the environment. Graph-based methods using nav-meshes typically trace a single path representing the shortest route between the listener and the sound source [7][8]. However, basing the sound source localization on the shortest path can cause a jarring effect if the shortest path changes due to a moving sound source or listener. These methods are also incapable of providing sound source ambiguity caused by occluding structures such as walls.

The spatial sound framework introduced by Cowan [4, 5] (subsequently called "Graph-based Realtime Audio Framework", or GrAF), is capable of providing believable spatial sound for both geometric (polygonal models or voxels) and non-geometric environments (2D, hand painted scenes). The virtual environment can be automatically converted into a 3D graph when geometry is available, or manually plotted for non-geometric environments. Replacing complex virtual environments with a 3D graph consisting of interconnected nodes greatly reduces the computational requirements. In addition, each node in the graph can be processed in parallel, making use of modern graphic processing unit (GPU) architectures.

The framework provides propagation methods that can include the natural ambiguities that occur in the real world, if desired. To accomplish this, every node in the graph is populated with both distance and direction data. The result is that every possible path connecting the listener to the sound source is explored, and the perceived direction is based on a weighted average of all available paths. Ray tracing and beam tracing methods can also model sound propagation with multiple paths. However, the efficiency is negatively affected by geometric complexity, and the computation time increases linearly with each added sound source [9][10][11][12]. GrAF abstracts away the geometric complexity (polygon count) by replacing the detailed environment with a graph. The graph is fully populated with data and serves as a look-up table for sound sources, which allows for an unlimited number of sounds with no additional processing required by the GPU.

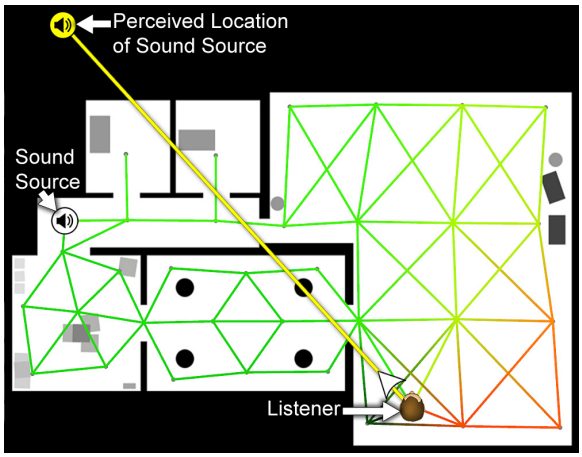

Figure 1: The spatial sound framework uses a graph to calculate the perceived distance and direction of the sound source.

Figure 1 shows a simple graph used to simulate an environment comprised of several connected rooms. GRAF has calculated the perceived direction and distance for every node in the graph based on the listener's position and orientation. The graph can now be used as a look-up table for any number of sound sources. The perceived direction and distance information can be used to play each sound at a virtual (perceived) location using a standard sound engine such as FMOD. For a more detailed overview of the framework including performance results, see [4].

3 Using GrAF for NPC AI

The initial and primary focus of GrAF was to render spatial sound for human players/listeners. However, the framework is also capable of providing a sense of hearing for any number of NPCs without any additional computation required by the GPU. Using this technology, NPCs can react to sound in a far more realistic manner, even becoming confused and being misled. Players must therefore take into account the acoustics of the environment when attempting to sneak up on enemy NPCs.

Since GrAF is capable of estimating occlusion and path length attenuation for a human listener, this same system can also be used to estimate how well NPCs are able to hear sounds originating at the player's location (such as sounds made by the player such as footsteps, or the firing of a weapon). The amount of sound energy lost along the path is direction-independent, meaning that the same amount of energy would be lost if a sound travelled from point A to point B as would be lost travelling from point B to point A. Therefore, we can use the NPC's position to determine how well the player can hear sounds made by the NPC, and we can reuse that same path length information to determine whether or not the NPC is capable of hearing sounds made by the player. No additional processing is required by the GPU; all of the data required by the NPCs was created as a by-product of the graph-based method.

It is also possible to calculate the perceived direction to/from the listener (player) from every point in the environment. The graph stores the direction that the listener would perceive the sound to be coming from for each node. However, the perceived direction is based on the last-leg of the journey, or the last obstacle that the sound had to pass before reaching the listener. This direction cannot simply be reversed to find the direction that an NPC would perceive sound originating at the listener's location. Instead, the NPC must find the nearest node in the graph, and use the path length of each of the neighbours to estimate the direction as shown in equation 1.

$$\hat{V} = \left\| \sum_{i=0}^{c-1} (P_n - P_i) \times L_i \right\|, \qquad (1)$$

where, $\hat{V}$ is a normalized vector pointing in the perceived direction of a sound originating at the listener's position. $\boldsymbol{P_n}$ represents the position of the nearest node to the NPC, and $\boldsymbol{P_i}$ represents the position of the $\boldsymbol{i^{th}}$ neighbouring node. $\boldsymbol{L_i}$ is the path length stored in the $\boldsymbol{i^{th}}$ neighbouring node.

In this way, the method can simulate a human-like perception of sounds originating at the player's location for any number of NPCs. By following the perceived direction of sounds made by the player, NPCs are effectively performing pathfinding, assuming that there are no obstacles to sound. The method is essentially a GPU-based pathfinding technique capable of calculating the path taken by the sound from every point in the environment to the player, and from the player to every point in the environment simultaneously and in real-time [4].

3.1 Environmental Factors

Knowing the direction that the NPC perceives a sound to be coming from, along with the estimated path length, may be sufficient to simulate rudimentary AI. The NPC will simply move in the opposite direction of sound propagation for a short distance before switching back to a wandering behaviour. The player would have to repeatedly make noise in order to guide the NPC to their location. This may be an acceptable method to simulate low-level animal intelligence, but it is insufficient to simulate the thought process of a human or other high-level intelligence opponent. Upon hearing a sound made by the player, enemy NPCs should estimate the location where the sound originated from, travel to that location, and then begin an organized search of the surrounding area. An NPC's ability to estimate the

origin of a sound depends on many factors, including the type of environment, the characteristics of the sound, the amount of background noise present, and the individual characteristics of the NPC. This localization accuracy, and the threshold of hearing of each NPC can be customized to suit the type of character they are meant to portray.

Other environmental factors should also be considered in order to simulate real hearing. We know that reverberation can have a negative effect on the listener's ability to locate a sound source [13]. More specifically, reflections with a short delay and amplitudes similar to the direct signal interfere with our ability to accurately localize sound sources [14]. However, the ratio of the direct to reverberant energy provides an important distance cue for sound sources that are greater than one meter from the listener [15]. In addition, reverberation may provide additional information about the environment where the sound originated, including the size of the room and the reflectivity of the walls and other surfaces. A listener in a familiar environment may know which room a sound originated from based on the acoustics of the room, even if the path between the listener and sound source is indirect.

Having additional information about the environment such as the average reflectivity of the materials present, or the room size, allows the system to estimate how useful reverberation is for sound localization. In audio middleware FMOD, room size is a value that ranges on a scale between 0 (e.g., a small enclosed space such as a hallway or a tunnel), and 1 (an open space), and it is used to describe how open or enclosed an environment is. A GPU-based algorithm that estimates both reflectivity and room size in real-time for use with FMOD and other sound engines is described by Cowan and Kapralos [16]. Other environmental factors, such as background noise and acoustical occlusion can reduce the listener's ability to detect a sound, in addition to interfering with their ability to accurately locate a sound source. All of the factors discussed above were taken into account when designing this system to approximate human hearing for artificial characters in a virtual environment.

3.2 Audibility

An important consideration is to determine whether the NPC is able to hear a sound. This hearing ability can be calculated as follows: First, the amplitude of the sound is decreased following the inverse square law (Equation 2).

$$A_1 = A_0 + [1 - (1 - R_s) \times R_r] \times 20 \times \log_{10} \frac{1}{D_p \times 10^{-3}}, \quad (2)$$

where, A_1 is the amplitude attenuated by distance, A_0 is the source amplitude, R_s is the room size, R_r is the reflectivity of the environment and D_p is the distance of the shortest path between the sound source and listener. Note that the path distance (D_p) is multiplied by 10^{-3} to convert from meters output by the graph, to kilometres. Using this formula, sound attenuation generally follows the inverse square law except when the environment is very enclosed such as a tunnel (R_s is close to 0) with highly reflective surfaces (R_r is close to 1). The room size R_s and reflectivity R_r are based on the room size and reflectivity properties of the nearest node to the player (the player is the sound source that NPCs are listening for).

Next, the attenuation caused by the medium (M), in this case air, must be accounted for (Equation 3). Here we assume M to be equal to 1 dB/km (10^{-3} dB/m) for sounds with a frequency of 200 Hz (roughly the frequency of male voices) at sea level with a temperature of 21 degrees Celsius and 50% humidity [ISO 9613-1, 1993].

$$A_2 = A_1 - M \times D_p, \quad (3)$$

where, A_2 is the amplitude remaining after attenuation due to distance and the medium (M) has been applied. A_1 is the output from equation 1. Dp is the path distance returned by the graph in meters. This attenuation is dependent on atmospheric pressures, temperatures and the medium.

The amplitude is reduced further by the amount of acoustical occlusion present along the path taken by the sound. The occlusion is calculated by the spatial sound framework based on the actual distance travelled by the sound, taking into account twists and turns caused by occluding structures. An occlusion value of 1 implies that the source is unoccluded, while

a value of 0 implies that no sound is able to reach the listener. Equation 4 takes the output from Equation 3 and further attenuates the amplitude based on the path occlusion output by the graph.

$$A_3 = A_2 \times \left[1 - \left(1 - O_p\right) \times (1 - R_r)\right], \qquad (4)$$

where, A_3 is the resulting amplitude after factoring in the path occlusion O_p. A_2 is the output from Equation 3 and represents the amplitude remaining after attenuation due to distance and the medium have been applied. Note that the occlusion is reduced for reflective environments. Even without direct sound, much of the sound energy would be added back in the form of reverberation.

Beginning with the source amplitude (A_0), the three formulae above (Equations 2 to 4), describe the reduction of sound energy that is able to reach the NPC. Next, the NPC's ability to hear the resulting sound based on the amount of background noise and the hearing ability of the individual listener (NPC) is accounted for. When the effect of background noise is considered, it is important to note that the presence of a sound does not prevent a listener from hearing another sound, even if that sound has a lower intensity. However, the presence of background noise does dull our sense of hearing making other sounds appear less intense [17]. Equation 5 is a useful approximation used to reduce the amplitude further based on the background noise level.

$$A_4 = A_3 \times \left(\frac{A_3}{\sqrt{A_3^2 + L^2}}\right), \qquad (5)$$

where, A_4 is the perceived amplitude of the sound due to the presence of background noise level in decibels (L). A_3 is the attenuated and occluded amplitude from the previous steps. This formula can be used to simulate locations where a consistently load background noise is generated by something in the environment (e.g., a waterfall, music, machinery) that impairs the NPC's ability to hear. If the perceived amplitude is greater than the listener's individually assigned threshold of hearing, then the listener has heard the sound and will react to it (most likely by moving to the perceived location).

3.3 Localization Error

The next step is to estimate the localization error based on the intensity at the listener's position and the direction that the listener is facing. Localization error is decreased as the sound intensity increases up to approximately 70 dB [18]. Very loud sounds may also be difficult to localize due to the reverberant energy they generate. Reverberation causes an increased localization error due to the reflected sound waves carrying conflicting directional information [19]. To approximate the effect intensity has on sound localization, Equation 6 incorporates a parabola with an axis of symmetry located at 70 dB.

$$E_1 = (0.1 \times A_4 - 7)^2, \qquad (6)$$

where, E_1 is the amount of localization error in degrees, and A_4 is the amplitude of the sound at the listener's location in dB.

Source localization error ranges between 2.75° for sounds that are in front of the listener, and up to 20° for sounds that are behind and above or below the listener under ideal conditions (young listeners seated in an anechoic chamber) [20]. Equations 7 and 8 provide an efficient approximation of localization error based on the direction the NPC is facing relative to the direction of the perceived sound.

$$w = \frac{\hat{V_l} \cdot \hat{V_s}}{2} + \frac{1}{2}, \quad (7)$$

$$E_2 = E_1 + w \times E_f + (1 - w) \times E_b, \qquad (8)$$

where, w is a weight factor used to linearly interpolate between the forward facing error (E_f) and the error factor for sounds located behind the listener (E_b). Note that E_f is equal to 2.75°, and E_b is equal to 20° for ideal listeners. V_l is a normalized direction vector pointing in the direction that the listener is currently facing. V_s is the normalized direction that sound appears to be coming from based on the path(s) taken by the sound. E_1 is the output localization error in degrees given by Equation 6.13 which is based on the perceived intensity of the sound. E_2 is the combined localization error due to the sound intensity and the direction that the listener is facing relative to the sound.

If the path between the sound source and listener is unoccluded, then the path is a straight line. A path contains occluding geometry if the path taken by the sound has a greater length than the straight line distance. Occluded paths contain twists and turns that further obscure the origin of the sound. The localization error caused by occlusion can be approximated by comparing the length of the path to the straight line distance between the listener and the sound source. If we assume that the worst case for sound source localization occurs when the path contains one sharp turn, then we can assume that the localization error caused by occlusion is between zero and our worst case depicted in Figure 2.

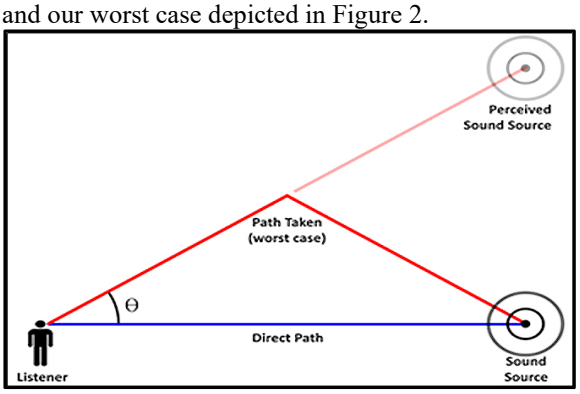

Figure 2: The worst case example, all of the occlusion is due to one sharp turn midway between the listener and the sound source. The listener may perceive the sound source as originating in a different direction due to propagation-based sound localization.

The angle represents the maximum amount of localization error caused by path occlusion. The average error due to path occlusion is assumed to be half of the maximum or worst-case error. This error level is added to the two previous error calculations based on the amplitude and the relative direction (Equation 9).

$$E_3 = E_2 + \tan^{-1}\left(\frac{\sqrt{D_P^2 - D_A^2}}{D_A}\right) \times \frac{180}{2\pi}, \qquad (9)$$

where, D_P is the path length or distance travelled by the sound returned by the graph, and D_A is the actual straight line distance between the listener (NPC) and the sound source (player). E_2 is the amount of error caused by the direction of the sound combined with the error introduced by background noise. The total amount of localization error (E_3) can be used to determine the size of the area that the NPC must to search while looking for the source of the sound. A smaller search area implies that the NPC is more confident about their estimate (Figure 3).

The location of the sound source is estimated for each NPC based the amount of localization error. The worst estimate will place the sound source in the perceived direction (V_p) with a distance equal to the path length (D_p). The best estimate will place the sound source in the correct location which is the straight line direction to the sound source (V_a) with a distance equal to the actual straight line distance (D_a) between the sound source and the listener.

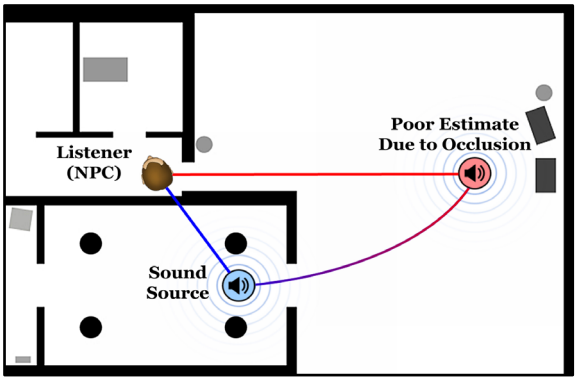

Figure 3: The diagram above contains a listener, a sound source, and poor estimate of the sound source's location caused by occluding geometry. The NPC pictured here will estimate the sound source to be located at some point on the curved line based on the accuracy calculated above.

$$w = \max\left(1 - \frac{E_3}{90}, 0\right), \qquad (10)$$

$$\widehat{V_e} = \widehat{V_a} \times w + \widehat{V_p} \times (1 - w), \qquad (11)$$

$$D_e = D_a \times w + D_p \times (1 - w), \qquad (12)$$

$$L_e = L_l + \widehat{V_e} \times D_e, \quad (13)$$

where, $\boldsymbol{w}$ is a weight factor used to linearly interpolate between two normalized vectors; the direction that the sound is perceived to be arriving from ($\boldsymbol{V_p}$), and the actual direction vector pointing directly to the sound source ($\boldsymbol{V_a}$). $\boldsymbol{V_e}$ is a normalized direction vector pointing toward the location where the NPC estimates the sound source to be, and $\boldsymbol{D_e}$ is the estimated distance between the listener and the sound source. $\boldsymbol{D_p}$ is the length of the path taken by the sound in order to reach the listener, and $\boldsymbol{D_a}$ is the actual straight-line distance between the listener and sound source. $\boldsymbol{L_l}$ is the location of the listener (NPC), and $\boldsymbol{L_e}$ is the position where the listener estimates the sound source (player) to be located.

The size of the area to be searched depends on the localization error (the NPC's confidence), and the resolve of the NPC to find the source of the sound. For example, a lazy guard may only check the location where they estimate that the sound originated from before returning to relaxed state, whereas a more tenacious guard will continue to search a wide area around the estimated source location. Equation 14 can be used to decide on the size of the area to be search based on the localization error, and the tenacity of the NPC that heard the sound. Note that the Equation 14 was designed to provide a convenient circular search area. This formula was arrived at through informal testing, and is provided for ease of implementation. Some programmers may find it difficult to construct a search behaviour for irregularly shaped areas.

$$r = \frac{E_3}{90} \times T \times D_p, \quad (14)$$

where, $\boldsymbol{r}$ is the radius to be searched in meters, $\boldsymbol{E_3}$ is the estimation error from Equation 9, $\boldsymbol{T}$ is the tenacity or resolve of the NPC to find the player, and $\boldsymbol{D_p}$ is the path length provided by the graph. If $\boldsymbol{T}$ is set to 0, the NPC would only check the estimated location, if $\boldsymbol{T}$ is set to 1, the NPC would search a wide area around the estimated location before giving up.

The characteristics of an individual listener may also be incorporated into the model. 10 dB can be used as the threshold of hearing (amplitude of normal human breathing recorded from a distance of 1m) for ideal listeners (NPCs with excellent hearing) [21]. However, higher values can be substituted to simulate NPCs with hearing loss, whether temporary hearing loss or permanent. The threshold of hearing may also be lowered to simulate the superior sense of hearing in some animals such as dogs. Localization error may be increased for NPCs wearing helmets or headphones. Additional background noise may be added for a short period of time after a loud explosion is heard by an NPC to simulate temporary hearing loss. Background noise may also be dynamic based on the NPC's proximity to sound sources in the environment. For example, it should be far easier for a player to sneak up on an NPC who is standing near loud machinery, or who is discharging a weapon.

NPCs can use the output from this algorithm to search for the source of a sound in a more realistic manner. In a graph system, the environment is simplified for the NPCs by representing it as a series on interconnected nodes which can be searched efficiently using a path finding algorithm such as A*. The output from the method described here provides an area to be searched. Nodes located close to the center of the search area have a higher probability of containing the sound source (the player). Nodes outside of the search area can be assigned a probability of 0% or a probability close to 0%. The NPC could use this information to prioritize unsearched nodes based on the probability of the sound originating at that location, and their distance from the node. Nodes are considered searched if the NPC has an unobstructed line-of-sight with the node which results in the probability at that node being set to zero (there is 0% chance that the player is present there). The NPC might return to a more relaxed patrolling state once all of the nodes have been searched.

4 Discussion

We have introduced here a method to provide more realistic NPC behaviour through the creation of a more believable and true-to-life auditory sensory ability. The graph-based spatial sound framework, GrAF, introduced by Cowan [4], produces a graph that is completely filled with simulated acoustical occlusion data. While calculating the occlusion between the listener (player) and each node in the

graph for the purpose of rendering spatial sound for a human listener, the algorithm has provided enough information to simulate a sense of hearing for NPCs. This information is a by-product of the graph-based sound propagation method. Repurposing this data for artificial intelligence requires no additional processing on the GPU, and very little extra processing on the CPU. The method introduced here is capable of providing a realistic sense of hearing for any number of NPCs in real time which can result in them behaving in a more realistic manner.

As described in the Introduction, the realistic behaviour of NPCs can be a critical component to creating a believable and immersive simulation or game. At present, the focus for realistic NPCs has been on their visual acuity rather than their auditory ability. Part of the reason for this visual-based AI focus has been the processing power involved in auditory signals. By piggy-backing on a framework that is already calculating spatial sound propagation, we can allow an AI system to use auditory sensory information by NPCs without significantly increasing processing needs.

There are some current limitations to the system described. More specifically, frequency-dependent effects have not been considered by the spatial sound framework. Where frequency is required for calculations, a frequency of 1,000 Hz has been assumed. Constructing an algorithm whereby frequency is not required as an input was done to improve ease of implementation as many game sounds contain a variety of frequencies or change frequency during playback. The frequency of the sound effects may be taken into account by supplying an average frequency of a similar sound taken from a look-up-table. In the future, sound files could be parsed in order to calculate their average frequency. In addition, some of the values used by the formulae were arrived at through informal listener testing. Using the formulae presented here will provide a believable sense of hearing for NPCs. However, extensive user testing is required to fine-tune the formulae and constants before this method could be used to predict audibility and localization accuracy for real-world environments. The algorithm presented here simulates human behaviour for NPCs by providing them with a realistic and customizable sense of hearing, it cannot be used to predict actual human behaviour.

The next stage in our research will be to conduct user studies to determine if users can detect a difference in the quality of the AI of NPCs by adjusting the NPC ability to localize sound. We believe that even if average game players cannot determine a difference in quality of NPCs with higher auditory artificial intelligence, such a difference will matter when it comes to real-world simulations for such applications as police or military training. We believe that realistic auditory behaviour is an important and overlooked aspect of current artificial intelligence in games and simulations.

References

[1] G.N. Yannakakis. “Game AI Revisited”. *CF’12*, May 15–17, Cagliari, Italy (2012).

[2] R. Arrabales, J. Muñoz, A. Ledezma, G. Gutierrez & A. Sanchis. “A machine consciousness approach to the design of human-like bots.” In *Believable bots* (pp. 171–191). Springer, Berlin, Heidelberg (2013).

[3] R.M. Lara-Cabrera, C.C. Noguiera-Collazo, & A.J. Fernández-Leiva. “Game Artificial Intelligence: Challenges for the Scientific Community.” *CEUR Workshop Proceedings.* 1394. 1–12, (2014)

[4] B. Cowan. *A Graph-Based Real-Time Spatial Sound Framework*. Ph.D. dissertation, Faculty of Science, Ontario Tech University, (2020).

[5] B. Cowan, B. Kapralos & K. Collins. “Does Improved Sound Rendering Increase Player Performance? A Graph-Based Real-Time Spatial Sound Framework.” *IEEE Transactions on Games*, doi: 10.1109/TG.2020.3000261.

[6] Resonance Audio. Discover Resonance Audio. Retrieved July 25, 2019, from https://resonance-audio.github.io/resonance-audio/discover/overview

[7] M. Beig, B. Kapralos, K. Collins and P. Mirza-Babaei. "An Introduction to Spatial Audio in Virtual Environments." *The Computer Games Journal* vol. 8, 1–16, (2019).

[8] J.F. Guay. "Real-time sound propagation in video games." In *Game Developers Conference GDC Vault* (2012).

[9] N. Tsingos, C. Dachsbacher, S. Lefebvre, & M. Dellepiane. "Instant sound scattering". In *Proceedings of the 18th Eurographics conference on Rendering Techniques* (111–120). Eurographics Association. (2007).

[10] M. Jedrzejewski, M. *Computation of room acoustics on programmable video hardware*, Master's Thesis, Polish-Japanese Institute of Information Technology, Poland (2004).

[11] N. Röber, U. Kaminski, & M. Masuch. "Ray acoustics using computer graphics technology." In *Proceedings of the 10th International Conference on Digital Audio Effects*. Bordeaux, France. (2007).

[12] Nvidia Corporation: Developer Resources. Retrieved July 15, 2019, from https://developer.nvidia.com/vrworks/vrworks-audio

[13] C.M. Zannini, R. Parisi, & A. Uncini. "Binaural sound source localization in the presence of reverberation." In *the 17th International Conference on Digital Signal Processing (DSP)* 1–6, (2011).

[14] F. Ribeiro, D. Ba, C. Zhang & D. Florêncio. "Turning enemies into friends: Using reflections to improve sound source localization." In *International Conference on Multimedia and Expo* IEEE (ICME) 731–736 (2010).

[15] B.G. Shinn-Cunningham. "Distance cues for virtual auditory space." In *Proceedings of the IEEE-PCM,* Vol. 2000, 227–230, (2000).

[16] B. Cowan, & B. Kapralos. "A GPU-based method to approximate acoustical reflectivity." *Journal of Graphics, GPU, and Game Tools*, 15(4), 210–215, (2011).

[17] E.A. MacPherson & J.C. Middlebrooks. "Localization of brief sounds: effects of level and background noise." *The Journal of the Acoustical Society of America*, 108(4), 1834–1849, (2000).

[18] R. J. Davis, & S.D.G. Stephens. The effect of intensity on the localization of different acoustical stimuli in the vertical plane. *Journal of Sound and Vibration*, 35(2), 223–229, (1974).

[19] C. Giguere, S.M. Abel. "Sound localization: effects of reverberation time, speaker array, stimulus frequency, and stimulus rise/decay." *Journal of the Acoustical Society of America*, 94 (2), 769–776, (1993).

[20] J.C. Makous & J.C. Middlebrooks. Two-dimensional sound localization by human listeners. *The journal of the Acoustical Society of America*, 87(5), 2188–2200, (1990).

[21] National Institute on Deafness and Other Communication Disorders NIDCD). (2010, June 7). I Love What I Hear! Common Sounds. Retrieved July 23, 2016, from https://www.nidcd.nih.gov/health/i-love-what-i-hear-common-sounds