VIRTUAL ACOUSTICS WITH GAME ENGINES

Massimiliano Masullo, Hasan Baran Firat and Luigi Maffei

Department of Architecture and Industrial Design, Università degli Studi della Campania “Luigi Vanvitelli”, Italy
email: massimiliano.masullo@unicampania.it

Architectural acoustics have a new direction as virtual acoustics. There are in literature several studies which have investigated the ways to obtain immersive acoustic environments. Beside the endeavours to achieve more successful point-based auralizations, scholars are trying to reproduce real-time immersive-3D acoustic environment. In addition, today we know that audio and visual perception cannot be analysed separately and it is a well-known fact that the visual perception is significantly augmented by matching sound stimuli. Through the different methods, this preliminary study will focus on utilization of the game engines in acoustic studies to create 3D immersive models and to work both visual and audio dimensions. Visualization of the virtual environment is the primary purpose of the lots of game engines. Whereas they achieved serious progress on quality of visuals, they need time to attain more successful audio realism. The most leading game engines basically work with their own audio engines that provide real-time synthesis, dynamic DSP effects and physical propagation models, but their compliance with outdoor and indoor physical acoustic laws need to be verified. Nevertheless, there are interesting developed audio features especially by the middlewares like Wwise, Steam Audio and other plugins. This paper investigates these developments and the audio features of Unreal Engine 4 and its related audio tools with focus on sound propagation. Comparisons between simulated audio phenomena and physical based acoustic models are presented.

Keywords: Virtual Acoustics, Virtual Reality, Game Engines

1. Introduction

In last decades, lots of methods and techniques have been developed to measure and simulate the sound field in several conditions. Since the Kroknstad, Strøm and Sørsdal study [1] of 1960s, the advancements in computer and acoustics sciences have led to the development of room and building acoustic software which calculate impulse responses of even complex shaped rooms by geometrical acoustics based algorithms. However the latter algorithms yields correct results as long as the dimensions of the room are large compared to wavelengths and if broadband signals are considered [2]. That’s why to overcome this problem most of acoustic simulation methods are using wave-based method for the frequencies below the critical frequency and hybrid methods of geometrical acoustics for the simulations above the critical frequency [3]. The possibility to obtain simulated impulse responses for indoor (or outdoor) scenarios has contributes to the development of several studies on auralization, term introduced by Kleiner et al.[4] to describe the convolution in room acoustics as an acoustical analogy to visualization. Although, in some complex scenes and with large volumes, still may need long calculation times, acoustic simulation software allows to obtain accurate auralizations.

More recently a new field of research called Virtual Acoustics (VA) emerged. VA investigates the reproduction of the sound environment and of its characteristics to provide the listeners to sensation of being there, immersed in a simulated or recorded sound, as in the real world.
Even though these studies are relatively new, several Virtual Acoustic systems have been investigated by different research groups. Pioneering works have been carried out by Savioja, Lokki, Schröder and Vorländer. The research group of Helsinki University of Technology implemented an auralization engine called as DIVA (Digital Interactive Virtual Acoustics) [5] which have been used over to the EVE Virtual Reality project after [6]. Another real-time simulation framework RAVEN (Room Acoustics for Virtual Environment) was developed at RWTH Aachen University by a research group of Michael Vorländer and Dirk Schröder [3]. An additional real-time auralization software environment, EVERTims, was developed by Katz et al. [7]. For our knowledge, it is the only free and open source software in market. Beside these, also REVES developed by INRIA should be considered. Its main focus area is large number of source in VR scenes [8]. The distinctive element of these researches is that they are focused on real-time auralizations to create a sonic immersive environment where you can move inside it without any pre-calculations.

After dark years for Virtual Reality (VR) between 80s and the end of 90s, thanks to the development of lots of cheap and wearable devices with great potential, this technology has obtained more and more favour and now seem to be at the centre of the attention of market by several Industries. Despite major VR developers were concentrated on visual part of the software for years, the sonic part of these virtual environment started to arouse more and more the interest of researchers. Especially researches on human perception which consider both aural and visual stimuli have showed how visual perception and auditory perception interact each other [9-11]. Understanding and perception of surrounding environment is incontestably holistic. That’s why vision, sound, olfactory and haptic should be investigated simultaneously.

Today major VR development software (game engines) like Unity3D, CryENGINE and Unreal Engine 4 (UE4) have achieved relatively successful visual rendering while they are still working to improve the quality of aural reproduction. Although the use of game engines for indoor/outdoor multisensory researches could provide more holistic experiences for subjects, the parametric approach which they use in describing the physical phenomena with regard to sound, and lighting, make Virtual Environments very limited to reproduce physically reliable environments.

For this purpose, some basic sound functions of one of the most popular game engine were investigated. Experimental sessions were carried out to measure sound propagation phenomena in UE4 and its related plugins and middlewares. Because of the insufficient documentation experimental studies gain importance. This preliminary study provides an introduction to sound propagation phenomena in game engines.

2. Game Engines

Analyzing the most popular game engines may be observed that they have many similarity, some of them highlight their visual renderings whilst other shine out for their flexibility, providing easy Graphical User Interfaces or allowing to generate customized functions by coding.

Among the different game engines, we have chosen UE4 because, besides it has widespread usage and a large user of community, it has an internal engine for light and audio simulation and it is feasible to integrate with other audio tools. Since the first version of UE, showcased in 1998 as “first-person shooter game Unreal” this game engine has reached the current release Unreal Engine 4.18 (UE4). This release has full C++ source code access and Blueprint visual scripting features.

While it is possible to generate your content with Blueprint visual scripting feature where you can create your model from GUI of UE4, you can also access the complete C++ source code. This C++ code access gains importance particularly in case of creating scenes for various different condition. The visualization and the interaction with virtual environments is the primary purpose of game engines. In the last years UE4, and the other competitors, achieved serious progresses on these topics reaching unexpected levels of graphical detail and of fluidity of images. On the contrary, less convincing seems to be the advances concerning the audio simulation. The main reason is the
qualitative or parametric approach that these game engines use which aims more to reproduce audio effects than of physical phenomena.

In this paper is presented a preliminary experimental aiming to verify quantitatively the respect of the basic physical phenomena of sound, by the Audio Engine of UE4 [12] and of two additional tools used to create more real sound.

2.1 Audio Implementation Alternatives for UE4

As Garner outlined [13], for sound designers lots of alternatives to increase the realism of soundscape and the perceived fidelity of audio in VR exists. Audio Source Development Kits (SDKs) produced by VR firms (e.g. Oculus Audio SDK), third-party plugins (e.g. Steam Audio) and middleware tools that can create sound objects and events separately, (e.g. FMOD, Wwise – Wave Works Interactive Sound Engine by Audiokinetic) are the three major options.

Middlewares are important especially because of their user-friendly GUI and because they allow to design own audio in a separate application which can be integrated with lots of game engines. Other solution is the third-party plugins that can be installed every project individually. With these, like with Steam Audio (v.2.6 beta) already integrated into UE4, it is possible to arrange audio effects from GUI of UE4. Lastly, Audio SDKs gather different plugins, tools and libraries and most importantly provide access to source code, enabling designers to define their own setup.

All of these tools can be used at once or in different procedural audio\(^1\) work flow as well.

In this study, because of their prominent place in market and potential relevance to acoustic studies, Wwise (WW) and Steam Audio (SA) were chosen for a main analysis.

3. Physical Sound Propagation

3.1 Audio Engine of UE4

This section describes some default audio settings and audio components that UE4 provides to designers to reproduce various audio effects. In particular, the software allows to build simple or combined sounds in the form of Sound Cue, editable by its Editor or by other integrated tools and libraries. This process to create sound content is called procedural audio.

UE4 supports 16-bit wave files at different sample rate (e.g. 22050, 44100 Hz). Each imported sound (mono or stereo) yields a simple sound node that can be referenced by a Sound Cue for playback. The sound level reproduced by UE4 can be adjusted with a linear scale controller from 0.0 silent to 1.0 full volume, although it has no limits and it is possible to set volume above 1.0 causing distortions of sounds. To work with dB scale, it is necessary to use a converter between dB scale and linear scale.

Within the Sound Mix asset itself, accessible from the Content Browser, several properties can be modified. EQ Settings can be used to adjust the gains of four frequency bands. As the EQ Settings of multiple Sound Mixes cannot be combined, an EQ Priority allows to control which active mix’s properties are applied at any given time [15]. The implementation of EVERTims groups octave bands into three larger bands (32 Hz-250 Hz, 500 Hz-2 kHz and 4 kHz-16 kHz) using IIR shelving filters as well in order to maintain real-time performance on current hardware [7].

For the directivity of sound sources, UE4 implements only a function which mimics this phenomena by introducing an attenuation cone which concentrates an amount of energy radiated from the front and back of sound source in a certain direction. A similar approach is used by WW.

Other acoustic features in UE4, like reverberation, reflection and occlusion, which are not a subject for this study should be also mentioned. UE4 uses a sort of binary-space partitioning (BSP) techniques

---

\(^1\)Procedural audio is a common term to define applications to create more realistic audio design which is originated from procedural generation. More precisely Farnell explains it like; Procedural audio is non-linear, often synthetic sound, created in real time according to a set of programmatic rules and live input [14].
by the name of Audio Volume, which allow to create regions named volumes to modify some acoustics properties of the sound field.

In this preliminary study two simple phenomena that characterize the sound propagation in free field condition: 1) Distance Attenuation ($A_{\text{div}}$) and 2) Air Absorption Attenuation ($A_{\text{atm}}$) were evaluated. The methodology of measurements and the experimental results are showed and commented in following sections.

### 3.2 Methodology and Measurements

Free-field conditions were modelled in UE4 to measure **Distance Attenuation** ($A_{\text{div}}$) and **Air Absorption Attenuation** ($A_{\text{atm}}$) using different tools: Wwise, UE4 Audio Engine, BP_S Attenuation (MacOSX), and SA (Windows).

The measurements layout consists of a sound source (S) and 15 receivers (R) positioned at different distances (1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192 m) doubling the distance from 1 m. The wave files (44.1 kHz, 16 bit) used for the tests were: a white noise for **Distance Attenuation** and pure tones at each central frequency band (63, 250, 500, 1000, 2000, 4000, 8000 Hz) for **Air Absorption Attenuation**.

The measurements were carried out in the 5m × 5m × 5m anechoic chamber of the Department of Architecture and Industrial Design. By means of a workstation (Intel Core i7 CPU 2.9 GHz processor, Radeon Pro 560 4 GB, Intel HD Graphics 630, 1536 MB graphic card), the sound stimuli were played back via headphones (Beats Solo 3 Wireless Headphone A1795) and recorded with a calibrated Mk1 Cortex Manikin connected to a Symphonie 01dB soundcard and the software dBTrig.

#### 3.2.1 Distance Attenuation ($A_{\text{div}}$)

UE4 allows to simulate the free-field sound propagation [12] by six different attenuation functions: **Linear, Logarithmic, Natural Sound, Log Reverse, Inverse** and **Custom**. These attenuation functions are provided for Sphere, Capsule, Box and Cone attenuation shapes. The Attenuation functions (except **Custom**) are described by the following equations:

\[ \text{Linear} = 1.0 \cdot f - \left( \frac{\text{Distance}}{\text{Falloff Distance}} \right) \]  
\[ \text{Logarithmic} = 0.5 \cdot f - \log \left( \frac{\text{Distance}}{\text{Falloff Distance}} \right) \]  
\[ \text{Inverse} = 0.02 \cdot f / \left( \frac{\text{Distance}}{\text{Falloff Distance}} \right) \]  
\[ \text{LogReverse} = 1.0 \cdot f + 0.5 \cdot \log \left( 1 - \frac{\text{Distance}}{\text{Falloff Distance}} \right) \]  
\[ \text{Natural Sound} = 10.0 \cdot f \left( \frac{\text{Distance}}{\text{Falloff Distance}} \right)^{\left( \frac{\text{dB Attenuation at Max}}{20} \right)} \]

Where:

- $f$: Attenuation scale factor  
- $f = \frac{\text{Falloff Distance}}{\text{Inner Radius}}$

**Inner Radius**: minimum distance from sound source where attenuation will be applied

**Falloff Distance**: maximum distance from sound source where attenuation will be applied;

**Distance**: current distance of receiver to the sound source;

**dB Attenuation at Max**: maximum dB attenuation at Falloff Distance.

Each of the five previous functions only run for distances greater than the **Inner Radius** and less than the **Falloff Distance**.
For testing the Attenuation functions, two different factors scale: $f_1=100/1$ and $f_2=1000/1$ were considered. For each receiver, the average (left/right) sound level of measurements was calculated using the UE4 functions: *Linear*, *Logarithmic*, *Natural* (60 dB), *Natural* (40 dB) and *Log Reverse*. Where the values of 60 dB and 40 dB for Natural functions were set according to physical attenuation level at *Falloff Distance*.

The results of the measurements and the SPLs of physical spherical free field decays were plotted in Figure 1 (for scale factor $f_1$ and factor $f_2$).

*Inner Radius* was set to 1 meter and first 1m was not measured due regard being had to near-field effect [18].

![Image](image1)

Figure 1: Comparison of the SPLs measurements for six UE4 attenuation functions and spherical free field decay, as function of distance, for $f_1$ (left) and $f_2$ (right)

As can be seen in Fig. 1, for *Logarithmic*, *Log Reverse* and *Linear* functions, the sound levels above the *Falloff Distances* was due only by the background noise level ($L_{eq}=25.2$ dB) in the chamber.

### 3.2.2 Air Absorption Attenuation ($A_{atm}$)

The attenuation due to atmospheric absorption during propagation through a distance is calculated by atmospheric absorption coefficient for each frequency band in ISO-9613. This atmospheric absorption coefficient changes as the temperature, humidity and pressure change [19,20].

In UE4, only some options are provided to simulate a given *Air Absorption Attenuation* (Figure 2) with the distance by unspecified Absorption Method (Linear or Custom).

![Image](image2)

Figure 2: Air Absorption Attenuation Options of UE4

These options are not based on the ISO-9613 but seem to be based on a Low Pass Filter (LPF) that create the effect of air absorption at high frequencies. Experiments carried out using several set-ups has showed that Air Absorption Attenuation do not provides any changes in sound pressure level.
3.3 Audio Tools (Wwise, Steam Audio and Blueprint Scripting)

In another experimental session, the Distance Attenuation and Air Absorption Attenuation obtained with audio tools were also measured and compared with the physical laws.

For Distance Attenuation, the physically based sound propagation option of SA, a customized logarithmic curve in WW and a Blueprint Script prepared by authors, were tested.

Results in Figure 3 show as, beside a near field effect that can be observed up to firsts 2 meters, the three tools have only small deviations from the physical distance attenuation curve.

![Figure 3: Distance Attenuation Curves of Acoustic Tools for UE4](image)

At the same time, for Air Absorption Attenuation, the default option of SA and a low pass filter provided by WW were also tested. Despite WW provides also a Parametric EQ filter to mimic the air absorption attenuation, this option was not used in this study because it presents a limited Falloff distance of 1000 m. The measurements made for each frequency band from 63 to 8 kHz showed that, likewise WW, SA implements different filters (see Figure 4) which, unfortunately, are not coherent with curves defined in ISO-9613 for different air conditions. (see Figure 5).

![Figure 4: Left) Low Pass Filter of Wwise-UE4; Right) Air Absorption Attenuation with SA](image)
4. Conclusions

The results of this preliminary work have showed how the Audio functions of one of the most popular Game engine (UE4) doesn’t allow to describe correctly any of two physical sound propagation phenomena observed: Distance Attenuation and Air Absorption Attenuation.

On the other hands, while an alternative and reliable representation of Distance Attenuation can be calculated by audio tools, like WW, Steam Audio and Blueprint scripts, at least for distances ranging from the inner radius and a falloff distance, for Air Absorption Attenuation, also the additional tools seems to be inadequate to simulate appropriately this phenomenon.

Entire audio features should be studied with a physical approach to use UE4 or any other game engines as an integrated platform for multi-sensory (visual-aural) perception analysis.

REFERENCES


6 The Eve: A Virtual Environment at HUT. [Online.] available: http://eve.hut.fi


