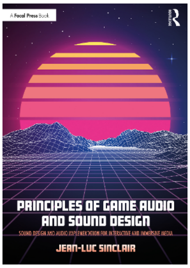
CHAPTER SAMPLER

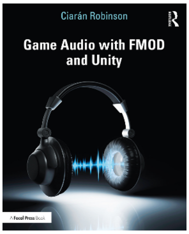
Game Audio and Sound Design

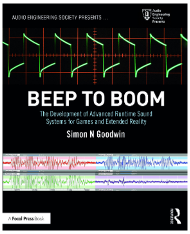
Routledge
Taylor & Francis Group

www.routledge.com

Contents







20% Discount Available

You can enjoy a 20% discount across our entire range of Routledge books. Simply add the discount code **UML20** at the checkout.

Please note: This discount code cannot be combined with any other discount or offer and is only valid on print titles purchased directly from www.routledge.com. Offer ends on 31st December 2020

THE ROLE OF AUDIO IN INTERACTIVE AND IMMERSIVE ENVIRONMENTS

Learning Objectives

The purpose of this chapter is to outline the major functions performed by the soundtrack of a video game, as well as to layout the main challenges facing the modern game audio developer.

We shall see that audio plays a multi-dimensional role, covering and supporting almost every aspect of a game or VR environment, from the obvious, graphics animation, to the less obvious, such as narrative, Artificial Intelligence and game mechanics, to name but some. All and all, the soundtrack acts as a cohesive layer that binds the various components of a game together by providing us with a consistent and hopefully exciting sensory experience that deals with every sub system of a game engine.

1. Inform, Entertain, Immerse

What is the purpose of audio in games? What makes a player turn up the volume in a game instead of streaming their favorite music playlist?

Games have come a long way since the days of the Atari 2600 and its embryonic soundtracks, the blips and noises still in our collective memory today. Newer, better technologies have come online, giving sound designers new tools and more resources with which to create the soundtracks of future games. Yet, even with the recent technological advances, crafting a compelling soundtrack remains a tricky affair at best, reminding us that technology isn't everything, and that, at its core, the issues facing the modern sound designer have at least as much to do with the narrative we strive so hard to craft as with the tools at our disposal. So perhaps we should begin our investigation not so much by looking at the tools and techniques used by professionals but by understanding the aims and challenges gaming confronts us with, and how to best tackle them.

Understanding these challenges independently from the technology involved will allow us to ultimately get the best out of the tools available to us, whatever those may be, whether we are working on a AAA game for the latest generation of dedicated hardware or a much more humble mobile app.

If we had to sum up the purpose of sound in games and interactive media we could, perhaps, do it with these three words: inform, entertain, immerse. The role of the sound designer and audio engineer in interactive media is to pursue and attain these goals, establishing a dialogue between the player and the game, providing them with essential information and data, that will help them navigate the game. Perhaps a simple way to think about how each event fits within the overall architecture of our soundtracks is through this simple equation:

Data + Context = Information

It is easy to understand the *entertain* portion of our motto. The soundtrack (a term that refers to music, dialog and SFX) of a AAA game today should be able to compete with a high-end TV or film experience. We expect the sound design to be exciting, larger than life and original. That is a challenge in itself, of course. Additionally, however, in order to create a fully encompassing gaming experience, it is also important that we provide useful feedback to the player as to what is happening in the game both in terms of mechanics and situational awareness. Using the soundtrack to provide gamers with information that will help them play better and establish a dialog with the game is a very powerful way to maximize the impact of the overall experience. Indeed, as we shall see, even a simple, mobile arcade game type can be significantly improved by a detailed and thoughtful soundtrack, and the user's experience vastly heightened as a result. Effective aural communication will also certainly greatly contribute to and enhance the sense of immersion that so many game developers aspire to achieve.

In a visually driven media world we tend to underestimate – or perhaps take for granted – how much information can be conveyed with sound. Yet constantly in our daily lives we are analyzing hundreds of aural stimuli throughout the day that provide us with information on our surroundings, the movement of others, alert us to danger or the call of a loved one and much, much more. In effect, we experience immersion on a daily basis; we simply call it reality, and although gaming is a fundamentally different experience, we can draw upon these cues from the real world to better understand how to provide the user with information and how to, hopefully, achieve immersion.

Let us take a closer look at all three of these concepts, inform, entertain and immerse, first in this chapter, then in more detail throughout the rest of this book as we examine strategies to develop and implement audio assets for a number of practical situations.

1. Inform: How, What

In a 3D or VR environment sound can and must play an important role when it comes to conveying information about the immediate surroundings of the user. Keeping in mind that the visual window available to the player usually covers between 90–120 degrees out of 360 at any given time, sound quickly becomes indispensable when it comes to conveying information about the remaining portion of the environment. It should also be noted that, while the visual field of humans is about 120 degrees, most of that is actually peripheral vision; our actual field of focus is much narrower. The various cues that our brain uses to interpret these stimuli into a distance, direction and dimension, will be examined in more detail in a future chapter, but already we can take a preliminary look at some of the most important elements we can extract from these aural stimuli and what they mean to the interactive and immersive content developer.

a. Geometry/Environment: Spatial Awareness

In a game engine, the term geometry refers to the main architectural elements of the level, such as the walls, stairs, large structures and so on. It shouldn't be surprising that sound is a great way to convey information about a number of these elements. Often, in gaming environments, the role of the sound designer extends beyond that of creating, selecting and implementing sounds. Creating a convincing environment for sound to propagate in is often another side of the audio creation process, known as *environmental modeling*. A well-designed environment will not only reinforce the power of the visuals but is also a great way to inform the user about the game and provide a good backdrop for our sounds to live in.

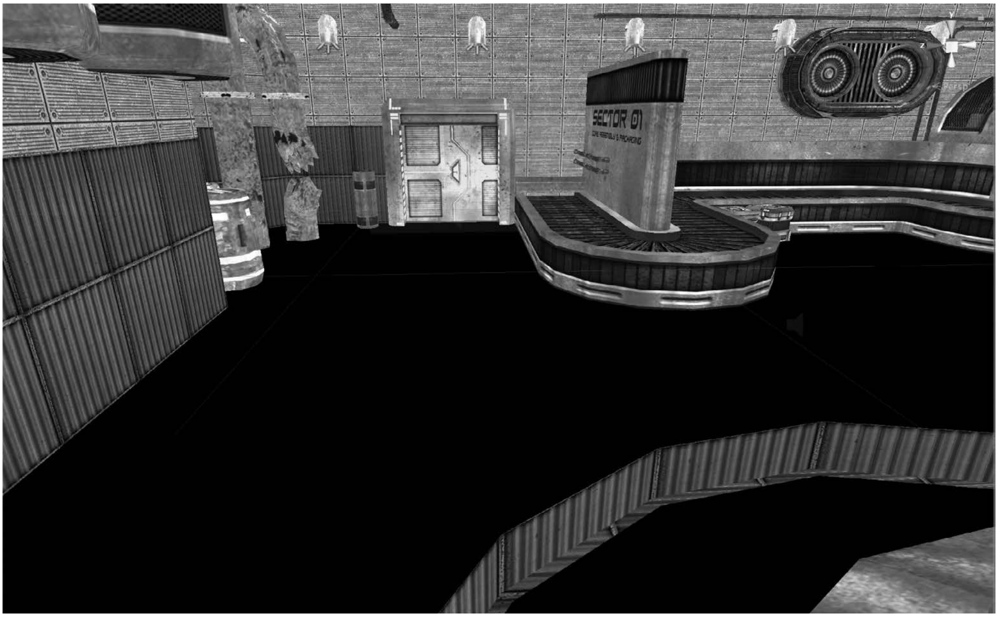


Figure 2.1

Some of the more obvious aspects of how sound can translate into information are:

- Is the environment indoors or outdoors?
    - If indoors, what is the order of the size of the room we find ourselves in?

  - If outdoors, are there any large structures, natural or man-made, around?

- Do we have a clear line of sight with the sound we are hearing, or are we partially or fully cut off from the source of the sound? We can isolate three separate scenarios:

  1. We are fully cut off from the audio source. The sound is happening in an adjacent room or outside. This is known as *occlusion*. There is no path for the direct or reflected sound to get to the listener.
  2. The path between the audio source and the player is partially obstructed, as in a small wall or architectural feature (such as a column for instance) blocking our line of sight. In this case the direct audio path is blocked, but the reflected audio path is clear: that is known as *obstruction*.
  3. The direct path is clear, but the reflected sound path isn't, blocking the reverberated sound: this is known as *exclusion*.

Each of these situations can be addressed and simulated in a soundtrack and provide the user with not just an extremely immersive experience but also valuable information to help them navigate their environment and the game itself.

b. Distance

We have for a long time understood that the perception of distance was based primarily on the amount of dry vs. reflected sound that reaches our ears and that therefore reverberation played a very important role in the perception of distance.

Energy from reverberant signals decays more slowly over distance than dry signals, and the further away from the listener the sound is, the more reverb is heard.

Additionally, air absorption is another factor that aids us in perceiving distance. Several meteorological factors contribute to air absorption; the most important ones are temperature, humidity and distance. The result is the noticeable loss of high frequency content, an overall low pass filtering effect.

Most game engines, Unity being one of them, provide us with a great number of tools to work with and effectively simulate distance. It does seem, however, that, either due to a lack of knowledge or due to carelessness, a lot of game developers choose to ignore some of the tools at their disposal and rely solely on volume fades. The result is often disappointing and less-than-convincing, making it difficult for the user to rely on the audio cues alone to accurately gauge distance.

c. Location

The perception of the location of a sound in terms of direction in 360 degrees is a little more complex a mechanism, as it in fact relies on multiple mechanisms. The most important are:

- Interaural time difference*:* the time difference it takes for sound to reach both the left and right ears.
- Interaural intensity difference: the difference in amplitude between the signal picked up by the left and the right ear.
- The precedence effect: in a closed space, the precedence effect can also help us determine the direction of the initial sound source. It was demonstrated by Dr Helmut Haas in 1949 that humans, when confronted to under certain circumstances, will determine the location of a sound based on the first arriving wave.

As outlined with these principles, our ability to discern the direction a sound comes from is dependent on minute differences in time of arrival and relative intensities of signals to both ears. While some of these phenomena are more relevant with certain frequencies than others (we almost universally have an easier time locating sounds with high frequency content, for instance), it is almost impossible to determine the location of a continuous tone, such as a sine wave playing in a room (Cook '99). A good game audio developer will be able to use these phenomena to their advantage.

The process currently used to recreate these cues on headphones is a technology called Head Related Transfer Functions, which we shall discuss in Chapter four.

Another somewhat complimentary technology when it comes to spatial audio is ambisonic recording. While not used to actually recreate the main cues of human spatial hearing, it is a great way to compliment these cues by recording a 360-degree image of the space itself. The Unity game engine supports this technology, which their website describes as an 'audio skybox'. Ambisonic and their place in our sonic ecosystem will also be discussed further in upcoming chapters.

d. User Feedback and Game Mechanics

This might be less obvious than some of the previous concepts discussed up until now, as in some ways, when successfully implemented, some of the features about to be discussed might not – and perhaps should not – be noticed by the casual player (much to the dismay of many a sound designer!).

On a basic level, audio based user feedback is easily understood by anyone who ever had to use a microwave oven, digital camera or any of the myriad consumer electronics goods that surround us in our daily lives. It is the Chime Vs. Buzzer Principle that has governed the sound design conventions of consumer electronics good for decades – and TV quiz shows for that matter.

The simplest kind of feedback one can provide through sound is whether an action was completed successfully or not. The Chime Vs. Buzzer Principle is actually deceptively simple, as it contains in its root some of the most important rules of sound design as it relates to user feedback:

The chime almost universally symbolizes successful completion of an action, or positive feedback. It is a pleasant, musical sound that we associate with immediate action and positive sentiments. The buzzer, of course, is noisy, unpleasant to the ear and associated with negative feedback and negative sentiments. Both these sounds have the benefit of being easy to hear, even at moderate levels in a somewhat crowded or noisy environment, although the chime appears to achieve similar results while remaining pleasant to the listener.

These qualities, being easy to hear in a noisy environment, easy to understand when heard (also known as legibility), make them prime examples of the specific demands that user feedback sound design requires.

Sound can provide much more complex and subtle feedback as well. Adding a low tone to the mix when entering a room can induce an almost subliminal sense of unease in the player; a sound can inform us of the material that something is made of even though it might not be clear visually. There are many variations of the Chime Vs. Buzzer Principle in gaming. Contact sounds, the sound the game makes if you hit a target, for instance, are one such great example, but there are far too many for us to list here. As you can see, there are many ways to use the Chime Vs. Buzzer Principle in your games, and coming up with creative ways to take advantage of our innate understanding of this principle provides the game developer with endless opportunities for great sound design.

Additionally, the mix itself is an effective way to provide information to the player. By altering the mix – for instance the balance between music, dialog and FX – or even by changing the relative balance between sound effects, the game can attract the attention of the player and focus it on a specific element or, in turn, distract the attention of the player.

2. Entertain

The focus of this book being on sound design and not composition, we will think of music in relation to the sound design and overall narrative and emotional functions it supports.

a. Sound Design

We all know how much less scary or intense even the most action-packed shots look when watched with the sound off. If you haven't tried, do so. Find any scary scene from a game or movie, and watch it with the sound all the way down. Sound allows the story-teller to craft and compliment a compelling environment that magnifies the emotional impact of the scene or game, increasing the amount of active participation of the gamer. An effective combination of music and sound design, where both work together, plays a critical role in the overall success of the project, film or game.

Sound design for film and games remains still today, to an extent, a bit of a nebulous black art – or is often perceived as such – and one that can truly be learned only through a long and arduous apprenticeship. It is true that there is no substitute for experience and taste, both acquired through practice, but the vast amount of resources available to the student today makes it a much more accessible craft to acquire. This book will certainly attempt to demystify the art of sound design and unveil to students some of the most important techniques used by top notch sound designers, but experimentation by the student is paramount.

As previously discussed, sound supports every aspect of a video game – or should anyway. If we think of sound as simply 'added on' to complete the world presented by the visuals, we could assume that the role of sound design is simply to resolve the cognitive dissonance that would arise when the visuals are not complemented by sound.

Of course, sound does also serve the basic function of completing the visuals and therefore, especially within VR environment, allows for immersion to begin to take hold, but it also supports every other aspect of a game, from narrative to texturing, animation to game mechanics. A seasoned sound designer will look for or create a sound that will not simply complete the visual elements but also serve these other functions in the most meaningful and appropriate manner.

b. Music and the Mix

While this book does not focus on music composition and production, it would be a mistake to consider sound design and music in isolation from each other. The soundtrack of any game (or movie) should be considered as a whole, made up of music, dialog, sound effects and sometimes narration. At any given time, one of these elements should be the predominant one in the mix, based on how the story unfolds. A dynamic mix is a great way to keep the player's attention and create a truly entertaining experience. Certain scenes, such as action scenes, tend to be dominated by music, whose role is to heighten the visuals and underline the emotional aspect of the scene. A good composer's work will therefore add to the overall excitement and success of the moment. Other scenes might be dominated by sound effects, focusing our attention on an object or an environment. Often, it is the dialog that dominates, since it conveys most of the story and narrative. An experienced mixer and director can change the focus of the mix several times in a scene to carefully craft a compelling experience. Please see the companion website for some examples of films and games that will illustrate these points further.

Music for games can easily command a book in itself, and there are many out there. Music in media is used to frame the emotional perspective of a given scene or level. It tells us how to feel and whom to feel for in the unfolding story. I was lucky enough to study with Morton Subotnick, the great composer and pioneer of electronic music. During one of his lectures, he played the opening scene to the movie *The Shining* by Stanley Kubrick. However, he kept changing the music playing with the scene. This was his way to illustrate some of the obvious or subtle ways in which music can influence our emotional perception of the scene. During that exercise it became obvious to us that music could not only influence the perceived narrative by being sad or upbeat or by changing styles from rock to classical but that, if we are not careful, music also has the power to obliterate the narrative altogether. Additionally, music has the power to direct our attention to one element or character in the frame. Inevitably, a solo instrument links us emotionally to one of the characters, while an orchestral approach tends to take the focus away from individuals and shifts it toward the overall narrative.

Although we were all trained musicians and graduate students, Subotnick was able to show us that music was even more powerful than we had thought previously.

The combination of music and sound can not only be an extremely powerful one, but it can play a crucial role in providing the gamer with useful feedback in a way that neither of these media can accomplish on their own, and therefore communication between the composer and sound design team is crucial to achieve the best results and create a result greater than the sum of its parts.

3. Defining Immersion

Entire books have been dedicated to the topic of immersion – or presence – as psychologists have referred to it for several decades. Our goal here is not an exhaustive study of the phenomenon but rather to gain an understanding of it in the context of game audio and virtual reality.

We can classify virtual reality and augmented reality systems into three categories:

- Non-immersive systems: typically, simple Augmented Reality systems that affect one sensory input. Playing a 3D game on a laptop is a common example. This is the type of system most people are familiar with.
- Semi-immersive systems: typically allows the users to experience a 3D world while remaining connected to the real world. A flight simulator game played on a multiscreen system with realistic hardware, such as a flight yoke, would be a good example of such a system.
- Fully immersive systems: affect all or most sensory inputs and attempt to completely cut off the user from their surroundings through the use of head mounted displays, headphones, and additional systems such as gaming treadmills, which allow the user to walk or even run though a virtual environment.

An early definition of presence based on the work of Minski, 1980 would be:

The sense an individual experiences of being physically located in an environment different from their actual environment, while also not realizing the role technology is playing in making this happen

We in the gaming world tend to think of presence or immersion as a rather novel topic, one that came about with games and virtual reality. Truly, however, the concept has been part of conventional medias such as literature for hundreds of years. Narrative immersion happens when a player or reader is so invested in the plot that they momentarily forget about their surroundings.

There is no doubt, however, that games and virtual reality have given us a new perceived dimension in the immersive experience, that is, the possibility to act in an environment, not simply having the sensation of being there. So, what are the elements that scientists have been able to identify as most likely to create immersion?

The research of psychologist Werner Wirth suggests that successful immersion requires three steps:

1. Players begin to create a representation in their minds of the space or world the game is offering.
2. Players begin to think of the media space or game world as their main reference (aka primary ego reference).
3. Players are able to obtain useful information from the environment.

Characteristics that create immersion tend to fall in two categories:

1. Characteristics that create a rich mental model of the game environment.
2. Characteristics that create consistency amongst the various elements of the environment.

Clearly, sound can play a significant role in all these areas. We can establish a rich mental model of an environment through sound by not only 'scoring' the visuals with sound but by also by adding non-diegetic elements to our soundtrack. For instance, a pastoral outdoor scene can be made more immersive by adding the sounds of birds in various appropriate locations, preferably randomized around the player, such as trees, bushes etc. Some elements can be a lot more subtle, such as the sound of wood creaking, layered every once in a while, with footsteps over a wooden surface, for instance. While the player may not be consciously cognizant of such an event, there is no doubt that these details will greatly enhance the mental model of the environment and therefore contribute to creating immersion.

Consistency, this seemingly obvious concept, can be trickier to implement when it comes to creature sounds or interactive objects such as vehicles. The sound an enemy makes while it is being hurt in battle should be different than the sound that same creature might make when trying to intimidate its enemies, but it should still be consistent overall with the expectations of the player based on the visuals and, in this case, the anatomy of the creature and the animation or action. Consistency is also important when it comes to sound propagation in the virtual environment, and, as was seen earlier in this chapter, gaming extends the role of the sound designer to modeling sound propagation and the audio environment in which the sounds will live.

Inconsistencies in sound propagation will only contribute to confusing the player and cause them to eventually discard any audio cue and rely entirely on visual cues.

Indeed, when the human brain receives conflicting information between audio and visual channels, the brain will inevitably default to the visual channel. This is a phenomenon known as the Colavita visual dominance effect.

As sound designers, it is therefore crucial that we be consistent in our work. This is not only because we can as easily contribute and even enhance immersion as we can destroy it, but beyond immersion, if our work is confusing to the player, we take the risk of having the user discard audio cues altogether.

It is clear that sensory rich environments are much better at achieving immersion. The richness of a given environment maybe given as:

- Multiple channels of sensory information.
- Exhaustiveness of sensory information.
- Cognitively challenging environments.
- Possessing a strong narrative element.

Additionally, while immersion can be a rather tricky thing to achieve, it is rather easy to break. In order to maintain immersion, research suggests that these elements are crucial:

- Lack of incongruous audio/visual cues.
- Consistent behavior from objects in the game world.
- Continuous presentation of the game world – avoid commercials, level reset after a loss.
- The ability to interact with objects in the game world.

While some of these points may be relatively obvious, such as the lack of presence of incongruous elements (such as in-game ads, bugs in the game, the wrong sound being triggered), some may be less so. The third point presented in this list, 'continuous presentation of the game world', is well illustrated by the game *Inside* by Playdead studios. *Inside* is the follow-up to the acclaimed game *Limbo,* and *Inside*'s developers took a very unique approach to the music mechanics in the game. The Playdead team was trying to prevent the music from restarting every time the player respawned after being killed in the game. Something as seemingly unimportant as this turns out to have a major effect on the player. By not having the music restarted with every spawn, the action in the game feels a lot smoother, and the developers have removed yet one more element that may be guilty of reminding the player they are in a game, therefore making the experience more immersive. Indeed, the game is extremely successful at creating a sense of immersion.

It is important to note than the willingness to be emotionally involved is also an important, perhaps crucial, factor to achieving immersion. This is something that developers have no control over and that pre-supposes the desire of the user to be immersed. This is sometimes referred to as the 'Fan Gene'. As a result, two users may have wildly differing experiences when it comes to the same experience, based, partially, on their willingness to 'be immersed'.

2. Challenges of Game Audio

In spite of the improvements that each new generation of hardware brings with every anticipated release, developers are forced to come to one ineluctable conclusion: no matter how new, exciting, revolutionary, even, each new generation of tools is, we are almost always at some point contending with finite resources. It could be said that developers working on mobile gaming today are facing similar challenges as their peers did when developing games on the first generation of gaming consoles. In that regard, the range of technologies available to us today requires the modern developer to deal with a massive range of hardware and capabilities, demanding a level of expertise that is constantly evolving and increasing.

1. Implementation

It is impossible to understate the importance and impact of implementation on the final outcome, although what implementation actually consists of, the process and its purpose often remain a somewhat nebulous affair. In simplistic terms, implementation consists of making sure that the proper sounds are played at the right time, at the right sound level and distance and that they are processed in the way the sound designer intended. Implementation can make or break a soundtrack and, if poorly realized, can ruin the efforts of even the best sound designers. On the other hand, clever use of resources and smart coding can work their magic and enhance the efforts of the sound designers and contribute to creating a greater sense of immersion.

Implementation can be a somewhat technical process, and although some tools are available that can partially take out the need for scripting, some programming knowledge is definitely a plus in any circumstance and required in most. One of the most successful third-party implementation tools is Audiokinetic's Wwise, out of Montreal, Canada, which integrates seamlessly with most of the leading game engines such as Unity, Unreal and Lumberyard. The Unreal engine has a number of tools useful for audio implementation. The visual scripting language Blue Print developed by Epic is a very powerful tool for all-purpose implementation with very powerful audio features. As a sound designer or audio developer, learning early on what the technical limitations of a game, system or environment are is a crucial part of the process.

Because the focus of this book is to work with Unity and with as little reliance on other software as possible, we will look at these concepts and implementation using C# only, although they should be easy to translate into other environments.

2. Repetition and Fatigue Avoidance

We have already seen in Chapter one that the first generations of gaming hardware did not rely on stored PCM data for audio playback as is mostly the case today but instead used on-board audio chips to synthesize sounds in real time. Their concerns when it came to sound therefore had more to do with number of available voices than trying to squeeze as many samples as possible on a disc or download. Remember that the Atari 2600 had a polyphony of two voices.

The 1980s saw the rise and then dominance of PCM audio as the main building blocks of game soundtracks. Audio samples afforded a level of realism that was unheard of until then, even at the low resolutions early hardware could (barely) handle. Along with increased realism, however, came another host of issues, some of which we are still confronted with today.

Early video game systems had very limited available RAM, as a result of which games could ship only with a small amount of samples. Often these samples were heavily compressed (both in terms of dynamic range and data reduction), which severely reduced the fidelity of the recording or sound, making them hard to listen to, especially over time. In addition, since so few samples could be included, they were played frequently and had to be used for more than one purpose. In order to deal with listener fatigue, game developers early on developed techniques that are still relevant and in use today, the most common being randomization.

The use of random and semi-random techniques in sound and music, also known as Stochastic techniques, had been pioneered by avant-garde composers such as John Cage and Iannis Xenakis in the 1950s and 1960s. These techniques, directly or indirectly, have proved to be extremely helpful for game developers.

The use of random behaviors is a widespread practice in the gaming industry, which can be applied to many aspects of sound.

Randomization can be applied to but is not limited to:

1. Pitch
2. Amplitude
3. Sample Selection
4. Sample concatenation – the playback of samples sequentially
5. Interval between sample playback
6. Location of sound source
7. Synthesis parameters of procedurally generated assets

(Working examples of each of the techniques listed in the following and more are provided in the scripting portion of the book.)

The most common of these techniques is the randomization of pitch and amplitude, often built into game engines, such as Unreal, in which it's been implemented as a built-in feature for iterations. Pitch and amplitude randomization might be a good start, but it's often no longer enough to combat listener fatigue. Nowadays developers rely on more sophisticated techniques, often combining the randomization of several parameters. These more advanced, combinatorial techniques are sometimes referred to as procedural, a term in this case used rather loosely. In this book, we will tend to favor the stricter definition of the term procedural, that is, real time creation of audio assets, as opposed the real time manipulation of existing audio assets. The difference between procedural asset creation and advanced stochastic techniques are sometimes blurry, however. These more advanced random or stochastic techniques are certainly very important, and their usefulness should not be underestimated.

3. Interactive Elements and Prototyping

One of the challenges that even very accomplished sound designers coming from linear media tend to struggle with the most initially when working in gaming is the interactive elements, such as vehicles, machines, weapons and other devices the user may interact with. Interactivity makes it difficult to predict the behavior of a game object and therefore cannot be approached in a traditional linear fashion. How can one design sounds for a vehicle not knowing in advance how the user will interact with it? Simple things such as acceleration, braking sounds and the sound of tires skidding when the vehicle moves at high speed are all of a sudden part of a new equation.

The answer when addressing these issues is often prototyping. Prototyping consists of building an interactive audio model of the object, often in a visual environment such as Cycling74's MAXMSP, Native Instrument's Reaktor or Pure Data by Miller Puckette, to recreate the intended behavior of the object and test in advance all possible scenarios to make sure that our sound design is on point and, just as importantly, that the sounds behave appropriately. For instance, in order to recreate the sense of a vehicle accelerating, the engine loop currently playing back might get pitched up; inversely, when the user is slamming on the breaks the sample will get pitched down, and eventually, in more complex simulation, another sample at lower RPM might get triggered if the speed drops below a certain point and vice versa.

Working with interactive elements does imply that sounds must be 'animated' by being pitched up, down, looped and processed in accordance with the circumstances. This adds another layer of complexity to the work of the sound designer: they are not only responsible for the sound design but also for the proper processing and triggering of these sounds. The role of the sound designer therefore extends to determining the range of the proper parameters for these actions, as well as the circumstances or threshold for which certain sounds must be triggered. The sound of tires skidding would certainly sound awkward if triggered at very low speeds, for instance. Often, these more technical aspects are finely tuned in the final stages of the game, ideally with the programming or implementation team, to make sure their implementation is faithful to your prototype. In some cases, you might be expected to be fluent both as a sound designer and audio programmer, which is why having some scripting knowledge is a major advantage. Even in situations where you are not directly involved in the implementation, being able to interact with a programmer in a way they can clearly comprehend, with some knowledge of programming, is in itself a very valuable skill.

4. Physics

The introduction and development of increasingly more complex physics engines in games introduced a level of realism and immersion that was a small revolution for gamers. The ability to interact and have game objects behave like 'real-world' objects was a thrilling prospect. *Trespasser: Jurassic Park*, released in 1998 by Electronic Arts, is widely acknowledged as the first game to introduce ragdoll physics, crossing another threshold toward full immersion. The case could be made that subsequent games such as *Half Life 2*, published in 2004 by Valve Corporation, by introducing the gravity gun and allowing players to pick up and move objects in the game, truly heralded the era of realistic physics in video games.

Of course, physics engines introduced a new set of challenges for sound designers and audio programmers. Objects could now behave in ways that were totally unpredictable. A simple barrel with physics turned on could now be tipped over, dragged, bounce or roll at ranges of velocities, each requiring their own sound, against any number of potential materials, such as concrete, metal, wood etc.

The introduction of physics in game engines perhaps demonstrated the limitations of the sample-based paradigm in video game soundtracks. It would be impossible to create, select and store enough samples to perfectly cover each possible situation in the barrel example. Some recent work we shall discuss in the procedural audio chapter shows some real promise for real-time generation of audio assets. Using physical modeling techniques we can model the behavior of the barrel and generate the appropriate sound, in real time, based on parameters passed onto us by the game engine.

For the time being, however, that is, until more of these technologies are implemented in production environments and game engines, we rely on a combination of parameter randomization and sample selection based on data gathered from the game engine at the time of the event. Such data often include the velocity of the collision and the material against which the collision occurred. This permits satisfactory, even realistic simulation of most scenarios with a limited number of samples.

5. Environmental Sound Design and Modeling

In creating the soundtrack for a large 3D game or environment, one should consider the resulting output as a cohesive whole instead of a collection of sounds playing somewhat randomly on top of each other. This kind of foresight and holistic approach to sound design allows for much more engaging and believable environments and a much easier mix overall. The soundtrack of a game is a complex environment, composed of many layers playing on top of each and changing based on complex parameters determined by the gameplay. In a classic first-person shooter game, the following groups or layers of sounds could be playing at any single time over each other:

- Room tones: drones, hums.
- Environmental sounds: street sounds, weather.
- Dialog and chatter.
- Foley: footsteps, movement sounds.
- Non-player characters: AI, creatures, enemies.
- Weapons: small arms fire, explosions.
- Machinery: vehicles, complex interactive elements.
- Music.

This list does gives us a sense of the challenge that organizing, designing, prioritizing and playing back all these sounds together and keeping the mix from getting cluttered represents.

In essence, we are creating a soundscape. We shall define soundscape as a sound collage that is intended to recreate a place and an environment and provide the player with an overall sonic context.

In addition to having the task of creating a cohesive, complex and responsive sonic environment, it is just as important that the environment itself, within which these sounds are going to be heard, be just as believable. This discipline is known as environmental modeling and relies on tools such as reverberation and filtering to model sound propagation. Environmental modeling is a discipline pioneered by sound designers and film editors such as Walter Murch that aims at recreating the sonic properties of an acoustical space – be it indoors or outdoors – and provides our sounds a believable space to live in. The human ear is keenly very sensitive to the reverberant properties of most spaces, even more so to the lack of reverberation. Often the addition of a subtle reverberation to simulate the acoustic properties of a place will go a long way in creating a satisfying experience but in itself may not be enough. Environmental modeling is discussed in further detail in this book.

6. Mixing

The mix often remains the Achilles' heels of many games. Mixing for linear media is a complex and difficult process usually acquired with experience. Mixing for games and interactive media does introduce the added complexity of unpredictability, as it isn't always possible to anticipate what to expect sonically in an interactive environment where events may unfold in many potential ways. We must teach the engine to deal with all potential situations using a carefully thought-out routing and rules of architecture for the game to follow. In most situations the game has no or little awareness of its own audio output.

Our challenge is, as it is so often in game audio, twofold: ensure a clean, crisp and dynamic mix while making sure that, no matter what, critical audio such as dialog is heard clearly under any circumstances and is given priority. Discussing the various components of a good mix is beyond the scope of this chapter and shall be addressed in detail in Chapter twelve.

7. Asset Management and Organization

A modern game or VR simulation requires a massive number of audio assets. These can easily range in the thousands, possibly tens of thousands for a AAA game. Managing these quickly becomes a challenge in itself. Game engines, even third-party software such as Wwise, should be thought of as integration and creative tools rather than asset creation tools. The line between the two is not always an absolute one, but as a rule you should only import in the game engine and work with polished assets ready to be plugged in as quickly and painlessly as possible. While you can fix some issues during the implementation process, such as amplitude or pitch adjustments, you should avoid consistently relying on adjusting assets in the game engine for matters that could have been taken care of sooner. This tends to cost time and create unnecessarily complex projects. It is much more time-efficient to make sure all assets are exported and processed correctly prior to importing them.

An asset delivery checklist, usually in the form of a spreadsheet, is a must. It should contain information about the following, but this list is not exhaustive:

- Version control: you will often be dealing with multiple versions of a sound, level, game build etc. due to fixes or changes. Making sure you are working with and delivering the latest or correct file is obviously imperative.
- Deadlines: often the work of the sound design team is split up into multiple deadlines for various assets types in order to layer and optimize the audio integration and implementation process. Keeping track of and managing multiple deadlines is a highly prized and useful organizational skill.
- Consistency and format: making sure that all the files you will be delivering are at the proper format, sample rate, number of channels and at consistent sound levels across variations, especially for sounds that are related (such as footsteps for instance), quickly becomes challenging and an area where it is easy to make mistakes.
- Naming convention: dealing with a massive number of assets requires a naming convention that can easily be followed and understood by all

the team members. The naming convention should be both descriptive and as short as possible:

Hero_Fstps_Walk_Wood_01.wav
Hero_Fstps_Walk_Metal_02.wav
Hero_Fstps_Run_Stone_09.wav

Deciding on a naming convention is something that should be carefully considered in the preproduction stages of the game, as it will be very inconvenient to change it halfway through and could cause disruptions in the production process. Keep in mind that audio files are usually linked to the engine by name.

Conclusion

The functions performed by the soundtrack of a video game are complex and wide ranging, from entertaining to providing user feedback. The goal of an audio developer and creator is to create a rich immersive environment while dealing with the challenges common to all audio media – such as sound design, mixing and supporting the narrative – but with the added complexities brought on by interactive media and specific demands of gaming. Identifying those challenges, establishing clear design goals and familiarity with the technology you are working with are all important aspects of being successful in your execution. Our work as sound designers is often used to support almost every aspect of the gameplay, and therefore the need for audio is felt throughout most stages of the game creation process.

The Basics of Nonlinear Sound Design

In this chapter we provide an overview of sound design for game audio as a process. This is a great starting place for those new to the industry or intermediate sound designers looking for a refresher. Our goal is to offer a broad perspective on concepts such as nonlinearity, tools and skills for the game sound designer, as well as foundational components of game sound. We will explore these topics in depth in later chapters.

Many of the concepts discussed in this chapter are entry level and can be understood by novice sound designers. However, we will quickly move on to more intermediate topics in later chapters. If you at any point have trouble with the material in later chapters, we suggest you return to this chapter to refamiliarize yourself with the basics. You can also use the glossary and index as references.

WHAT IS NONLINEAR SOUND DESIGN?

To answer the question "What is nonlinear sound design?" we must first look into how games and other nonlinear media differ from linear productions like film and TV. In the "Game Audio" section of Chapter 1 (page 12) we briefly mentioned some differences between the two processes and here we will discuss further the challenges that come with nonlinear media. In linear media sound and visuals are synced and printed, which means there is no chance for change during viewing. The nonlinearity of games puts the player in control of the gameplay, which means audio must adapt to changes in run-time through pre-planned events.

For some, game audio is an afterthought, but the role of sound in games is much more than ornamental; it sets the mood, adapts to changes in the **game state** (a description of the current contexts and systems in the game, similar to a **snapshot**), provides feedback on player input, and focuses the player's awareness. You can probably recall a moment in your favorite game or film where sound was used to grab your attention – "Hey, look over here!" Sound gets our attention in our everyday lives, and it does so in games as well. In short, nonlinear sound helps shape the direction of gameplay in a very tangible way. Viewers may notice new details about the sound in a film or television show, but the sequence and character of game sound fundamentally changes *every time the game is played*. This begs the question: "*How do we design sound for games?*"

Let's imagine ourselves as the audience of a film. In this scene the main character rushes into traffic on a busy street. The car horns are blaring, drivers are screaming, and general street noise is present underneath everything. In this scenario what is drawing our *focus*? What is the role of the audio when supporting that focus? When we watch films our focus is usually decided for us. The camera makes the choice for us (for the most part) by focusing on important characters, objects, or events. Our focus is on the character in this sequence, and the role of the audio is to then set the scene and fill out details about the environment.

Now let's imagine a new scene, this time we are the developers of a video game. In this scene a huge battle has just occurred and the last few enemies have been defeated. Succeeding in this battle opens up a new area of the map. How do we change the player's *focus* so that attention is drawn to this new area? The challenge is that in a video game the player can choose where to look and what to pay attention to. We don't have the luxury of always controlling specific camera angles – we need a method of drawing the focus of a player to their next task or objective. The answer of course is sound! If we want our players to *choose* to explore a new area without controlling the camera we can plant a sound effect in that location. This is how sound should function in a nonlinear environment. Audio cues should give players enough information to *inform* their choices, but not *choose for them*.

The freedom players have to make their own choices is what sets games apart from linear media. The player's actions affect the game state and in turn the game provides information and feedback to the player. Game sound functions in the same way. Sounds trigger in reaction to the player, and those sounds then influence the player's actions. These sound triggers (user interface [UI], music, dialogue, or sound effects) are critical in guiding the player through the game world.

As sound designers this interactivity between the player and the game creates a fundamental challenge. Imagine a linear scene where a character is walking through the forest. She pauses briefly, listening to some rustling in the brush ahead of her. Then she continues on. As sound designers for this scene we already know exactly what happens and when. We simply have to create a list of necessary **assets**, design or source the sounds, and place them on a timeline in perfect synchronization with the visuals. If this scene were in a game we would have much more to think about as sound designers. First, we would need to create a list of necessary assets and create them, just as we did before. However we can't just place them on a timeline. We need to *implement* them into the game engine so that each footstep triggers in sync with the animations. On top of that, we have to

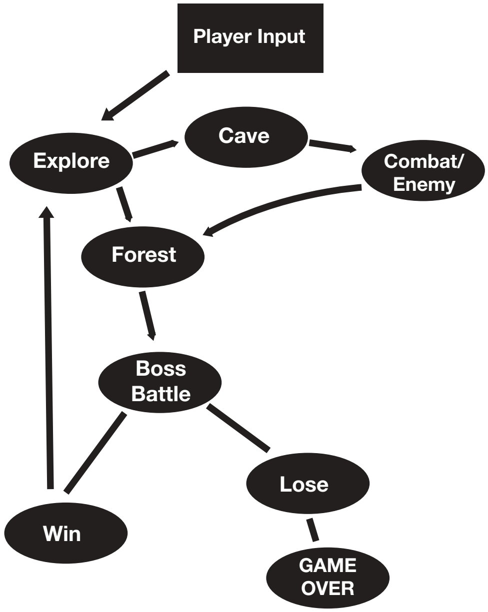


FIGURE 2.1 Player input changing game states.

make sure to include appropriate **aural cues** so that the player can take appropriate actions toward her objectives. In many cases the two scenes would sound *very* different side by side. In linear media, focus should be placed on the sounds that create the mood of the scene without getting in the way of the action. In nonlinear media the sounds need to react to and influence the action.

There are many strategies and tools that we have as sound designers to aid us in creating this kind of nonlinear audio. With regards to the example above we would likely use ambient loops and **random playlists** of sounds to ensure that the audio is detailed and does not cut out to silence. Because player actions are unpredictable we would also have to set this up as an **event** (system of adaptive or interactive sound) rather than on a timeline. This is so that the **game engine** (the framework within which the game is built) can "understand" how the audio is triggered and synchronized to the visuals in real-time (see Figure 2.1).

THE SOUND DESIGN PROCESS

Now that we have a better understanding of how nonlinear sound differs from linear sound let's explore sound design as a process. From a creative perspective, the job of a sound designer is to tell a story through sound that immerses the player in the game environment. The goal is to produce unique sounds that complement or enhance the gameplay and visuals, set the tone, and inform the player of potential actions.

There are three main phases in the sound design process. The first two are **sourcing** (recording, generating, and licensing), and **designing** (**layering** and manipulation). These two phases have been adopted from the tradition of film sound. However, the **implementation** phase is derived from film sound mixing but entails challenges and methods unique to games. As mentioned earlier, this involves organizing sounds into adaptable events which are triggered during **runtime** as the game state changes or based on the player's input. Below we provide a quick explanation of these three phases but in Chapter 3 we discuss the specific details.

Sourcing

Sourcing refers to the process of retrieving the necessary components of a sound effect for use in a game. There are many ways to source sound effects, making this an interesting task. Sounds can be sourced through licensing, designing synthetic sounds from scratch (synthesis), or through custom recordings.

When sound designers license sounds it means that they have paid for the **media license** to a single audio file or a library of audio files. In many cases the audio is in the form of **source recordings** to be used as basic elements of a sound to be combined and designed later on. Other times the audio is more of a complete sound effect that will be shaped or trimmed to fit the needs of the game. In either case licensing sounds is usually a **non-exclusive agreement**, which means that other sound designers can use these same sounds in their projects as well. For this reason licensing sounds is a very quick and easy way to get started, but it can also lead to somewhat generic sounds if you aren't careful.

Let's imagine we have been assigned a project and our first task is to create player character footsteps for a third-person adventure game. It might seem tempting to purchase a sound-effect library and deliver the sounds "as is" to the programmer for implementation. This is a quick and easy way to handle the task, right? The problem with this process is the client is likely assuming they are working with a sound *designer* and not a sound *plucker*. Anyone can choose a sound from a library and place it in game, but games need to be immersive, and sound is a large factor in that immersion. It is the sound designer's responsibility to shape layers of sound into a unique asset that perfectly suits the action in game *and* supports the gameplay. This will not only result in a better fitting sound, but there will also be a slimmer chance that the licensed sound will be recognized in other games.

With library sounds you also have to consider the quality of the source. There are many options for library sounds ranging from no cost to thousands of dollars. No-cost or low-cost libraries run the risk of lesser quality assets, which would need clean-up and restoration before becoming a good candidate for in-game use. It is equally important to consider the type of license, and your rights to use the assets in a commercial project. For this reason we recommend reading the license agreement fully to make sure your project is legally viable for use with the library sound *before* you purchase it.

This isn't to say that you should always avoid licensing sounds. Library sounds are used very commonly in a range of projects from indie to AAA. It's perfectly fine to use a properly licensed library sound effect when time and project budgets don't offer the opportunity to record a new source. It is also beneficial to license sounds when the intention is to edit or process the sound heavily, and layer it with other sounds so that it fits the game more adequately. This doesn't mean you won't run into a situation where a single sound can be pulled from a library, cleaned up and used singularly in game.

Generating sound through synthesis is another way to source sounds. These kinds of **synthesized sounds** can be used on their own or combined with **mechanical sounds**. Plenty of games have retro themes where synthesized sounds are the perfect complement to the art. If you are tasked with creating the sound for a spaceship starting up and taking off, mechanical sounds like power tools or a hair dryer are great sources. But adding synthesized elements can help add movement to the engine and bring it to life. Cyclic sounds such as sub base tones can be generated from scratch, and they will make the engine sound weighty and powerful while the hair dryer or other tonal sources sit up top in a higher **frequency range**. Of course, synthesized sounds (even by themselves) always sound great with retro-styled artwork.

The important thing to remember about synthesized sounds is that they are by definition synthetic. This means they can be tricky to mix in with more organic or natural sound sources. To achieve the best results, be intentional about the layers of synth elements that you add. Adding a synthetic bass layer is often more than enough to add some interesting element to a sound. Don't overdo it, and always pay careful attention to how the event sounds as a whole so that you can properly balance the synthesized elements with the organic ones. We will discuss sound design using synthesis later on in Chapter 3, under the section titled "Sourcing Sound through Synthesis" (page 58).

The process of custom recording is a great way to source unique material to fit your game. It will give you full control over the material that you record and use in the design process, and it offers the greatest flexibility over the recordings themselves. If you are contracted to create sounds for a racing game, you may decide to record custom sounds if the developer is specific about the authenticity of vehicle sounds and the budget allows. By choosing to custom record you could get the *exact* sounds of each model of car present in the game. Not only that, but you could record any aspect of the car that you wanted. You could record **pass-bys** on a racing track, engine revving, mechanical noises, or even put the cars up on blocks and record the engines in isolation. This would come in particularly handy if the game offered players multiple perspectives during a race because there is a vast sonic difference between driving a car in third person and first person. You also have control over the number and types of microphones you use and just about every detail of the recording process is under your control.

Recording custom sounds is in many ways an ideal way to source audio for your game, but it does have its disadvantages as well. Recording custom sounds can be expensive and time consuming. If footsteps in snow are needed, there is a list of items that need to be taken into consideration before a custom recording is even possible; the most obvious item on the list being "Is there snow?" If you are in a location without snow on the ground you would have to consider the costs of traveling to a location with snow. You can of course simulate the sound of footsteps in snow by placing a lot of salt in a bucket and stepping in, or placing a piece of carpet over gravel and walking on top, but it may not yield the sound you are aiming for.

Should you decide to record the sound yourself you also have to consider if you have the proper equipment. In future chapters we will get more specific on microphone techniques, but for now let's just say you will need to think about the microphone, microphone placement, polar pattern, handling noise, room or environment control, clothing noise, breathing, recorder settings, proper walking (gait and heel-to-toe movement), proximity to the microphone, and more.

Regardless of how you source your sounds, you will have to consider how they fit the visuals. The best choice of course is the one that fits appropriately with the game. For example, in a game scene where the snowy terrain is hard and compact it won't make sense to use source sounds of deep footfalls in soft snow. The footfall must sound harder to be true to the animation. Sourcing is the foundation of your game sound, so make sure to search for fitting material before you move on to the design phase.

Sound Design as a Process

The design process is where you put your mark on the assets by editing the source, adding additional textures, layering, and processing. This is the stage where designers add detail and depth to the sound effect as a whole, and continue the process of molding it to fit the needs of the game.

The design process must start with preparing the sound for use in game. This is often referred to as cleaning up or editing the audio. Any source audio that is custom recorded, or comes from a lower quality sound library will likely need to be cleaned up. Try to think of how a game would look if the artist didn't polish up the shading or left the coloring inconsistent. When all the artwork came together, the game would be a mess! Assets would be unclear and ill defined. Colors would blend into each other and players would have a tough time walking around let alone traversing levels. The same effect happens with sound when it's not properly polished or mixed before placing it in game. If the character footsteps were recorded outdoors, and they are left unedited players might hear bird chirps or car engines triggering with each step. These are unwanted artefacts present in the source sound, and they *must* be edited out *before* they are used in the asset. There are a lot of great restoration tools to help aid in this clean-up process. Using de-noise, de-click, or spectral editing tools such iZotope's RX can speed up the workflow and help deliver quality sounds. As we dive further into editing in "Effects Processing as a Sound Design Tool," Chapter 3, page 96, we will further explore **plugin effects** and their use as a sound design tool.

Quality sound effects almost always involve layering of some kind. After you clean up your source, you can begin layering sounds to build an asset that contains all of the sonic qualities necessary to fit the visuals in game. A good example of this type of layering is weapon-fire sound effects. An effective gun-fire sound will typically be built from at least two or three layers. Typically the base layer will be the main firing sound. This layer will be the foundation of the asset, and characterize the sound as a whole. Designers will commonly add a secondary layer, usually in the lower or low-mid frequency range which adds some **punch** to the sound so that players really feel the impact of the weapon. Finally, a mechanical layer can add detail to the sound, and convey information regarding the materials that the weapon is made of.

While layering is an important part of the process, you will probably find yourself needing to use some creative plugin effects to further manipulate the sound. A plugin is software that processes audio in some way. This aspect of design falls under audio editing techniques. After you have the appropriate layers in place, plugins can help make the sound cohesive and powerful. Plugins like EQ, echo, and **compression** can effectively "glue" your layers together, and make sure the asset as a whole is clear and polished. Other plugins like **transient shapers** can add dynamics and punch. Similarly pitch shifters and **modulation** plugins can add some interesting aspects to the sound, and really stretch the sonic boundaries further from the source material. There are many more plugins than these, and this is a crucial step in the design process. As mentioned above, we will dive further into plugin types and usage in later chapters.

Implementation

The implementation phase is the only phase that is entirely unique to games, and it is a highly technical process. Asset implementation plays a huge role in making sure the sound design adapts to (and informs) the gameplay. Effective audio for games is often considered 50 percent creative source and design, and 50 percent implementation.

By definition, implementation is the process of carrying out a design and putting it into action. For our purposes, the "design" is referring to the audio asset and the "action" is the game engine. When it's time to implement, our assets have already been created and now they need to be integrated into the game engine with a set of instructions to control their behavior. There are a variety of proprietary and non-proprietary game engines. Larger development teams may choose to create their own proprietary engine while smaller teams are likely to work with third-party engines such as Unity or Unreal. It is important to understand which engine the game is being built in before you begin the implementation phase. These engines each have their own process for integrating audio and scripted events.

Implementation requires a practical understanding of how sound is hooked up to the game engine. It also requires knowledge of the myriad options for integration. Having this knowledge will allow you to implement audio in a more realistic and immersive way. Let's say that in testing a game you find that only one footstep asset plays regardless of terrain type. In the days of *Super Mario Bros.* this might have been fine, but modern games have advanced technically and visually. As sound designers it is our duty to keep the audio experience up to par. An understanding of scripting and how sound can be implemented will provide opportunities for more effective integration solutions. In this case, delivering six to eight footsteps with each terrain type will offer more variety and help avoid listener's fatigue for your player. The player character walk cycle will sound noticeably more realistic. For additional variety you could write a quick script to randomize volume and pitch on the footstep sounds. It would also be beneficial to add cloth or armor movement sounds to the footsteps, or you could **bake** them into the assets themselves. We will further explore the pros and cons of this in Chapter 8 and discuss how audio middleware offers the sound designer more control through a graphical user interface.

The Sound Design Process Summary

Based on the information presented here we can conclude a few things:

- Game audio design and implementation (nonlinear audio) needs to be approached in a different manner than linear media.
- The recipe for a good sound designer starts with a passion for audio and games, mixed with a well-trained ear, a mastery of the tools, and a solid understanding of implementation and game engines.
- There are three phases of game audio as a sound designer: *sourcing, designing*, and *implementation*.
- Good *source* sounds need to be clean and fit the aesthetic of the game. Source material can be licensed, synthesized, or custom recorded.
- Regardless of how the sound is sourced the assets must be manipulated and processed to uniquely fit its position in the game during the *design phase.*
- *Implementation* requires a broad understanding of nonlinear audio, the specific gameplay mechanics of your project, and a practical understanding of the integration options.

CHALLENGES IN NONLINEAR SOUND DESIGN

Triggering and Game Sequence

Sound design for games as a process wasn't created by completely reinventing the wheel. Many techniques and practices have been adapted from film sound and traditional audio post-production. These post-production techniques gave game audio a bit of a headstart, but interactive experiences come with their own creative and technical challenges that are not present in linear media. Previously we learned that the player's actions in game can change the game state. We can break this down further by examining play styles and preferences. Some players speed through a game like they are in a lightning bonus round while others like to linger and explore, taking in the scenery. Some players will pick up gameplay mechanics quickly, while others might try and fail many times before succeeding. In many games players can even make choices which affect the outcome of the story itself. These are just a few examples of how play styles can create totally different outcomes in a game. As sound designers we have to account for these possibilities. The challenge is how to design sound as the sequence of events triggered in a game varies from player to player and session to session.

Keeping up with the player's unpredictable choices can be likened to working on an assembly line where graphics, sound, and programming do their best to keep up with the changing demands. An example of adaptive sound design is player movement from terrain to terrain in a game environment. The terrain is tagged with a material type which scripting and audio events can adapt to allowing for smooth transitions as the player connects with various floor types while moving about the world. Another example might be reverb zones positioned to change the sense of space as the player moves from room to room or area to area. We address these challenges in Chapter 8 where we discuss perspective, resource limitations, repetition, and the mix.

EXPLORING IMMERSION AND INTERACTIVITY

How do we define "**immersion**" in a game? Generally speaking, immersion means "achieving a state of deep engagement." It's safe to say that in games the goal is to engage the player into the game world so that she feels immersed, as if she is fully and completely part of the experience. To do this, let us first examine the elements of gameplay. Each of the following elements plays a role in immersing the player into the world of the game.

Game Elements

- Story/narrative
- Game mechanics

- Environmental interaction
- Graphics/art style
- Cutscene transitions
- Music and sound design

While sound might not be the first thing that comes to mind when you ask yourself what immerses you in a game, we are going to focus on the importance of sound and how it provides the player with a feeling of being surrounded by the world.

Just like a good film production, good game production will make the player forget about reality and draw them into the game world. This phenomenon called "suspension of disbelief"[1] isn't new or specific to games. It was coined by poet Samuel Taylor Coleridge in 1817. This ability to immerse with sound becomes even more important with virtual reality as we will later discuss in "Audio for New Realities: VR," Chapter 9.

Sound, just like visuals, can enhance the user experience by eliciting emotions, supporting the game's theme, and heightening intense moments. Pairing sound and visuals can enhance the experience and ensure the necessary information is received by the player. Visual cues provide a vast amount of information at any given time in a game. Sound helps the visuals relay information since the player's eyes cannot be in multiple places at once. A human ear can receive numerous sounds at one time from all around the environment. The brain works to dissect the load of aural information and break it down into several recognizable components. In an RPG (role-playing game) collectible card game there are various sounds to provide feedback to the player that don't require them to look away from where they are focused. An example is a timer ticking away to inform the player who might be looking over their deck that time is getting low.

Because sound is such a big part of our everyday life it is an important part of visual media. A truly immersive in-game soundscape provides the player with a feeling of being dropped into the middle of the virtual scene. Sound design done right will add to the experience but when sound is implemented poorly the users really notice. A solid understanding of how sounds interact in real-world spaces will help the designer to sonically recreate them. Unless an abled-hearing person is entombed in an **anechoic chamber** they aren't without sound, not even for a single moment.

Sound without visuals can be processed and spatially located even when not in the field of view. As humans we've learned that birds can be heard chirping during the day and crickets stridulating at night. With outdoor soundscapes audio designers can cycle the time of day with the sound of birds or crickets, which will provide the listener with the expected sonic information.

Psychology of Sound

Understanding the psychology of emotive sound design will help you to create immersive soundscapes. There are plenty of studies that show how happy and upbeat music can have a positive boost on mood while tense music can have the opposite effect on the listener. Luxury brands have been using sound to influence consumers for years. A luxury automobile manufacturer might work with a sound designer to create an engine design that gives the user a sense of power. In games, the audio designer should explore the psychology of sound to implement the perfect emotive influence on the player.

Little Nightmares, a puzzle-platformer horror adventure game developed by Tarsier Studios, has a soundscape that toys with the player's emotions. The story is that of a little girl cloaked in a yellow raincoat who is trapped in a mysterious vessel which is home to wildly creative and somewhat grotesque creatures. The ambient soundscape makes the player feel vulnerable and afraid. If this were a happy tale with fuzzy teddy-bear-like **NPCs** (non-player characters) a different approach to asset choice and implementation would be required.

Audio plays, podcasts, and radio shows are proof that great sound design doesn't always need visuals to evoke emotion and tell a story. A narrative isn't even necessary as we can often hear a singular sound and process it as positive, neutral, or negative sonic feedback. With audio and its implementation as a partner, the visuals and narrative can take on a whole new level of immersion.

Audio designers for *Rockstar's Red Dead Redemption* put massive amounts of thought and work into the detailed soundscape that drops the player into the Old West. It can be difficult to find the perfect balance to transition the listener from an indoor to outdoor setting, or to introduce new musical themes. As creatives we must tame the urge to cram as much of our work into the project as possible. There are plenty of scenarios of games that require minimal sonic elements. The sound of silence can play a big role in provoking emotion.

A soundscape near a campfire in the Old West will require sound to enhance the visuals we see in the scene but also requires sounds off in the distance, beyond what we can see, to fill out the soundscape. The audio designer should examine the scene and determine which sonic assets will make the world feel natural and alive.

Sounds that come from specific sources *within* the scene are called **diegetic**. Diegetic sounds, like **PC** (**Player Character**) movements are just one part of the soundscape however. The rest is ambience. As well-practiced listeners we expect to hear a variety of aural cues, even if they are not visible. These ambient sounds are **non-diegetic** (not visible in the scene), and they serve the function of filling out background noises. Without light prairie wind or crickets and coyotes in the background, the scene would will feel incomplete. Of course, with a plethora of sound emitting from the scene a well-balanced mix will be necessary to avoid any muddiness.

Fire up *Red Dead Redemption 2* and ride a horse for a few moments in a non-combat game state. The tiny details that include creaking of the saddle's leather and the clinking of the buckles heighten the experience. Imagine how the experience would differ if the personality of the horse was sonically characterized by random neighing instead of distinct hardware details shaping the character.

The total level of immersion in a game is a result of each of the elements, listed above, working together. Audio is sometimes overlooked as an immersive tool, but the reality is that audio is a core element of immersion. The reason for this is that audio intersects with each of the other elements in important ways. It may be "invisible," but without audio the other elements cannot stand up on their own when it comes to immersion.

Let's take a look at how audio connects and supports each of these game elements to create an immersive experience.

Story/Narrative

To create an immersive soundscape, the audio designer should focus on supporting the game design. Sound should provide the necessary information to the player to guide her through the game or support various **gameplay mechanics** and **narratives**.

Let's imagine we are back in a forest scene. We hear birds, wind, and maybe even some animals making sounds off in the distance. This sets the locale and helps us identify with the graphics of the forest. But what is the emotion or *mood* in this *particular* forest? If the birds are happily chirping away and the wind is light and wispy we might feel that there is no danger and think of this as a safe area. This location can be turned into a darker place without swapping the visuals. Changing the sounds so the birds chirping become owls hooting or hawks screeching, and modifying light wind to a darker howl would be evidence of potential danger. This demonstrates sound as a powerful element that can modify the player's perception of the environment, even with the same visuals in place.

Either way, the *character* of the audio causes us to absorb details about the environment, which *immerses* us into the setting. This is a very basic example, but this idea can be taken to levels of extreme detail in more complex games. The more audio information the player has to absorb, the higher the level of immersion is likely to be. By extension, if the level of immersion is high the player has a greater chance at being successful in gameplay tasks. The player will also have a higher likelihood of being impacted emotionally by the narrative of the game.

Game Mechanics

Game mechanics themselves are an important aspect of most games. The mechanics of a game determine *how we interact with it.* Audio creates immersion in this area by offering *expected* aural feedback to the player. By this we mean that in our daily lives when we perform an action, we expect certain sounds to occur. If they don't, it is a strange and attention-grabbing experience. This is also true in games. When we unlock a door in a game, we *expect* to hear some clicks and a door open. When we shoot a gun in a game, we *expect* to hear a click and a firing sound that reflects the material and quality of the weapon. The more detailed the gameplay mechanic, the more detailed the audio must become to reflect this.

This is not only true of realistic games. This is in some ways more important for abstract games. Because most of us have opened and closed countless doors in our lives, we already are familiar with what to expect. The mere suggestion of the sound of a door opening or closing will allow our brains to fill in the gaps, and evaluate the experience as natural. In other words it takes a *really bad* door sound to take us out of an otherwise immersive experience. By contrast, in a game that is completely abstract, the mechanics are completely new to us. Our brains don't exactly know what to expect, they just expect *something*. In these cases, it is even more important for sound designers to use sounds that simultaneously reinforce the abstract visuals of the game while grounding the player in some kind of familiar sound to associate with. An example might be an otherworldly portal opening to another dimension. Players have literally no idea what this should sound like because it doesn't exist, so the design needs to suggest this surreal quality of the portal. At the same time, whatever sound source is being used to design the portal sound must also *feel like a door opening*. By that we mean that the **gesture** of the sound should in some way emulate more realistic sounds, like the opening of a door or a gate. There are many ways to do this (we will explore some of them in later chapters), but to put it simply it usually involves volume and pitch **envelopes** as these are two characteristics of sound that our ears are very attuned to.

Environmental Interaction

The sounds making up the environment in a game can set the mood, offer a sense of locale, and suggest to the player specific emotions at points throughout the game. This is how audio impacts the story of the game, as mentioned above. Sound can also direct attention and actively draw players through an experience. We mentioned earlier that sounds are great ways to attract the attention of a player. This is one method of teaching players where their next objectives are, but sounds like this also drastically increase the level of immersion in a game. If you want to introduce the player to an area of the game or to provide a sense of placement of an object in the world, simply set a **positional sound emitter** on the object with proper attenuation settings (see Chapter 8 for more information on sound emitters).

Reverb reflections or **delay slaps** of triggered sounds can provide a sense of acoustics to let us know if we are outdoors, in a tunnel or a cave, or in any other location. Sonic details such as this spatial and acoustic information add another dimension of immersion in addition to task-related information. This is especially true for first-person or VR (virtual reality) games because the player's perspective is literally the **3D space** of the game scene. Our ears are sensitive to false spatial and acoustic information, so accurate spatial and acoustic information is key to a deeply immersive game environment.

It's also important to remember that the more interactive the environment is, the more immersive a game is. This goes hand in hand without audio. If something in the environment makes a sound, it tricks our brains into thinking there is more depth than there actually is to a game scene. Keep this in mind as you plan your audio design. The more detailed the environment, the more immersion you will likely achieve.

Graphics/Art Style

Sound can provide information about materials and objects as well as setting the mood or providing a sense of space. This is integrated very heavily into the graphics or art style of a game. The sound needs to fit in some way with the *mood* of the art style. Games that are highly realistic and three dimensional usually call for highly realistic and spatialized audio. Artistically rendered two-dimensional games usually have more creative freedom for designers. For example an enemy zombie with a detailed and grotesque visual style may be designed with gruesome levels of audio detail. This would work fine if the game is meant to be a horrifying experience. Another option is for the audio design to be ironic, which would add some fun and lightness to the zombies. The sounds would be less threatening or even funny. This would work well if the game is more of a casual experience. Either way, there must be some palatable logic and coherency to the audio design for the game to be immersive.

Audio also has to provide relevant detail on the objects and environments themselves so that they convey practical information to the player. A large robot walking across a metal catwalk will have movement and footstep sounds attached. These footsteps will communicate the weight of the bot and provide a sense of how powerful the character might be. This is highly relevant to the gameplay because it influences how the player might treat a combat situation. If the sound is menacing and heavy, the player will take that to mean the robot is a serious threat. Situations like this can be made to either supply the player with valuable information to succeed in an upcoming task, or can be exploited to add a sense of surprise and challenge to a scene.

ESSENTIAL SKILLS FOR SOUND DESIGNERS

At this point you may be wondering if you have what it takes to be a Sound Designer for games. The good news is if you have a passion for audio with an interest in recording, collecting, and breaking down sounds then you are on the right path to *learn* what it takes. That said, sound design for games can get pretty complicated. There are a variety of essential skills that apply specifically to sound design that should be on your radar to master if you are serious about working in the industry. We have listed a few such examples below. For more information on general game audio tools and skills refer to "Essential Soft Skills and Tools for Game Audio," Chapter 1, page 16.

Skills for the Game Sound Designer

A Well-Developed Ear

An indispensable attribute for a game sound designer is the ability to listen to sound critically and analytically. An understanding of how sound interacts with the real world is key to translating that into game environments. If you aren't listening and analyzing it can be harder to understand how sound should work in virtual worlds and more difficult to understand how to recreate sounds.

Having a good "ear" for sound doesn't mean having perfect hearing. The ability to listen to a sound and break it down, picking apart frequencies and being aware of noise are more useful abilities for our purposes. Training your ear takes practice, just like learning an instrument or building your composition skills. You can practice listening to how effects change a sound source by applying them yourself and analyzing the results. There are quite a few apps and websites that offer ear training for sound designers that help flex your frequency muscle memory.

SoundGym[2] offers gamified learning and practice with frequency, space and time, quality, and dynamics. It can be really handy when you want to manipulate sound with effects if you have a good understanding for how the effects alter the sound. For example, SoundGym's Filter Expert game plays audio with various EQ settings applied. You can bypass the effect to listen to the original as you decide which frequencies were affected. In a way it is like reverse engineering the effects of equalization. It will help you gain a solid understanding of how EQ affects a sound, which in turn gives you confidence in using EQ. A well-trained ear should be able to listen to a sound and hear whether it needs to have 200 Hz–400 Hz reduced by 6 dB, or perhaps it needs a change in the room size of the reverb, or the **pre-delay** adjusted, and so on.

Reading about building critical listening skills can help get you started but putting them into practice is the best way to develop your ear. Practice by going to a location and sitting still while you listen. Bring a field recorder with you if you have one, although you can certainly use your smartphone with a recording app. Record the sounds you are hearing and make some notes about the sound on a technical and emotional level. How does the sound make you feel about the environment? Cheerful bird chirps in a park create happiness and peace while a wolf howl in the same setting promotes a mood of fear. On a technical side try to guess the frequency ranges of sounds and later check the recordings against a spectrum analyzer to see how close you came.

Consistency and Practice

Although this is not something often discussed, arguably the most important skill for new game audio designers is the ability to *consistently* and *frequently* create sounds within a given timeframe. Great sound design can start from inspiration but in reality that is the first step in a long process which requires intentional and consistent practice to bring great sound design to fruition. The best way to develop this essential skill is to set aside a minimum timeframe, and (without any expectations or judgment about what you are doing) challenge yourself to design a sound *every single day* within that timeframe. Just as musicians must practice their instrument daily, audio designers need to stay in practice and build their skills. The biggest challenge you may find is being tempted to seek perfection in your own work. The important thing is not to micromanage every detail, but to learn how to harness your creative abilities quickly and consistently to meet deadlines and to raise your output capabilities to the highest possible degree.

Thinking Outside the Box

A sound designer should be technically inclined and have a willingness to solve problems. Being creative and thinking outside the box are also necessary attributes. Often in-game audio issues that are seemingly simple need a bit more research and experimentation. For example being tasked with capturing gameplay with computer sound seems pretty straightforward, but some software solutions require additional routing of the systems internal audio channels to capture computer audio. Being able to work with your audio interface to route the computer audio, plus a line-in, will offer you the option to add commentary to the screen captures. This is a very basic example and you will find much more challenging tasks ahead, but it outlines the kind of self-starter problem-solving skills required to work on a team.

Rule Breaker

By this we mean being willing to step outside of standard workflow to experiment with processing and editing sound. Forget the general "rules" for effects chains and try reordering your plugins to hear how a pitch-shifter will influence a compressor and then reverse them and listen to how the affected sound changes.

When trying to place sound in fantasy worlds, experiment with more interesting reverb and delay plugins like Valhalla SpaceModulator or FXpansion Bloom to go beyond what the player might be expecting to hear based on the visuals of the room.

Microphone and Recording Techniques

A sound designer can certainly generate source from synthesizers and/or sound-effect libraries as we discussed above, but often additional source is needed for adding additional details for a unique sound. The best way to generate these details is by recording them yourself. Therefore it is essential for sound designers to understand which microphones best suit the sound they are trying to capture as well as the placement that will get the sound they are after (see "Essential Soft Skills and Tools for Game," Chapter 1, on the companion site and "Microphone Choice and Placement," Chapter 3, page 75, for more details). If your options for microphone selection are limited, you should have a solid understanding of how to get the best sound from the mic(s) you do have access to.

Manipulating Sound

This includes the ability to clean up or restore raw source, trim, layer, and process audio into a high-quality and unique asset for use in game. Good sound design goes beyond pulling a sound from a library. It's all about gathering source that fits into a palette which fits the visuals and narrative. It's great to have a load of plugins but it's important to know what to use and when and how to use it. In Chapter 3 we explore in detail using effects processing as a sound design tool, layering and restoring source.

Music Theory

A sound designer should understand the basics of tuning, rhythm, and harmony. A good sound designer should be able to distinguish when something is in or out of tune and time and how to fix it when necessary. Some sound effects might have tonal elements which would need to be tuned to the games background music or other musical elements. When tones clash the user will pick up on this sound and it can make for an uncomfortable in-game experience.

Looping

The ability to create a seamless loop is an essential skill for both sound designers and composers, resource management being a big factor for using looped assets in games. In order to manage memory and CPU usage on the game platform, loops are used to continue a sound or musical track until the game state changes or an event is stopped. Ambience, diegetic and non-diegetic music and positional sound emitters (see Chapter 8) all make use of loops to manage resources.

Achieving a smooth loop is a task that sounds easier to achieve than it is. In order to provide an immersive experience for the player we want to avoid pops and clicks or gaps in our loops. Editing the start and end points of the file at the zero crossing will help you achieve a smooth loop. In "Essential Soft Skills and Tools for Game," Chapter 1, we asked you to review some basic skills in the Sound Lab (companion site). These resources included *looping music* and *sound effects*. Take some time to review the looping assets tutorials. This will be useful for beginners as well as intermediates or professionals in the industry who may be looking for a different technique to implement into their workflow.

File Formats

Just as a graphic artist understands compression and file formats, an audio designer must have a solid understanding of file formats and their encoding processes to master the delivery of files and to meet the implementation needs of a project.

For example, did you know mp3 files add a space to the start and end of the file during the encoding process? This is true and explains why you may have encountered gaps of silence at the start and end of mp3 files. You will want to look to other formats for delivery of looped assets to avoid gaps in the loop.

Let's quickly explore why the encoding process adds silence to the assets. The encoding process of mp3 files works in a way that it needs to fill blocks of data. If your file doesn't exactly fill the blocks of data (and it usually doesn't) the process adds silence to fill the space. This leaves you with silence at the start and end of the file. When an asset is played on loop you will hear that silence in an awkward gap. You can, however, speak to your programmer and see if they are knowledgeable in compensating for the gap when triggering the asset in game. So all hope isn't lost for mp3 files, but unless you confirm your programmer has a plan you should try to avoid this format for loops.

Implementation

In "Essential Soft Skills and Tools for Game Audio," Chapter 1, on the companion site, we briefly discussed middleware and game engines and in Chapter 8 we dive deeper into the process. For our intended purposes here we want to express how essential it is to be familiar with and understand what it means to implement audio into a game. A game sound designer will at some point land a project which requires working with a game engine. Understanding how to install, run, and add sound to a scene in a game editor is essential to the process. Further understanding of the availability of audio middleware and audio engine plugins and the benefits of each will help you along in the process. Having the ability to integrate your sounds avoids the "asset cannon" approach, and this could have a more positive impact on the final product.

Mixing

Another part of implementation is the final mix of all sound and music instances in game. Mixing a game goes beyond volume control, it also includes ensuring resource management and optimization of sounds per platform, managing the perspective of sound and balancing frequencies (see "Dynamic Mix Systems," Chapter 8, page 286 for detailed information).

Command of Essential Tools

In "Essential Soft Skills and Tools for Game Audio," Chapter 1, on the companion site we reviewed the basic tools necessary for designing and implementing sound effects. Command of these tools is necessary in the competitive world of game audio. Even though it's possible for sound designers to work without much outboard gear, a solid understanding of routing and signal flow is a key skill. Setting up multiple monitoring solutions and controlling them through the computer's audio devices or the audio interface mixer is necessary for listening on different speakers. This is important because players listen on vastly different sound systems, so a game mix needs to sound clear and compelling on all of them.

Those who are looking to get into game audio often question if they have the right tools and setup to get started. The studio or environment sound designers work in can vary in size, shape, acoustics, and equipment. A state-of-the-art studio isn't necessary to design AAA sounds. In a talk at the Game Developers Conference (GDC) in 2012, Darren Korb, audio director and composer from Supergiant Games, explained how he created the sounds for *Bastion* as a one-man team on a shoestring budget by recording in his closet.[3]

Time, practice, and experimentation can result in well-polished sounds regardless of location and equipment. A properly treated room will allow for less time and energy cleaning up noise from sounds, but in the end you can make do with what you have. Don't use your lack of space or equipment as an excuse to delay getting started.

Research and Practice

There are plenty of sound designers who have YouTube channels or blogs with great tips on sound design and implementation. In Chapter 1 we directed you to the Sound Lab (companion site) where we discussed many learning resources. Schedule some time each week to read a blog post, watch a video, or a take an online course to improve, build, and expand your skills in sound design.

* * *

Let's wrap up this section with some tips for aspiring sound designers.

- Start experimenting with different microphones and different sound sources.
- Undertake further research to gain a solid understanding of key fundamentals of sound such as physics, loudness, dB, and frequencies.

- Start recording and building your own SFX library. Invest in a field recorder and library sounds for any source you that don't have the means to capture.
- Build up a database of future source material in your head. Always be listening to and analyzing sounds. Remember the sounds you might want to record later. Record them on the spot if you can, otherwise make a note and go back later to capture them.
- Keep your technical game well oiled. Know your DAW, multiple game engines, and audio middleware as if they are your best friends. Keep up with the latest technological advances and continue to improve your workflow.
- Understand game audio theory and integration systems.

* * *

The Sound Lab

Before moving on to Chapter 3, head over to the Sound Lab to wrap up Chapter 2 with some exercises and further reading suggestions.

NOTES

1 S. Böcking, "Suspension of Disbelief."
2 Sound Gym, "Audio Ear Training and Learning Center for Producers and Engineers."
3 D. Korb, "Build That Wall: Creating the Audio for Bastion."

BIBLIOGRAPHY

Böcking, S. (2008). "Suspension of Disbelief." *The International Encyclopedia of Communication*. Retrieved from https://onlinelibrary.wiley.com/doi/abs/10.1002/9781405186407.wbiecs121

Korb, D. (August 18, 2014). "Build That Wall: Creating the Audio for Bastion." Retrieved from https://youtu.be/jdnMqjLZz84

O'Donnell, M. (August 24, 2014). "Riven Foley." Retrieved from https://youtu.be/-FU_gMFW7Uk

Rayne, C. (November 16, 2013). "Playstation 4 Audio DSP Based on AMD's PC TrueAudio Technology." Retrieved from www.redgamingtech.com/playstation-4-audio-dsp-based-on-amds-trueaudio-technology

Sound Gym. "Audio Ear Training and Learning Center for Producers and Engineers." Retrieved from www.soundgym.co/

Usher, R. (April 18, 2012). "How Does In-Game Audio Affect Players?" Retrieved from www.gamasutra.com/view/feature/168731/how_does_ingame_audio_affect_.php?page=2

3 Game Audio vs Film Sound

Learning Outcomes

By the end of this chapter, you will be able to:

- Recognize the major differences between audio for games and audio for film.

If you have any background in film and TV sound, you will find that there is a definite skill set overlap with game audio (and if you're working on cut scenes, essentially they are the same job).

However, game audio has its own set of challenges that you should be aware of before we start:

Event Triggering

A computer game is non-linear. You can't predict when, or even if, the player will perform an action. These are not problems we have to deal with in film – every time you watch a scene, the same events will happen at the exact same time. In a game, every play-through will be different – we have no precognition.

So how do we approach this? We need to set up interactive "Events" that are triggered by the game. These are configured to respond differently, according to all the possibilities available to the player.

An Event could do something as simple as causing a sound effect to play, or it could trigger a complex sequence of actions (such as changing the mix settings, effect parameters, and music score).

In our game, we will be configuring Unity to trigger FMOD Events on actions such as:

On Collision.Enter

Triggered when one Game Object collides with another. For example, when the player collides with a Crate, a sound effect will be played.

On Trigger.Enter/Exit

Triggered when one Game Object enters/exits an area. For example, we will use this to change the mix settings when the player enters and leaves the Fort.

On Input.GetButtonDown

Triggered when a specified button is pressed. For example, when the player presses the left mouse button, a Rocket will be fired, and FMOD will play a sound effect.

On Object.Start

Triggered as soon as the Game Object is added to the scene. For example, when a rocket is instantiated (this just means that an instance of the Rocket is created), the rocket trail sound effect will be triggered.

We'll look at several other ways to trigger Events as and when they are needed in our game.

Event Variables

We also need to change the sound according to how the action happens – will the player brush against the edge of the door frame when exiting a room, or will they walk headfirst into the wall? Our sound could change to reflect this. We can use several different techniques here:

Event Conditions

We can trigger different FMOD Events according to states and conditions in the game. For example, the player's footsteps will trigger a different Event according to the surface they are walking on.

Parameter Control

We can use game variables to control the Events. For example, the velocity of an object could affect playback – perhaps the faster the object is moving, the higher the pitch?

Our First Person Shooter game doesn't have anywhere we could easily implement this, but there's a bonus chapter (Parameters 2) where we'll take a look at how this can be done.

Repetition

One of the core tenets of game sound is "repetition destroys immersion." If you hear the same sound again and again, then it distracts the player, and detracts from the game.

This can be particularly noticeable with dialogue (or indeed, any vocal sound effects). Games based on film licenses are particularly guilty of this. They often have limited vocal sample assets, and when you hear the same wisecrack over and over again, the repetition quickly becomes very obvious.

Although you may be tempted to throw a Wilhelm scream into your game, remember that what might seem quite funny on the first play through, will get less funny every time (yes, I'm looking at you *Star Wars Battlefront 2*…).

The Wilhelm Scream

The Wilhelm scream has been heard in hundreds of films, games, and TV programs. Originally it was part of the Warner Bros sound library (there are actually six versions of the scream), but it gets its name from the film *The Charge at Feather River*, where the Wilhelm character gets shot in the leg with an arrow.

It became a running joke to fit the scream into films, which inevitably made its way over to games. However, though you might get away with using it in a cut scene, every time the player hears it, it can break their immersion, and take them out of the game – doing the exact opposite of what you're trying to achieve!

Randomization

One approach to this is to add elements of randomization. For example, multiple samples can be placed in one Event, and when the Event is triggered, one of these samples is randomly selected and played.

One of the first examples of this we'll come across will be for our footsteps Event. We'll need several footstep samples – but not too many!

Take a quick walk across the room, and listen to the sound of your feet – there's unlikely to be much difference from one step to the next. If your footstep samples change too much, it will sound as if the game avatar is swapping shoes at every stride.

There's an art to getting this right – hitting the goldilocks zone for the right amount of variation.

We also need to consider that file space is at a premium in a game. For example, as I write this, the over-the-air size limit for iOS apps is 150MB. Any larger, and your iOS device will have to connect to a wi-fi network.

Developers try to keep their apps under this limit (if you have to wait until you connect to a wi-fi network before you can download this great game a friend tells you all about, then you're far less likely to get around to installing it).

150MB sounds like quite a lot of file space – enough for 15 minutes of uncompressed "CD quality" audio. However, this needs to be shared with all of the game assets – and graphics tend to have priority over sound… We need to get as much use as possible from every audio asset we put into the game.

For example, we could add randomization to the Event playback – we could add slight pitch and amplitude variation – not enough to be immediately noticeable, but enough to give a subtle difference every time the sample is played (we'll experiment with this with our explosion sound effects).

As an exercise, try taking some footage of a fighting game, and replacing all of the punch sound effects using only four audio files. You can layer these, change the playback speed/pitch and amplitude, but don't use any effects processing or EQ. This will give you great practice in squeezing every last drop from your audio assets.

Duration

How long is a video game? Well, a typical "Triple-A" title might come in at around 30 to 40 hours. And then you have MMORPGs (Massively Multiplayer Online Role-Playing Games) such as *World of Warcraft*, where players can spend thousands of hours in the game world.

Obviously, there isn't 40 hours of soundtrack scored and recorded for every game… One solution to this would be to loop the music. However, as anyone who has worked in retail over the holiday period will tell you, hearing the same tunes again and again quickly becomes irritating!

We can use randomization techniques to add variation to the musical arrangement, but we also need to use Events to control the music progression. This way we can use the music to inform the player, and add to the emotion of the game. For example, in *Mario Kart 8*, there are points along the course that trigger different music sections – the score always becomes more upbeat and intense as you start the last lap.

We'll keep things quite simple in our game. There will be several "pickups" scattered around the level. Each one the player finds adds to a score parameter, which controls the music arrangement.

The Music of *GTA V*

In *GTA V*, once you get in a car, you can tune your radio from a selection of stations, offering different genres of music. The DJ announcements also help build the game world, reporting on events happening in the city of Los Santos.

This almost diegetic approach works well, as listening to music while driving a car is something that goes together naturally (by the time I finished playing the game, I'd heard most of the music and DJ announcements more than once, but this issue has been fixed by offering additional DLC radio stations).

For the main missions in the game, a dedicated score that interacts with the in-game events is used. This gives the missions a more cinematic feel, and helps them stand out from the rest of the gameplay.

The interactive score was also arranged linearly, mixed, and released as "The Music of Grand Theft Auto V – Volume 2." Definitely worth a listen…

Diegetic Sound

The term "diegetic" is used to refer to sounds that originate within the game world – as opposed to "mimetic" (mimetic sounds originate outside the game world – though usually we refer to these as "non-diegetic").

Examples of diegetic sounds in our game will include:

- Explosion sound effects
- Footsteps sound effects
- Background ambience
- Sentry Drone alert and scan sound effects

In most games, narration, music and interface sound effects will be non-diegetic. Obviously there are exceptions to this – the first example that comes to mind is the music in the Fallout games. As you wander through the post-apocalyptic landscape, you often come across radios playing music by Bing Crosby and Nat King Cole.

Virtual reality offers a new set of challenges in game music. The aim of VR is immersion – the player should feel they are truly present in the game world. Anything that detracts from this should be avoided.

Now consider what it's like when you walk around listening to music on headphones… essentially, you are listening to non-diegetic music. This has the effect of distancing you from your environment.

But we still want to use music in the game – it is an almost essential tool for adding emotion. We're still in the early pioneering days of VR, and there isn't yet a "standard" way to do this, but one technique is to use elements of diegetic music. For example, in the Playstation VR demo street luge game, music comes from the stereos of the cars that the players pass as they roll down the mountain.

Dialogue can be another problem – if your character speaks, how do you make it sound as if it's coming from the player? One solution has been to process the recordings using impulse responses of a human skull!

As you work through this book, we'll encounter other areas and aspects unique to game audio, but for now, let's jump to Chapter 4, where we'll take our first proper look at FMOD.

The Essence of Interactive Audio

Let's pretend you are strapped in on the starting grid of a Formula 1 motor race. Twenty-four cars of a dozen designs roar as the lights turn green. Ninety-six tyres of various types and infinitely variable loads are poised to emit any or all of eight subtly inflected sounds, including peel, skid, scrub, bumps and roll transitions, not to mention brakes, suspension and other parts. Seconds later, they'll be jostling for position on the first corner.

In F1 and similar games, all this is orchestrated in absentia by the audio team, performed by the audio runtime system, conducted by the player. It might play on a single phone speaker, stereo headphones, surround-sound speakers or more than one of those at a time. Interactive audio might be rendered more than once, in a local split-screen view or remotely on the screens of competing players or in headphones for a virtual reality (VR) player and simultaneously on speakers for a watching audience.

Or imagine you are in a shell-hole—for practice, as penance or for kicks. Between the bomb-blasts, small-arms fire and cries of friends and foes, buffeted mud squelches round your calves as you squirm. Hundreds of explosions large and small echo above and around you. Sounds shift realistically in your speakers or headphones as you warily scan the parapet. The ground around you thumps with each impact. Will you know which way and when to jump? How soon and how surely can you know when an incoming shell has your number on it? Without accurate audio—not just cinematic immersion—you won't last long.

More prosaically, imagine you're trying to find a public toilet. Normal headphones, in conjunction with binaural sound synthesis, GPS, gyros and an augmented reality system in your phone, can tell you which way to go, whatever you're looking at, whatever you're trying to find.

A rally car splashes across a ditch. In Codemasters' *DiRT2* game, this "event" triggers 24 voices of fresh sound effects to play. Milliseconds later, as the positions, velocities, amplitudes and echoes of all those sounds are finely adjusted, another dozen voices chirp up. That's what happens when you give a sound designer a budget of hundreds of voices and need to sell millions of copies on all major console and PC platforms.

That's how modern games work. Whether the player is racing, exploring, shooting or playing a sport, convincing, immersive and informative sound is essential. The same techniques apply to virtual, augmented and extended realities, in education or training as well as entertainment. Each soundfield is tailored for a single listener who controls the camera, the view-point (first or third person, close or distant) and chooses the listening environment. All these parameters they can change on a whim.

Propelled by commercial and aesthetic competitive pressures, gamers represent a diverse and wealthy global community of early adopters, while extended reality introduces new markets in education and simulation. Interactive audio is a superset of prior sound technologies. This book explains what it needs and what it lacks.

Size of the Challenge

The author was Central Technology Group Lead Programmer on the *DiRT2* driving game, which has grossed around $200M since 2009, primarily on PlayStation and Xbox consoles; it's still selling on PC and Mac. More than 50 programmers worked on that game for more than a year, plus a similar number of full-time testers, dozens of designers and more than 100 graphic artists. This book directly addresses the roles of the eight sound designers and five audio programmers.

The concepts are relevant to anyone interested in creating interactive media and the many differences between that and the old passive media of TV, cinema and recorded music. This book draws on decades of game development experience, encompassing music, sport and shooting genres and even VR space exploration. It includes war stories, deep geek detail, analysis and predictions.

DiRT and *Formula 1* (F1) games are hardly the tip of the iceberg. Top-selling "triple-A" games, like Rockstar's *Grand Theft Auto 5*, cost hundreds of millions to develop but generate operating profits of billions of dollars.[1] In its first five years *GTA5* sold more than 90 million worldwide, at prices higher than any movie. Such success is only possible by designing it to suit all platforms, current and future, rather than picking one console or PC configuration.

GTA5 audio benefits from audio adaptations demonstrated by the author at the 2009 Audio Engineering Society (AES) Audio for Games conference in London. Such scalability is a prerequisite of sustained global sales, and it's been achieved by constant technical innovation, including the adoption and refinement of advanced techniques described here.

This is a book about managing complexity in a way that suits the customers, the designers, the platforms and current and future genres of entertainment and training. It's a book about curves, synergies and neat tricks. It's also about having fun making interesting noises and understanding and playing with psychoacoustic principles.

It's Different for Games

A misperception, slowly abating, concerns the superficial resemblance between games and passive media like TV and movies. Those are made for a mass audience, compromised to suit a generic consumer and set in stone before release. But each game is live, one time only.

The experience is never the same twice. The more it varies, the greater its lasting appeal and the more it involves and teaches the player. Whether it's limited to a few dozen sounds or the hundreds of concurrent samples modern computers can mix, it is a designed experience, dependent upon categorisations and decisions made long before but mediated by the player. There is no 'final cut.'

> The ear never blinks.

The runtime system necessarily embeds the experience of sound designers, engineers and live mixers so it can "finish" the product on the fly. This draws on the asset-selection and balancing skills of designers, just as movies might, but demands more variety, flexibility and dynamic configuration, because the players call the shots. Audio is more demanding than graphics because all directions are equally important in a game, the ear never blinks and there's no "persistence of hearing" akin to the persistence of vision which smooths out the flickering of video.

Perception is multi-modal. Georgia Tech research, published by MIT in 2001, established that high-quality video is more highly rated when coupled with high-quality sound.[2] Good graphics also make poor audio seem worse! Even if audio is not explicitly mentioned, it has a profound influence on perception.

Psychoacoustics

Sounds have multiple dimensions: pitch, intensity, spread, timbre, distance, direction, motion, environment. Our brains interpret these according to psychoacoustic curves learned over lifetimes in the real world. Each aspect, and the changes in each dimension, must be plausible, sympathetic to the whole and perceived to be smooth, progressive and repeatable, to maximise the informational content and forestall confusion.

Whether they can see it or not, audio tells the player the distance and direction of each sound, how it is moving, and what sort of environment surrounds it and the listener. More deeply, it identifies threats, opportunities, choices and risks which will influence future events, in whatever way the player chooses to interpret them. Learning by finding out is more personal and persuasive than following some other's story. It relies on coherence, consistency and flexibility, because any glitch might break the spell and destroy the suspension of disbelief which turns a simulation into a lived experience.

Like any artistic representation, the results can be symbolic, realistic or hyper-real. Games and VR often target a remembered dream state more than strict reality, in which auditory salience works as a filter rather like depth of field in a film. But whereas cinema auteurs pick a subset of the sounds to fit their pre-ordained story and spend arbitrary time compiling each scene, interactive media must continuously identify and include cues that help the listener create a new, unique story of their own, not once but many times over, without inducing monotony or revealing gaps that might burst the bubble of immersion and agency.

Strict realism is just a start. The most realistic *Richard Burns Rally* would only be playable by Richard Burns. In a real *F1* race you'd rarely hear anything but your own engine, but in a game that's not good enough. Interactive audio has the capability to be symbolic, extracting just the essence or key cues in a scene, or hyper-real—reproducing the remembered synaesthetic experience, augmented by imagination, not the prosaic reality. If you're ever unlucky enough to be in a car accident you may find the crash disappointingly dull by game standards—but games and VR are meant to be fun, thus all the more memorable.

What's New?

This book explains how modern interactive audio systems create and maintain consistent and informative soundfields around game players and consumers of extended reality products. It complements books for sound designers and game programmers by filling the gaps between their accustomed models of reality.

It's not a book about sound design, though it contains many tips for sound designers, especially related to interactivity. It's not a book for mathematicians, though it builds on and refers to their work. It's not even a book about psychoacoustics, though of all the related fields, that one informs the content most of all. There are many excellent books about all those subjects, as the references reveal.

This book pulls together concepts from those fields and decades of practical systems design and programming experience to explain how they fit together in modern games and extended realities. It presents a layered approach to implementing advanced audio runtime systems which has been successfully applied to platforms ranging from obsolescent telephones to the latest game consoles, arcade cabinets and VR rigs.

This is a practical book for designers, programmers and engineers, boffins, inventors and technophiles. The focus is on runtimes—the active parts that ship to the customer—rather than pre-production tools, though Chapter 24 surveys free and commercial tools for content-creation. It tells how to do things, with tested examples, but more significantly it explains why those things are useful and how they fit together.

Starting from the first computer-generated beeps and the basic challenges of volume and pitch control, it traces the development of audio output hardware from tone generators to sample replay, the layering power of multi-channel mixing and the spatial potential of HDMI output, building up to the use of Ambisonic soundfields to recreate the sensation of hundreds of independently positioned and moving sound sources around the listener.

The author has pioneered the interactive use of 3D soundfields but is well aware from research and direct experience that few listeners enjoy a perfect listening environment. One of the greatest changes taking place in media consumption in the 21st century is the realisation that there's no correct configuration. Figure 1.1 shows the preferred listening configurations of more than 700 console and PC gamers.[3] Every listener benefits from a custom mix, and interactive audio systems deliver that as a matter of course.

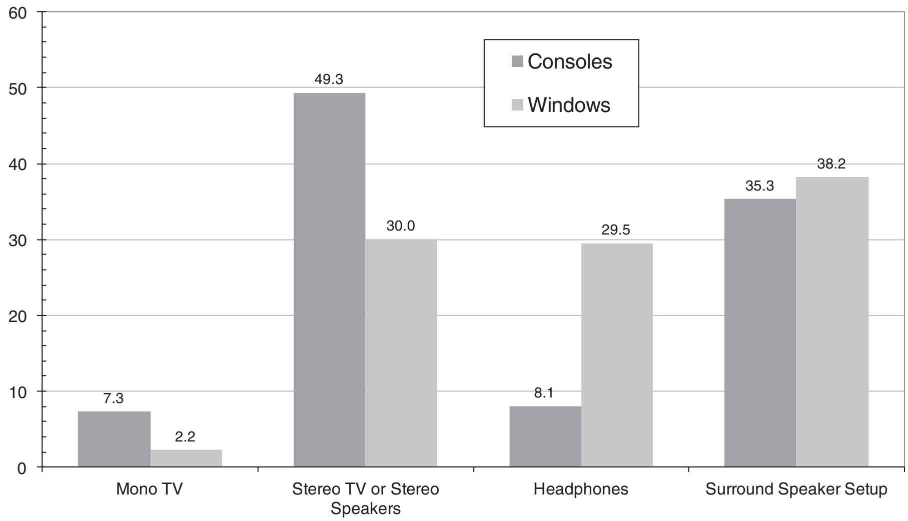


Figure 1.1: Listening preferences in percentage by output format

Since then phones and tablets have shifted the goalposts—although Apple TV and many Androids support HDMI and 7.1 output, most listeners use mono speakers. Live analytics provide specific usage information. Over 40 million sessions, barely 3% of mobile players of *F1 Race Stars* used headphones; 17% had the audio muted! The story was similar in *Colin McRae Rally* for iOS, with 8% on headphones and 12% unable to hear the co-driver. It's nice to know that 88% were listening, even if mostly in mono.

Single-Player Optimisation

Traditional movies deliver a generic mix for a mass audience in a cinema, aiming to create a sense of immersion so that those in the cheap seats still feel part of the party. Immersion is easy—it involves little more than spreading ambience around the listener. But interactive media is much more demanding. It focusses on each individual listener without requiring them to be locked in a halo-brace centred in an anechoic irregular pentagonal room with a wire-mesh floor—even if that would deliver the most technically perfect reconstruction available with commodity components to the golden ears of an ideal listener.

Some expensive aspects of VR—like head tracking, which adjusts the experience every few milliseconds to account for movement of the client's ears and eyes—are not needed for conventional gaming. Here the player's thumb directs the senses of their avatar; looking away from the screen is a recipe for disaster. However control is achieved, audio challenges remain. The "sweet spot" in which sounds are perfectly balanced may be small, especially if space is tight and speaker positions are compromised, but a lone player is strongly incentivised to move into it.

Playing and exploring together is still fun even if the shared environment compromises individualised perception. So this book reveals techniques to deliver multiple soundfields in a single listening space for split-screen multi-player games and ways to mix adaptively for multiple simultaneous listening environments, like a single-player VR experience shared by remote passive observers or others waiting their turn in the same room.

This concept of "multiple endpoints" marks out new media platforms and modern ways of listening. Custom mixing and spatialisation technique can create additional coherent soundfields at low marginal cost. We'll explore the pros and cons of headphone and multi-loudspeaker listening while showing how even a single mono loudspeaker can inform active listeners about the movement and relative position of 3D sounds in their vicinity.

Dynamic Adjustment

One of the greatest strengths of interactive audio—as opposed to TV, radio, cinema and other media designed to place a ready-made mix before the ears of many passive consumers—is the continuous conversation that takes place between active listeners and a dynamic audio rendering system. Interactivity is a cyclic process.[4] Unlike old media, interactive audio adjusts continuously to hints from the player as well as dynamic changes in the simulated world.

> All directions are equally important.

The player, not the director or camera operator, decides where to look and listen. All directions are equally important when all can be interchanged with a flick of the player's finger; our ears perceive sound from all directions, without blinking or blind spots, even though their response is coloured by direction and environment. Just as the brain integrates uneven responses to sounds in a complex environment by small head movements, filling in the gaps from short-term memory, sedentary players derive the same benefit by twitching their thumbs.

Learning Through Play

Even if they're incomplete or not quite correct, players learn to interpret multiple environmental cues to their advantage by experiment. The power of such learning stems from reinforcement, positive and negative, and the tight feedback loop between the input devices—gamepads, mice, track balls, accelerometers or keyboards—and output devices like screens, speakers, headphones and haptic emitters like rumble motors, butt-kickers and "force-feedback" steering wheels.

Optional sub-woofers are used in cinema for low-frequency effects and often to economise on the low-frequency response of more numerous speakers through "bass management." These deliver both sonic and haptic feedback. We still need to know what the client expects, drawing on their larger experience of real-life listening and their equipment's capabilities, to decide which aspects of the sound to model and prioritise.

Scalability

Rally is a motor sport that sets a single driver against the clock and a complex environment. It may be simulated on a handheld phone or a console rig that dominates a room. This book compares Rally games made in the same studio a few years apart, showing how they were customised for their platforms.

A current PC or console game might mix 250 voices—individual spatialised sounds—from an hour of short samples in RAM and hours more swappable or streaming from disc, all with filters and layers of reverberation driving a 2D or 3D 7.1-speaker rig. Contrast an obsolescent iPhone struggling to play a dozen sounds at once with no reverb or filters and in mono at half the sample rate and video stuttering along at a cartoon-like pace.

In the heyday of iOS 7 Apple would not list a title on their store unless it ran on their end-of-life iPhone 4, even though that model was about a seventh the speed of the next one up and 50 times slower than the latest iPad or phone. In the past decade game developers have switched from chasing tens of millions of high-end gaming PCs and consoles to supporting hundreds of millions of battery-powered handheld devices, where the best were 100 times more capable than the least. To reach such audiences you must innovate on the latest devices without forgetting the also-rans. Discoverability is all, in the biggest and most competitive fashion markets.

So *Colin McRae Rally* mobile was playable on iPhone 4, cheap Androids and even BlackBerry phones, yet the same game got a second Apple Store feature on the strength of its 60-Hertz video and 48-kHz stereo sound on high-end iOS 8 devices. That configuration also plays extra collision, environment and surface sounds, and allows the user's music to override the game's. The iPhone 4 was limited to 24 kHz mixing, 15-Hertz video, reduced effects and playing sounds from the two wheels nearest the listener.

Scalability matters. To reach the biggest market and to stay there you need to reach the minnows at the start without short-changing the nascent whales. You might find you're making the same game, in varying priority order, on iPhones and iPads, innumerable semi-compatible Android phones and set-top boxes like the NVIDIA Shield, PCs in myriad configurations, plus recent Sony, Microsoft and Nintendo consoles. Sometimes even macOS.

Memory capacities vary from half a gigabyte (GB), mostly allocated to other things, to many times that amount. Physical game media hang on for high-end consoles, while one Blu-ray disc can hold 200 GB of compressed data, but downloads will predominate in future and with no such limit. However much RAM you get, you can't load everything into it—not least because of the loading time—players are impatient people. So we discuss how to organise, compress, index and track game audio data.

This book explains how to transparently handle different formats—sample rates, codecs, multi-channel sets, resident and streaming—so game designers don't need to worry about platform details, yet the products make the best of each configuration. Even early in the 21st century it was not uncommon for a game to come with over 150,000 samples; all it takes now is a smart commentary system and internationalisation to drive this over a million.

Pervasive Interactivity

New forms of content delivery, 3D graphics and sound mean that even passive media is sometimes authored with game-centric tools like Unreal Engine and Unity. As customers come to expect custom experiences, dynamic systems find use in film and TV, and by architects, trainers and planners, not just those seeking to bottle up interactive thrills.

After years making do with hand-down technologies like MP3 and Dolby surround, designed for the average expectations of passive consumers, the tide has turned. Naturally interactive and scalable schemes like Ambisonics,[5] context-sensitive mixing and Virtual Voice management are being shared in the opposite direction, with the realisation that film and TV are subsets of interactive media. Even if you're doing nicely in the old media world, it pays to learn how you can empower your customers and ease your workflow, especially in pitching and prototyping, by borrowing ideas from games and XR.

Audio Runtime Systems

The audio runtime system is the part of a simulation which creates soundfields around its users, enabling them to find their way around and make sense of the simulated environment using the skills they've learned in real life. It sits under the "game code," called from the "update thread," and above the "audio pump" that spits out freshly mixed blocks of multichannel sampled sound every few milliseconds. It does the job of a live mix engineer as well as resource management.

How high above that pump the runtime sits depends upon the layering and the pre-existing audio capabilities of the host. This book shows how the layers fit together and how to evaluate existing components to see if they fit your needs. It explains how to test those and decide just what your product needs, rolling your own if desirable.

The main purpose of the layering is to provide a consistent set of advanced capabilities on every platform, large and small, yet tailored to take advantage of the unique features or limitations of each platform and take any help—in terms of standard audio components, like mixers, panners, filters, resamplers and reverbs—which is predictable enough to be of value. There are two deeper interfaces in this layer cake.

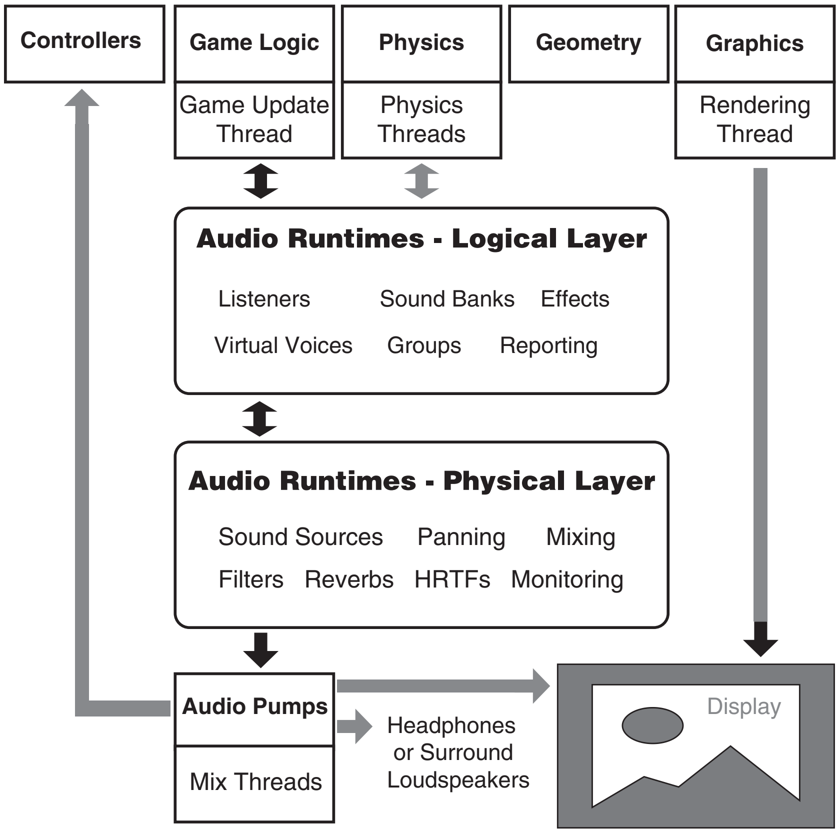


Figure 1.2: Layers and threads in the audio runtime system

Runtime Layers

The top layer is an object soup of Virtual Voices, which play samples generated on the fly, loaded piecemeal or in organised Banks. The voices behave rather like independent threads, all gathered together in Groups and mixed via Listeners—virtual microphones—to whatever audio outputs the player selects. All in 3D. This layer works out what to play when there's more than the hardware supports—hence the term "Virtual Voices." It handles distance effects, Doppler pitch shifts, frames of reference, memory management, profiling and reporting. It provides a common interface for game programmers so they can make one game for many platforms, which makes allowance for the strengths and weaknesses of each without dumbing down the strongest or over-taxing the weak.

This logical layer rides on top of a simpler interface which can be bolted in a few days onto any multichannel platform audio sub-system capable of mixing memory-resident samples and outputting them, like XAudio or OpenAL. At this point, if the platform is mature, you can take advantage of tightly coupled and optimised machine-specific implementations of basics like pitch conversion, filtering and reverb—providing you know how to translate the controls in a platform-independent way or can explain the differences in a way which will thrill rather than puzzle hard-pressed designers. You grab what works and fix what's left.

Typically, you get optimised hardware/DSP/codec and reverb access, quite often using hardware which would not otherwise be available to you. If all goes well, assuming good documentation or willingness to experiment and rely on the results, you get the benefit of several platform-experts working on your audio team, albeit distantly. And if not, there's always a fallback.

Platform Abstraction

If you lack all or some of this physical layer—generally provided by the operating system maker—you can augment or replace it with your own code. This book explains by example the steps involved in building a generic cross-platform mixer, as well as where to find them off the shelf, perhaps for free. The requirements are more relaxed than for a full application programming interface, since all we really need at this layer is multiple sampled voice replay with pitch and volume controls.

Software mixing won't be as fast as hardware, but it's easier to keep it working consistently across all the platforms. It is a stable core, decoupled from its clients by the logical layer, in which piecemeal or wholescale optimisations can be performed as your profile timings and production schedule dictate.

The customer-friendly stuff like Virtual Voices and microphones, distance and Doppler effects, event logs and priority systems is all done upstream, in the logical layer. It's most of the code on any platform and shared between them all.

What You Get

In other words, we give you, or your heirs and successors, a quick way to get a sophisticated, scalable audio system running on almost any platform—after the first—and a production-engineering framework to optimise the heck out of it afterwards without everything breaking.

This book explains the organisation and implementation of a system to manage hundreds of virtual 3D voices with Ambisonic positioning, multiple directional listeners, group updates and reporting, distance and Doppler effects, group and per-voice filters, delays, reverberation and optional asset streaming. The mixing stage supports as many voices as your hardware and other needs of the title can handle, with live decompression, parametric equalisation, advanced Ambisonic panning and click suppression, for mono, stereo, binaural and surround-sound outputs to multiple endpoints.

Key aspects are illustrated with short program snippets expressed in a slim subset of C++, the standard language of portable real-time systems, which can be converted to ANSI C, C# and related languages. The code has been tested with Visual Studio 2017 for Microsoft x86 and x64 architectures, Apple LLVM version 9.0.0 (clang-900.0.39.2) on macOS and GCC 5.4.0 on Ubuntu 16.0.4 compiling for ARM64 with -std=c++11.

The caret ^ symbol is used in this text to denote exponentiation, aping languages like Algol, TeX, Google's calculator and code representations of exponentials when a superscript is not available; 2 ^ 3 = 2^3 = 2 * 2 * 2. Unfortunately C has no exponential operator, relying on the library function powf(), e.g. 2^3 = powf(2.f, 3.f), which is why I opt for the commonly used ^ shorthand. The separator .. is consistently used, as in programming languages like Pascal and Ada, to denote a subrange. For example 1..4 denotes values between 1 and 4. The prefix m_ introduces variable values which are class members. Empty parentheses after an identifier indicate that it's a function.

Sections of This Book

The book is in four sections. First it traces the history of interactive audio generation, showing the fundamental audio issues of timing, mixing, volume and pitch controls and how they've been tamed in software and hardware. The following section considers what a modern audio runtime system needs to do and the team roles which make it possible.

The third major section, from Chapter 15 onwards, explains implementation options and their rationales in detail. Chapter 24 lists concepts and resources, free and commercial, which can help you build and maintain a flexible, portable, extensible audio runtime system without reinventing wheels. You don't have to use everything in this book to derive full benefit, but you should consider most of it.

The references cite many relevant books and research papers, but in such a new industry some insights are only available online. Inevitably links go stale in the lifetime of a book, so you may need to use Google to find some. The Wayback Machine[6] tracks old websites.

Topics are introduced in logical sequence, to minimise the need for cross-references, but once a term is introduced its meaning is taken as read. For best results, read the book first from start to finish. Later you can cherry-pick concepts and implementation details to suit your application and audience. There are plenty to choose from.

References

[1] *GTA5:* www.forbes.com/sites/insertcoin/2018/02/08/at-90-million-copies-sold-gta-5-further-cements-its-inexplicable-legendary-status/#429cd6de2e22

[2] *Interactions in Perceived Quality of Auditory-Visual Displays; Russell L. Storms and Michael J, Zyda, Presence, ISSN 1054-7460, Volume 9 Number 6, December 2000*

[3] *How Players Listen; Simon N Goodwin, AES 35th International Conference, Audio For Games, 2009:* www.aes.org/e-lib/browse.cfm?elib=15172

[4] *Understanding Interactivity; Chris Crawford, 2000, ASIN B0006RO8ZA, no ISBN*

[5] *Periphony: With-Height Sound Reproduction; Michael A. Gerzon, JAES, ISSN 1549-4950, Volume 21 Number 1, February 1973, pages 2–10:* www.aes.org/e-lib/browse.cfm?elib=2012

[6] *Wayback Machine*: https://archive.org