# ABSTRACT

My master thesis deals with a study on, and a demonstration of, the potential of procedural audio in computer games. Furthermore, this thesis examines the influence of procedural audio on the job of an audio designer.

## A Historical Perspective

First, important aspects of the development of computer games are described through a historical perspective. This description focuses on the evolution of techniques influencing design and implementation of audio in computer games: Starting from an experimental and programmer-based process in the 1970s where the sound was generated via sound synthesis – and evolving to controlled processes, where specialists from fields such as film and music today take part in the creation of audio for computer games. This is due to the fact that audio is handled in a similar sample-based manner in all three fields. This allows specialists from one field to be involved in audio production in the other fields as well.

## The Limitations of Sample-based Audio

The use of sample-based audio locks audio to a linear, time wise progression. The precise time-bound nature of linear media, such as film and music, goes against the principle of interactivity which is crucial to computer games. This results in a lot of resources being spent on making linear audio work within the non-linear context of the computer game. In addition, the sample-based technology results in an ever-increasing need for storage capacity. Computer games are made to entertain for many hours and therefore require huge amounts of audio in order to avoid repetition. This leads to an increased use of sample libraries.

In order to present a qualified alternative to the existing approach I analyze the typical work process of an audio designer – and the ways in which it is conditional on the tools involved.

Through this analysis I find that the development process is marked by a clear distinction between the production phase and the implementation phase.

This waterfall-like approach makes it hard to move back to the previous phase in order to correct or adjust the produced audio. This inflexible work process is a result of the tools used. The production tools handle audio in a linear fashion, which necessitates resources being spent on trying to hide the linear starting point.

## Embracing the Medium

Having exampled the problems related to the sample-based approach I seek to clarify how we have ended up in the current situation by analyzing the genealogy of computer games as a medium.
The analysis is based on the term *remediation*, coined by Bolter and Grusin. The analysis leads me to conclude that remediation of filmic aesthetics has lead to the reuse of work processes and tools – which have been developed in relation to working with the linear film medium. Remediation is inevitable, so it might be worth examining alternative influences for the development of audio in computer games, in order to optimize the development process according to the fundamental principles of the medium. In this way it might be possible to create audio designs that utilize the potential of the medium.

The use of sample-based audio limits the possibilities of auditive variation, and at the same time it hinders the work of the audio designer. This brings me to conclude that audio production for computer games should take advantage of the non-linear nature of the computer in order to produce audio designs that exploit the potential of the medium. In this way, the work process related to the creation of computer game audio would also be optimized.

## Procedural Audio

Consequently I describe an approach where audio is not locked sequentially to time, but varies within a defined setting. The setting is defined through a number of parameters and the computer game audio is generated in real-time according to these parameters. The parameters can be changed on the fly and can vary according to events like player interaction, or react adaptively according to internal game processes. This makes the audio capable of adjusting to the current action taking place, rather than the action foreseen by the game developers when using sample-based audio.
I define audio produced from such an approach as *procedural audio*. Hereby I mean audio that is generated synthetically in real-time according to parameters that can be affected by both player and game world.

## Implementing Procedural Audio: a practical example

Having defined an alternative to sample-based audio I describe techniques that can be used when developing procedural audio. I use a couple of these techniques in a practical demonstration of the potential of procedural audio in computer games.

In light of the acquired knowledge I implement a prototype of a game-like scenario using the game engine Virtools. The prototype lets one navigate a tube-shaped object around a floor via a mouse. The audio for the scenario is generated in real-time through the implementation of three different sound synthesis techniques in the visual programming environment Max/MSP[1].
The audio of the prototype is a product of the action taking place - changes in the scenario cause parametric changes that are the foundation of the audio generation.
Reflecting on this work, I examine how it could impact the competences, tasks and role of the audio designer.

[1] The used synthesis techniques include: Modal synthesis, physical modeling, and subtractive synthesis.

## Results and conclusion

Through a combined theoretical and practical approach I find that procedural audio can be implemented in computer games. It requires continuous data from the game engine to be mapped to sound synthesis parameters. In this way it is possible for the audio designer to create responsive audio designs that behave realistically as opposed to sample-based audio which primarily sound realistic. In addition, procedural audio reduces the needed storage capacity, because the audio is generated when needed.

Procedural audio designs make use of continuous data from the game engine. This requires the audio designer to plan which data should be mapped to sound, and how. Mapping the parameters is an important task for the procedural audio designer, and it allows for a great number of possibilities. I found that rigid mappings between game parameters and sound synthesis could limit the audio designer’s creative possibilities: If the shape of an object dictates its sound, the influence of the audio designer will be compromised. Instead the procedural audio designer can find inspiration in real world mappings, and base the synthesis methods on more abstract interpretations of these events.

The work process of the audio designer changes when working with procedural audio. The methods behind procedural audio are rooted in the *synthesis paradigm*, whereas the methods of audio designers working with sample-based audio are related to the *sequencer paradigm*. In the sequencer paradigm the computer is used to arrange audio in a linear time-wise fashion, whereas the computer is used as a generative audio machine in the synthesis paradigm. Therefore, the procedural audio designer must be able to alter his perception of audio. Procedural audio is an auditive material that is created and shaped in real-time on the basis of parameters from the game engine. This requires the audio designer to be able to understand and use paradigms that are normally in use by programmers, while maintaining the well-known role of developing efficient, aesthetic and communicative audio designs for computer games.

To balance this, the procedural audio designer can carry out his work in abstract, visual programming environments. These allow him to work with familiar concepts, which can increase the creativity.

The change of work paradigm also affects the entire work process of the audio designer. The waterfall-like approach known from the development of sample-based audio in computer games is replaced by an iterative process that allows changes and corrections to be made along the way.
Procedural audio has technical consequences that relate to the *variable cost* of procedural audio: The more complex the sound is the more work it requires to produce, contrary to sample-based audio that has a *fixed cost*. Because of the variable cost, procedural audio will affect the computer's CPU more than sample-based audio. This only had little effect on my prototype, because of its limited size and complexity.
On a larger scale it might be necessary to generate procedural audio from a dedicated *Audio Processing Unit* (APU), in a similar manner to the introduction of the *Graphical Processing Unit* (GPU), when 3D graphics were introduced in the 1990s.

During my practical work I experimented with different synthesis methods[2]. Among these were subtractive synthesis, through which I found that efficient and responsive procedural audio in computer games does not necessarily require the use of advanced synthesis methods. This result has been a cause for wonder if we need 'realistic' audio based on recordings of what is visually depicted, or do we just as much need audio that behaves realistically in an interactive context? The latter is hard to achieve with a sample-based approach, since audio here is arranged according to time.
This question is not answered in my thesis, but it could prove to be a relevant question to future work with procedural audio in computer games, since it could clarify in which contexts procedural audio is beneficial.

The use of more complex sound synthesis may require the audio designer to understand and use programmer-related paradigms, which may alienate the audio designer from the process. This could be solved by letting the audio designer work with *modal synthesis*, where prefabricate parts are put together to build the needed sound.

Very few games make use of procedural audio today, but the game Spore might be able to change that. Even though the game is not to be launched until September this year, it has been subject of extensive media coverage. The interest for Spore is mainly due to its widespread use of procedurally generated content, regarding graphics, animations and audio. If the game is a hit it may increase the interest for procedural audio in other computer games. Furthermore, rumour has it that EA and Sony are planning to make use of an Open Source audio engine, based on the visual programming environment PureData in next generation game consoles and games. If this proves to be true, the interest for procedural audio may increase in the years to come.

## Critique

An alternative method for acquiring knowledge of synthetically generated audio in computers could have been based on analysis of comparable fields, such as software-based musical instruments. Several of these utilize synthesis techniques such as physical modelling as opposed to sample-based software instruments.

My relatively simple prototype does not benefit from the direct potential of procedural audio. The primary advantage of procedural audio is its ability to create great amounts of content on the basis of very limited data. A rather limited prototype like mine will therefore not show such benefits. This requires more elaborate testing in complex environments.
Finally, I would like to address my delimitation of this master thesis. I have focused my work on procedural generation of sound effects, but most computer games also include music and speech. Therefore, this thesis does not deal with the possibility of producing entire audio designs via a procedural approach. Algorithmic composition and speech synthesis are comprehensive studies that are covered elsewhere. In combination with such studies I hope that my work can contribute to the future study of procedurally generated computer game audio.

[2] Physical modeling, modal synthesis, and subtractive synthesis.