# Samples Homogenization for Interactive Soundscapes

**Jorge Garcia Martin**

MASTER THESIS UPF / 2011

Master in Sound and Music Computing

Master Thesis Supervisor:
Jordi Janer
Department of Information and Communication Technologies
Universitat Pompeu Fabra, Barcelona

To my parents and brother.

# Acknowledgements

Firstly, I would like to thank my supervisor Jordi Janer for his academic guidance, help, contributions and for all the brief but valuable advice to focus the direction of this thesis within the existing projects at the Music Technology Group: the Soundscapes platform and Freesound. I also have to thank Stefan Kersten for his guidance, shared contributions in this thesis, and all the software engineering, OSC and SuperCollider tips. Thanks to Sergi Jordà for all the comments and help related to some Pure Data and HCI concepts. Cheers to Graham Coleman and Vincent Akkermans for their handy feedback and comments. I also want to thank Xavier Amatriain and Pau Arumí for their software engineering classes and attitude. At this point I am grateful to Xavier Serra, for bringing me the opportunity of being in Barcelona as a Sound and Music computing student and learn from the aforementioned individuals, the rest of teachers of the Master, and the Computer Science undergraduate courses.

I also have to mention the valuable feedback and comments that I received from industry professionals, who also shared their vision and helped me to develop a better understanding of some of the fields related to this thesis topic: Amaury La Burthe (AudioGaming, France), Jeanine Cowen (Berklee College of Music, US), Leonard J. Paul (VideoGameAudio, US), Steve Johnson (SCEA, US), Erik Petterson (Lionhead, UK), Andrew Quinn (Splash Damage, UK), Gianni Riciardi (Ubisoft, Italy), Kenny Young (Media Molecule, UK), Damian Kastbauer (Lost Chocolate Lab, US), Will Goldstone (Unity Technologies) and Iñigo Quilez (Pixar, US). Special thanks to Andy Farnell (DSP scientist, UK) and Nicolas Fournel (SCEE, UK) whose work and publications related to procedural audio and audio analysis in part inspired this thesis topic. I also want to take this opportunity to mention George Sanger and Max Matthews for being pioneers in the fields of Game Audio and Computer Music, helping to build the path for a lot of us.

I can't forget to mention the persons that motivated me the most in the last decade. Especially Daniel Ferreira, for being so authentic in sharing his passions about synthesizers, arts and electronic music, and for being my colleague in the Clubbervision project for several years. Thanks to Enrique Rendón, from the Universidad Politécnica de Madrid, for being part of the academic seed that encouraged me to pursue a career in the fields related to computer graphics, games and interaction. Thanks a lot to Josué Gomez, for all his friendship and support while sharing his passions about music, voice acting, radio, media and broadcasting over the years. Cheers to Julio Galarón, for being such a geek and passionate guy in terms of technology and life

in general. Thanks to Fer and all the crew at Electronic Arts EIS in Madrid for the opportunity of working there and supporting my first steps in the games industry.

Additionally, I can't thank enough the Hispasonic community for being a valuable online resource and point of contact for all-things Audio and Music technology related. And most recently, to Designing Sound blog for sharing such amount and quality of articles and interviews. Additionally, I want to send kudos to the Game Audio Network Guild, the Game Audio Pro group in Yahoo, the Interactive Audio Special Interest Group, the Procedural Audio group at Mendeley, StackOverflow and Stratos, for their support to the community. Cheers also to all the people, companies and institutions involved in the worldwide Music Hack Day events where I participated.

Because of these last two years in Barcelona, I have to be very grateful to all the SMC Master colleagues and friends that I shared moments with, and specially to whom I have probably enjoyed the most important moments and adventures of my learning, fun, pain and laughter while doing the Master: Alvaro, Andrés, Imanol, Panos, Felix, Adrian, Zuri, John, Alexis, Stelios... (lots of names missing here). Thank you guys! Also, I want to send huge thanks to all of my friends in Barcelona and Catalunya. Thanks to David, Irene, Manu, Esther, Robin... and my flat mates during last year: Alejandro, Lucre and Silvia.

Here I want to thank Victor for his lifelong friendship and support since we met at the primary school, more than 15 years ago now.

I also have to thank my mother for her inspiring attitude at doing meaningful and funny things in life. Thanks to my father as well, for always pushing the bar up and motivating me to advance on both at personal and professional level. Thanks to my brother, for his love and support, bass playing inspiration, electronics motivations and music recommendations.

Finally, very special thanks to my partner Elisa, for her loving support and understanding during several months in the distance, all the travels and unforgettable moments shared together.

**Jorge Garcia Martin**
**September 2011**

# Abstract

This thesis presents the challenges and current state of the art related to soundscapes modeling and design. The concept of soundscape comprises various fields of knowledge and craft: from scientific to technical and artistical. Moreover, current scenarios demand new digital content creation (DCC) paradigms focused on user generated content (UGN) platforms and on-line repositories of sound assets, like Freesound. The work carried out focuses on DSP methodologies that can be used to support the creation and design of soundscapes. We stress their relationship with the nature of interactive and immersive environments.

One application area is explained for the use-case of "sound concepts" that need to be modeled using material from various recordings, carried out using different setups or sound design techniques. An homogenization algorithm is presented with the aim to reach a certain homogenization across groups of samples by means of auditory-based features. Then, a study about the transformations that can be applied using this model are presented as equalization methods. To support the development and evaluation of the techniques presented, a prototype in Matlab and the integration of a SuperCollider server with the game engine Unity 3d have been carried out. Additionally, some applications and use-cases are also mentioned within the contexts of interactive (non-linear) and linear sound design.

**Keywords:** Audio, DSP, Sound Design, Games, Simulation, Soundscapes, Analysis, Synthesis, Interaction, Content-based audio transformations, Cognition, Perception.

*"The definition of insanity is, perhaps, using that quote."*

George Sanger, The Fat Man @ O'Reilly blogs

# Contents

# List of Figures

# Chapter 1

# Introduction

## Motivations

Designing and modeling a soundscape are tasks that comprise various scientific, technical and artistic fields of knowledge. Traditionally, the concept of soundscape has been playing a major role in different areas (like psychoacoustics, environmental studies or electroacoustic composition), but it also has considerable importance in mainstream media, games, art installations and interactive applications. Moreover, current scenarios demand dealing with new digital content creation (DCC) paradigms focused on user generated content (UGN) platforms and on-line repositories of assets, like Freesound[1] .
Historically, interactive audio was dominantly the domain of games, but increases in processing power and wider availability of high-level audio APIs have opened up more possibilities for research, training, and educational applications (as defined by the AES Audio for Games website). In addition to this, open virtual worlds demand an enormous amount of assets to sonify each of the objects, characters and interactions that a user can experience. Dealing with thousands of static audio samples is also a great problem within new tendencies on procedural animation [2], mobile platforms or cloud computing. Current trends are driven towards a procedural audio approach, that allows flexibility, pseudo-automatic generation of assets and better integration with other software components (in the case of games, physics engines, for instance [3]). Hybrid physically-based synthesis and spectral modeling synthesis techniques based on samples can help out to define future techniques for generating assets models. Also, content-based audio transformations and new adaptive digital audio effects research offer a stimulating framework to build up future technologies. There are various open fields in industry and research that motivated the choice of this thesis topic:

- Environmental studies related to urban planning and health.
- Sonic Interaction Design, and the influence that audio feedback has over the user in various contexts, like virtual reality and games.

[1] http://www.freesound.org

- Data sonification as an interactive and non-interactive medium.
- Adaptative digital audio effects (A-DAFX), offering a gestural control and mapping framework for DSP algorithms.
- Game Audio (VR, games, simulation and sound for interactive media).

Hence, one of this thesis' aims is to be a link between recent areas of research and industry. Moreover, the design and development of high-level tools to support the work carried out by soundscape designers (more appropriately named sound designers and/or sound implementers) is important to ensure that the message and the content fit quality standards and technical requirements.

## Goals

- To research on digital signal processing and the needed techniques for samples homogenization in order to have timbrical and level coherence in groups of related sound samples.
- Integration of a soundscape generation system within a commercial simulation/games development engine: Unity 3d[2].
- To develop and publish a set of demos and libraries of the methods presented.

## Outline

This thesis is structured in various parts, each of them having a certain dependency with the others, though each chapter can be read separately. Firstly, an introduction to the problem domain is presented. The scientific background defines part of the ground truth for the research and studies on soundscapes, the related technologies, and some projects that present the concept under different perspectives. Then a more particular state of the art scenario is presented in the fields of games, virtual reality and interactive applications focusing on the interactive nature of audio in those environments, with recent research references in the area. The part of soundscape systems and audio engines is a show-and-tell of the available commercial products and platforms that are also coming from academia. An overall approach related to procedural audio and its relationship with sound synthesis is also defined. Also, the relationship of content-based audio transformations and the work done in adaptive digital audio effects is presented within the context of interactive audio in the soundscapes modeling context. Then the algorithms and methods developed are explained, as well as the details for the demos and use-cases studied.

[2] http://www.unity3d.com

# Chapter 2

# State of the Art

## 2.1 Problem domain

### 2.1.1 Scientific background

The concept of soundscape has been defined several times across different disciplines, but the major impact has its origin in the studies of Raymond Murray Schafer starting in 1969 at Simon Fraser University about echological acoustics [4]. It focuses on the combination of sounds that arise from an immersive environment: a soundscape is an acoustic environment or an environment created by sound. Models to analyze and generate a soundscape have been developed in the last years, and in particullar the work on sound taxonomies by William Gaver helped to develop a framework of terminologies and categorization of sounds [5]. The use of taxonomies helps to understand how to model and shape methods for synthesizing sound objects according to certain physical attributes. As stated in the thesis by Nathaniel Finney in 2009 [6], the variability of physical attributes for resynthesizing sound objects is a growing field of research for improving the interactivity aspect in Virtual Environments (VEs). Also the relationship between the soundscape concept and its importance in media has been noted by Gerard Roma et al. in 2010 within the context of the Freesound project [7].

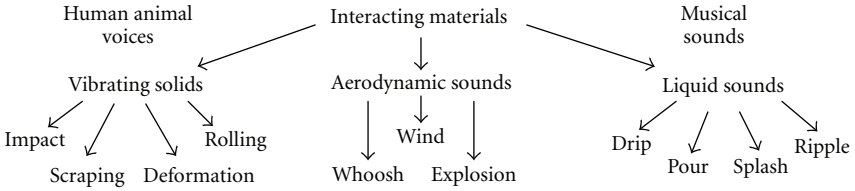


Figure 2.1: William Gaver taxonomy for enviromental sounds

As mentioned in the studies by Linda O'Keeffe, within a real world all the senses are exposed to information, sight, sound, smell and touch. Within a synthetic environment, we are currently exposed to an overriding visual experience and minimal sound information. So, an analysis from a sociological perspective is also needed because sound is an inherent physical concept and we

are always exposed to it, even if we are not focusing our listening to a certain source. This also influences the way a soundscape is modeled, hence the surrounding world is intrusive to its design. Thus, an interdisciplinary approach is needed, for instance, to properly define the relationship between reality and experience. By other side Milena Droumeva also reviews that link in the context of games production defining an acoustic communication framework [8]. The studies by Stefania Serafin and Turchet et al. in walking interactions [9] expose the role of the context in recognition of walking sounds and the nature of interactive contexts.

### 2.1.2 Structure and causality

The studies by Andy Farnell debate the concept of procedural audio and introduce it as "sound as a process, as opposed to sound as a product" [10]. Part of this view is oriented towards a bottom-up approach, taking a scientific analysis of physical phenomena as starting point. More details about the structure and causality about procedural audio will be discussed in the following chapters, taking into account the context of this thesis. A debate in this area is becoming important, due to the appearance of dedicated conferences and workshops like the DAFX 2011[1] on Versatile Sound Models for Interaction in Audio Graphic Virtual Environments.

### 2.1.3 Methods for sound design

The theories and research carried out by Liljedahl et al. in the field [11], reference the importance of methods and frameworks for soundscapes modeling from a scientific perspective. Since the work of a sound designer also comprises the understanding of human perception, aesthetics and semiotics, a scientific approach can highly benefit the quality of the audio, and our perception of what sounds right in various contexts. Taking into account the properties that are part of the sound (for instance, the place in the space), they also claim the lack of generic, well-proven or established design principles. The CLOSED[2] project carried out at IRCAM aimed at defining a parametrizable framework to assist in the sound design, keeping in mind the problems traditionally raised from the sound design process.

Cecile Picard et al. [12] proposed another approach more suitable for content creation, having a focus on both the engineering and scientific fields. They use sound sequences based on onset detection and on models from recordings and similarity measurements between a model and sound grains. Cecile Picard also reflects the importance of physics of sound and sound perception in the context of soundscapes modeling in her PhD thesis [1]. Another approach comes from more industry oriented backgrounds, and the social responsibility of games development by Susan Luckman & Karen Collins [13] , Andrew Quinn [14] and David Sonnenschein [15]. A labeling is carried out based on the nature of the sounds within the context: dynamic, diegetic and non-diegetic.

[1] http://dafx11.ircam.fr/wordpress/?page_id=224
[2] http://www.closed.fr

## 2.2 Audio in games, interactive applications and virtual environments

The "interactive audio" term can be related to several fields that range from the virtual reality (VR) perspective, games, toys, art installations, educational applications or simulation. The strategies for sound rendering depend on the context, and as will stated in the following sections, they vary depending on the scope and even on the budget of the application scenario. Advances in computer graphics and software APIs for simulation now allow realistic physics (PhysX[3], Havok[4]) and animations (Naturalmotion[5]), but there are still technology shortcomings related to audio, because traditionally it has been computationally expensive and relegated as a secondary need in interactive applications. An important fact to mention is the coupling of the animations from a virtual environment, to sound. New trends on procedural animation demand a more flexible approach about sound generation. Moreover, another important fact related to the topic of this thesis, is the flexibility of the tools that are aimed at content authoring by artists and designers. Having, for instance, a proper event distribution (and a data model) that allows flexible parameter controls for the artists is key to make the most of the tools.

The traditional approach for sound in interactive is playing pre-recorded or synthesized samples [1]. On top of that, digital signal processing effects and manipulation are carried out in order to give a sense of space (by means of reverberation or sound propagation models), distance (simulating roll-off and attenuation curves), dynamic range compression, etc. This framework leads to some problems. The first problem is related with memory space, since all the samples have to be stored, although each time more this is not a big problem, yet it is a major issue for the applications running over mobile devices. Another issue is repetition, even though it can be avoided using cheap techniques like pitch-shifting, it doesn't provide good results (it can cause distortion), and depending on the target audio quality, it can become a problem. Physics models, synthesis techniques and DSP algorithms [16] bring out new possibilities to solve the aforementioned problems, but also lead to take a step back and learn how to integrate these new paradigms in certain applications to make them useful or practical.

On the other hand, the use of game technology can benefit the development of other not so related applications, like musical instruments. As an example the project done by Shane Mecklenburger at STEIM in Amsterdam[6], in part uses the Unity 3d game engine. This engine provides a streamlined production and deployment pipelines for games across several platforms. Its free version has been also on the raise from the independent games development scene since 2005, providing more affordable alternatives to major productions technology and game engines, like Unreal. Another related project has been done by Benjamin Vigoda and David Merrill at

[3] http://developer.nvidia.com/object/physx.html
[4] http://www.havok.com
[5] http://www.naturalmotion.com
[6] http://www.steim.org/projectblog/?p=2390

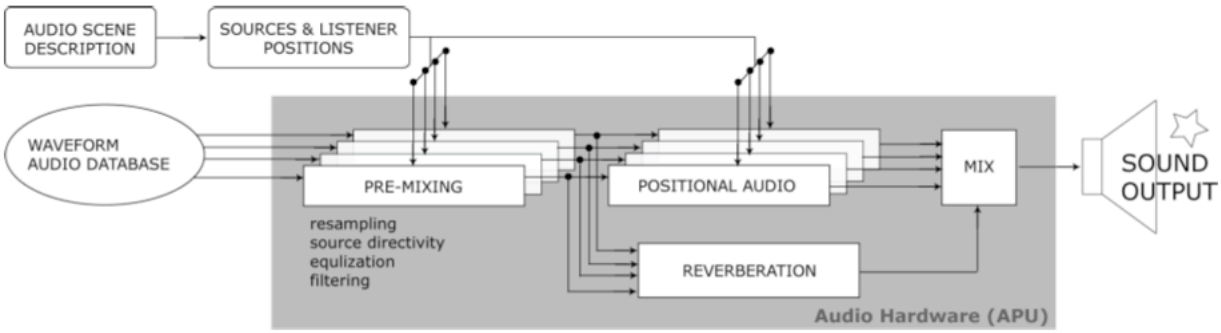


Figure 2.2: Traditional framework for interactive audio [1]

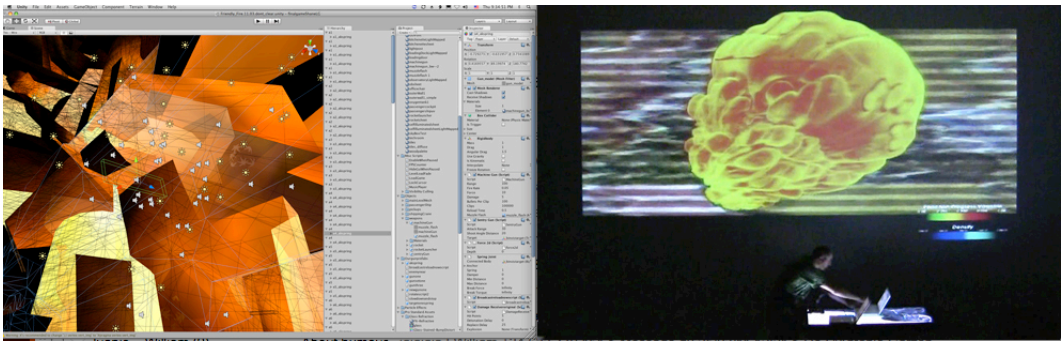

Figure 2.3: STEIM Amsterdam project

the MIT Media lab [17], called JamiOki-PureJoy, bringing out the use of a game engine in a live performance and musicians collaborative context. If talking about a more industry-driven context, most of the technological advances about interactive audio can be followed at annual conferences like Game Developers Conference (San Francisco, US), Audio Engineering Society Audio for Games (recently at London, UK), or Develop conference (Brighton, UK). Part of the material from these conferences can support the explanations in the following chapters from an industrial stand point, encouraging the reader to perceive an overall perspective apart from the purely theoretical and scientific background. The work carried out at the American Association IASIG[7] also brings out some guidelines and encouragement for the upcoming needs and trends in interactive audio education.

### 2.2.1 Soundscape systems, audio engines and tools

There are several available resources for authoring and simulating virtual environments coming from academia, research and industry. UDKOSC[8] is a project from Robert Hamilton at CCRMA in Stanford and expands the Unreal Engine 3 with the OSC pack by Ross Bencina[9].

[7] http://www.iasig.org/
[8] https://ccrma.stanford.edu/wiki/UDKOSC
[9] http://www.rossbencina.com/code/oscpack


Figure 2.4: UDKOSC, Seeking and tracking target projectiles.

The Tapestrea soundscape generation system, from Perry Cook et al. 2006[10], provides a platform for generating soundscapes using an analysis-transformation-resynthesis approach. The Soundscape system by Zach Poff and N.B.Aldrich[11] offers a basic functionality about panning, sources placement, and synthesis with a focus on sound design. The recent book on audio programming (at the time of this writing) from Richard Boulanger et al. also reviews some related topics to audio engine design [18].

If looking at the other side of the barrier, we can find the typical audio engines and middleware products used in the games industry like: FMOD[12] (from Firelight Technologies and included in the Unity 3d toolset), Audiokinetic Wwise[13] (company stated by some former researchers and workers of McGill university in Canada), the Miles Soundsystem[14], BassLib[15] or the open source 3D audio API OpenAL[16]. The American company Valve Software also includes a dedicated system for generating soudscapes in their Source Engine[17]. The French company Genesis Acoustics[18] also has a product for sound rendering in the context of car simulators. Moreover, game developers and educational programmes are mostly using the well-known products, like the above mentioned (UDK, Unity, Wavelab, SoundForge, ProTools, Max/Msp...), middleware (Wwise, Fmod...) and custom in-house tools or frameworks. This chapter also

[10] http://taps.cs.princeton.edu/
[11] http://www.interactivesoundscapes.org/demo_playing.html
[12] http://www.fmod.org
[13] http://www.audiokinetic.com/
[14] http://www.radgametools.com/miles.htm
[15] http://www.un4seen.com/
[16] http://connect.creativelabs.com/openal/default.aspx
[17] http://developer.valvesoftware.com/wiki/Soundscape
[18] http://www.genesis-acoustics.com/en/audio_simu-17.html

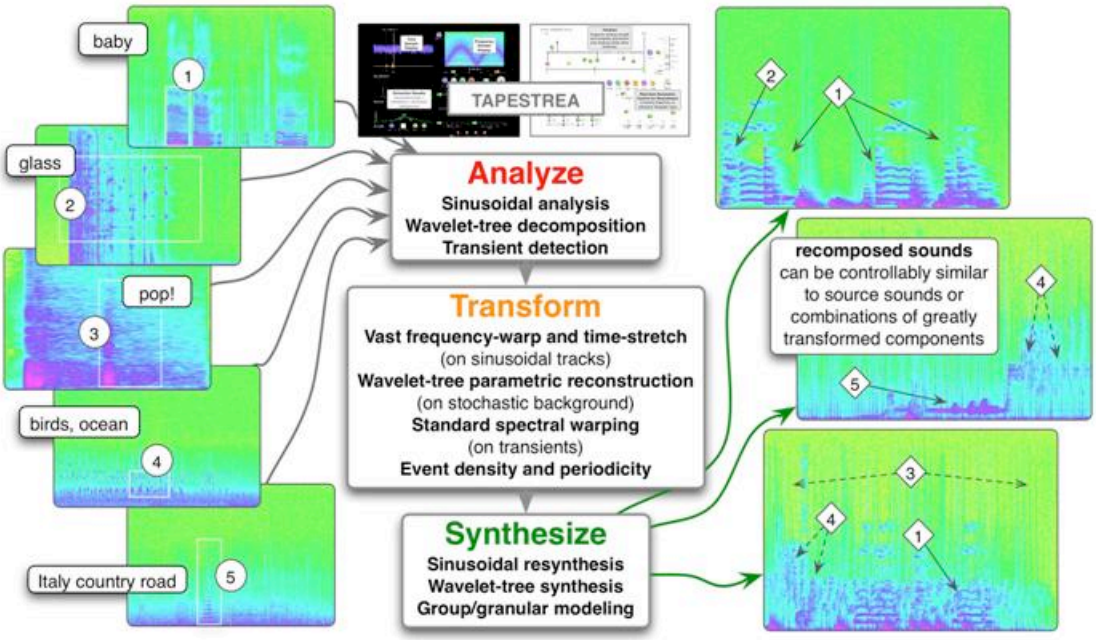


Figure 2.5: Tapestrea soundscapes system

presents a list of available resources coming from independent developers, community contributions, and research centers. For instance, several free and open source frameworks and tools are available from the websites of CCRMA[19] (Stanford, US), CIRMMT[20] (McGill, Canada), Centre for Digital Music, Queen Mary[21] (University of London, UK), Music Technology Group[22] (Universitat Pompeu Fabra, Spain) or IRCAM[23] (Paris, France). Their research output can fulfill current and future game audio technology needs in various fields: i.e. social gaming, audio games, sound synthesis, sound processing, audio analysis or procedural audio. Audio programmers, game designers and sound designers can easily benefit from their resources while prototyping or developing a game.

**DSP Dataflow modeling**

These tools allow quick prototyping and modeling Digital Signal Processing (DSP) algorithms, generating code that can be used within an audio engine or as standalone plugins or applications. Examples that fall in this category: Clam, JSfxGen, Synthedit, SonicFlow, LibPd or Faust.

**Audio and Music programming languages**

There are specific languages with dedicated methods and semantics for audio and music generation. Some of them allow generating code and portions of algorithms that can be

[19] https://ccrma.stanford.edu/
[20] http://www.cirmmt.mcgill.ca/
[21] http://www.elec.qmul.ac.uk/digitalmusic/
[22] http://mtg.upf.edu
[23] http://www.ircam.fr/

used at preproduction or prototyping stages. Examples: CSound-WinXSound, SuperCollider, Chuck.

**RESTful web APIs**

There is a salient amount of web services that allow carrying music analysis, voice synthesis and MIR (Music Information Retrieval) techniques to existent or user generated audio. Examples: Echo Nest, Bmat Ella, Freesound API and more, from the list at Programmable Web.

**Communication Protocol Libraries**

The Open Sound Control protocol and TUIO protocol are widely used to bridge communications between applications and input devices. Libraries that implement these kinds of communication over TCP or UPD are useful to link components within a game engine. Examples: Oscpack, Bespoke OSC, UnityOSC, UDKOSC and TUIO.

**Programming APIs**

Developing the core DSP algorithms of an audio engine can be a though labour, but less painful if some external libraries are used. In this category we can find libraries for analysis, synthesis and audio processing. Examples: Beads Java, STK, FFTW, IRCAM forumfree, JAudio, SOX, Zen Garden, FEApi, libXtract, Camel Framework, rtAudio, IDMIL mapping tools, UGen, rtMidi, OpenFrameworks audio, libsndfile, PortAudio, MPEG-7 Multimedia Software Resources.

**Standalone tools and plugins**

Beyond the typical audio editor functionality, it is useful to extract and visualize characteristics and features from an audio signal, in order to asses the best analysis parameters or salient characteristics that can be exploited in interactive. Examples: IRCAM tools, Smartelectronix, BeatRoot, Wavesurfer, Praat, Sonic Visualizer, Vamp plugins, Octave, SMS tools, FluidSynth or SPEAR.

**Online content creation platforms and tools**

There are online platforms and systems that can be used as audio engines for prototyping and testing new DSP algorithms, sound design techniques, audio transformations or creating new static content. Examples: Marsyas, as3sfxr-b or Tapestrea.

An updated list including links and pointers to these (and additional) resources can be found in this thesis at the Appendix A: Digital Resources.

### 2.2.2 Procedural audio and sound synthesis

The term "Procedural Audio" has been defined at several places across different disciplines. One of the most relevant definitions was declared in 2007 when Andy Farnell published the first articles [19] that served as a preface for his book Designing Sound [10] where he describes the concept from a scientific point of view. Procedural Audio stands for "Sound as a Process, as

opposed to Sound as a Product" and is non-linear, often synthetic sound, created in real time according to a set of programmatic rules and live input. He also labels the term as adaptive, generative, stochastic and algorithmic. Further definitions are also expressed with typical AI and Machine learning techniques (hidden markov models or neural networks) and their relationship with the ancient times of game audio, which was completely procedural, and the demoscene heritage[24].

Additionally, Stefan Bilbao [20] studies tackle the problem and the concept of procedural audio by describing physical phenomena with sets of equations, and also defines their correspondence to traditional synthesis approaches. His book has a special focus on time domain finite difference methods presented within an audio framework. It covers time series and difference operators, and basic tools for the construction and analysis of finite difference schemes, including frequency-domain and energy-based methods, with special attention paid to problems inherent to sound synthesis. The methods from Perry Cook in his book "Real Time sound synthesis for interactive applications" [21] give an overall explanation about the basics of Digital Signal Processing, but also talks about physical models based on tubes, air cavities or membranes. A brief exploration in wavelets, spectral analysis and statistical modeling and estimation is exposed. He also describes some application scenarios in virtual and augmented reality, animation and gaming, and digital foley.

Charles Verron did a broad study on sound rendering and generation of soundscapes for his PhD thesis [22], mentioning the various approaches and techniques that can be used in terms of filtering, FFT, analysis and spatialisation. He also presents a study in different controls and mapping that can be arised from different use-cases and application scenarios. On the other hand, the PhD thesis from Cècile Picard [1] presents a good review of the state-of-art procedural models and synthesis techniques oriented to animation. She also presents some solutions to physically-based contact sounds, and mentions some of the problems from current games production scenarios.

Recent presentations at conferences from Nicolas Fournel [23] express the future and current industry needs in terms of procedural audio. Bottom-up (based in synthesis and physical models) is compared to the top-down approach (based in samples) which sometimes is more appropiate to real-time applications (needing less CPU resources or specialist knowledge). He also remarks the importance of audio analysis in the development of procedural models and high-level audio tools aimed at sound designers working for game studios. The studies and presentations from Leonard J. Paul offer a practical view to procedural Sound Design [24] for video games[25] (having worked in some prototypes using Pure Data, the Source Engine and Unity).

The French company Audio Gaming has been founded by former workers and collaborators

[24] http://en.wikipedia.org/wiki/Demoscene
[25] http://www.videogameaudio.com

of IRCAM. By the date of March 2011 they presented a new product to generate audio in real time dependant on gestures coming from game controllers (like the WiiMote), AudioGesture[26]. The company AudioKinetic[27] offers to game developers a toolset and audio engine to simplify the workflow during pre-production and development periods. This company has links to McGill University in Montreal, Canada. Their product Wwise includes a procedural audio toolset called SoundSeed (for Air-based and Impact-based sounds, at the time of this writing). It uses basic synthesis algorithms and modal synthesis to generate variations from a single (or various samples) in real-time. The German developer PSAI[28] offers a novel gesture-driven adaptive music system as well. Their products seem (as of March 2011) more oriented to the casual games market. Other middleware companies also offer solutions based on dataflow models and synthesis: Gigantic software[29], and Play all[30].

Additional academic and industrial references, articles and videos related to the fields of procedural audio and sound synthesis for video games can be found at the Game Audio Relevance website[31].

## 2.3 Content-based audio transformations

The studies from Amatriain et al. [25], present an overview of the possible frameworks and capabilities that arise from the concept of "Content-based Audio transformations". The importance of gestural control reflects the awareness of the analysis-synthesis process that can be exploited in various contexts, taking also into account the user input.

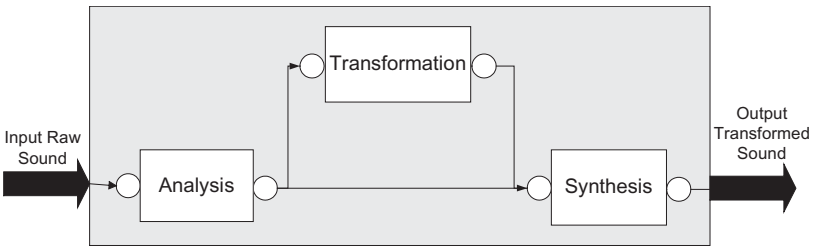


Figure 2.6: Transformation process based on analysis-synthesis model.

When talking about content-based transformations, it is implied that some sort of mapping between low-level parameters and higher level ones is being performed. As also stated in [25], the level of abstraction of the final controls has a lot to do with the profile of the target user. An expert user may require low-level, fine tuning while a naive user will prefer high-level, easy

[26] http://www.audiogaming.net/products/audiogesture
[27] http://www.audiokinetic.com
[28] http://www.homeofpsai.com/
[29] http://www.giganticsoftware.com/
[30] http://www.playall.fr/
[31] http://gameaudiorelevance.iasig.org

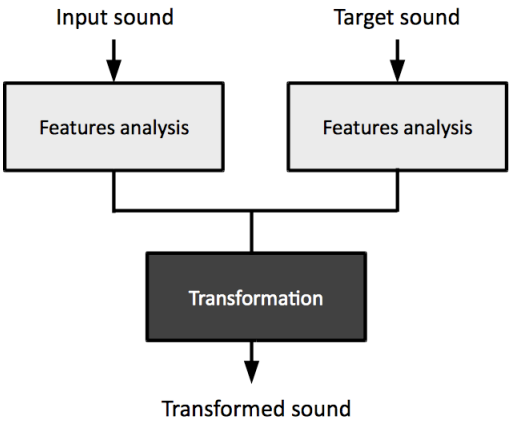


Figure 2.7: General diagram for source-target transformations.

to grasp parameters.

The presentations by Nicolas Fournel in audio analysis [26], present the importance of transformations of samples and the creation of assets models using different methods. Additionally, it also presents the importance of analysis in different use-cases like adaptive-mixing and game audio engines that can be aware and spectrally informed of different content, for instance, using audio features.

The work carried out by Lloyd et al. for the Project Salzburg at Microsoft Research [27] showcases some audio transformations[32] applied to samples. Yet it is a limited approach, they reached good results in terms of bringing timbrical variety and disk space savings for game consoles. They also use a sinusoidal plus residual decomposition [28] and implement this procedural system by means of Wwise plug-ins.

A new category of digital audio effects (DAFX) was defined by Verfaille et al. in 2006 as Adaptive DAFX[29]. As also mentioned by the studies of Amatriain et al., a gestural or user input control is introduced as a time-varying control computed from sound features modified by specific mapping functions. Commonly used effects can affect different characteristics of the input sound, like loudness, time, pitch or timbre. Actually timbre is one of the most interesting characteristics that can be manipulated, and it was defined as the feature that includes the widest category of audio effects: vibrato, chorus, franging, phasing, equalization, spectral envelope modifications, spectral warping... etc.

[32] http://il.youtube.com/watch?v=NgQrV3bszTg

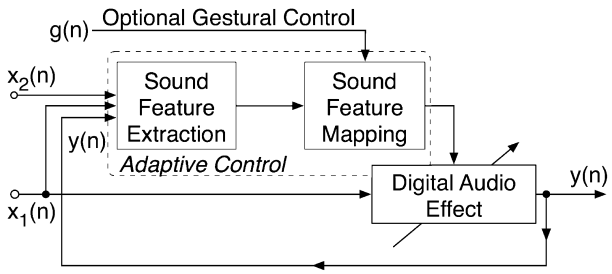


Figure 2.8: Adaptive Digital Audio Effect diagram.

### 2.3.1 Excitation patterns

If talking about feature analysis for filtering and equalization, one of the most popular models of excitation patterns is the Gammatone filter bank originally proposed by Roy Patterson et al. in 1992 described as Equal Rectangular Bandwidths (ERB). Gammatone filters were conceived as a simple fit to experimental observations of the mammalian cochlea, and have a repeated pole structure leading to an impulse response that is the product of a Gamma envelope and a sinusoid. One reason for the popularity of this approach is the availability of an implementation by Malcolm Slaney [30], as a Matlab Toolbox[33]. The auditory filter bandwidths increase with center frequency, and constitute what is known as the upward spread of masking. The upward spread of masking refers to the fact that low frequency sounds are more effective at masking higher frequency sounds.

The work of Sebastian Vega [31] studied part of the LDSP (Loudness Domain Signal Processing) framework introduced by Alan Seefeldt. The LDSP framework consists of transforming the audio into a perceptual respresentation using a psychoacoustic model of loudness perception. In this domain the signal is processed according to some desired transformation (automatic gain control and equalization are some of the possibilities) and then the model is inverted to obtain the processed audio. This is done because, as we know, the auditory system introduces some non-linearities that dictate the way we perceive sound. By processing the audio in a perceptual domain the idea is that we actually perceive what we aimed for when processing the audio, as we are taking the response of the auditory system into account.

The auditory filtering concept is in part analog to the constant-Q transform (Brown and Puckette, 1992), but in that case, the bank of filters is of a constant ratio between the center frequency and bandwidth. With the appropriate configuration, the center frequency of the filters of the constant-Q transform can directly correspond to musical notes.

Another approach is based on the works of Stevens, Volkman and Newmann in 1937 that defined MEL, a perceptual unit based on pitch comparisons. We can compose a bank of triangular-

[33] http://cobweb.ecn.purdue.edu/~malcolm/interval/1998-010/

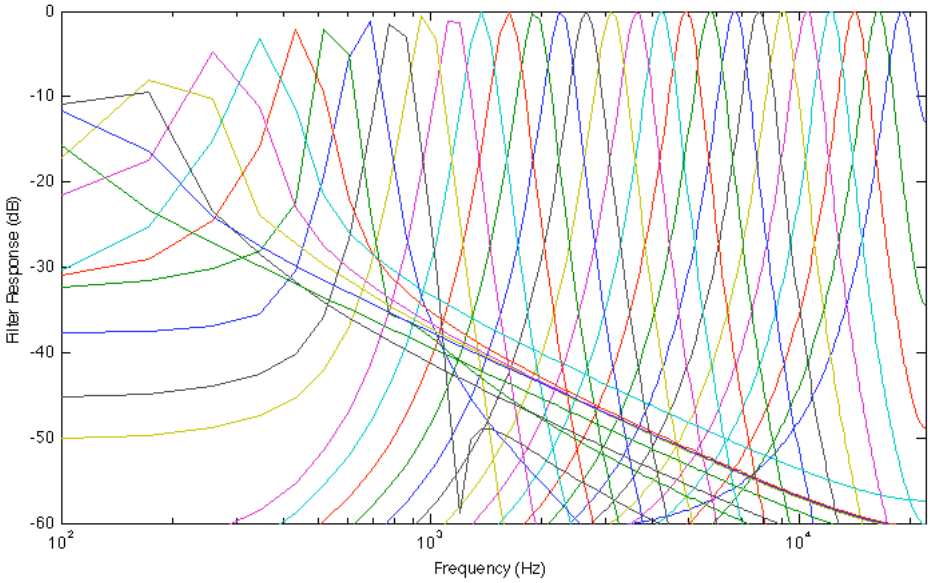


Figure 2.9: Auditory Gammatone filters. Magnitude response for 30 bands, sampling rate of 44100 Hz and low frequency limit set at 50 Hz.

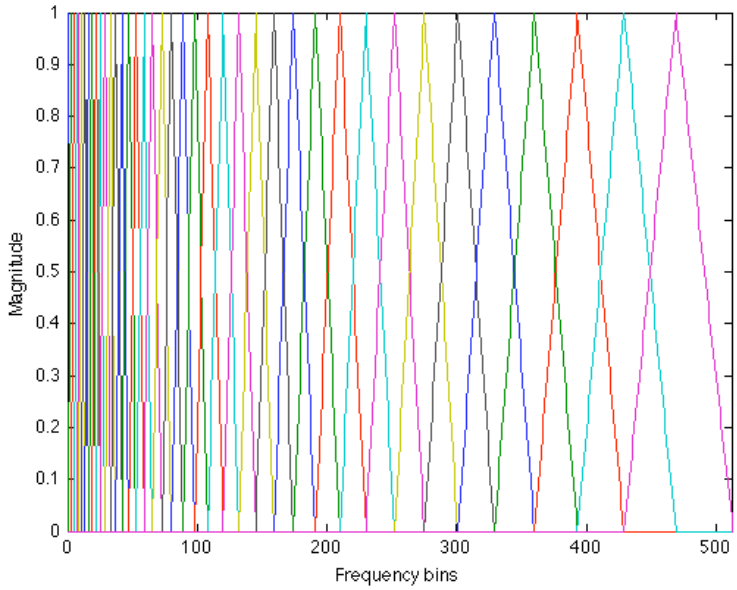


Figure 2.10: MEL-scaled filter bank magnitude response for 40 bands.

shaped filters that symmetrically overlap. If we compare it with the Gammatone filter bank, the influence of each band is only related to its direct neighbors, suggesting that the MEL filters is the better choice when modifying filter band gains in order to keep the changes locally restricted. This approach is also related to the Mel-Frequency-Cepstral-Coefficients (MFCCs), that measure the power spectrum of a sound in the MEL scale and are used in various applications like timbre characterization or audio compression.

## 2.4 Preliminary survey

### 2.4.1 Methodology

Taking advantage of the 41st AES conference in Audio for Games at London that took place between 2nd and 4th February of 2011, we sent a survey to key industry professionals (based on their position, relevance and visibility/contributions in the community) with diferent backgrounds and levels of expertise. We got response from 12 Audio Programmers, Sound Designers, employees of middleware companies and Educators in Europe and US.

The key points for the assessment were focused on the tools used at their departments and to rate the fields that are important in the state of the art of game audio. We also asked about the current needs within companies workfow and about the future trends in the field. The live form can be accessed at http://www.jorgegarciamartin.com/AESevaluation. Details about the layout of the form can be found at Appendix B.

### 2.4.2 Key responses and highlights

#### Results about the middleware and tools used

The major middleware players (Wwise and FMOD) have their presence, but a notable fact is that in-house developments are prevalent. This can be justified to certain game development needs that are tied to the specific workflows and singularities of each project.

The majority of the professionals contacted possess more than 5 years of experience working in games, or related fields. Hence, it can be stated that the responses are not totally biased by certain projects and could be taken as a close image to the realities of the industry. Integration tools, audio analysis, procedural audio and adaptative mixing are the fields rated as the most important to deal nowadays, and in the years to come.

#### Selected responses about searching, editing and annotating samples, and average time of integration

“I think the important thing to note is that you always need to create/edit a sound with prior knowledge of what the implementation method is going to be, otherwise you don’t know how to edit the sound. I don’t mean edit as in ”sound effects editing” so much as the specifics of

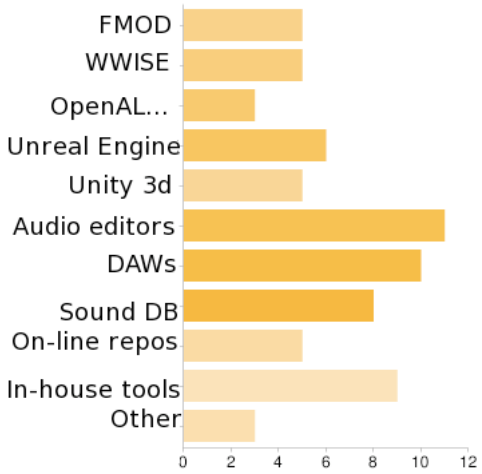


Figure 2.11: Tools survey results

how the sound will work in the game and how you need to create or "edit" the sound to make it fit that playback method, so SFX editing and how you "edit" a sound in to its component parts happen at the same time. (...) That's how I work anyway, I guess some people might use temp sounds and try one thing and then iterate on it, but that's not a good use of your coders time, so I prefer to solve my implementation strategies up front."

"I have found that the context and the way the sound plays back to be quite important. I believe that having a sound that responds dynamically to the game to be more important than the sample itself. After working on a game which uses the same specifications as the original NES and reproducing many different real-world sounds, I have found that approximating the sound can be more than enough for games that don't require a high degree of realism. I tend not to be interested in games that attempt to portray realism too closely as they often fail quite horribly."

"(...) Day to day I'm spending more time exporting sound elements which are used to construct sounds at runtime rather than full sounds. (...) A large amount of time is always going to be spent searching for the right raw sound effects because of the limitations on the number of words that are available to describe sounds. Tools like Basehead and SoundMiner make this job easier and quicker but it still takes time to find the perfect elements."

"(...) Length of integration really depends on the type of sound, the team, the manner of integrating etc. (...) If I wanted to hook up a simple torch it might take me 15 min to search a library and edit/loop/master the file, and then a few min to integrate it into a Scream or FMOD bank, and another few to tag it ingame, and then a few to build it all to check ingame."

"Creating a game sound FX from sample libraries is a task that can require from few minutes to several days. The time spent it's not totally related to the tool used, but in my opinion

it depends from:

- Quality of the library used. With the term "quality" I mean how much the library overall sound is close to the sound you're searching for.
- How much the sound you're creating is important for the game (is it a sound placed on an important gameplay element or it's a common soundfx?).
- Which is the level of "originality" you want to achieve for that sound. Are you searching for something un-heared before? or will it be a sound you've heard several time in several games?
- The time/budget available to complete the game"

"In my role as a technical sound designer, I primarily take content created by a sound designer and implement it in a system created for playback in-game. Ideally the creation of the system for playback requires the most time, and integration is as easy as adding the WAV files to the system, and iterating on the results. The integration time depends heavily on the tool and pipeline, not all processes are created equal. So, I guess one could say that the integration should be minimal if the system in place for playback has been properly set-up."

### Selected responses about future and current needs in game audio engines

"Open DSP frameworks for procedural audio with hybrid capabilities, resynthesis, granular extraction, behavioural audio methods for excitation modes."

"Audio engines which are more aware of what they are playing. Better integration with other game subsystems (animation, physics, etc...) and more real-time control over sound effects."

"From my past experience as lead audio I would say: ease of integration for repetitive tedious tasks, flexibility in terms of dependency with other teams work (animation, textures, ...), and of course procedural audio tools."

"The biggest bonus to a dedicated engine is the ability for sound designers and composers to have some control over the finer details of the implementation. Being able to tweak levels, effects, timing, all of that is what separates a poor sounding game from a great sounding game. Having to pass requests for tweaks back and forth with a programmer just slows things down and ultimately creates a lot of busy work for everyone."

"(...) There's all kinds of DSP and stuff that hasn't been exposed to designers on the tool side, so one thing I would look for is not only what an engine supports but the ease with which its features are accessed. The less work an audio programmer needs to do, the better, and the more willing they are to help when you are trying to do something cool. It would have to be able to efficiently handle lots of cleanly but heavily compressed files. (...) If I needed a lot of

procedural, synthetic, generated style of audio in a game then that would probably be a big deciding factor as well."

"Things I currently dream about for the future of game audio engines: better parametrization of values based on real-time game data, further use of synthesis and procedural, DSP for both modification of existing samples as well as monitoring/analysis, a physical model of reality as it pertains to sound propagation and frequency attenuation over distance, reverberation based on geometric features in addition to the ability to modify properties creatively."

### Selected responses about game audio tools leaks and needs

"Intuitive controls that make repetitive things easy, and difficult things accessible."

"Tools with powerful sample editing features are the best because they can bring "real" elements (samples recorder from real life) to an unreal sound world. So I like any kind of sample modification tools and plugin that can heavily change the structure/harmonical content of a previously recorded material."

"I would look for easy flexibilty within the tool -how easy it is to recycle waveforms or parts of waves into other sound events- repitching, applying different envelopes to change their shape etc."

"The tools must be the same quality as DAWs etc used in other industries. Otherwise I will just use Pro Tools or Logic for the creation of the assets. Once the game audio tool includes a subset of game specific functionality then I will look to it to take over some of my main DAW tasks."

"With tools like Wwise the daily flow has improved a lot, its really the small things inside these tools that could improve your flow a little."

"If the process of mixing and exporting the sound elements from the DAW could be integrated somehow into the Integration tool, then the process of going back and forth between DAW & Tool testing different elements would be avoided and testing sped up. "

"Realtime update of audio data is something which none of the middleware solutions support. It's tricky to solve, but not impossible. You can hack it in to existing solutions by re-starting the audio engine, but it isn't pretty. Anything like this which removes dead-time from a pipeline is a real winner."

"Easier way to make static sounds dynamic e.g. creating procedural models from existing samples."

### 2.4.3 Preliminary conclusions

From the responses received, certain facts can be raised:

- The tools used for editing are basically the standard for sound design in linear media (Soundforge, Wavelab, ProTools...etc), and specifc middleware (Wwise, FMOD) or custom in-house tools build at development studios and platform manufacturers (i.e. gaming consoles).

- Audio engines are not yet very aware of what is happening in a game. There is a need of better code integraton with game data (physics, animations...). The use of audio analysis tools to support the content creation would be very valuable.

- Ease of integraton is very valued. Additionally, there is a need to decrease the integration time.

- In interactve environments the focus is on asset models and sample events, not just samples or recordings.

- Real-time update of audio data would be really valuable, but it is hard to implement. In most of the cases the typical workflow comprises rebooting certain software components to test changes.

- The collaboration between the roles of Audio Programmers and Sound Designers is crucial. Also, defining a proper pipeline and workfow together is key, having always in mind the technical and production constraints of games development.

- The paradigm of sonic realism vs. what is needed: it depends on the game, schedule and budget.

- All participants agreed that a hybrid scenario based in samples, synthesis models and procedural techniques is the path that will be likely to be predominant in the future.

# Chapter 3

# Samples Homogenization

The design of a soundscape presents several challenges to sound designers. For instance, there are various use-cases where certain processing of sound samples could be helpful. When designing a "sound concept" [32], or "sound object" described by a group of sound samples, various approaches can be taken. If using for instance, a set of different microphones and recording gear, combined design techniques (like sampling and synthesis), or sampled material coming from different sound libraries or field recordings: overall we can find a miriad of different timbres and levels.

## 3.1 Analysis and filter parameters estimation

Initially, our efforts are focused on reaching a certain homogenization across samples computing the excitation of a filter bank and then processing all the samples to reach a certain and meaningful transformation point. This approach is also related to the phase vocoder[1], whose early applications were time scale modification and frequency shifting. As expressed in the state of the art, various excitation patterns could be used, but here we are focusing on features analysis coming from the excitation of Gammatone and MEL-spaced filter banks.

### 3.1.1 Filtering gains estimation

After generating the needed filters to implement the analysis and filtering stages, we make use of the RMS value as the analysis feature that represents the gain contribution of a certain band $m$ across a windowed frame of $N$ samples:

$$G_m = \left(\frac{\sum_{i=1}^{N} x_i^2}{N}\right)^{\frac{1}{2}} \tag{3.1}$$

Afterwards we compute the mean for each of the previous excitations across $n$ frames using $k$ filters. A normalization of the mean values is carried out in order to compute a weighting that helps to sort the frames by relevance depending on their energy contribution.

[1] https://ccrma.stanford.edu/~jos/sasp/Phase_Vocoder.html

$$W_n = \frac{mean(Fr_n)}{max(Fr)} \tag{3.2}$$

Thus, the weighted gains across frames are computed as:

$$WG_{m,n} = G_{m,n} \cdot W_n \tag{3.3}$$

And the source gain excitation for a certain audio file is:

$$Gs_m = \frac{\sum_{m=1}^{k} WG_m}{\sum W_m} \tag{3.4}$$

Then we compute the mean of the excitations related to a filter $m$ for all the files in the dataset. We use it as the target excitation we want to reach, namely the "homogenization point". At the end, a scaling factor for each file can be composed as a scaling matrix:

$$S_{m,file} = \frac{Gt_m}{Gs_{m,file}} \tag{3.5}$$

Optionally, a matrix correction can be applied to the scalings, as explained in the next sub-section (Iterative gain estimation). Finally the gain transformations yield from the multiplication of the scaling to the corresponding gain of each filter, that is, the final gain for each filter applied to the homogenized file:

$$Gt_{m,file} = Gains_m \cdot S_{m,file} \tag{3.6}$$

This approach is based on a weighted average across frames analised using the Short-Time Fourier Transform (STFT) as mentioned in [33], hence, an additional audio feature can be raised: frames with more spectral content contribute more to the gain estimation (filter bank excitation) of an audio file. For instance, in short sound samples with a short attack time (impact or strike-like), giving more weight to the bands contribution from the first miliseconds of the sound, would provide a more relevant timbrical representation of the excitation.

Listing 3.1: Homogenization algorithm pseudo-code

```
filesList = dir('sourceDir/*.wav');
filterbank = generateFilterbank(FFTsize, samplerate, numFilters);

for(i = filesList.length; i >= 1; i--)
{
  file = filesList[i].name;
  energies = computeExcitation(file, filterbank);
  numFrames = size(energies);
  framesMean = mean(energies);
  weights = framesMean/max(framesMean);
```

```
  for(j = 1; j <= numFrames;   j++)
    weightenedenergies[j] = energies[j] * weights[j];

  for(k = 1; k <= numFilters; k++)
    excitationProportions[k] = sum(weightenedenergies[k]);

  sourceExcitations[i] = excitationProportions/sum(weights);
}

targetExcitation = mean(sourceExcitations);

for(k = 1; k <= numFilters; k++)
{
  for(i = filesList.length; i >=1 ; i--)
    scalings[k, i] = targetExcitation[k] / sourceExcitations[k, i];
}

for(i = filesList.length; i >= 1; i--)
  computeTransformation(filesList[i].name,
                destinationFile, scalings[:,i], filterbank);
```

---

### 3.1.2 Iterative gain estimation

The overlap of each of the filters in the filter bank (as it happens both for the Gammatone and MEL-spaced configurations), causes a deviation from the computing of the gains. In order to fix this, a gain matrix correction can be applied using a variation of the methods proposed in [31]. Given a gain matrix A that characterizes the overlap across filters, we can correct the gains as:

$$Cg = A^{-1} \cdot Gains \tag{3.7}$$

This method has the issue of computing a correct gain matrix A, which is highly tied to the filter type. Moreover, in the case of the Gammatone filter bank, there is a noticeable complex influence across bands. In order to deal with the filter overlap we designed another approach based on iterative filtering. We observed that filtering iteratively and computing again the scalings, there is a convergence of the scaling values. We defined a threshold of 2% as exit condition. Depending on the sound, it will take more or less iterations to reach the maximum transformation point (ideally, the spectral envelopes of source and target overlap).

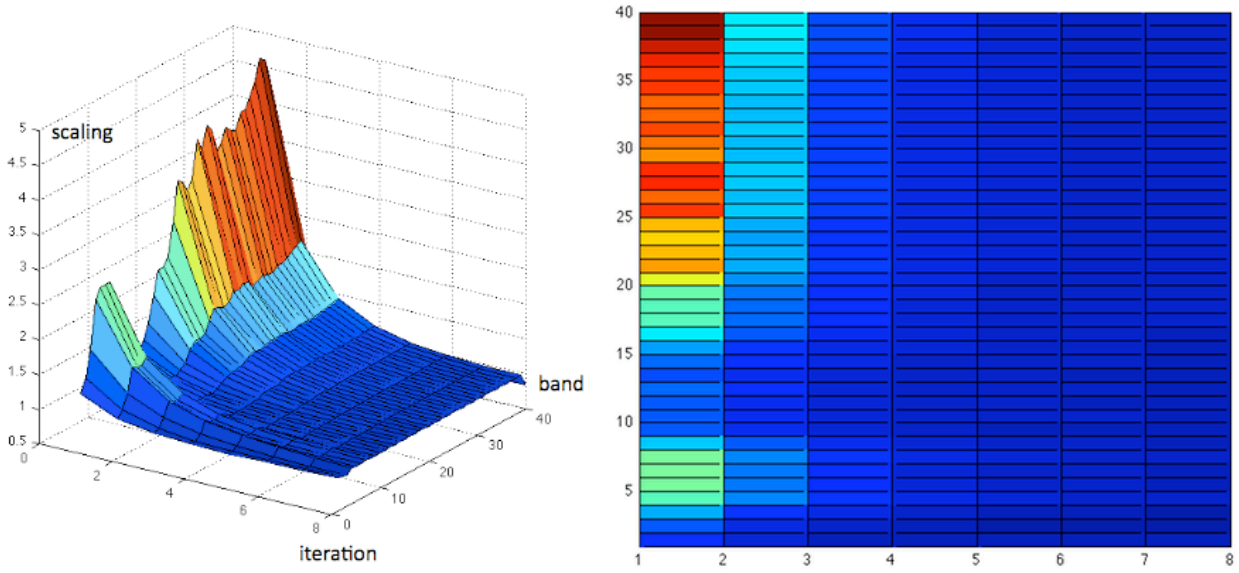


Figure 3.1: Transformation scalings convergence at 2% (8 iterations) over a file.

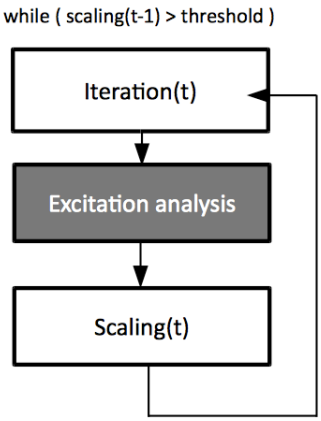


Figure 3.2: Iterative filtering approach, high-level diagram.

The overall scaling to be compared every iteration is the mean of all the scalings minus one. Additionally, we can also measure the error for each of the iterations using a least-squares approximation.

Listing 3.2: Scalings mean computing pseudocode

```
currentMeanScaling = mean(abs(scalings[iteration] - 1));
```

Listing 3.3: Error computing across iterations pseudocode

```
error[iteration] = sum((target - source[iteration - 1])^2);
```

## 3.2 Experiments

Auditory filtering is a powerful tool. In the previous sections we have defined an approach to carry out an homogenization of groups of samples. In order to verify the method, we created different experiments using the Matlab Auditory toolbox [30]. Firstly, we homogenize groups of samples. Then a transformation from source to target is performed within the context of this thesis. Finally, an inverse process to the homogenization is presented.

### 3.2.1 Homogenization

For interactive and immersive environments, the coherence of the different sound sources presented in a soundscape is important. One of the tasks of the sound designer is to select, edit and mix the sounds for a "sound concept". To carry out the homogenization process we used different datasets:

- 6 alarm and siren sounds from the 100 non-speech sounds dataset[2].
- 7 segmented footsteps over ice from the East West Samples Blue Box collection[3]. They are steps recorded without highly noticeable environmental or background ambience. The overall timbre, spectral envelope, and temporal distribution is different across samples.
- 4 female speech phrases retrieved from Freesound[4]. 3 of these files have been recorded using different set-ups at different levels, whilst the remaining is synthesized speech.

Figure 3.3 shows the resulting waveforms and spectrograms for the alarm and siren dataset. We used here the Gammatone filter bank from the auditory toolbox with a standard low-frequency cut at 50Hz and a FFT size of 1024 samples to homogenize the input files one time, and then to apply again the algorithm to the resulting files. By visual inspection we can observe some levels compensation, as well as more timbrical coherence at low frequencies.

For the steps over ice dataset a FFT size of 1024 samples has been used, with a hanning window of 512 samples (11 ms at 44100 Hz) and a hop size of 256 samples, using the MEL-spaced filter bank. After some initial tests, we used 24 MEL-spaced filters to reach a certain compromise between filter overlap and computation time, having also into account the bark-bands number[5]. The resulting homogenization after the 4th pass can be observed in figure 3.4. The homogenization is influenced by the differences in levels of the recordings. We also observed that carrying a preliminary segmentation using onset information or manual editing, the homogenization across the different units (in this case, the isolated steps), improves.

[2] http://www.cse.ohio-state.edu/pnl/corpus/HuCorpus.html
[3] http://www.eastwestsamples.com/details.php?cd_index=36
[4] http://www.freesound.org
[5] https://ccrma.stanford.edu/~jos/bbt/Bark_Frequency_Scale.html

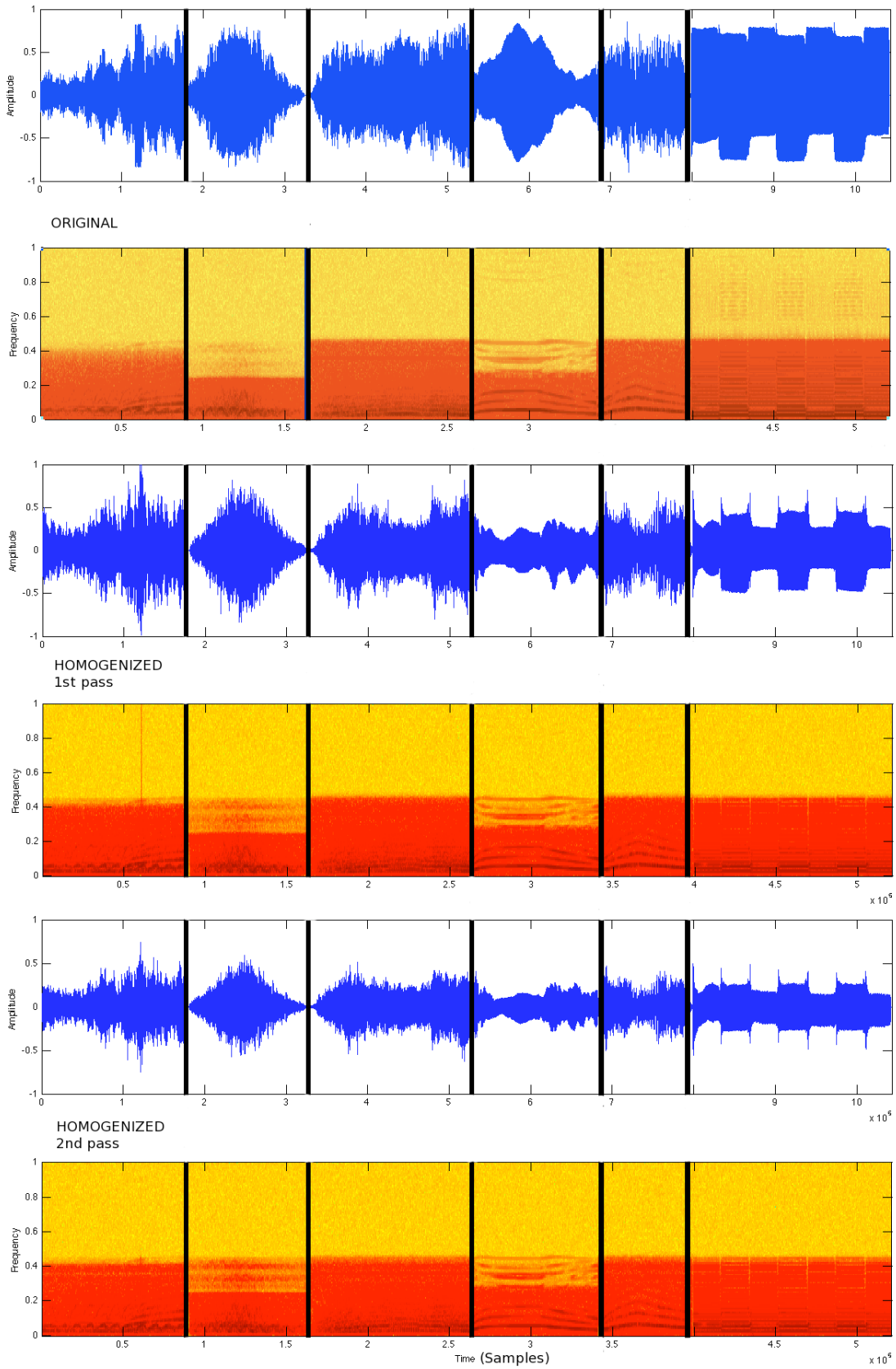


Figure 3.3: Comparison of the homogenization of 6 audio files containing sirens recordings, using a Gammatone filter bank of 30 bands. Applied the algorithm two times (2 passes) over the dataset.

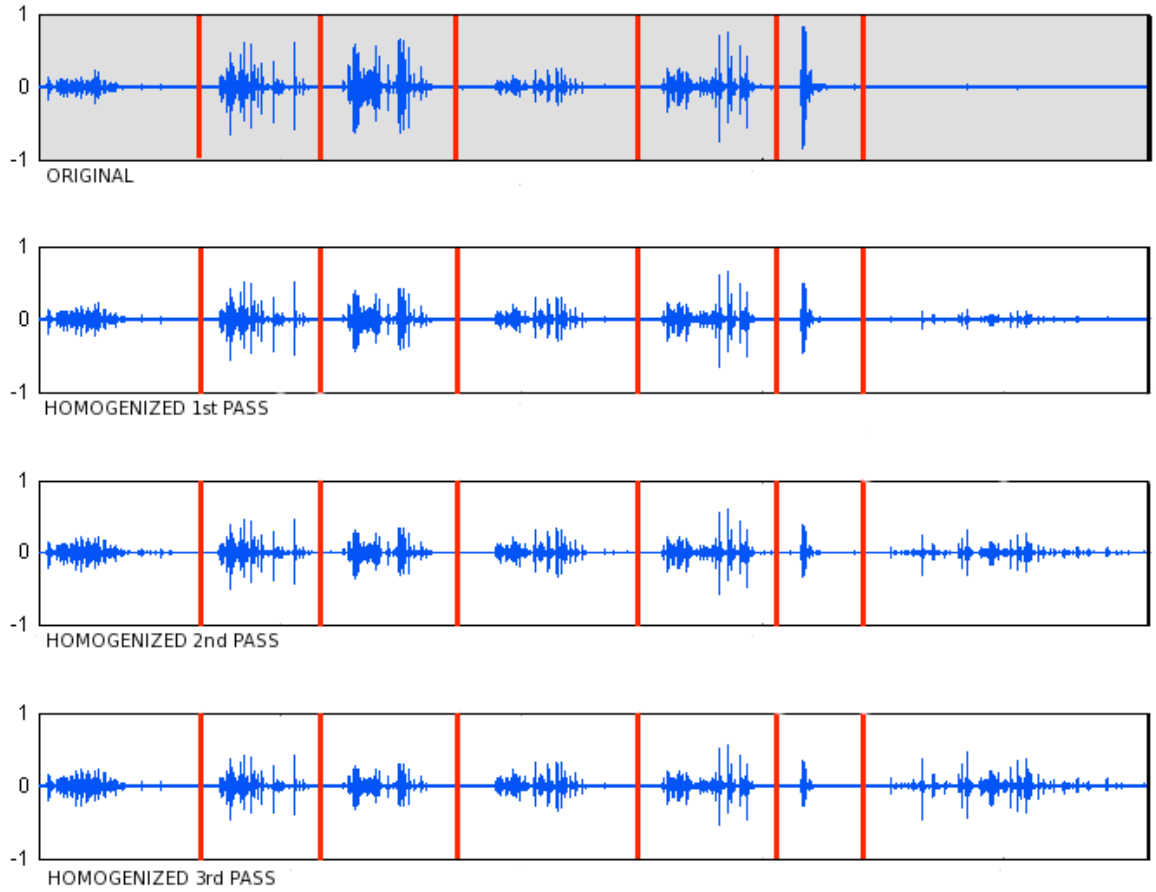


Figure 3.4: Comparison of the homogenization processes (file waveform) for 7 footstep sounds over ice, using a MEL-spaced filter bank of 40 bands. Applied the algorithm three times over the dataset.

The following table shows the RMS values (in dB) after the homogenization computed using the MIR toolbox[6], as well as the standard deviation for the dataset, at each of the passes (algorithm iterations). We can observe a decrease of the standard deviation values.

| File | 1 | 2 | 3 | 4 | 5 | 6 | 7 | st.dev. |
|---|---|---|---|---|---|---|---|---|
| Original | -16.9021 | -12.2844 | -11.2013 | -16.9383 | -12.7417 | -10.1947 | -28.1186 | 6.1622 |
| Homogenized 1st pass | -16.2460 | -12.9748 | -12.9910 | -15.8359 | -13.3277 | -12.8782 | -21.2361 | 3.0698 |
| Homogenized 2nd pass | -15.7970 | -13.4547 | -14.0266 | -15.4472 | -13.7942 | -14.3580 | -17.8939 | 1.5494 |
| Homogenized 3rd pass | -15.5776 | -13.7339 | -14.5819 | -15.3091 | -14.0855 | -15.1389 | -16.2702 | 0.8816 |

As it can be raised from the data analysis above, the algorithm can be applied iteratively until a certain standard deviation threshold is reached (it would depend on the application). The homogenizations of the sound datasets, including the speech recordings, can be retrieved from the digital material placed at the dedicated website for this thesis (please refer to A).

[6] https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox

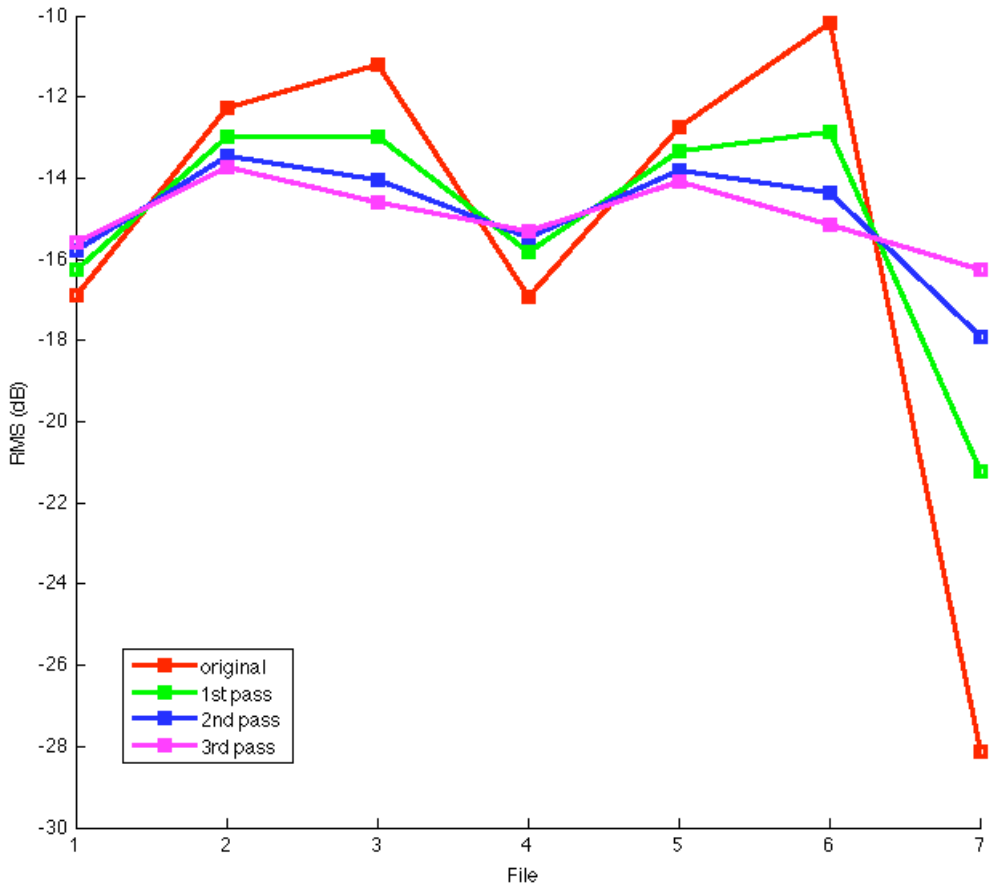


Figure 3.5: Comparison of RMS (dB) values from 7 files of a footsteps dataset before (original) and after applying the homogenization.

### 3.2.2 Source to target transformations

We can also compute one direction transformations from a source to a target filter bank excitation. For this experiment, we carried out a transformation computing the filter excitation of a slow-paced water stream and transforming it to the excitation of a medium water stream (both files also from the Blue Box sound library). We also computed the spectral envelope of the resulting samples to assess variations in the overall spectrum by following some of the techniques presented in [34], like the cepstral envelope[7]. From it, we can observe certain match of the spectral envelopes of the transformed and target sound, when some iterations are applied. In order to compute the cepstral envelope, we need to first carry out a LPC analysis (in this case we used a 24th order linear predictor to compute the coefficients of the filter, with hamming windowing of 600 ms).

[7] https://ccrma.stanford.edu/~jos/sasp/Spectral_Envelope_Cepstral_Windowing.html

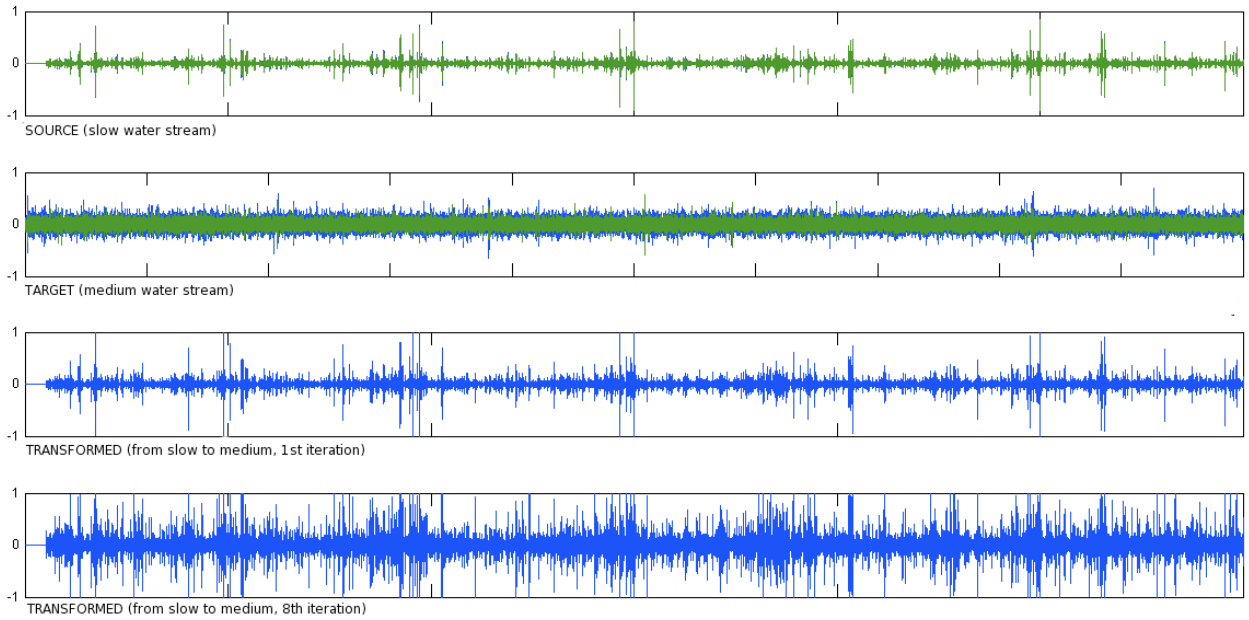


Figure 3.6: From top to bottom: source slow water stream, target medium water stream, transformed from slow to medium (first iteration), transformed from slow to medium (eight iteration).

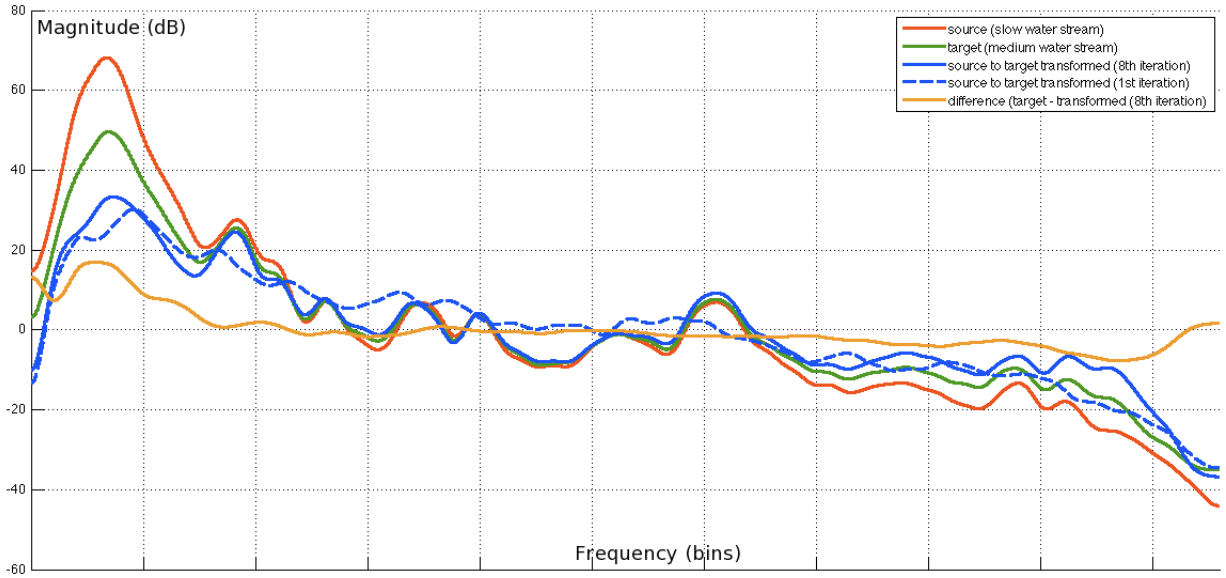


Figure 3.7: Cepstral envelopes comparison for water stream sounds.

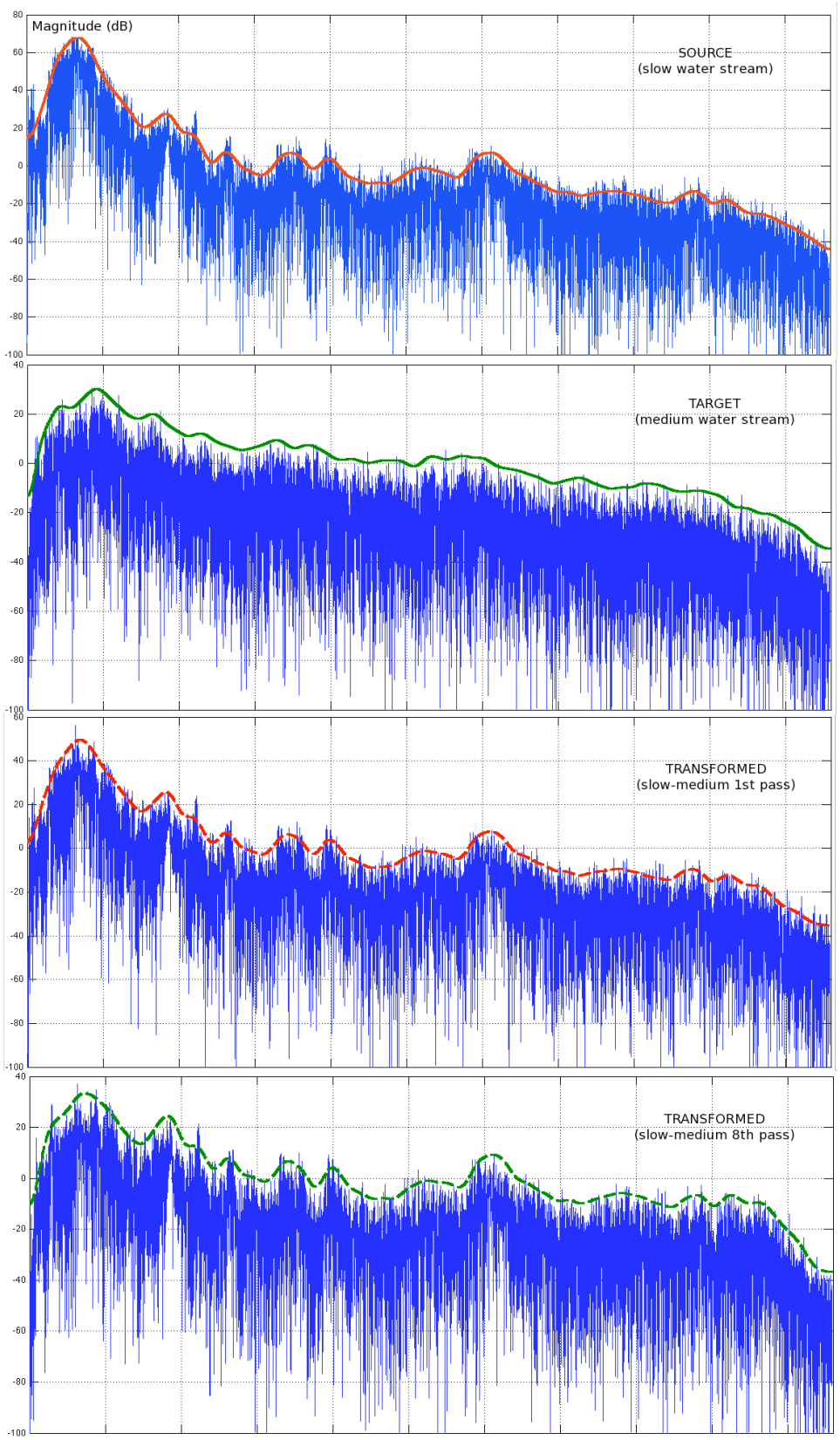


Figure 3.8: FFT spectrum and cepstral envelopes, from top to bottom: source slow water stream, target medium water stream, transformed slow-medium 1st pass, transformed slow-medium 8th pass.

### 3.2.3 Timbre variations

A third transformation can be carried out based on the auditory filtering approach. The aim is to learn from the equalization process, taking the inverse approach to homogenization: generate different timbres from a single source file, namely "variation approach". We took three different files as dataset:

- Snare drum sound.
- Step over ice.
- Keychain shake.

After generating different permutations changing the gain of the most contributing bands excitation, namely -influence bands- and the different amount of transformation (ranging from 0 to 1) an ulterior analysis can be carried out. Our first approach is to compare spectral centroid and spectral rolloff for a footstep sound over ice and for a snare drum sound. The aim is to characterise timbre changes, so we would need to compare with more features. In the plots, the original file scattered in red. We also carried out an MFCC (Mel-frequency cepstral coefficients) computation for some variations. Additionally, we could also compute a cepstral envelope to compare the overall spectrum changes.

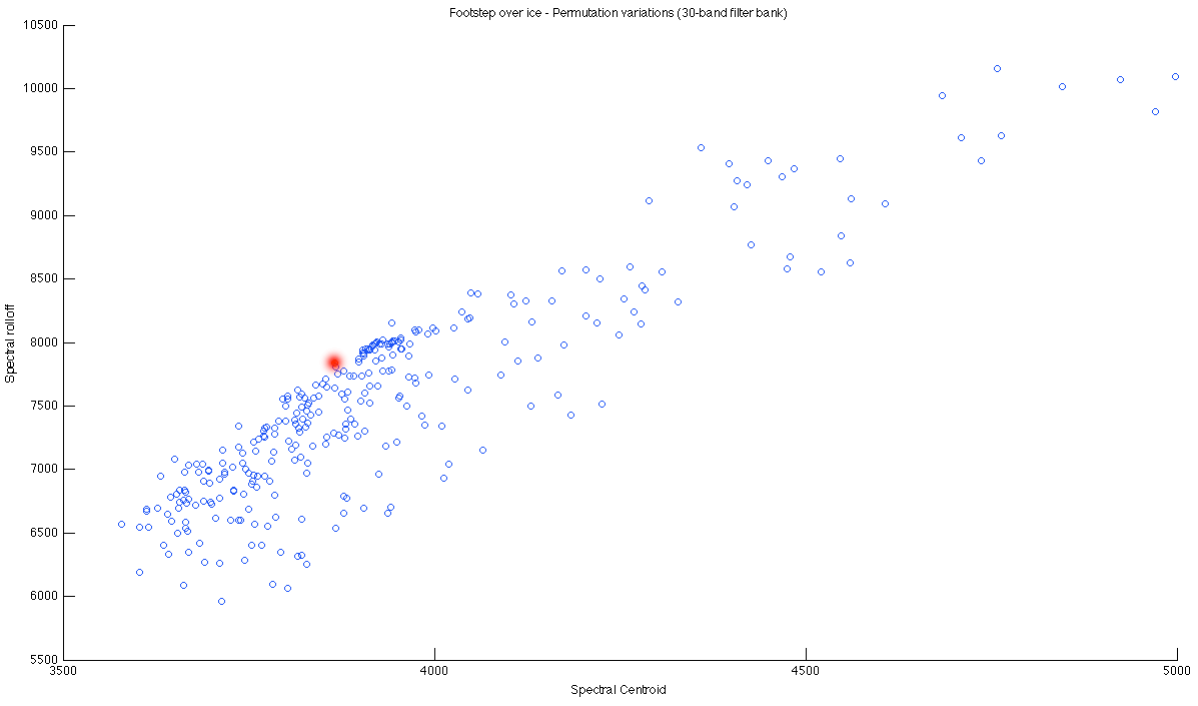


Figure 3.9: Footsteps over ice transformations (spectral centroid vs. spectral roll-off). Original sound scattered in red.

We propose to further evaluate this approach, since these timbre variations would be of advantage in certain scenarios like the development of high-level tools for sound designers, and

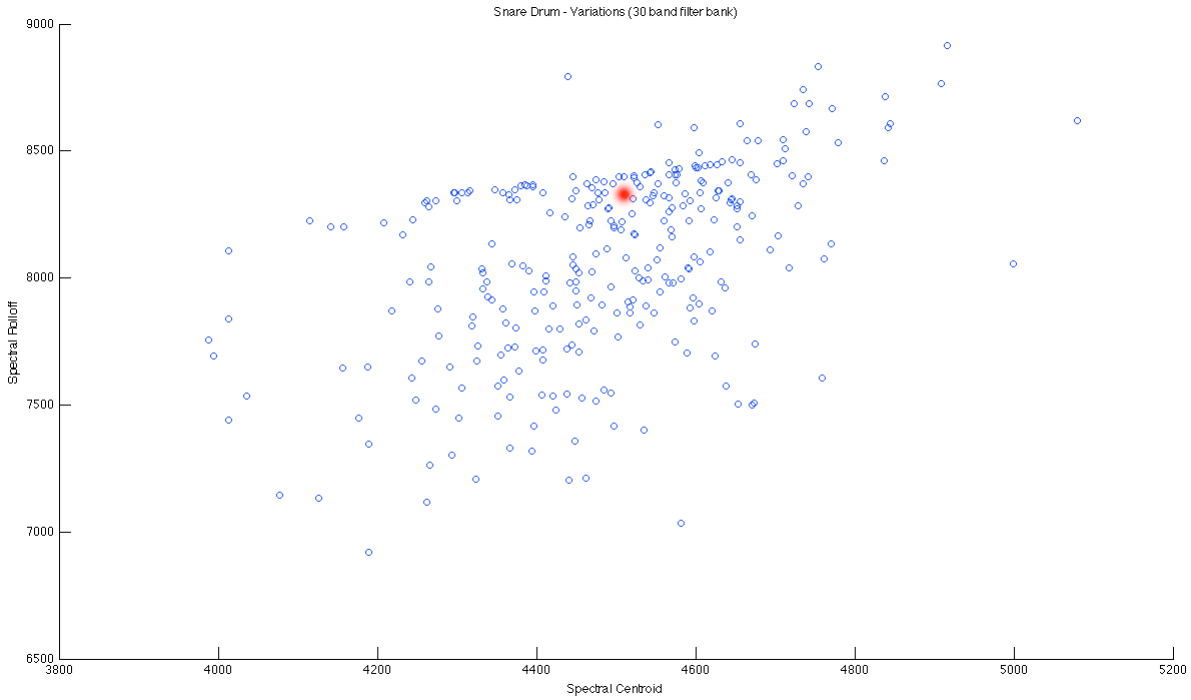


Figure 3.10: Snare drum transformations (spectral centroid vs. spectral roll-off). Original sound scattered in red.

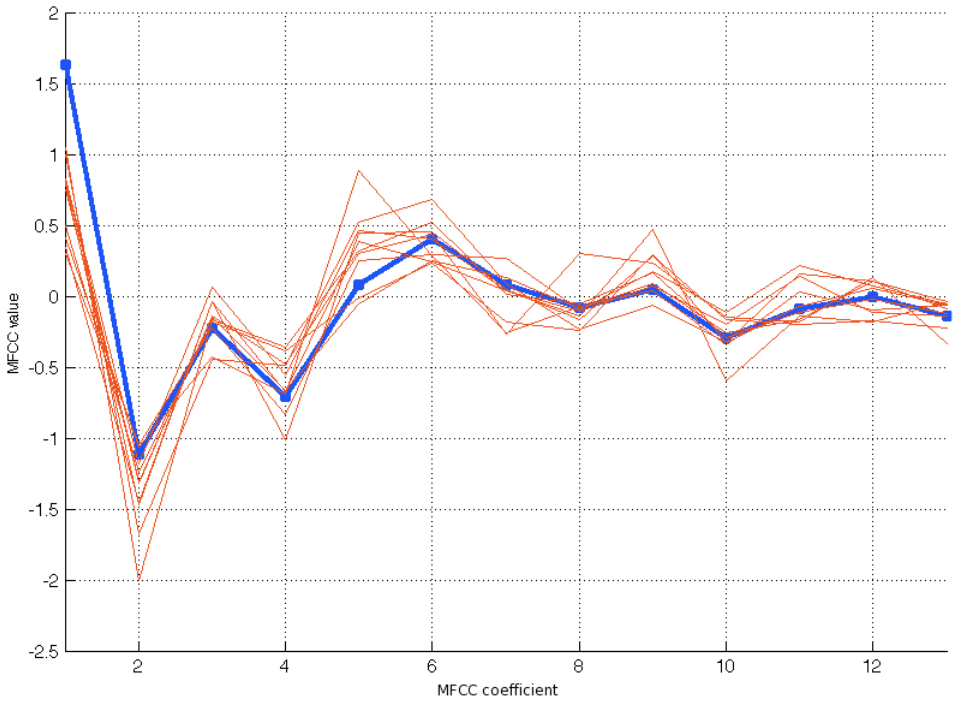


Figure 3.11: MFCC computation for 8 timbre variations (red) across 13 coefficients of an original step over ice sound (plot in blue).

synthesis models equalization. One use-case would be the mapping of the influence bands and amount of transformation parameters to a 2D plane where the sound designer can explore the timbre space. Additional computation like i.e. neural networks (self-organizing maps) would help out to define a more interesting exploration across the timbre space, in combination with additional audio features. In order to understand these experiments better, an additional user interface in Matlab has been developed (please head to the Prototype Implementations chapter for further details).

## 3.3 Evaluation

### 3.3.1 Listening tests

We conducted a listening test with 6 subjects (it can be accessed here[8]), asking them to compare footstep sounds in terms of similarity with regard to the ground material and the recording location. We took recordings of footsteps on three different ground materials, *concrete*, *dirt* and *grass*. The recordings were hand-segmented into individual footsteps, creating a pool of single footsteps for each category.

From this pool we randomly chose three examples for each pair of transformation, e.g. *concrete* to *grass* or *dirt* to *dirt*, yielding 27 examples in total. We then doubled the set of examples by adding for each pair of target sound and processed source sound, the pair of target sound and unprocessed source, in order to have a baseline for the rated similarity. The order of presentation of the 54 sound examples was randomized and each sound was presented four times in a row, with 300 ms of silence between the footsteps. Our subjects were asked to rate the pairs of sounds in terms of their similarity in terms of ground material on a scale from 1 (*completely different*) to 5 (*same material*). In a binary choice they should also decide whether the two sounds have been recorded in the same location or not.

The sample homogenization between source and target was carried out with the following parameters: 512 samples of FFT window length for the filterbank computation and a threshold of 0.02 for the rate of change in the filter gains between iterations, resulting in 6 to 12 iterations, depending on the source and target sounds. Each sound was recorded to one channel, with a sample rate of 44.1 kHz and a resolution of 16 bits.

### 3.3.2 Results and conclusions

For this preliminary test we observed and measured slightly better similarity in material and a significant improvement in sensation of recording location (a factor of 0.75 for transformed and 0.39 for non-transformed sounds), though, the results to this listening test aren't conclusive because the number of participants isn't enough (below 10 participants).

[8] http://www.jorgegarciamartin.com/ListeningTest/page0.html

In order to further complete the evaluation of the methods presented in this thesis, we also present additional evaluation procedures:

- Detailed comparison of MEL and Gammatone filter banks. Carry out metrics of computation times and spectral envelopes for different filter bank configurations (first one would be number of filters).
- Homogenize a group of samples and compare it in an interactive context (i.e. using the prototypes mentioned in the next chapter).
- Sonification of a short film/animated movie using transformed sound samples from the "variation approach".
- Compare the methods presented with the results provided by commercial applications like Wavelab meta-normalizer[9] and DUY Magic spectrum[10].

NOTE: Text portions from this section are a contribution from Stefan Kersten in the published paper for the AudioMostly 2011 conference [35].

[9] http://www.steinberg.net/en/products/wavelab.html
[10] http://www.duystore.com/com/magicspectrum.html

# Chapter 4

# Prototype Implementations

## 4.1 Platform integration

This thesis makes use of the existing MTG's Soundscape Modeling Technology[1]. This generative system [36] aims at simplifying the authoring process, but offering at the same time a realistic and interactive soundscape. A sample-based synthesis algorithm is driven by graph models (see figure below). Sound samples can be retrieved from a user-contributed audio repository. The synthesis engine runs on a server that gets position update messages and the soundscape is delivered to the client application as a web stream. The system provides standard format for soundscape composition. The system includes an authoring module to create the soundscapes, and the actual audio generation engine. All code has been implemented in SuperCollider, and is available under the GNU-GPL license. The system has dependencies on other SuperCollider packages (GeoGraphy, XML).

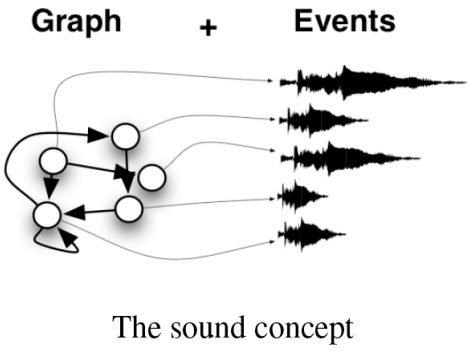


Figure 4.1: Soundscape system graph models and sound concepts

In online virtual environments, the audiovisual content render is typically achieved by a standalone application on the client side. Our interest is here, rather than deploying a large-scale efficient system, to provide a flexible platform for soundscape design and generation that is both application-agnostic and is focused to user accessibility. We intend to foster user-generated

[1] http://mtg.upf.edu/technologies/soundscapes

content, and the web has become commonplace as a collaborative repository of media content. As for the actual audio rendering, in spite of the lower efficiency, a server-side architecture offers advantages, since it does not require any specific software installation by the user.

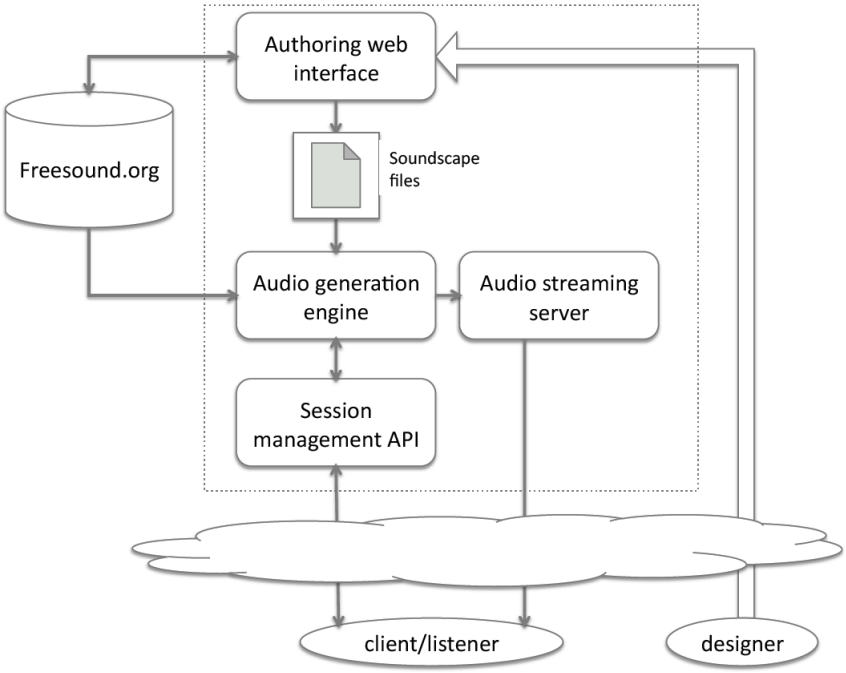


Figure 4.2: Soundscape system architecture

Interaction with the soundscape running in the server is done through a web API. This allows client applications to add listeners and obtain personalized streams given the coordinates of each listener ("position" and "rotation"). A web server implemented a Twisted framework[2] and it provides an HTTP interface for external Internet clients, which translates to OSC (OpenSoundControl [37]) calls for controlling the streaming server. The web server is also responsible for maintaining client sessions. In the current implementation, a fixed pool of streaming URLs is used, and so the number of clients is bounded. Finally, this server module streams the listener output produced by the soundscape generation in the MPEG1 Layer 3 format.

### 4.1.1 Game engine integration

The Unity 3d engine integration with the Soundscape system comprises two modules:

- Communication layer that allows sending and receiving messages from and to the Soundscapes server in Supercollider. This includes an implementation of the OSC protocol in C# mono .NET and some handler and helper classes.
- Exporter that allows serializing data from the game engine in the formats needed for the Soudscapes server: a KML annotation file and a XML sound concepts database file. Additionally, a SuperCollider script is generated in order to boot the server.

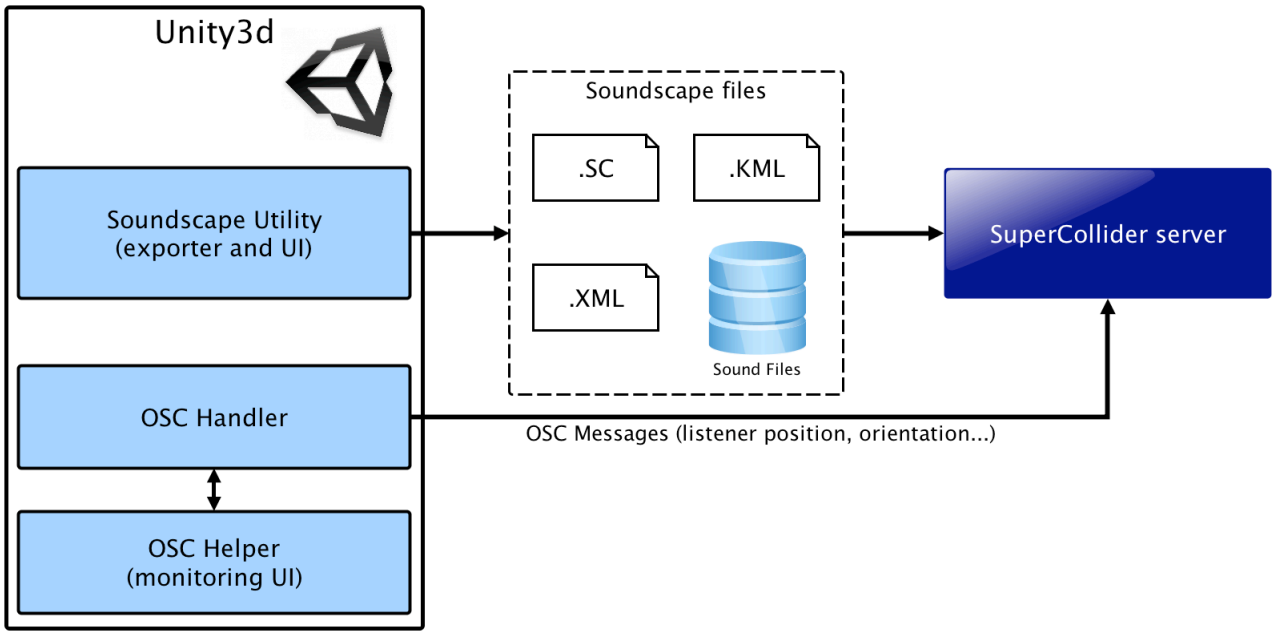


Figure 4.3: Systems integration overview

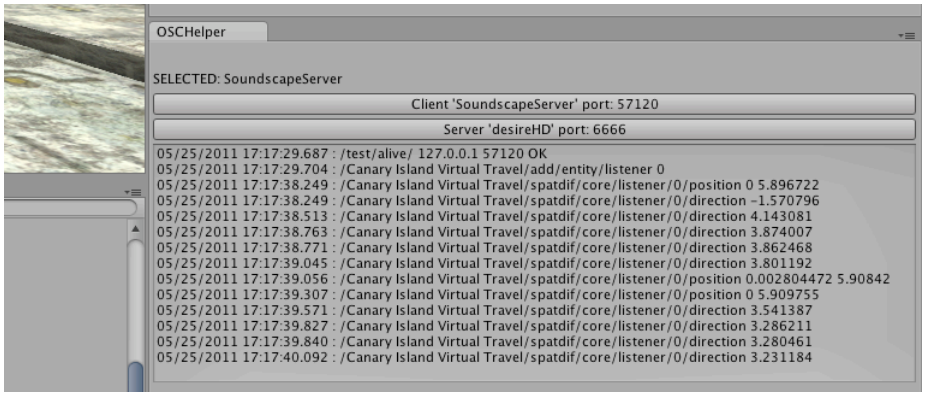


Figure 4.4: OSC Helper UI

The OSC protocol implementation source code can be accessed from the links in the Appendix A. A mapping of the listener position and orientation has been carried out in order to compose the needed messages for the soundscape server (figure above). In order to correctly generate the needed files for the server, the sound designer should follow these steps[3]:

- First, a layer to contain all the related geometry for the soundscape server must be defined. With the current version of Unity (at the time of this writing, 3.3), this can be done at the top-right corner of the editor at the "layers" drop-down menu.
- It is needed to create one Unity GameObject for each of the "sound concepts". It can be an empty GameObject, or for helper purposes, whatever geometry object could be

[2] http://twistedmatrix.com
[3] To understand the process, a prior knowledge of how the Unity game engine works is required

used. In order to serialize the object, it must be included in the layer previously defined (choosing the appropiate one from the GameObject Layer drop-down menu).

- Afterwards, every GameObject for each "sound concept" should contain at least an Audio Listener component. Every Audio Listener component should have assigned an Audio Clip (a sound file, previously imported from the Unity Asset Manager). Every Audio Clip attached to a GameObject will be serialized and packed under a "sound concept" folder.
- If a GameObject is defined as a "sound concept", a ConceptExtension script should be attached to it (drag-and-drop from the Extensions folder in the Unity Project window).
- If a GameObject is defined as a "zone", a ZoneExtension script should be attached to it (same process as previous point).
- Then, all the needed soundscape parameters (defined at the Soundscape server) should be filled in each of the scripts, before exporting.
- Finally, the Soundscape utility (it can be found at Window menu - Soundscape utility) can be used to export all the soundscape data to a desired destination folder, having also to define the soundscape name, width, height, and the layer where all the soundscape GameObjects are placed. Afterwards, pressing "GENERATE KML+XML+SC" starts the export process.

As future work for this system, we propose to implement the DSP homogenization algorithm integrated within the engine (for instance, as a backend process in a Unity script), so the sound designer just has to import the audio files, and then the timbre and level compensation can be automatically carried out by running a process, for each of the "sound concepts". A real-time resynthesis algorithm can also be implemented in SuperCollider in order to generate variations, i.e. from a single footstep sound, as presented the chapter 3.


Figure 4.5: From left to right: Soundscape Utility UI, scene with various sound concepts, same scene with the sound concepts filtered by the soundscape layer.

## 4.2 Matlab prototypes

In order to support the evaluation of the DSP algorithms presented, and to showcase what a high-level tool could offer to a sound designer, a prototype in Matlab has been built using GUIDE[4]. It also supports to carry out parametric homogenizations of samples thanks to a comprehensible User Interface (Figure 4.6). When booting up the application, it allows selecting input, target and output folders:

- Input folder acts as the source folder for the files to be processed (format accepted: mono .wav at 44100 Hz).
- Target folder serves as the target files to fit the homogenization. If selected, files in this folder are analyzed and used as target for the homogenization instead of computing the "homogenization point" (average filter excitation across samples, which is the standard case we defined in the homogenization method).
- Output folder allows selecting the destination path to store the resulting "homogenized" or "transformed" samples (output format: mono .wav at 44100 Hz).

After selecting the chosen configuration, the user can homogenize the input, select the number of filters of the filter bank to be used in the computation, transform the input excitation to the target excitation, select the number of transformation steps to be generated (if target is selected), or select the filter bank model (Gammaton or MEL-spaced). Additionally, it is possible to plot the magnitude response of the filter bank. The status area shows the current file that is being analysed/processed, and alerts the user when the process has been completed. The user can also listen to the files from each folder (PLAY button), and compare the waveform and spectrograms from the input, target or output folders (COMPARE SELECTED button). As future work for this tool, we propose to automatically segment the input files in a pre-process step, so to improve the homogenization.

With the aim to support the evaluation of the auditory filtering approach in the inverse scenario to homogenization, an additional User Interface has been developed (Figure 4.7). It allows loading a single sound file and applying a random equalization across a number of user-defined influence bands (sorted by energy). Also, the amount of transformation (range 0 to 1) can be selected with a slider. Hence, different timbres from a sound can be generated and assesed in the time and frequency domain by means of comparing their spectrogram and waveform. The User Interface also permits to change the input and ouput computation values such as FFT and Hop size.

[4] http://www.mathworks.es/help/techdoc/ref/guide.html

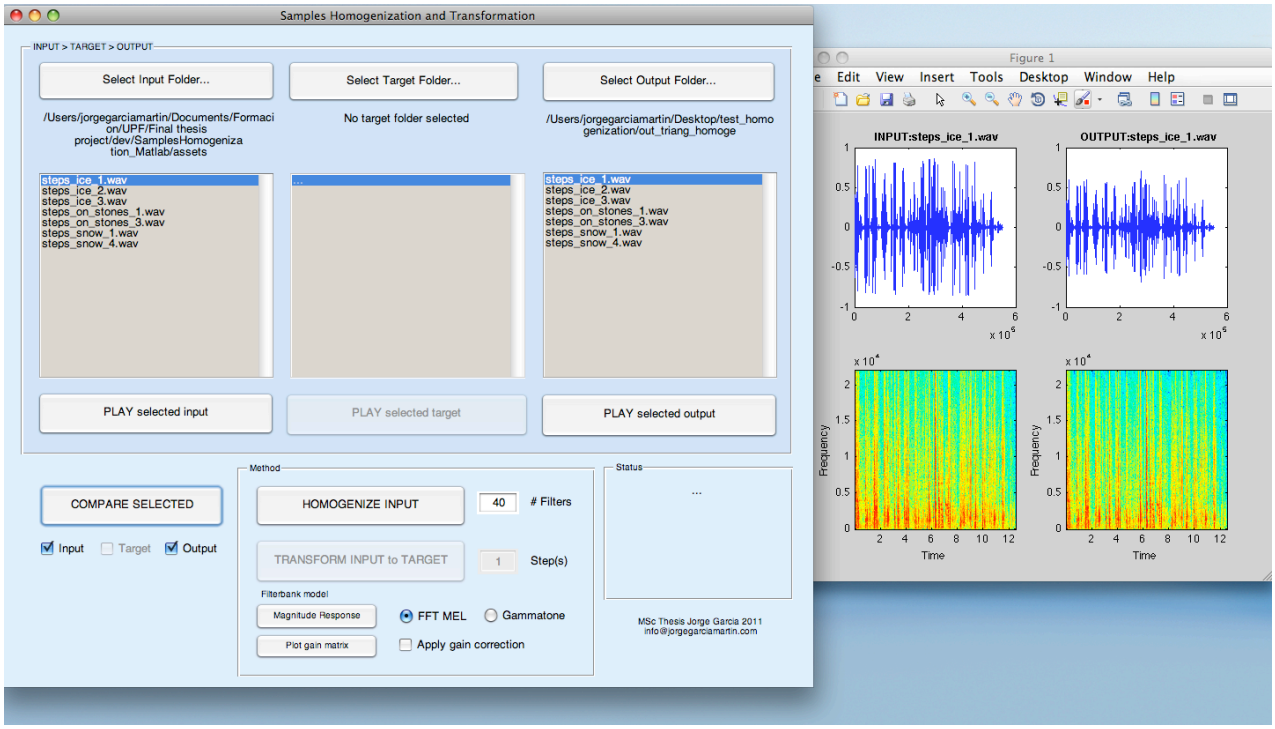


Figure 4.6: Matlab GUIDE prototype for the samples homogenization method.

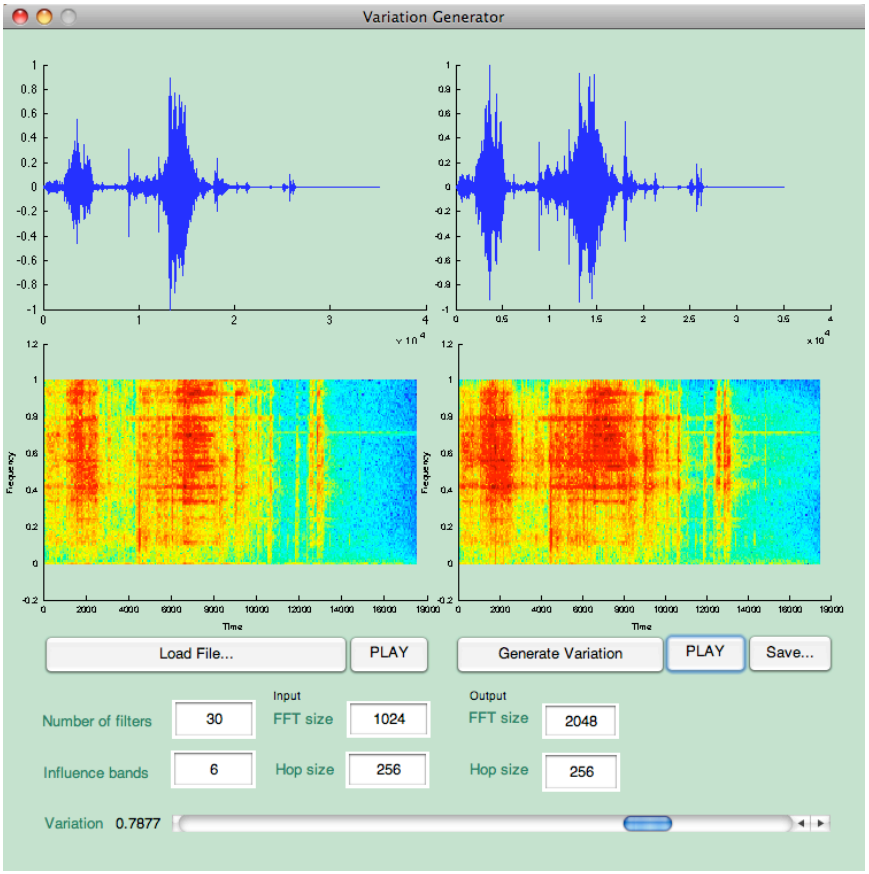


Figure 4.7: Matlab GUIDE prototype for the samples variation approach.

# Chapter 5

# Conclusions

## 5.1 Assesment of the results

The work carried out for this thesis leads to some results based on the auditory filtering approach. Various applications can be arised from the methods presented:

- Adaptive mixing for dynamic soundscapes.
- Combination of samples, "layering" taking into account masking.
- Synthesis models equalization (expanding the timbre space).
- Music productions. Normalization/equalization of studio recordings.

## 5.2 General conclusions

- This thesis is mainly focused on DSP, but the methods presented would highly benefit from i.e. machine learning techniques as statistical modeling, clustering or neural networks (like self-organizing maps or sparse coding).
- Auditory filtering is a powerful tool. It can be used in various application areas, but it is limited by the inherent nature of the filter banks used.
- The transformation paths and direction are important. From the experiments carried out so far, it is needed to asses and learn how to make the transformations because the results are different (i.e. it is not the same transforming steps over snow to steps over ice as over ice to over snow). They depend on the sound and need of adding constraints to be usable.
- The different mappings or semantics depend on the application. Sound designers can benefit from these techniques if proper mapping and/or high-level audio tools are developed at production scenarios (linear or non-linear sound design, which highly relies on the implementation).
- Reutilization of samples or recordings from online repositories like Freesound, could stimulate the usage of these techniques.

## 5.3 Future work

- **Research on additional mapping controls for the transformation algorithms.** It would be tightly related to the application, and depending on the scenario, defined by the sound designer. Maybe some research over the Sonic Interaction Design field could help to develop better frameworks, for instance, to improve sound mappings to synthetic animations.

- **Onset segmentation from field recordings.** It would be useful to segment (isolate) footstep sounds. Different approaches are reviewed by Simon Dixon in [38] and can be combined with additional features analysis (spectral flux... etc).

- **Comparison with Spectral Modeling Synthesis (SMS) and resampling methods.** Evaluation of using SMS transformations [28], changing timbre by moving/scaling partials... etc. Also carrying the homogenization based on resampling [39].

- **Assesment of samples self-simmilarity and similarity measures for transformations.** It could comprise the analysis of transformed sounds, in order to find patterns or fractal components.

- **Learning the different equalizations using Gaussian-Mixture models.** As presented in the conclusions above, we can also use some statistical clustering methods (like the used in speech recognition) to improve the equalization algorithms presented.

- **Algorithms performance measures and optimizations.** If talking about real-time implementations of the algorithms presented, computation metrics would need to be carried out to measure performance and different optimizations depending on the target hardware (need of SIMD vectorization, cache and memory management, power management, scalability, parallelization...).

## 5.4 Summary of contributions

- Carried out a state of the art review contributing to the current liasion between research and development communities. Exploration of the game audio arena from academic and industrial perspectives.

- Researched methods that can support and simplify sound design processes.

- Developed a Matlab tool for sample homogenization, and a library for integrating Unity 3d and an external soundscape generation system.

- Contributed to the AudioMostly 2011 international conference with a co-authored paper [35] that includes part of this thesis output.

# Bibliography


[1] C. Picard-Limpens. *Expressive Sound Synthesis for Animation*. PhD thesis, Universitè de Nice - Sophia Antipolis, 2009.

[2] R. Skovbo. Automatic semi-procedural animation for character locomotion. Master's thesis, Aarhus University, 2009.

[3] E. Mullan. Driving sound synthesis from a physics engine. Technical report, Sonic Arts Research Centre, Queens University Belfast, 2009.

[4] R.M. Schafer. *The new soundscape*. Universal Edition, 1969.

[5] W. Gaver. How do we hear in the world? explorations in ecological acoustics. *Rank Xerox Cambridge EuroPARC and Technische Universiteit Delft*, 1993.

[6] N. Finney and J. Janer. Autonomous generation of soundscapes using unstructured sound databases. Master's thesis, Universitat Pompeu Fabra, 2009.

[7] G. Roma, J. Janer, S. Kersten, M. Schirosa, P. Herrera, and X. Serra. Ecological acoustics perspective for content-based retrieval of environmental sounds. *EURASIP Journal on Audio, Speech, and Music Processing*, pages 1–11, 2010.

[8] M. Grimshaw. *Game Sound Technology and Player Interaction: Concepts and Developments*. Igi Global, City, 2010.

[9] L. Turchet, R. Nordahl, and S. Serafin. Examining the role of context in the recognition of walking sounds. In *Proc. of Sound and Music Computing Conference*, 2010.

[10] A. Farnell. *Designing Sound*. 2010.

[11] M. Liljedahl and J. Fagerlönn. Methods for sound design: a review and implications for research and practice. In *Proceedings of the 5th Audio Mostly Conference: A Conference on Interaction with Sound*, AM '10, pages 2:1–2:8, New York, NY, USA, 2010. ACM.

[12] C. Picard, C. Frisson, J. Vanderdonckt, D. Tardieu, and T. Dutoit. Towards user-friendly audio creation. In *Proceedings of the 5th Audio Mostly Conference: A Conference on Interaction with Sound*, AM '10, pages 21:1–21:4, New York, NY, USA, 2010. ACM.

[13] S. Luckman and K. Collins (ed). From pac-man to pop music: Interactive audio in games and new media. aldershot: Ashgate. 1(1):127–128, 2009.

[14] A. Quinn. What is so special about interactive audio? Technical report, Leeds Metropolitan University, 2010.

[15] D. Sonnenschein. *Sound design: The expressive power of music, voice and sound effects in cinema*. Michael Wiese Productions, 2001.

[16] O. Mayor, J. Bonada, and J. Janer. Audio transformation technologies applied to video games. In *41st AES Conference: Audio for Games*, London, UK, 2011.

[17] B. Vigoda and D. Merrill. Jamioki-purejoy: A game engine and instrument for electronically-mediated musical improvisation. In *Proceedings of the 7th international conference on New interfaces for musical expression*, pages 321–326. ACM, 2007.

[18] R. Boulanguer and V. Lazzarini. *The Audio Programming Book*. MIT Press, 2010.

[19] A. Farnell. An introduction to procedural audio and its application in computer games. *Audio Mostly Conference*, (September), 2007.

[20] S. Bilbao. *Numerical Sound Synthesis*. John Wiley and Sons, 2009.

[21] P. Cook. *Real Sound Synthesis for Interactive applications*. A K Peters/CRC Press, 2002.

[22] C. Verron. *Synthèse immersive de sons d'environnement*. PhD thesis, CNRS-LMA, Marseille, France, 2010.

[23] N. Fournel. Procedural audio for video games: Are we there yet? *Sony Computer Entertainment Europe, Game Developers Conference presentation*, 2010.

[24] L. J. Paul. Procedural sound design. In *GameSoundCon San Francisco*, 2010.

[25] X. Amatriain, J. Bonada, A. Loscos, J. Arcos, and V. Verfaille. Content-based transformations. *Journal of New Music Research*, 32, 2003.

[26] N. Fournel. Audio analysis: The missing link. *Sony Computer Entertainment Europe, Develop Brighton presentation*, 2010.

[27] B. D. Lloyd, N. Raghuvanshi, and N.K. Govindaraju. Sound Synthesis for Impact Sounds in Video Games. *ACM*, 2011.

[28] X. Serra and J. Smith. Spectral modeling synthesis. *International Computer Music Conference*, 1989.

[29] V. Verfaille, U. Zolzer, and D. Arfib. Adaptive digital audio effects (a-DAFx): a new class of sound transformations. *IEEE Transactions on Audio, Speech and Language Processing*, 14(5):1817–1831, September 2006.

[30] M. Slaney. Auditory toolbox. *Apple Computer Company: Apple Technical Report*, 1993.

[31] S. Vega and J. Janer. Content-based processing for masking minimization in multi-track recordings. Master's thesis, Universitat Pompeu Fabra, 2010.

[32] J. Janer, S. Kersten, S. Mattia, and G. Roma. An online platform for interactive soundscapes with user-contributed content. *AES 41st proceedings in Audio for Games*, pages 4–9, 2011.

[33] U. Zolzer. *DAFX: Digital Audio Effects*. John Wiley and Sons LTD, 2002.

[34] A. Robel and X. Rodet. Efficient spectral envelope estimation and its application to pitch shifting and envelope preservation. *Proc of the 8th International Conference on Digital Audio Effects DAFx05*, pages 30–35, 2005.

[35] J. Garcia, S. Kersten, and J. Janer. Towards equalization of environmental sounds using auditory-based features. *Audio Mostly conference, Coimbra (Portugal) 2011.*

[36] M. Schirosa, J. Janer, S. Kersten, and G. Roma. A system for soundscape generation, composition and streaming. In *XVII CIM - Colloquium of Musical Informatics*, Turin, 2010.

[37] A. Schmeder, A. Freed, and D. Wessel. Best practices for open sound control. In *Linux Audio Conference*, Utrecht, NL, 2010.

[38] S. Dixon. Onset detection revisited. In *9th Int. Conference on Digital Audio Effects (DAFx-06), Montreal, Canada,*, 1996.

[39] G. Coleman and J. Bonada. Sound transformation by descriptor using an analytic domain. In *International Conference on Digital Audio Effects*, Espoo, Finland, 2008.

# Appendices

# Appendix A

# Digital Resources

Main website for the material related to this thesis (documentation, slides, audio, videos, links...)
http://www.jorgegarciamartin.com/MSc

UnityOSC, an Open Sound Control protocol implementation and helper classes for Unity 3d (C# mono .NET). Released under the GNU L-GPL license[1].
http://www.github.com/jorgegarcia/unityosc

List of open source resources, libraries, websites and references related to this thesis topic
http://liferetrieval.blogspot.com/2011/07/about-processes.html

Link to the origin of the introduction quote and discussion at O'Reilly blogs
http://blogs.oreilly.com/digitalmedia/2006/10/the-definition-of-insanity-is.html

If you have any questions or comments regarding the material listed above, please feel free to drop an email to info@jorgegarciamartin.com.

[1] http://www.gnu.org/licenses/lgpl.html

# Appendix B

# Preliminary Survey

The following form can be accessed at http://www.jorgegarciamartin.com/AESevaluation

---

(*) Mandatory fields

PRELIMINARY DATA *

- Company / institution name
- Role / position / level
- Years in the industry

QUESTIONNAIRE

Multiple choice: What tools/middleware do you currently use in your job or are used in your department? * If not listed, please write them at the "Other" field

- FMOD
- Wwise
- Other Frameworks/Libraries (Miles, BassLib, OpenAL ...)
- Unreal Engine, UDK
- Unity 3d
- Editors (Wavelab, Soundforge, Audition ...)
- DAWs (Nuendo, ProTools...)
- Sound databases (SoundMiner, BaseHead, Audio Finder ...)
- On-line repositories (freesound.org, soundcloud.com ...)
- In-house tools and/or engine
- Other:

Please rate the fields/techniques that you consider to be key in game audio

development and game engines * e.g. Mark the level of importance you consider (1 the less, 5 the most)

- Procedural Audio
- Adaptative Mixing
- Soundscapes modeling
- Integration tools
- Audio Analysis
- Virtual Instruments / Effects
- Generative music
- Voice synthesis / recognition
- Sound Synthesis (physical, spectral modeling)

Free text-box: In Sound Design tasks, do you consider that a lot of time is spent searching and annotating the appropiate samples? And, if it's your speciality, what is the average time you use to integrate a sample into a game? Do you think it heavily depends on the tool you are using?
* Please respond as detailed as you want

Free text-box: Please describe briefly what you mostly use now or will look after in Game Audio engines * e.g. Samples integration, pipeline issues, assets...

Free text-box: Please describe briefly what you mostly use now or will look after in Game Audio tools. * e.g. Remark what would or what improves your daily workflow

Single choice: Please select the best suitable choice from the following options *

- Synthesis models will be the only one techniques used for authoring audio in interactive media in the future.
- A hybrid scenario based on samples, synthesis models and procedural techniques is the path to follow.
- The sample based approach will continue to be the standard.

---

# Appendix C

# Listening Test

The following test can be accessed at http://www.jorgegarciamartin.com/ListeningTest/page0.html

---

```
Listening test

Please listen carefully to each pair of footstep sound examples and answer
the following questions by filling out the appropriate cells in the accompanying
spreadsheet (see below).

    * How similar are the materials the footsteps are performed on (1: completely
    different - 5: same material)?
    * Have the sounds been recorded in the same location (yes/no)?

Please take note of the number preceding each sound example and fill the
corresponding row in the accompanying form.
```

---