

Procedural 3D Audio for AR Applications

Angeliki Skandalou
John Jeremy Ireland

Supervisor:
Michael Rose



Kongens Lyngby 2017

Abstract

This is the first paragraph

THis is the second

THird paragraph of abstract

Four paragraphs is enough I guess

Acknowledgements

We would like to express our gratitude and appreciation to our supervisor for his support and guidance throughout this thesis work. Several discussion sessions and advice helped us take the most out of this project and make this study possible.

We would like to express special thanks also to the rest of our classmates who did their thesis at the same time under the same supervisor and offered us their advice. Furthermore, we would like to thank **To do** name of the recording guy (1) for providing us with the acoustics room to perform recordings. And last but not least to our family and friends whose support throughout this thesis was invaluable.

List of Figures

2.1	Sinusoidal Additive Synthesis Algorithm [4].	5
2.2	Frequency Response of a Band-pass Filter [1].	6
2.3	Filter-based Modal Synthesis Algorithm [4].	6
4.1	The drumstick used to struck objects during recordings.	11
4.2	The eleven objects used in the thesis.	12

List of Tables

2.1	Acoustic effects of source attributes [6].	4
2.2	Derivation of data used in modal synthesis.	4

Contents

1	Introduction	1
2	Theoretical Background	3
2.1	State-Of-The-Art	3
2.2	How physical attributes affect sound	3
2.3	Modal Analysis	4
2.3.1	Data Extraction	4
2.4	Modal Synthesis	5
2.4.1	Sinusoidal Additive Synthesis	5
2.4.2	Filter-based Modal Synthesis	5
2.5	3D Audio	7
3	Method	9
3.1	Chuck language	9
3.2	PureData	9
3.3	Heavy Compiler	10
3.4	Unity®	10
3.5	Overview	10
4	Measurements	11

4.1	Recordings	11
4.2	User tests	12
4.2.1	Stimuli	12
4.2.2	Procedure	13
4.2.3	Participants	13
5	Implementation	15
5.1	Impact Sounds	15
5.1.1	Sinusoidal Additive Synthesis	15
5.1.2	Filter-based Modal Synthesis	15
5.2	Rolling Sounds	15
5.3	Scratching Sounds	15
5.4	Assignment of Different Materials	15
5.5	Changing the Size	15
5.6	User Interface	16
6	Results & Discussion	17
6.1	Results	17
6.2	Discussion	17
6.2.1	Bugs	17
6.2.2	Which Synthesis Method Is Better?	17
6.2.3	Why our work can be used in VR/AR?	17
6.2.4	Did we manage to achieve what we wanted?	17
6.2.5	How can we improve our work?	17
7	Conclusion	19

A Results of tests to users	21
B User Guide to our product	23
Bibliography	25

Abbreviations

ASW Apparent Source Width.

AVIL Audio Visual Immersion Lab.

BRIR Binaural Room Impulse Response.

CS Compressive Sensing.

DOA Direction of Arrival.

ERB Equivalent Rectangular Band.

HATS Head And Torso Simulator.

HOA Higher Order Ambisonics.

HRTF Head-Related Transfer Function.

IACC Inter-Aural Cross Coherence.

ILD Inter-Aural Level Difference.

ITD Inter-Aural Time Difference.

STFT Short-Time Fourier Transform.

WFS Wave Field Synthesis.

Nomenclature

Ω_{LS} Vector containing directions of Loudspeakers in reproduction.

Ω_L Grid of directions used for the CS algorithm.

Ω_s Subvector of Ω_L containing only the prominent directions after CS processing.

$\check{\mathbf{H}}$ Combined transfer matrix for mixed-norm problem.

$\check{\mathbf{p}}$ Combined measurement pressure vector for mixed-norm problem.

\mathbf{x} Combined amplitude.

ℓ_p Norm-p.

\mathbf{H} Transfer Matrix for plane waves impinging on rigid sphere.

\mathbf{p} Measurement vector for the pressure on the spherical array.

\mathbf{x} Amplitude vector for plane waves impinging on the sphere.

$\tilde{\mathbf{p}}$ Pressure vector reconstructed from prominent plane waves.

B_n^m Ambisonics coefficients.

L Number of plane waves in a discrete grid of directions.

LS Number of Loudspeakers in reproduction.

N Truncation order for the spherical Harmonic Functions.

P_n^m The associated Legendre polynomials of the first kind.

Q Number of sampling points on the spherical microphone array.

R_0 Radius of reproduction area.

Y_n^m Spherical harmonic Functions.

Ω Angular Dependency on both azimuth and inclination angle.

λ Regularization factor for natural field HOA processing.

\mathbf{B}_N Ambisonics coefficients vector truncated at order N.

\mathbf{S} Loudspeaker signals resulting from HOA decoding.

\mathbf{W} Vector containing radial functions W_n .

$\mathbf{Y}_N(\boldsymbol{\Omega}_L)$ Spherical harmonics vector truncated at order N for all measurement angles in vector $\boldsymbol{\Omega}_L$.

\mathbf{p}' Residual pressure.

ε Noise parameter for Compressive Sensing Algorithm.

a Radius of microphone array.

“**w/ Residual**” Exploiting the residual pressure (full implementation of signal path in Figure ??).

“**w/o Residual**” Residual pressure is neglected (only upper path in Figure ??).

Introduction

Immersion and all these stuff that makes our thing good. Why we are doing it and what do we want to give to the community?

Audio in interactive projects like video games and VR/AR applications, plays a significant role for user immersion and realism. Visual and acoustic experiences are interconnected and lacking one of them spoils the whole experience.

The most difficult task is to produce realistic virtual sounds inside the application, difficult to distinguish them from the real ones. This can be achieved not only by playing back a realistic sound, but also by taking care of the environment effects and the context. For example, striking a nail on a board when it still vibrates from the previous struct, produces a different sound that gets added to the previous one [4].

Stuff about 3d sound as well.

Why is our method better than others? (eg wavetable)? And why we think this is the future of the audio in video games?

Stuff about sound in general

Sounds are strongly related to our everyday life and the ways we understand things. Through our experience, we can visualize an event by only hearing the sound it produces (e.g. a car approaching). The vibration caused by the collision of two objects produces sound that depends on the collision force, the duration of the interaction and the changes over time of it, but also on the size, shape, material and texture of the two objects. All these attributes form a unique sound and the sound waves produced from the interaction give the information to the listener [6].

Theoretical Background

Short overview of the theory parts

This is a way to link to explanations Direction of Arrival (DOA)

THis is a todo: **To do** todo test (2)

THis is smth done:

2.1 State-Of-The-Art

Wildes and Richards (1988) defined the angle of internal friction (ϕ), a shape-invariant acoustical parameter that heuristically categorizes sounds into material categories, as

$$\tan(\phi) = \alpha/\pi f, \quad (2.1)$$

where $\alpha = \tau_e$ is the damping coefficient, with τ_e being the time for the vibration amplitude to decay to $1/e$ of its original value after the object is struck and f is the vibration frequency[8].

2.2 How physical attributes affect sound

To do Note smwhere that we're using only vibrating solid objects (not liquids).Gaver[6] pg10 has info (4)

Impact sounds consist of an excitation that dampens over time. Hence, the amplitude of the oscillation depends only on the damping. On the other hand, scraping sounds consist of continuous supply of energy that adds to the amplitude value on top of the damping of the oscillation throughout the object interaction. In addition, the force of the interaction plays the most significant role for the amplitude of the oscillation. The stronger the force, the bigger it imposes the amplitude value to be - and the louder the sound [6].

Furthermore, the material of the interacting objects affects their vibrating oscillation. Damping factor is a material-specific attribute and the bigger it is, the faster the objects lose energy and thus the oscillation lasts shorter. For example, wood has way bigger damping factor than metal and this is why they produce a “thud” and a “ringy” sound respectively.

Moreover, the configuration of the object also affect its vibration. The size of it determines how high or low pitched sound it will produce. More specifically, the smaller the object the more high pitched will be the sound.

Table 2.1 shows the acoustic **information** affected by each physical attribute.

Source	Effects on the Sound Wave
<i>Interaction</i>	
Type	Amplitude, spectrum
Force	Amplitude, bandwidth
<i>Material</i>	
Restoring force	Frequency
Density	Frequency
Damping	Amplitude, frequency
Homogeneity	Amplitude, frequency
<i>Configuration</i>	
Shape	Frequency, spectral pattern
Size	Frequency, bandwidth
Resonating cavities	Spectral pattern
Support	Amplitude, frequency, spectrum

Table 2.1: Acoustic effects of source attributes [6].

2.3 Modal Analysis

In this thesis we are using solid objects that are struck in different ways to produce sound. These ways could be falling on the floor or colliding with another object. The sounds produced can be impact, rolling or scratching sounds. When an object is struck, the forces applied cause deformations to it, emitting sound waves through the vibration of its outer surfaces [14].

Modal analysis studies the response of models under excitation. It uses the 3D model of an object to calculate its modal modes (vibration modes). There are multiple ways to do this, with the most accurate being FEM (Finite Element Method). The objective of FEM is to calculate the natural frequencies of a structure when it vibrates freely.

Another method for modal analysis is the “Example-guided”, where data get extracted using example recordings of the objects being struck. Using a suitable algorithm it is easy to extract features from the recordings such as the fundamental frequency and its harmonics and the frequency peaks of the signal.

2.3.1 Data Extraction

Modal analysis is performed before modal synthesis, to extract the necessary data. Modal synthesis is the sum of damped oscillators each corresponding to a modal frequency, as it will be discussed further below. The data needed for synthesis and their **origin** are shown in the table 2.2.

Symbol	Description	Derivation
A_n	Initial amplitude	Modal analysis
d_n	Damping	Material properties
f_n	Modal frequency	Modal analysis

Table 2.2: Derivation of data used in modal synthesis.

Since every different point being struck produces different deformations on the object, we need matrices of size N (N being the number of struck points of the object). More specifically, we need a vector \mathbf{f} of size \mathbf{N} corresponding to the modal frequencies of every point, a vector \mathbf{d}

of size \mathbf{N} corresponding to the **damping ratios** and a matrix \mathbf{A} of size $\mathbf{N} \times \mathbf{K}$, where K is the number of modal frequencies calculated in one point, which corresponds to the amplitudes of each mode in every point of the object. All the above gives the modal model which can be symbolized as $\mathbf{M} = \{\mathbf{f}, \mathbf{d}, \mathbf{A}\}$ [14].

2.4 Modal Synthesis

In the modal synthesis part, using the data extracted above, we synthesize the struck sound corresponding to the object. There are different ways to synthesize impact sounds, two of them being “Sinusoidal Additive Synthesis” and “Filter-based Modal Synthesis”. The former uses exponential damping and the latter band-pass filters where the damping is the Q-factor of the filter.

2.4.1 Sinusoidal Additive Synthesis

At a struck point k when vibrating in mode n , the impulse response of the model is:

$$y_k = \sum_{n=1}^N A_{nk} e^{-d_n t} \cos(2\pi f_n t) \quad (2.2)$$

if $t > 0$ and $y_k = 0$ if $t \leq 0$ [14].

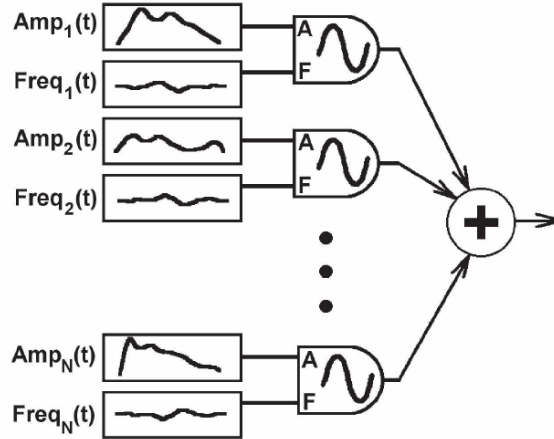


Figure 2.1: Sinusoidal Additive Synthesis Algorithm [4].

2.4.2 Filter-based Modal Synthesis

Band-pass Filters

At this point we will give some basic description of the band-pass filter since it is widely used in this thesis. Band-pass filters (BPFs) take a signal as input and give only a range of it as output, attenuating the rest of the frequencies. This range depends on the central frequency f_c . A filter of this kind is a result of a cascading of a low-pass and a high-pass filter circuit.

The passing range or “band” of frequencies is called **Bandwidth (BW)**. Defining as 0db the resonant peak, we can find the two cut-off frequencies (f_{c_LOWER} and f_{c_HIGHER}) at -3dB. The range between them is the bandwidth (equation 2.3). In figure 2.2 we can see the frequency response of a BPF. [1].

$$BW = f_{c_HIGHER} - f_{c_LOWER} \quad (2.3)$$

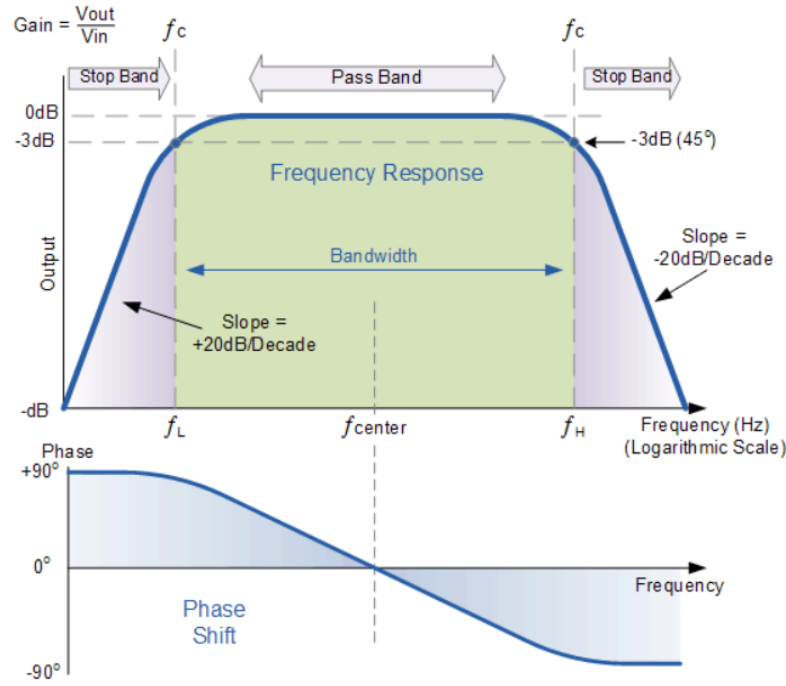


Figure 2.2: Frequency Response of a Band-pass Filter [1].

Synthesis

This method is also additive, since we are adding the outputs of a number of band-pass filters. To synthesize a sound using this method, we use as many filters as the modal frequencies. The filter takes as input an impulse, the center frequency which is the modal frequency and a **Quality factor (Q-factor)** which specifies the bandwidth of the filter. The Q-factor is calculated heuristically, depending on the material of the sound and is inversely proportional to the bandwidth ($Q = f_c/BW$), so the lower the Q-factor, the wider the bandwidth and vice-versa. Hence, more and less frequencies respectively will be included in the **audible** range. We call the above structure a *resonator*, which also includes a multiplication with the corresponding amplitude, taken from the A matrix.

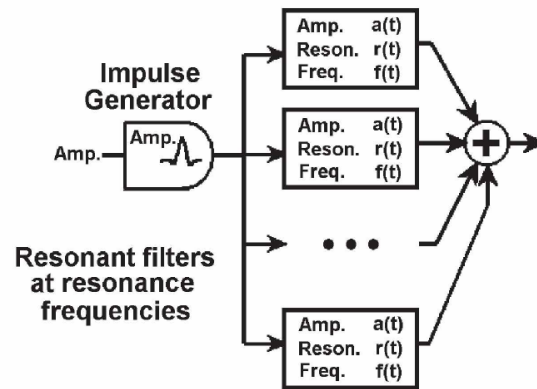


Figure 2.3: Filter-based Modal Synthesis Algorithm [4].

2.5 3D Audio

In VR/AR applications, the location of the incoming sound plays as significant role as the sound itself.

To do maybe name this chapter smth else?? (5)

A combination of the methods described in Chapter 2 is proposed in the present study.

3.1 Chuck language

Chuck language is a music programming language, made for “real-time sound synthesis and music creation” as mentioned in their website [7]. It’s biggest advantage is the way it manipulates time. More specifically, the user specifies how long a sound will last, independent of other sounds that may play at the same time.

Modal features extraction code

We used the ChuckK language at the starting point of our thesis to identify and extract the peaks of the recorded “wav” files. The algorithm used in this part of the thesis is made by Perry Cook for the course “**Physics-Based Sound Synthesis for Games and Interactive Systems**” held by *Perry Cook* and *Julius O. Smith* at **Kadenze Academy** [11].

First, we define *ten* to be the total number of peaks identified. After some trials, we found out that *five* peaks are already enough. However, the authors recommend to use twice the sufficient number, hence *ten* for our case. Afterwards, the algorithm having taken a recording as input, computes its histogram and identifies its peaks. The frequencies where peaks occur are the modal frequencies. Finally, the algorithm finds the maximum value of the signal on each peak, calculating the corresponding amplitude of each mode.

3.2 PureData

Pure Data (pd) is another music programming language. It is open source and the main difference from ChuckK language is that pd is a visual or “patcher” programming language, using objects instead of code, linked together to form a sequence [10].

Re-synthesis patch

For reasons that will be explained below, we had to attach only one pd patch to every 3D object in the demo. Therefore, all synthesized sounds (impact, rolling and scratching) are being synthesized under one main pd patch.

The main part of the re-synthesis patch is the resonator. Using band-pass filters it gives the impulse response of an object struck at a specified point. It also clips the signal to give more brightness and harmonics. To do develop more (6)

3.3 Heavy Compiler

Heavy is a compiler that generates audio plugins from pd patches in interactive sound and music applications [5]. In this thesis we used it to compile pd patches into Unity audio plugins. Heavy's interface is their website where users upload patches and then are able to download the corresponding plugins and put them into their applications. The plugins we used consist of DLL files and a C# (Unity code) script that allows communication of the plugins with the rest of the script and also enables the sound card to play sounds.

3.4 Unity®

3.5 Overview

Measurements

4.1 Recordings

As mentioned above, we used a sample recording of a wooden stick hitting every different area on the objects, to extract the necessary data. We separated the objects into areas depending on their shape and tried to keep the struck sound as uniform as possible inside the same area.

As the hitting object we used a drumstick shown in the figure 4.1. All recordings took place in the same room, using the same drumstick and the same hitting force. The only parameter changing every time are the point of the object being struck.



Figure 4.1: The drumstick used to struck objects during recordings.

We used eleven different objects of everyday life. The idea of choosing those objects came firstly from the need of both owning the object (to perform the recording) and also being able to model it for the demo (simple enough objects). Secondly, since we wanted to test the immersion of the synthesized sounds on users, we wanted to be sure that the sounds used are familiar to them. In figure 4.2 both real and modeled objects are pictured.



(a) Real objects.



(b) 3D models of the objects.

Figure 4.2: The eleven objects used in the thesis.

For the recordings we used a **To do** describe the device used for recordings (7). For the 3D models we used Maya Autodesk software [2] and exported them as FBX® files [3], a file format recognizable by Unity® software [13].

4.2 User tests

To examine the immersion of real-time produced and physics-based sounds on game players, we performed some *MUSHRA* tests to people. The stimuli of the test was both a recorded sound of a struck object and its corresponding synthesized sound using both sinusoidal and filter-based synthesis.

4.2.1 Stimuli

In the first phase of the test, participants were provided with a reference sound (the actual recording) and two testing sounds (the sinusoidal and filter-based synthesized sounds), and were asked to choose the one sounding closer to the reference. Afterwards, they heard only synthesized sounds and were asked to choose the one with the best sound quality. Lastly, we generated different stimulus giving the participants sounds from the same object but with different materials assigned to it and were asked to point out when material changes.

To do generate different stimulus sets, varying the size (8)

4.2.2 Procedure

Stimuli was presented to the participants through To do describe headphones (9), in a room with reduced external noise.

4.2.3 Participants

To do number of participants, age, gender, normal hearing, job (10)

Implementation

To do Here we can put pictures and codes snippets (11)

Instead of having a huge amount of different pre-recorded sound files, which need a lot of space, we eliminate the problem to a **2D look up**, where the algorithm matches the object first and then the exact point of the object that collided with some other object.

5.1 Impact Sounds

To do describe the patch, describe the sound (starts low, goes to max ampl and then decays etc) (12)

To do Put spectrogram pictures of the sounds to describe them (13)

5.1.1 Sinusoidal Additive Synthesis

5.1.2 Filter-based Modal Synthesis

5.2 Rolling Sounds

5.3 Scratching Sounds

5.4 Assignment of Different Materials

Different materials can be assigned to the objects made for this thesis. The designer is able to choose between *plastic*, *wooden*, *ceramic*, *glass* and *metal* by adjusting a slider on the interface.

Metallic or glass sounds are more “ringy” than wooden or plastic ones that are more “thud”. We achieve those sounds by changing the **Q-factor** of the **band-pass filters** used in the pd patch. Q-factor indicated the power loss in the filter. The higher the Q the less power is lost, so the resonator vibrates longer [15].

5.5 Changing the Size

In an application, the same object can appear in different sizes, so this thesis takes this into account. It is known that under the same excitation, the smaller the size of an object, the more high pitched sounds it will produce, because the sound waves travel a smaller distance.

Hence, we implemented a slider for the designer to choose the best sound that corresponds to the size of her object.

5.6 User Interface

Results & Discussion

6.1 Results

6.2 Discussion

6.2.1 Bugs

- you have to press apply on prefabs
- the procedure of putting the freq and ampl data in
- it is object specific
- lack of acoustical richness that might characterize synthetic signals [8]

6.2.2 Which Synthesis Method Is Better?

6.2.3 Why our work can be used in VR/AR?

6.2.4 Did we manage to achieve what we wanted?

6.2.5 How can we improve our work?

- take into account the environment (reverberation etc)
- make objects destructible

CHAPTER 7

Conclusion

This is the conclusion
4-5 paragraph approx

APPENDIX A

Results of tests to users

APPENDIX B

User Guide to our product

Bibliography

- [1] AspenCore, Inc. Passive band pass filter. http://www.electronics-tutorials.ws/filter/filter_4.html, Accessed: April 28, 2017.
- [2] Autodesk, Inc. Maya autodesk (version 2016). <https://www.autodesk.com/products/maya/overview>, Accessed: April 20, 2017.
- [3] Autodesk, Inc. Fbx file format. <https://www.autodesk.com/products/fbx/overview>, Accessed: May 2, 2017.
- [4] Perry R. Cook. *Real Sound Synthesis for Interactive Applications*. A. K. Peters, Ltd., Natick, MA, USA, 2002.
- [5] Enzier Audio, Ltd. Heavy (version r2017.02). <https://enzienaudio.com/>, Accessed: May 2, 2017.
- [6] William W Gaver. What in the world do we hear?: An ecological approach to auditory event perception. *Ecological psychology*, 5(1):1–29, 1993.
- [7] Ge Wang. Chuck programming language. <http://chuck.cs.princeton.edu/>, Accessed: May 2, 2017.
- [8] Bruno L Giordano and Stephen McAdams. Material identification of real impact sounds: Effects of size variation in steel, glass, wood, and plexiglass plates. *The Journal of the Acoustical Society of America*, 119(2):1171–1181, 2006.
- [9] D Brandon Lloyd, Nikunj Raghuvanshi, and Naga K Govindaraju. Sound synthesis for impact sounds in video games. In *Symposium on Interactive 3D Graphics and Games*, pages PAGE–7. ACM, 2011.
- [10] Miller Puckette. Pure data programming language. <http://puredata.info/>, Accessed: May 2, 2017.
- [11] P. Cook, J. Smith. Physics-based sound synthesis for games and interactive systems. <https://www.kadenze.com/courses/physics-based-sound-synthesis-for-games-and-interactive-systems-iv>, Accessed: May 2, 2017.
- [12] Zhimin Ren, Hengchin Yeh, and Ming C Lin. Example-guided physically based modal sound synthesis. *ACM Transactions on Graphics (TOG)*, 32(1):1, 2013.
- [13] Unity Technologies. Unity® software (version 5.5.1f1 personal). <https://unity3d.com/>, Accessed: May 2, 2017.

- [14] Kees Van Den Doel, Paul G Kry, and Dinesh K Pai. Foleyautomatic: physically-based sound effects for interactive simulation and animation. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 537–544. ACM, 2001.
- [15] Wikipedia. Q factor. https://en.wikipedia.org/wiki/Q_factor, Accessed: May 8, 2017.

To do...

- ☐ 1 (p. v): name of the recording guy
- ☐ 2 (p. 3): todo test
- ☒ 3 (p. 3): this is done
- ☐ 4 (p. 3): Note smwhere that we're using only vibrating solid objects (not liquids).Gaver[6] pg10 has info
- ☐ 5 (p. 9): maybe name this chapter smth else??
- ☐ 6 (p. 10): develop more
- ☐ 7 (p. 12): describe the device used for recordings
- ☐ 8 (p. 12): generate different stimulus sets, varying the size
- ☐ 9 (p. 13): describe headphones
- ☐ 10 (p. 13): number of participants, age, gender, normal hearing, job
- ☐ 11 (p. 15): Here we can put pictures and codes snippets
- ☐ 12 (p. 15): decribe the patch, describe the sound (starts low, goes to max ampl and then decays etc)
- ☐ 13 (p. 15): Put spectrogram pictures of the sounds to describe them