REAL TIME PROCEDURAL WIND SOUNDSCAPE
The effect of procedural wind soundscape on navigation in virtual 3D space

Bachelor Degree Project in MEB: Media Arts, Aesthetics and Narration
30 ECTS
Spring term 2014

Jóhannes Gunnar Þorsteinsson

Examiner: Anders Sjölin / Lars Bröndum
Abstract

Sound design with the help of procedurally generated sound in video games has seen a rise in the last few years given how that method gives us greater freedom in how sound reacts in real-time to the games, and the players. This research looks into if there is any difference in how procedural sound, in this case procedurally generated wind, affects the navigation of players in a three-dimensional world, as opposed to static sample-based sound design.

Keywords: procedural sound navigation
# Table of Contents

1 Introduction .................................................................................................................. 1

2 Background ..................................................................................................................... 2
   2.1 Procedural Audio ....................................................................................................... 2
   2.2 Sample Based Audio ............................................................................................... 2
   2.3 Pure Data ................................................................................................................. 3
   2.4 Open Sound Control ............................................................................................... 4
   2.5 Game Examples ....................................................................................................... 4
   2.6 Wind ......................................................................................................................... 4

3 Problem .......................................................................................................................... 6
   3.1 Method ..................................................................................................................... 6
      3.1.1 Template Level ................................................................................................. 7
      3.1.2 Level A, Sample based wind ........................................................................... 7
      3.1.3 Level B, Procedural Wind Soundscape .............................................................. 8
      3.1.4 Gathering Data ............................................................................................... 8
      3.1.5 Processing data .............................................................................................. 9

4 Implementation .............................................................................................................. 10
   4.1 Progression ............................................................................................................. 10
      4.1.1 The Sound Engine .......................................................................................... 10
      4.1.2 First Level Design, The Natural ..................................................................... 11
      4.1.3 Second Level Design, The Urban ................................................................. 12
      4.1.4 Third Level Design, Natural but orderly ....................................................... 13
   4.2 Pilot Study .............................................................................................................. 14
      4.2.1 Results from pilot study ............................................................................... 15

5 Evaluation ...................................................................................................................... 18
   5.1 The Study .............................................................................................................. 18
   5.2 Analysis ................................................................................................................. 19
   5.3 Conclusions .......................................................................................................... 23

6 Concluding Remarks ..................................................................................................... 24
   6.1 Summary .............................................................................................................. 24
   6.2 Discussion ............................................................................................................. 24
   6.3 Future Work ......................................................................................................... 24

References ....................................................................................................................... 26


1 Introduction

This research explores the possible practical applications of procedurally generated sound for wind soundscapes, along with its effect on in-game navigation for the end user as opposed to sample based sound design. What we normally hear when we play video games or watch movies is sample based sounds which means the playback of never changing pre-recorded (and pre-designed) sound effects. This is something game design inherited from cinema as it works quite nicely there as sounds do not need to be played in real time nor do they need to vary at all as there is no interactivity. Because of the nature of cinema, movies remain virtually the same each time you watch them. Interactive art like games on the other hand do not. Sample based sounds in games have been working out quite well in the past, but we are starting to hit a wall concerning the processing and size cost of those samples when the games start to grow. Sample based sound banks do not scale very gracefully as they have a fixed cost as each sample is always going to take the same amount of memory and processing power. This works well with products using small sound banks but when the number of sound effects start growing into the hundreds, thousands, or even more, things start getting hard to manage. Procedural sound on the other hand has a variable cost for each sample and therefor scales a lot more gracefully upwards.

To be able to implement procedural wind soundscape in a video game this research uses a visual programming language called Pure Data (Puckette et al, 2013). It allows you to program sounds, music, video or even games to name a few examples. Although its strongest point lies in the audio and digital signal processing. But it is just one part of the puzzle. To be able to connect a game engine and a Pure Data sound engine together Open Sound Control (Freed & Wright, 2009), more commonly known as OSC, is used to take care of communications between the two engines.

In essence, wind is at its core simply noise, but the main source of the sound behind wind comes from vortices that are created when wind hits obstacles in its path and can that create either cavity tones or aeolian tones. Cavity tones being generated in hollows and cavities while the opposite accounts for aeolian tones, like for example when a stick is swung around (Dobashi et al, 2003).

Does procedural wind soundscape affect the way people navigate a 3d game world, and to what extent? This research subjects players to a video game controlled from a first person perspective, where the objective is to collect a number of objects in a near open world. In the background the game collects location data about the player in the 3d world for the research.

Half of the participants played version of the level where a sample based wind sound is used for the main soundscape while the other half played a level with procedurally generated wind sounds. As an example, in the latter level the soundscape changes dramatically as a low pass filter is applied to the master output when a player finds shelter from the wind. Wind does not affect game play or game mechanics at all directly. The underlying wind sound engine was developed in Pure Data (Puckette et al, 2013) and implemented in Unity (Unity Technologies, 2005) via Open Sound Control messages (Freed & Wright, 2009). The wind sound engine is loosely based on wind patch designs from the book Designing Sound (Farnell, 2010).

---

1 Unity, a powerful all around game engine.
2 Background

2.1 Procedural Audio

Procedural audio is the act of creating and manipulating sound in real time with the help of algorithms and such tools. Sample based audio differs from that where sound is simply played back like a track on a CD. Andy Farnell goes over synthetic audio and sample based audio quite thoroughly in his book Designing Sound (2010). For those that were raised up in the 80's and 90's (and perhaps earlier even) do remember the sounds that the old video game consoles and computers created. Each console had a different feel to the audio as instead of sample based audio, these consoles actually had their own synthesiser chips built into them that produced sound effects and music in real time. When the progress of synthetic audio hit a wall in the 90's, sample based technology took a leap and won the race in interactive media because of its perceived realism, after that synthesised sound was simply discarded (Farnell, 2010). Even modern games that are made in the same style as the games from the late 80's most often emulate the synthetic sounds with sample recordings rather than making use of real time sound synthesis.

2.2 Sample Based Audio

“A recording captures the digital signal of a single instance of a sound, but not its behaviour”.

(Farnell, 2010)

This is why real time generated synthetic audio can be such an interesting fit for game audio. Games are built on interactivity. Game characters, objects, and even visuals in general are affected by player behaviour, sample based sound on the other hand, is simply triggered. Farnell (2010) said that sample based sound can be compared to the series of static 3D renders that were used to construct the game Myst (Cyan, 1993). In Myst, a player navigates a 3D world in first person view via pre-rendered static frames. By clicking on locations in view, the game loads a new frame rendered from the new location, giving the impression that the player has travelled a said distance. Various filters and effects can be added to the sample based sounds but that only fixes the problem with lack of interactivity but the problem with space needed for samples still remains.

Andy Farnell (2007) mentions in his paper Synthetic game audio with Pure Data (Puckette et al, 2013), that one of the main reason why a switch to procedural/synthetic audio might be a good idea is because of the sheer amount of samples needed to create realistic audio in modern games. Just looking at footsteps and how you need a good amount of footstep sounds for each character, every single surface the character walks on, and also sometimes footstep sounds depending on the weight of gear or the style of running/walking (Farnell, 2007).

When it comes to sampled wind we are left with several sound sources, maybe 10-30 seconds long each, scattered over the virtual game area with big volume attenuation zones telling the game where they can and can not be heard. They are then blended with other sound sources of various lengths to make sure that the user won't hear as easily that they are looped. For extra detail couple of sound sources for smaller sounds like rustling leaves and trash being
dragged along the scene by the wind is added but it's usually not connected to any proper objects in the game. When you hear the wind pick up speed you don't see the difference in the game world. Those two elements are completely disconnected. The virtual visual world and the audible world are two separate worlds, not really communicating with each other.

You could in theory implement wind sound with samples that reacts to changes in the game world by using clever combinations of samples, along with various filters and reverb effects that many game engines have built in. But just a single waveform loop of wind sound in our example can take between 500 to 1000 kilobytes, multiply that with the amount of samples needed to see how expensive it would be. When it comes to procedural audio on the other hand, the wind synth created for this research will most likely remain around 500 kilobytes. This number does not multiply according to amount of sounds played as they are being generated in real time with the 500 kilobytes of code. Procedural audio therefor has a relatively high start cost, but barely increases as it scales unlike sample based audio.

### 2.3 Pure Data

Pure Data (Puckette et al, 2013) often known as is an open source visual programming language used by musicians, visual artists, performers, researchers and developers to create software graphically in real time without the need of writing code. Pd was started by Miller Puckette who also developed a similar (but commercial) software Max (Cycling '74, 2014) prior to starting work on PD. Johannes Kreidler describes Pure Data in precise terms as “real-time graphical programming environment for audio processing” (Kreidler, 2009).
Figure 1  Example of a part of a wind synth patch in Pure Data

Although Pure Data (Puckette et al, 2013) and Max (Cycling '74, 2014) are relatively similar products the main difference lies in the fact that Max is commercial and proprietary while Pure Data is open source, and therefore a lot easier to customize for each usecase. Like for example embedding it into a game engine which is something that is theoretically possible with Max but practically too much hassle both technically and legally. Because of this difference in licensing this research will be using Pure Data rather than Max.

2.4 Open Sound Control

The simplest method of implementing audio in the Unity engine (Unity Technologies, 2005) would be using Open Sound Control (Freed & Wright, 2009), or OSC for a short. Using a library called libpd2 (Pure Data community & libpd contributors, 2010) is also a possibility but it actually embeds into the game and is therefore favourable for publishing but at the same time trickier to implement. OSC allows us to simplify the process considerably by sacrificing the option of embedding the Pure Data (Puckette et al, 2013) patches into the game. OSC simply acts as a method to send messages between the Pure Data patch and the game engine. The Pure Data patch will need to be launched individually which makes it not optimal for published games but perfect for audio prototyping and also a decent method for this specific research.

2.5 Game Examples

A good example of using Pure Data (Puckette et al, 2013) in games would be Maxis' work on embedding Pure Data in their game, Spore (Maxis, 2008) as an audio engine to drive audio patches (Brinkmann & Kirn, 2011). This allowed them to generate procedural audio for the monsters for example, and also allowed Brian Eno and others to write procedurally generated music for the game easily (Kosak 2008). According to Andy Farnell (2010) Sony might even embed Pd in their future game console designs. Another example being the game Fract OSC by Phosfiend Systems (2014), that is built with the Unity engine (Unity Technologies, 2005). Fract OSC is a first person game set in a synthesizer world where the player needs to solve puzzles and affect the game world by operating large machine synths and sequences. That kind of sound design would be extremely difficult and expensive to create with the sample based method.

2.6 Wind

The main reason for choosing wind sound for this experiment as opposed to any other sound was the simplicity of the core sound behind wind. At its core wind sound is simply a noise that is then shaped and adjusted until favourable results are achieved.

In van den Doel's (2001) experiment in real time rendering of aerodynamic sound it was mentioned that their audio algorithms are not derived using first principles from physical laws. The laws governing the sound of wind are so complex that such a simulation would simply be impossible (Doel et al, 2001). Therefore we will be basing our wind sound generator on the work showcased in Farnell's book Designing Sound (2010), which breaks up the sound into easy to understand components. At the core there is the wind speed and the

---

2 A library allowing for easy implementation of Pure Data in games and other applications.
generic wind sound created with simple white noise generator plus various filters. Other sounds that we need to focus on are the gusts, and the squall along with the high frequency whistling. Then there are the howling of pipes and other cavity tones along with the rustling of leaves in the trees.

Another thing to keep in mind is that wind is actually traveling from one place to another over a certain period of time, so if the wind is traveling from left to right, then your left ear will pick up the rise in wind intensity before your right ear. To achieve this the location and direction data from the game can be passed to the sound engine to decide when and how much to filter the left or right channel to achieve this effect (Farnell, 2007).

To connect this all together with the game world various filters along with volume control will be connected to each node of the synth, allowing the game engine to feed the position of the player into it to affect it. Therefore when the player is in cover, the sound gets dampened for example.
3 Problem

The main hypothesis is the following; with the help of procedural wind soundscape, players will traverse the 3d world differently, seeking cover in areas with less wind even if said wind has no direct effect or consequences on gameplay or game mechanics. From this hypothesis we can construct the main research question, which is the following:

Does procedural wind soundscape affect the way people navigate a 3d world, and to what extent?

Procedurally generated and computed audio is slowly gaining traction in the sound design field and a lot of research has been coming up looking into and analysing the strengths and weaknesses of this method of sound creation, see Synthetic Game Audio with Pure Data (Farnell, 2007), Foley Automatic (Doel et al, 2001) to name a few. There has been a lot of research based on the generation of ambience sounds like Sounding Liquids: Automatic Sound Synthesis from Fluid Simulation (Pamplona et al, 2009) which makes a good foundation for looking into the effect it has on players. Limited amount of research looking into the effect of generated sound on player has been made and therefor it seems like this is a good starting point for that kind of research.

3.1 Method

Sample collection used the purposive sampling methodology to make sure that one group is not too over represented. For example avoiding excessive numbers of males in the age range of 20-30. With this, the research is hoping to fill a certain quota of certain age groups, genders and such, but inside each group participants is selected randomly with probability sampling (Dawson, 2007).

Taking note that technology and equipment is an important factor of this research a large sample size of players is not really an option. One method to solve that would be to spread the research game over the internet to a lot of people and then gather the results through that. But the discrepancies in gear could contaminate the results. Some people only have speakers that can't emit the sounds properly, while other people might experience lag, low amount of frames per second or other technical difficulties. Therefore it is important to do the test on the same equipment for each participant. This limited the sample size considerably along with the fact that population is extremely scarce where the research was conducted.

Participants in the experiment were subjected to playing a short and basic open world video game where they needed to collect all objects to finish the level. Two versions of this level were made and participants were split into two groups where one group played level A while the other played level B. Level A features sample based wind sounds while level B uses procedural wind sounds. Navigational data is collected from the playthroughs and the objective of this experiment is to compare the differences in data between those two groups.

Along with data collection of coordinates of player movement, the play session was recorded using standard screen recording software. The focus on this research is on the quantitative data but it is kept in mind that not everything can be planned out and expected. Therefore, grounded theory methodology (Dawson, 2007) is used in mixture with the quantitative
method. Qualitative screen recording data is more thought of as an extra data just in case if something unexpected would be discovered during the course of the research. Therefore, given the fact that this is in the unknown there is no set rule on how or if the data will be processed or analysed, but it was simply be collected from all participants in the experiment.

Participants were able to quit at any time given the fact that this research is not looking into how well people navigate but rather how they navigate. So quitting early will not invalidate the data points collected.

3.1.1 Template Level
The template level that was constructed as the foundation for level A and B (covered in detail below) in this experiment. The level is constructed using Unity Technologies Unity engine (2005) as a square landscape featuring open areas and valleys for the player to navigate through. Three objects will be placed irregularly throughout the level and the objective for the player is to collect all objects and after that the game ends. To encourage exploration and to avoid players learning the simple scene by heart and taking therefor the shortest paths a fog is used to obstruct the view partially

The number of objects chosen is to simply allow for enough diverse navigational data to be collected while keeping the play session’s length to the minimum. One objects would introduce a rather linear gameplay or in a worst case scenario a long walk around the game world without any perceivable progression for the player. Two objects would only allow for 2 possible gameplay variants to be taken but with three you have six possible routes to be taken. The size of the level will be decided by the rule that each playthrough should take approximately 5 minutes on average to avoid boring the players while still managing to collect sufficient amount of data.

Given the fact that this research is looking into how people navigate rather than how well they navigate the objects were added to the game simply as a goal to drive the player to progress in the game session. The amount of objects collected or the time it takes for each participant are of no interest to this research as the wind sounds are not designed to lead the player towards his goals. A non-linear square open map was also chosen to allow for more freedom in choice concerning navigation.

3.1.2 Level A, Sample based wind
The wind soundscape used in this scene is constructed out of several samples of real and pre-designed wind recordings be placed accordingly. The sounds do not act along with the changes in the scene though but will focus on fidelity rather than interactivity.
3.1.3 Level B, Procedural Wind Soundscape

In level B, The Pure Data (Puckette et al, 2013) wind sound engine comes into play, using OSC (Freed & Wright, 2009) to allow it to talk to the game engine, and the other way around.

![Diagram showing the flow of data between the game engine and audio engine.]

**Figure 2** An illustration showing the flow of data between the game engine in light gray and the audio engine in black.

In certain areas where the player has a cover from the wind according to its wind direction, a low pass filter\(^3\) is applied at the end of the sound engine pipeline. Resulting in all of the higher frequencies being silenced and the wind sound being softened and damped considerably. Putting it more into the background.

3.1.4 Gathering Data

Quantitative data was gathered at a timed interval, every second, with the X and Y coordinates of the player in the game world. Each dataset also features an identifier telling if the data is from the sample based experiment level or the reactive wind one.

Biometrics were not measured simply as that is not a realistic scope for this research. Although with that said, comparing biometric data between those two scenarios could give some interesting results and is a worthy idea for future research in the field.

A metrics tool of this sort will only give us very raw numbers from the game but will not give us anything regarding the person, feelings, social factors and such. That is not the scope of the project although to be on the safe side, subjects will be asked a short close-ended questionnaire with the usual question of age, gender and perceived time spent on games each week to roughly estimate their game experience (Tychsen. 2008).

This is the quantitative data, but the research also gathered some qualitative data to mix in grounded theory methodology with our current quantitative methodology of choice. This data was not planned to be of any use but was collected just in case if a discovery would be made half way through the research requiring this data to be collected. This allowed for the

\(^3\) Low Pass Filter; An audio filter that allows only certain low frequencies to pass through.
possibility of making adjustments to the research if possible, as new information emerged (Neuman, 2004).

A standard screen recording software took care of recording each gameplay session. At the end of each session along with the closed ended questionnaire, one open ended question will be given to the participants on paper. “What did you think of the wind?”.

As noted above, this is not something that was necessarily planned to be used in the research but is just used as a failsafe mechanism in case if such data is needed. It’s also there to hunt for unexpected data that might be the spark for future hypothesis and research.

Another problem with game metrics is that there is no standard method on how to output data from games but it seems like game metric tools generally output log files of raw data, imported into a database system where researchers can then work with the data (Tychsen, 2008).

In our case, the data looks to be written into a file according to the CSV* format and will look like this.

<table>
<thead>
<tr>
<th>Time</th>
<th>X</th>
<th>Y</th>
<th>Cover</th>
<th>Experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.925262</td>
<td>74.87</td>
<td>113.747</td>
<td>0</td>
<td>WindSynth</td>
</tr>
<tr>
<td>4.175257</td>
<td>74.87</td>
<td>113.747</td>
<td>0</td>
<td>WindSynth</td>
</tr>
<tr>
<td>4.425247</td>
<td>74.87</td>
<td>113.09</td>
<td>0</td>
<td>WindSynth</td>
</tr>
</tbody>
</table>

Time column gives us a timestamp from the game engine, X and Y simply give us X and Y coordinates for the player. 0 being the starting point for the player and the center of the map.

Cover column is a boolean value simply telling us if the player is in cover (and triggered the low pass filter) or not.

Experiment column tells us which map version the player is playing. Sample based wind map being “Samples” and procedural sound map being “WindSynth”.

### 3.1.5 Processing data

To process the data the coordinates were used to create overlay path maps showcasing a heatmap displaying the paths players of each group took. Certain coordinate ranges will then act as certain hot zones where it was analysed how often players enter these zones. These zones are for example certain “wind free zones” where players can seek cover.

Given the fact that datapoints are samples of each quarter second we can also look into how long a player stays in certain areas and how fast she will travel through it.

Qualitative data like the gameplay session recordings were skimmed through at 2x the regular speed to help the researcher look at the data from a different point of view. For example seeing unique behaviours that are not as easily visible in the coordinate data. This unique behaviour can then be located in the coordinate data for further analysis.

---

* CSV; Comma-separated values. A simple file format convenient for tabular data
4 Implementation

Three variations of the experiment were created with the Unity engine from Unity Technologies (2005). First version focused on a rather random and natural looking level design set in a desert. Second iteration used only primitive shapes and straight lines, resulting in a rather planned but cold design. Third iteration took the middle ground and made compromises between the two extremes. Natural terrain set in a desert, but designed from a grid to begin with.

The game was created with Unity 5 (Unity Technologies, 2005) along with Pure Data Extended (Puckette et al, 2013) on the sound side. A connection was made between the two executables via Open Sound Control network messages (Freed & Wright, 2009).

A small pilot study was conducted on the third design with 4 participants. Two adults and two children. Three of them female and one male.

4.1 Progression

4.1.1 The Sound Engine
The wind sound engine is relatively basic in essence. At its core there is a noise source that flows through every part of the engine. On the other side, a low frequency oscillator is used to control the falling and the rising of the wind speed. That all is then seeded through the wind squall, gust, leave, howl, and whistling wire part of the engine.

![Diagram of the wind sound engine](image)

At the end, everything is filtered through a panning object to ensure the sound fits the 3D world. After that the sound flows through a reverb object to add a tiny bit of space into the sound.
At the top of this is the “pd OSC” object that receives OSC messages (Freed & Wright, 2009) from the video game telling the engine what to do. There are three basic types of messages. The first one tells the sound engine if the game has started or ended, raising/lowering the volume and turning on/off the engine when needed. The second one tells the engine when the player has entered cover from the wind, which in turn will initiate and change the value of a low pass filter on the whole soundscape. The third type of messages gives the sound engine information in what direction the player is facing and passes that to the panning mechanism of the sound engine, “pd anglePan”.

While designing the sound engine, and most importantly, the OSC connections between the game and the sound engine, a very important limitation of the Unity game engine (Unity Technologies, 2005) was discovered. Given the closed source aspect of the game engine it was literally impossible to access, modify, or even read information coming from the integrated wind physics engine in Unity. So that any communication between the physics wind engine and the sound engine were virtually impossible. One of the concepts experimented with to get around that limitation was to create a closed wind system outside the game world with an object inside, affected by the wind. By measuring movement of said object it would had been possible to roughly estimate the wind speed and then send that information to the sound engine. But after some experiments this was deemed to difficult, and unreliable.

Because of this limitation the two way connection along with the physics wind engine was therefor excluded from the final design.

4.1.2 First Level Design, The Natural

In the first implementation of the game world a focus was put on making it look natural with a pseudo-random placement of various natural formations like hills, mountains, covers and such. The layout, overall design and general atmosphere was slightly inspired by That Game Studio's Journey.
In rough, the level consisted of a large island with sand dunes and mountains. The wind flowed from one direction so the mountains allowed for some predictable covers although it was problematic to know when the cover ended the more the player distanced him or herself from the mountain.

Given the random design elements of said level it became increasingly more difficult to balance the amount of areas that would give cover and areas that would not. This could lead to more skewed data results as players could simply stay for longer periods in cover areas because there was larger area providing cover, or the other way around.

Also because of the natural design it became really complex to decide on what areas would give cover and what areas would not. This could result in players seeing a possible cover from wind only to realize afterwards that it would not give cover. For example judging if a sand dune is tall enough to give cover or not.

For the same reasons reading the statistics of such a chaotic level would proof to be challenging.

The mountains and rocks also proved to be too distracting visual cues but they could lead to players following mountains rather than following their ears. But that element was also used in That Game Studio’s Journey, as in that game a large mountain was seen in the distance, acting as a temptation for the players to travel there rather than somewhere else.

This level design on the other hand did have some interesting visuals which were bound to keep the participant interested in exploring.

4.1.3 Second Level Design, The Urban
Second iteration was of a level built from mostly primary shapes and straight lines like boxes, triangles, and such. This resulted in a aesthetically boring level but made data gathering and analysing a lot easier. In other words, It introduced less noise into the dataset than the natural terrain level. The main problem with this level though was that it was too predictable for the user. Also, given the fact how aesthetically uninteresting it was it lacked the motivation to explore.

![Figure 5](image-url) Overview of the level, with starting point on the lower level on the left.
Wind flowed from east to west, so lower floor was cover while upper platforms were non-cover. Erected cubes along the sides were placed there simply for aesthetic reasons.

Objects were placed on both upper and lower floor, but level progression was mostly linear from west to east.

### 4.1.4 Third Level Design, Natural but orderly
The third and final version combined elements from both former designs in an effort to strike a balance between predictable and chaotic. Like the second version a simple lower and higher level concept was utilized to indicate where there was cover and where there was not. Wind would come in from the east (right) to west (left). During the design process a grid was overlaid on the height map to ensure a relatively even proportions between the lower and higher level.

Like with the first design proposal, sea was used to mark the end of the level.

![Figure 6](image.png)  
**Figure 6** Overview over the final map. Starting point shown in red, with objects circled in blue.
Three objects were placed in three separate corners of the map, shown in blue. Two on the upper floor, and one on the lower. Fog would ensure that a player would never be able to see an object in distance from the position another object.

Figure 7  Contour drawing showing cover area in white, while non cover area is displayed in black.

The above image shows areas with cover from the wind in white. Level was divided into 81 squares, with approx 40 squares for each level.

4.2 Pilot Study
A pilot study with a tiny sample size of only 4 people was done to test run the whole setup to see if any flaws would come up in the code or the data. The data gathered was better than expected.
In the pilot study a young volunteer that played the procedural version of the experiment noted that she could hear the chime of the objects very well, while a second volunteer who played the sample based version of the experiment was oblivious to the fact that there was a chime at all. The reason for this seems to be that the wind sound designed for the sample based version of the experiment was either louder or simply contained frequencies that made the chime and other sounds less audible. Because of this the chime was taken out for future studies. This would result in finding the objects slightly harder, but at the same time it would even the playing field between the two groups.

Pilot study sample size was minimal as the core idea was to simply test the software and hardware rather than testing the results and calculations. Sample size was 4 people, two adults and two children, three females and one male. All with diverse experiences regarding video games and computers.

To begin with all participants were given information about how to navigate and move the character in the game world. They were also told how to lower and raise the volume if needed. Participants were also told that the goal of the game was to locate and collect three pink objects, but they could still quit whenever they wanted.

In the final iteration of the game, data is collected four times per second and written to a CSV file in either ResearchData/Samples or ResearchData/WindSynth respectively, with a file name indicating time and date. The data collected is time, X and Y coordinates, if player is positioned in cover or not, and which research group said data belongs to.

For example, one data entry could contain the following: “2,203855 74.87 113.747 0 Samples”. In other words, Second 2,203855, X coordinates 74.87, Y coordinates 113.747, 0 for non cover, and data taken from Samples research group.

Probably the main flaw noticed in the pilot study was the inclusion of chimes resonating from the objects the participants needed to pick up. At first this was thought of as a subtle way to help people orientate themselves in the game world and to find the cubes, but after observing the pilot study participants I noticed that those playing the procedural sound version of the study were quicker at finding the cubes simply because the procedural sound masked the chimes less. With the sample based sound the chimes were hard to hear making it a lot harder to navigate.

4.2.1 Results from pilot study

The results of the pilot study, although relatively pointless given the small sample size, were quite interesting. As 4 people played the game, 2 each version, a certain difference was expected in the ratio between people staying in cover versus not staying in cover. But what happened was that the ratio between cover/non-cover was almost identical between the two groups. In the samples version 45.4434% of the time was spent in cover, while with procedural sound, the number was 45.825%. An error in measurements was at first suspected to be the reason for this but after thorough testing everything seemed to be fine with the data gathering. This anomaly will therefore be considered a coincidence until more data will be gathered.
Table 1 Ratio between time spent in and out of cover for both the sample group and procedural sound group.

<table>
<thead>
<tr>
<th></th>
<th>Non Cover</th>
<th>Cover</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Samples</strong></td>
<td>54.5566%</td>
<td>45.4434%</td>
</tr>
<tr>
<td><strong>Procedural Sound</strong></td>
<td>54.175%</td>
<td>45.825%</td>
</tr>
</tbody>
</table>

Even though the ratio between groups was near identical, the paths taken were not, as indicated by the heat maps below. For some reasons players from the procedural audio group focused most of their exploration efforts in the north east while sample based participants explored the area more evenly. At a closer look one can notice the bright yellow and white spots in the upper right in the samples data and lower right in the procedural sound data. These peaks are most likely caused by a single player staying in one place, looking around and trying to get a sense of direction.

![Heat maps indicating time spent in each area. Left being for the samples version and right for procedural sound. Red color indicates less travelled areas while yellow and white indicate more travelled areas.](image)

**Figure 8** Heat maps indicating time spent in each area. Left being for the samples version and right for procedural sound. Red color indicates less travelled areas while yellow and white indicate more travelled areas.

Although not much can be read from this data, one thing that can be easily noticed is that by comparing the heat maps to the map itself one can see how the heat maps follow the valleys and lower areas of the map. Oddly enough the valleys are more visible in the sample based group than in the other. Because of the small sample size it can be theorized that this is caused by a single player favouring the valleys for the ease of navigation for example. This needs to be taken into consideration if the real data will mirror this behaviour.
Figure 9  Same heat map as before, but with warm colors representing covers.

Another way to visualize the data is to use the above heat maps and overlay the contour map of the wind covers over it. When analysing this we can for example notice on the Samples map that for some reasons the eastern most ridge was highly popular amongst players while in the procedural sound group, the valley west of it was more traversed.
5 Evaluation

5.1 The Study

The equipment used in the study was a 11.6” Acer C7 laptop running Ubuntu 14.04, Shure SRH440 Headphones, OUYA Gamepad Controller for player control, and Behringer UCA202 Audio Interface for added audio quality, and the ability to connect a second pair of headphones for researcher to monitor the audio.

8 people participated in the study, 4 in each control group. 5 males and 3 women participated in the age range spread well between 14 and 38 years old. In the questionnaire afterwards participants noted roughly how much time was spent each week on video games and answers ranged between 0 and 99 hours, with the average counting 19 hours per week. People were also asked about their estimated experience with video games on the scale of 1 to 5, with 1 being little to no experience and 5 being a lot of experience.

<table>
<thead>
<tr>
<th>Amount of people</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Little to no experience</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Before each trial players were instructed on how to control the avatar, but the game featured a standard twin stick gamepad control scheme where the left analog stick allowed for forward/backward movement along with strafing to the sides. Right analog stick allowed for 360 degree head movement and avatar orientation. Players were also told that they could walk everywhere except that the sea was out of bounds for them. This was added to the instructions after it was noticed during the pilot study that some people did not realise it was possible to walk up slopes, which seriously reduced the number of possible routes to take. Players were also told that they were supposed to search for 3 stationary pink cubes and touch them to pick them up. They would get a notification sound when one would be picked up.

To ensure that participants would not feel forced or obligated to finish the study, participants were told that they could stop the experiment at any time. To avoid participants feeling guilt for not finishing the experiment, they were told that the amount of cubes collected had no effect on the study, nor the data.

Some of the playthroughs were recorded on video by standard screen recording software, along with audio microphone, but it was later considered to be not important enough as it could introduce lag and slow down the game. The audio was supposed to catch any remarks the players would make but throughout the study, with the exception of one person from the pilot study, players mostly stayed silent.
It was also avoided to tell people to describe verbally their playthrough while playing as that could negatively affect the audial experience, although they were free to do so if they wished.

5.2 Analysis

As the raw data featured thousands of entries of coordinates the best way to present it is simply looking at the seconds spent in cover and non cover, along with percentage between time spent in cover and non cover.

<table>
<thead>
<tr>
<th>Player</th>
<th>Non Cover sec</th>
<th>Non Cover %</th>
<th>Cover sec</th>
<th>Cover %</th>
<th>Total Playtime sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Player 1</td>
<td>112.75</td>
<td>66%</td>
<td>58.5</td>
<td>34%</td>
<td>171.25</td>
</tr>
<tr>
<td>Player 2</td>
<td>421.75</td>
<td>68%</td>
<td>199.25</td>
<td>32%</td>
<td>621</td>
</tr>
<tr>
<td>Player 3</td>
<td>353</td>
<td>32%</td>
<td>766.25</td>
<td>68%</td>
<td>1119.25</td>
</tr>
<tr>
<td>Player 4</td>
<td>513.25</td>
<td>66%</td>
<td>270</td>
<td>34%</td>
<td>783.25</td>
</tr>
<tr>
<td>Total</td>
<td>1400.75</td>
<td>52%</td>
<td>1294</td>
<td>48%</td>
<td>2694.75</td>
</tr>
</tbody>
</table>

The above table shows data gathered from individual players of the sample group which shows relatively similar results between each player except for a single extreme case, and difference in total time spent in game. In total the whole group spent 52% of their time out of cover, and 48% in cover.

Table 4 shows data gathered from individual players of the procedural audio based group. We notice near identical data in this to the sample group except with a slight increased time spent in non cover surprisingly, opposed to the predictions made before the study. Another interesting data point is player 6 who, just like player 3 from the sample group, seemed to play completely differently than other players, with 41% of the time spent in non cover and 59% of the time spent in cover. In total the procedural audio group spent 57% of their time out of cover, and 43% in cover. Which was 5% lower than the sample group before.
Table 4 Player data from procedural audio based group

<table>
<thead>
<tr>
<th>Player</th>
<th>Non Cover sec</th>
<th>Non Cover %</th>
<th>Cover sec</th>
<th>Cover %</th>
<th>Total Playtime Sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Player 5</td>
<td>110.25</td>
<td>70%</td>
<td>46.75</td>
<td>30%</td>
<td>157</td>
</tr>
<tr>
<td>Player 6</td>
<td>405.25</td>
<td>41%</td>
<td>579</td>
<td>59%</td>
<td>984.25</td>
</tr>
<tr>
<td>Player 7</td>
<td>213.25</td>
<td>66%</td>
<td>110.25</td>
<td>34%</td>
<td>323.5</td>
</tr>
<tr>
<td>Player 8</td>
<td>635.25</td>
<td>70%</td>
<td>278.75</td>
<td>30%</td>
<td>914</td>
</tr>
<tr>
<td>Total</td>
<td>1364</td>
<td>57%</td>
<td>1014.75</td>
<td>43%</td>
<td>2378.75</td>
</tr>
</tbody>
</table>

Most of the players have pretty much the same percentage difference between cover and non-cover. They stay usually around 65-70% in non-cover and 30-35% in cover. Two players, one in each group were the odd ones out though with 32% in non cover and 68% in cover, and 41% in non cover and 59% in cover. If we take these extreme cases out of the dataset the numbers will start looking even more similar between the groups. Indicating that the sound didn’t seem to have any effect at all on the participants. Even the extreme case individuals had similar ratio between time spent in non cover and cover.

Table 5 Total frequency calculations of sample groups

<table>
<thead>
<tr>
<th>Group</th>
<th>Non Cover sec</th>
<th>Non Cover %</th>
<th>Cover sec</th>
<th>Cover %</th>
<th>Total Playtime Sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Procedural Group</td>
<td>1364</td>
<td>57%</td>
<td>1014.75</td>
<td>43%</td>
<td>2378.75</td>
</tr>
<tr>
<td>Samples Group</td>
<td>1400.75</td>
<td>52%</td>
<td>1294</td>
<td>48%</td>
<td>2694.75</td>
</tr>
<tr>
<td>Total</td>
<td>2864.75</td>
<td>55%</td>
<td>2308.75</td>
<td>45%</td>
<td>5073.5</td>
</tr>
</tbody>
</table>

When everything is calculated together we see that the procedural group spent 57% of its time in non cover, and 43% of its time in cover. As opposed to the sample group which spent 52% in non cover and 48% in non cover. In other words, the sample group spent 5% more of their time in cover than the procedural audio group.
Table 6 Total frequency calculations of sample groups excluding extreme cases

<table>
<thead>
<tr>
<th></th>
<th>Non Cover sec</th>
<th>Non Cover %</th>
<th>Cover sec</th>
<th>Cover %</th>
<th>Total Playtime Sec</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Procedural Group</strong></td>
<td>958.75</td>
<td>69%</td>
<td>435.75</td>
<td>31%</td>
<td>1394.5</td>
</tr>
<tr>
<td><strong>Samples Group</strong></td>
<td>1047.75</td>
<td>67%</td>
<td>527.75</td>
<td>33%</td>
<td>1575.5</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>2006.5</strong></td>
<td><strong>68%</strong></td>
<td><strong>963.5</strong></td>
<td><strong>32%</strong></td>
<td><strong>2970</strong></td>
</tr>
</tbody>
</table>

If we look at the data excluding extreme case players the numbers look relatively similar with sample group players spending 2% more of their time in cover than the procedural audio group.

![Figure 10](https://via.placeholder.com/150)

**Figure 10** Heatmaps of paths taken by individual players. Upper row showcasing sample group and lower row showing procedural audio group.

Not much can be gathered from looking at heatmaps of individual players. But some difference becomes obvious after we calculate and render heatmaps for each group. One thing that is noticeable right away is that the paths taken by players in general tend to follow the map layout, but surprisingly the Procedural group seems at first sight to have a better overall spread over the whole map, while the sample group follows the layout more clearly.
Figure 11 Heatmaps of the paths taken by players. Procedural audio group on the left and Samples group on the right.

Figure 12 Heatmaps of the paths taken by players with cover information overlaid on top. Warm colours showcase cover while cold colours show non cover. Procedural audio group on the left and samples group on the right.
5.3 Conclusions

In short, players seem to be unaffected by the difference between a procedurally generated wind soundscape and a static sample based wind soundscape in a 3d game world seen from a first person point of view.

The results of the research were rather surprising as they did not give any definite answer. There seemed to be no difference between the two groups and the data favoured sample based audio rather than procedurally generated audio.

What we can read from the data though is that there is the possibility that visual cues affected the player more than the audio, for example player staying on high ground (non-cover) to be able to have better visibility and more chance of finding the objective.

Therefore, the methods of researching this subject need to be developed further and a different and less visually focused method needs to be designed.

Another problem is sample size but to mitigate that, a simpler game/experiment that would be more device agnostic, and preferably browser based, could increase the sample size considerably as it could be shared online to participants.
6 Concluding Remarks

6.1 Summary
The research question was the following: Does procedural wind soundscape affect the way people navigate a 3D world, and to what extent? To answer this question a level was created with near equal amount of area acting as cover from wind and not. 8 players were then split into two groups, with one group playing the level with procedurally generated wind sound and the other group played the level with regular sample based wind sound.

The procedurally generated wind sound would get affected by the position of the player, indicating if the player was in fact in cover from wind or not. While the sample based audio was simply a static loop of a wind recording.

Players were told to travel through the game world and collect pink cubes, while the game recorded the paths they took.

The data gathered from this experiment tells us no. In general, groups spent near equal amount of their play time in cover and given the fact how even the majority of the players were it can be assumed that with even larger sample group, the gap would become even smaller.

Players in total spent 55% of their playtime out of cover, and 45% of their time in cover. With the procedural group spending 57% of their time out of cover, and 43% of their time in cover. Sample group spent on the other hand 52% of their time out of cover, and 48% of their time in cover. Meaning they spent 5% more of their time in cover than the procedural group.

6.2 Discussion
Procedurally generated audio is an important step in the future of game audio, although we won’t know for sure how widely implemented it will be in the future. But given the high cost in space for sample based audio it is quite certain that game designers will start experimenting more and more with procedural audio where it fits.

The recent rise in amount of procedurally generated sandbox games like Minecraft (Mojang, 2015) and the currently unpublished No Man’s Sky (Hello Games, 2015), also indicates that procedurally generated sound has its place in the vast generated game worlds of the future. The experience needed to create realistic procedural audio on the other hand makes it very hard to see this happen in the near future except in isolated cases and very specific games, like Spore (Maxis, 2008).

The results from this research does tell us that procedural audio, or at the very least procedurally generated wind does not affect the navigation of players, but on the other side, at the very least it does not affect it negatively.

6.3 Future Work
There are many questions unanswered about procedural audio, but concerning navigation, the concept behind this research could be expanded in the future to get a better look at how audio can affect the behaviour of players. A future version of this research could for example
focus on eliminating the effect of visual cues and visual navigation for example by using a top
down 3rd person point of view, so no matter where you stand in the game world, it does not
alter your field of view and viewing distance. As it is theorized that in this research, people
stayed on the higher ground, and out of cover, to be able to see further.

To be able to fully implement procedural audio we will also need to research and learn of its
flaws. A research focused on looking into the workload and hours needed to design a single
sound as opposed to sample based sound design would give us many answers, and could lead
us to looking into how to optimize the workflow so that switching from sample based sound
design to procedural sound design would actually be realistic for companies.
References


Puckette, M. et al. (2013). *Pure Data* (Version: 0.43.4 Extended) [Computer program]. Available online: http://www.puredata.info

Pure Data community, libpd contributors (2010). *libpd* (Version: 0.8.2) [Computer program]. Available online: http://libpd.cc
