An efficient 3-D sound experience for mobile applications

Group 3 Aalborg University
Copenhagen, Denmark

Abstract—We developed an application for mobile devices that uses an efficient HRTF model based on to create 3D soundscapes. The HRTF model is based on a combination of filters and delays provided on theoretical basis by Brown et. al. [1]. The device orientation is used to set the angle between the sound and the user. This model has many possibilities on the limited hardware capabilities of the mobile devices in comparison to HRTF database based models. The prototype will present the user several sounds scattered in a limited area where the user can interact with

Index Terms—HTRF, mobile devices, soundscape, Pure Data, OpenFrameworks

I. INTRODUCTION

A lot of research on modeling 3-D sound has been done with fairly good solutions (references). The best results can be obtained by those models implementing 3-D sound using personalized HRTF functions which needs tedious measuring of impulse responses etc (reference). Since mobile applications as well as multimedia productions are usually aimed at a big number of users, implementing personalized HRTF functions would require measurements of the individual user which is almost impossible. Additionally, using huge databases with HRTF's for subjects having similar head diameters and pinnae structures (dunno, how those HRTF's work) are not feasible for mobile applications as memory space is limited. Therefore, the aim of this project was to develop an computationally efficient and general HRTF model, yet, keeping the quality of the 3-D sound as good as possible. We therefore used the model provided on a theoretical basis in [1] by implementing a combination of filters and delays in pure data and C. This audio engine was then embedded in a mobile application. Sensor data provided by the phone or tablet was used to compute from which direction the sound should appear to be coming from. With the application users should be able to discover a virtual sound space situated in Copenhagen. This means that our HRTF model should satisfy two major requirements.

1. It should provide direction cues that are good enough to guide the user to the position where a specific sound is situated.

2. The implemented HRTF model should provide and intuitive way of locating a sound source from an qualitative aspect.

The first aim of course is bound to accuracy of the GPS signal and orientation data provided by the individual mobile phone. We hypothized that using a general HRTF model should be of sufficient quality to satisfy the first requirement since sensory data can be rather noisy anyways which would be overfitted

by a very precise HRTF model.

===Something about the second hypothesis===

Taken both hypothesises together the user should experience a soundscape in at least 2-dimensions, azimuth and distance.

The purpose of this paper is...*(it should be short, specific and immediate)*

First, some state of the art research on audio augmented reality will be provided. That section will be followed by the presentation of our implementation of the audio engine as well as the main application. In the last part we will present the results obtained by testing our application and discuss those in light of the two hypotheses stated above.

A. State of the art

The concept of augmented audio reality (AAR) deals with techniques where a real sound environment is extended with virtual auditory environments. Nowadays, the progress in audio technology and computing predicts the introduction of completely new type of interactive audio applications []. Advances in mobile technologies have made it possible to create audio augmented spaces almost anywhere. For instance, spatial auditory displays that can provide the user with landmarks and are capable to attract the user's attention have been tested and introduced []. Many experiments, qualitative and quantitative researches have been designed so far to better understand the way in which people usually perceive multiple simultaneous sources differently placed and to increase the level of immersion in the experience.

1) Audio reality vs Virtual reality: First of all, the possibility to hear the natural acoustic environment around a user differentiates the concept of augmented reality audio from the traditional concept of a virtual reality audio environment []. In virtual reality, generally participants are abstracted from the natural environment and are surrounded only by a completely synthetic one (acoustic and/or visual). On the opposite, in augmented reality a virtual environment is *superimposed* on a real one. To be more specific, in a mobile audio augmented environment participants are able to interact with the virtual audio mixed with real vision and soundscape.

2) Augmented audio reality: The aim of AAR is to combine both real and virtual sound scenes so that virtual sounds are perceived as an extension to the natural ones. Virtual sound, as expressed in [], could be originated from another environment or artificially created. Two extreme possible augmentations are possible. The former is a perfect augmentation of the listener's auditory environment and is achieved when the listener is

unable to predict whether a sound source is part of the real or the virtual audio environment. The latter is a high-quality virtual auditory scene with some characteristic that are not possible in acoustic environments.

3) Previous work: Several work has been done in this field although this area is relatively new. *Audio Aura* (Mynatt 1995) was one of the first project to deal exclusively with audio in augmented reality system. It basically consisted in providing information to users as they travelled through their workspace. These information was triggered by particular locations in the workspace. The use of audio in *Audio Aura* is particularly interesting because most of its cues were associated to sounds from nature rather then recorded vocal, speech or synthetic sounds. Similar in approach to *Audio Aura* was the *Automated Tour Guide* (Bederson 1995). In both cases, triggers are readily identifiable, still and rarely changing. Those augmented sounds were associated to pre-determined locations in space and there was no need to determine the precise location of an individual. A successive work, *Hear&There*, was able to determine the location and head position of the user using the information from GPS and a digital compass []. A user could listen to these 'audio imprints' by walking into the area that a specific imprint occupied, which was triggered by proximity. The essential premise of *Hear&There* was that a physical environment has been augmented with audio. All the sounds and the data were gathered inside that system. Since a 'Field Use' has been developed, in which the user wear a hardware portable system, *Hear&There* has undoubtedly contributed to an improved definition of mobile augmented reality environment.

4) Human sound localization: The auditory system is used by the humans for several purposes in the daily life. One of this purposes is to provide the necessary information to localize sound sources in various dimensions, (width, height, depth) and it is even possible to guess the size of the source.

When a sound event occurs, the waves travel in all directions and , when they reach us, our brain compares the signals received by the left and right ears to localize it in the horizontal plane. The spectrum of the signal reaching each ear is different, since the amplitude and phase information differs. These binaural cues are called *interaural intensity difference* (IID) and *interaural time difference* (ITD). However, these cues are not enough to localize accurately the source since with this information the listener can not determine if the sound is in front, above or behind. This region of positions where all sounds yield the same ITD and IID is called *cone of confusion*. This ambiguity can be solved with the information provided by the filter effect caused by the pinnae, head, shoulders and torso, which modify the spectrum of the sound that reach the listener's ears. The sum of all these features are characterized by the Head Related Transfer Functions (HTRF) which are not only frequency and direction-dependent but also differ from person to person. That dependency makes therefore hard to generalize the spectral features among individuals. It is well known that using a HTRF from one person in another can significantly impair the perception due to the individual differences in the anatomy, but it has also been shown that some people localize better the sounds than others, and their HTRFs are suitable for a large group of listeners.

5) Localization/Lateralization: Headphones are much related to the concept of mobile and wearable application and have been used successfully in many virtual reality applications. Headphones, and earphones as well, make the ear receive the sound separately. That is advantageous because the whole scale of differences between the signals coming to ears, i.e. *interaural time differences* (ITD) and *interaural intensity differences* (IID), can be manipulated separately and individually [].

Generally speaking, when listening to sounds coming from headphones, the sound sources seem to be located inside the subject's head []. This is usually called *lateralization*, or intracranial, or 'inside-head-localization' (IHL). On the opposite, there is the effect of having the sound outside the heed, according to specific direction and distance, i.e. *localization* or 'outside-head-localization' (OHL). It has been necessary distinguish localization inside and outside the head. This difference in terminology has reinforced the assumption that there is a basic difference between perception of sound conveyed by headphones and sound that reaches the ear from an external and real source []. It has also demonstrated that a listener can make a clear distinction in headphones listening between localized (that is, sound inside the head) and lateralized sounds sources and that both these type can coexist in the listener's experience [].

6) Issues in headphones-conveyed sound: Headphones auralization often produces an incorrect localization of virtual sound sources [] and many issues could be experienced.

a) Externalization errors: externalization is related to the perception of auditory distance such that there is a continuum in perceived locations of sources from inside the listener's head to any external position. One of the most severe problem in AAR is a perceived effect of *lateralization* even in sounds that should be located in a real environment. To avoid such effect several techniques can be used. For instance, as expressed in [], the effect of a lateralized sound in headphone listening can be produced using amplitude and delay differences in two headphone channels corresponding to each source. The main goal of virtual acoustic synthesis should be to produce sounds that seem *externalized*, that is, outside the listener's body []. In order to make a sound source externalized and let the user be capable of a correct judgement of the distance of the sound more sophisticated binaural techniques are needed. In particular, spectrum differences in the two ear signal due to *head-related transfer functions*, HRTF's, play an important role. Moreover, acoustic cues such as the amount of reverberation and control of signal level are necessary for a successful auralization of a virtual sound source. These are explained into details in further sections.

b) Localization errors: localization error refers to the deviation of the reported position of a sound source from a measured 'target' location, i.e. the listener fails in matching the correct location of the sound. Localization errors can be divided in *azimuth* (deviations along the horizontal plane) and *elevation* errors (deviation from eye-level elevation) []. Dynamic cues related to head turning and other movements

of either a listener or a sound source should be taken into consideration []. These errors might come from some location accuracy. Determining the accurate position of the user of one of the most important task in an AR system owing to that system always produces output to the used based on his or her location in space []. That said, any location inaccuracy should be avoided using GPS receiver with high sensitivity and reliability. Here, the implementation design takes a crucial role as noted in [].

c) Reversal errors: sometimes called front-back or back-front 'confusion', that error refers to the judgement of a sound source as located on the opposite side of the interaural axis than the target position []. An informal proposal has been made, which tries to help the user in front-back discrimination on the basis of the familiarity of the effects on timbre cues, e.g. unique patterns in *early reflections* depending on the virtual sound location. Indeed, this has not yet been verified experimentally.

7) Conclusion: Although it is a recent field of research, several works and experiments in the context of audio augmented reality has been made so far. Most of research has been done mainly on the use of visual information with very little attention paid to the audio aspects.

The use of audio and its refinement in an augmented reality environment is a big deal. There are many issues and challenges that an audio environment presents []. For instance, providing a way for the user to orient himself or herself in an only-audio virtual space is even more difficult than providing visual cues. Another interesting aspects of audio, as also explained in [], is the fact that audio does not persist and it starts disappearing as soon as it has heard. That deals with a more attention due to the audio to capture every detail that may help the user retrieve a specific sound in a specific location.

Finally, with the advent of portable systems such as sensors-equipped smartphone and laptop a further development of *geolocated AR* has been made possible. That systems use locational sensing and user tracking technologies (e.g. Global Positioning System, or GPS) to let the user navigate through a audio-augmented space and to get a more defined listener's position, which is essential for a correct externalizaton. More information will be given in further sections.

II. DESIGN AND IMPLEMENTATION CHOICHES

also some argumentation why we used a HRTF model based on filtering rather than using a database (citations why database not efficient and why quality better with databases but good enough with filtering)

III. THE APPLICATION

A. openFrameworks

We decided to develop our application for Android platform. Nevertheless, we did not use the facilities and the built-in code provided by Android Studio, but instead we decided to use openFrameworks.

The reasons why we decided to develop using this programming environment was the need of a fast communication between the mobile platform and our program code. It works using C++ language, which implies that the code is compiled directly into assembly language and hence works very fast [] and does not need to be interpreted by a virtual machine as it is the case for a java-built environment such as Android Studio [].

Discussing all the steps involved in the building process is beyond the scope of this paper (and more information can be found in [] and []), but a short explanation of what is openFrameworks and the main operations used to achieve our application will be given.

1) Description: OpenFrameworks, by definition given on its website, is:

an open source C++ toolkit for creative coding []

Since it is entirely written in C++, distributed under MIT license and actually runs on five operative systems and four IDEs, it undoubtedly presents some kind of advantages. It gives the opportunity to several users to deal with code designed to be minimal and easy to grasp and use it on the favourite IDE. In that case, we use both OS X and Windows platform, and Xcode and Visual Studio as integrated development environment. That *simple and intuitive framework for experimentation* [] is designed to work as a general purpose glue and wraps together several commonly used libraries, such as *OpenGl, OpenCv, PortAudio* and many more. Nowadays, this is a popular platform for experiments in generative and sound art and creating interactive installations and audiovisual performances []. The current operative version is *0.9.0*.

2) Addons: its design philosophy claims for a collaborative environment. It thrives on the contributions of many people, and collaborate mainly on *addons* and projects. An *addons* is made of several snippets of code put together in order to extend openFrameworks functionality, bring some external framework and allow it to be integrated into openFrameworks project or make specific and complicated tasks easier and reusable in other contexts []. It also generally contains the library itself in a form that is ready to be linked to project binaries. In our case, as depicted in the next section, we used several *third-party* addons and built our own library called *ofxOrientation* to access sensors information about localization and orientation and provide the HTRF model with right values.

3) Main issues: openFrameworks requires such lots efforts to be mastered and its learning rate is quite steep. Its weakness is in fact due to the lack of documentation and people who don't have at least some programming background could feel quite uncomfortable. Hopefully, its community is continuously providing discussion and projects, while addons are released with some example to compile and have to be created using a bunch of accurate rules.

4) The openFrameworks project: all openFrameworks project have a similar structure of folders and files. The most important folder among then is for sure the *src* folder. It contains all the source codes and consists at least of `main.cpp` (containing the `main( )` function to let the operating system start the application), `ofApp.h` (containing declaration of the specific class) and `ofApp.cpp`, which contains definition of all functions declared in the previous file. All the methods in

that class are *event-handling* methods, that is they are triggered in response to events that happens inside the application such as mouse scrolling and program quitting.

To create a new project, we used the Project Generator wizard located in the same directory and directly provided by the environment. Such a way is simple and it is especially useful when dealing with several addons, which are automatically linked. Once we got the structure, we could access every source file in the *src* folder. At its simplest, working with an openFrameworks project is adding new code to the appropriate method, or just create a new one and declare it in the `ofApp.cpp`. In [] a further explanation of the main methods and their workflow is provided.

5) Compiling: soon...

B. Sensor data retrieving

Our application required two kind of sensors data: *GPS* and *orientation*.

- GPS is a way of determining location that has become amazingly common with the increasing of portable technologies. Any device that receives a GPS signal is called a GPS *receiver*. As explained in [] the receiver calculates its position by precisely timing the signals sent by the GPS satellites. We needed the GPS user position to calculate the distance to the sounds that were placed on fixed locations in the real environment.
- orientation usually is derived using a combination of field sensors, such as accelerometer and magnetometer. In this case, we needed orientation to define the angle the user is heading relative to the sound sources.

To access the GPS data we used *ofxMaps* and *ofxGeo* addons (reference to webpage/gitrepository). Reading the GPS position on an Android platform requires a listening mechanism and we implemented it in openFrameworks by calling the `ofRegisterGPSEvent()` in the `setup()` method of our main code. We also added a method to our application, in the `ofApp.h` precisely, to handle the updates from the Android OS system calls. It is named `locationChanged()` and adds an event handler for the dispatched event []. Moreover, the `startGPS()` method is called at the beginning and `stopPGS()` should call when quitting the application to avoid an overconsumption of phone battery.

(Something about smoothing if we used it, and maybe we have to look into the code how they used smoothing).

Since we are dealing with some kind of filtering depending on the sound sources position (e.g. the sound could appear in front of the user as well as behind), it is necessary to take into consideration the user's orientation. That is basically the angle created between the heading of the user and the geographic north pole. Initially, we assumed the user always faced to the north, i.e. the angle between the user and the geographic north pole[1] is equal to 0. If we can discard the deviation of the user's orientation, it is possible to define the angle between the user and the sound source as the *true bearing* [2]. This is, by definition

---

[1]There is a difference between *geographic* and *magnetic* north pole.

[2]It is possible refer to it simply as *bearing*.

---

the angle measured in degrees in a clockwise direction [3] *from the north line []*.

If we also assume a sound source as a point that radiates in every directions, we could easily calculate the bearing of the user from the sound source, as shown in (add figure). The bearing could be accessed via the *ofxGeo* addon, which contains a specific method called *GeoUtils::bearingHaversine* that uses the **'haversine'** formula:

$$\theta = atan2(sin(\delta\gamma)cos(\varphi_2))cos(\phi_1)sin(\varphi_2)$$
$$- sin(\varphi_1)cos(\varphi_2)cos(\delta\gamma))$$

where φ is the *latitude*, δ is the *longitude*.

Once we got the bearing, we can start supposing that the angle between the heading of the user and the sound source should be the sum of the bearing and the direction the user is facing. Doing so, we can retrieve it using a compass. In order to access the compass orientation we built our own addon accessing the 'SENSOR TYPE ORIENTATION' provided by the android API. Having this information we now had to differentiate between four cases: (some images and explanation about how we calculated the angle between our heading orientation and the sound). In order to smooth the sensor data (especially for the orientation) we filtered the sensory information by computing the median of a chunk of mediansize instances of the sensor output.

C. Audio engine

The audio engine was the most crucial part of our applications. As shortly mentioned in the introduction we decided to use a filter based HRTF model which claimed to be both 'efficient' and of reasonable 'quality' [**?**] in order to achieve a satisfying user experience and an efficient real time application. We basically followed the workflow (order of filter and delay lines) given in [**?**] and depicted in figure 1.

First, the monoaural sound which has been processed to add a range effect, goes to a head model and a room model. The binaural output of the former is then processed by the pinna model, and finally each one is added to the output of the room model to provide an externalization effect.

In the following formulas, the angles are measured in radians in the interaural-polar system, where θ is the azimuth angle and ϕ is the elevation angle. However, in our application the latter is no considered since all the sound sources are placed in the horizontal plane without elevation.

1) The head model: The path length difference from the sound source to the two ears and the shadowing effect produced by the head at the far ear, lead to a delay and a intensity difference in the sound arriving at the left and right ears. In order to estimate the time delay, the Woodworth's formulas are used [3] :

$$ITD = (a/c)[\theta + sin(\theta)] \ [0 \leq \theta \leq \pi/2] \qquad (1)$$

$$ITD = (a/c)[\pi - \theta + sin(\theta)] \ [\pi/2 \leq \theta \leq \pi] \qquad (2)$$

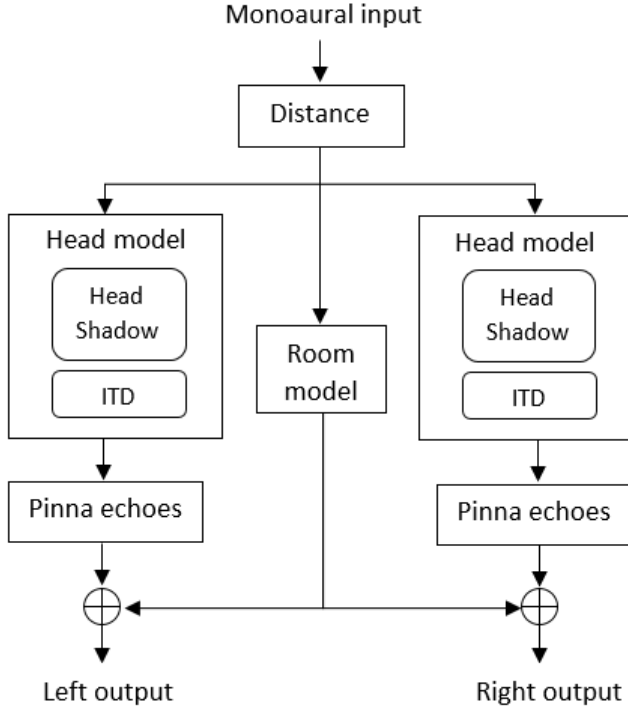[3]All bearings are measured in a horizontal plane.

Fig. 1. Components of the model provided on a theoretical basis in [?]

where a is the approximated head radius, θ is the azimuth angle in radians and c is the head radius. The time differences between the audio signal reaches the head and the ears are therefore

$$T_L(\theta) = \frac{a + a\theta}{c} \tag{3}$$

$$T_R(\theta) = \frac{a - a sin(\theta)}{c} \tag{4}$$

These formulas refers to a source in front of the head and on the right, with azimuths $0 \leq \theta \leq \pi/2$. If the source is placed on the right $(-\pi/2 \leq \theta \leq 0)$, the expressions are reversed.

The head shadow effect is characterized in the following analog transfer function:

$$H(s,\theta) = \frac{\alpha(\theta)s + \beta}{s + \beta}, \; where \; \beta = \frac{2c}{a} \tag{5}$$

====*Maybe this part in the appendix...?*====

Since this is an analog transfer function we had to derive the digital version by applying a bilinear transform applying the following substitution:

$$s = \frac{2}{T}\frac{z-1}{z+1}, \; where \; T \; is \; the \; sampling \; interval \; in \; seconds \tag{6}$$

Applying the substitution in equation 6 to equation 5, we get the following filter function in the digital domain:

$$H(z,\theta) = \frac{\alpha(\theta)(\frac{2}{T}\frac{z-1}{z+1}) + \beta}{(\frac{2}{T}\frac{z-1}{z+1}) + \beta} \tag{7}$$

To identify the filter coefficients we transformed Equation 7 and obtained the following frequency response[4]:

$$H(z,\theta) = \frac{2\alpha(\theta) + T\beta + z^{-1}(-2\alpha(\theta) + T\beta)}{2 + T\beta + z^{-1}(-2 + T\beta)} = \frac{Y(z)}{X(z)} \tag{8}$$

==

This gives us the filter coefficients $a_0 = 2\alpha(\theta) + T\beta$ and $a_1 = -2\alpha(\theta) + T\beta$ as well as $b_0 = 2 + T\beta$ and $b_1 = -2 + T\beta$. Since we could not use the filter function in Matlab which has as input values the filter coefficientes we had to isolate the output $Y(z)$ in Equation 8 as well as going from the frequency to the time domain. The result is given in Equation 9[5]:

$$Y[n] = \frac{a_0 X[n] + a_1 X[n-1] - b_1 Y[n-1]}{b_0} \tag{9}$$

2) The pinna model:

3) The room model: Finally, a room model is added to the two outputs of the pinna model...

D. User interface

include pictures of user interface etc.

IV. THE EXPERIMENT

The aim of our experiment was to investigate how fast users were able to locate and navigate to a single sound source using our 3-D audio engine compared to a simple panning and 2/d distance calculation which served as a baseline performance.

Additionally we assessed a qualitative analysis of how intuitive it was for subjects to locate the sound as well as investigating whether the sound seemed to be embedded in the real scenery. (look at the questionaire again)

(Here we can write a one paragraph preview of our results.)

A. Materials and methods

1) Participants: Data were collected from ... participants (... women and ... men; mean age: ..., ranging from ... to ... years) coming from different backgrounds.[6] ... students were already familiar with 3-D audio sound and/or audio navigation before conduction of the experiment. All other participants were naive. All participants reported normal or corrected to normal vision and no hearing deficits. Participants received no form of compensation other than gratitude.

2) Apparatus and stimuli: The experiment took place in a free field in "Valby Parken" (Coordinates: ...). The mobile device used was a Motorolla Moto G running with Android version 5.0.2. The device was connected to Sony MDR ZX600 stereo headphones. Only one person at a time was tested.

To create the auditory stimulus, digital vocal recordings of a male voice (age: ...) were collected using a ... . Recordings were done with ... bits of resolution for amplitude at a sampling rate of ... kHz. Auditory stimuli were normalized in peak amplitude so that at the minimal distance to the soundsource (7 meter) using the maximal level of volume on the mobile device, the soundlevel of each soundsource was at ... dB.

---

[4]For the derivation of Equation 8 see appendix A

[5]For the derivation of Equation 9 see appendix B

[6]However, ... were master students in Sound and Music Computing at Aalborg University Copenhagen.

3) Design: The experiment comprised two within subject factors at 2 levels (sound engine).

The sound engine was either our 3-D audio engine (3-D) or a simple panning and distance amplitude modulation (panning).

The starting point was fixed and the distance of the sound-source from that point was always 48 meters.[7] The sound could appear at any angle from the starting point.

For all trials the same soundfile was played. Each participant performed a total of 6 trials meaning 3 trials in each of the two conditions. Participants were encouraged to take a short break whenever they felt fatigued.

4) Procedure: The position (angle from the starting point) of the sound sources in the different trials were completely randomized within and between subjects independently from the sound engine. This ensured that participants could not predict the location of the sound in the next trial. The order of trials with the one or the other sound engine was not randomized within subjects. Nevertheless, half of the participants started the experiment with the 3-D audio engine while the other half started with the audio panning sound engine to counterbalance for effects related to training.

Each trial was initiated by the experimenter with a button press after which the sound source was played. No visual feedback about the location of the sound source or the already passed time to locate it was given. For localization, subjects had to rely solely on the auditory cues given via the headphones. /footnoteAdmittedly participants could have used the position of the starting point as some kind of referenced fixpoint. However, participants did know nothing about the possible distances of the sound source. Additionally, questions after the experiment about applied strategies did not reveal that participants used visual cues for localizing the sound source. As soon as the participants reached a radius of 7 meter around the sound source, an earcon was played indicating the success of the localization. The participants went back to the initial starting point and the next trial was started.

Before the experiment, participants were familiarized with the target sound as well as the earcons that were played when having reached the location of the virtual sound source. Additionally, all participants performed at least two practice trials with the sound engine they were tested with first before the actual measurements with that sound engine began. The location in the practice trials were fixed at the same positions for all participants to allow us to guide the subject when they got lost.

Participants were instructed to find the sound source as fast as possible. No information about the possible locations (distances and angles) was given to the subjects. Trials which lasted longer than 5 minutes were aborted and labeled as "not found".

5) Analyses: ... trials were excluded because the analyses of GPS data for those trials revealed very poor accuracy. The other trials which could be analysed summed up to ... trials for each of the two conditions in total.

The threshold chosen to correctly reject the null-hypothesis was always 5% ($\alpha = .05$).

WE PROBABLY DONT NEED THIS ((Reported p-values were computed using the Greenhouse-Geisser correction because Mauchly's test for sphericity was significant for several main and interaction effects (We have to check if that is the case in our experiment). Also the number of participants was relatively low reducing the power of Mauchly's test for sphericity [**?**, p. 72].)) Therefore, sphericity could not always be assumed even when Mauchly's test for sphericity was not significant [**?**, p. 72]. (Nevertheless, differences in significances (one test was significant the other was not) between sphericity assumed and Greenhouse-Geisser corrected values were rare.) WE PROBABLY DONT NEED THIS

B. Results

1) Quantitative analysis: We measured objective and subjective data on user performance.

For the objective measure we computed a two-sided t-test to compare the results for both sound engines. Analysis revealed that with the ... sound engine participants were significantly faster to locate the sound source (t(df) = 2.84, p = 0.01.).

If our results are not significant this could be because of low sample size -¿ "Numerous methodologists have cautioned that a small sample size implies low statistical power, that is, a high probability of Type II error" [**?**]

2) Qualitative analysis: Subjective analysis of the quality of the soundscape was investigated with a questionnaire participants had to fill out after each sound engine condition. The questions were aimed at examining ... of the soundscape. Therefore participants used a 10 point lickertscale. Evaluation of the questionaires showed that ...

V. DISCUSSION

VI. CONCLUSION

---

[7]However, because of poor GPS accuracies, the location of the soundsource was calculated with the GPS coordinates provided by the phone sensor instead of the actual fixed position in the free field. This ensured that the initial distance to the sound source was always the same.

REFERENCES   LIST OF FIGURES

REFERENCES

[1] C. Phillip Brown and Richard O. Duda, *An efficient HRTF model for 3-D sound*. Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. New York: IEEE. 1997

[2] Johannes M. Zmoelnig, *HOWTO write an External for puredata*. `http:// pdstatic.iem.at/externals-HOWTO/`, 2010.

[3] Neil L. Aaronson and William M. Hartmann, *Testing, correcting, and extending the Woodworth model for interaural time difference*. Acoustical Society of America, 2014.

[4] P.M. Delgado. *First steps on Pure Datas approach to real time audio processing. Compiling externals. Analysing lop* . https://pmdelgado.wordpress.com/2015/09/17/first-steps-on-pure-datas-approach-to-real-time-audio-processing-compiling-externals-analysing-lop/

APPENDIX A
DERIVATIONS

$$H\left(z,\theta\right) = \frac{\alpha(\theta)(\frac{2}{T}\frac{z-1}{z+1}) + \beta}{(\frac{2}{T}\frac{z-1}{z+1}) + \beta}$$

$$= \frac{\frac{2\alpha(\theta)(z-1)}{T(z+1)} + \frac{T\beta(z+1)}{T(z+1)}}{\frac{2(z-1)}{T(z+1)} + \frac{T\beta(z+1)}{T(z+1)}}$$

$$= \frac{\frac{2\alpha(\theta)(z-1)+T\beta(z+1)}{T(z+1)}}{\frac{2(z-1)+T\beta(z+1)}{T(z+1)}}$$

$$= \frac{2\alpha(\theta)(z-1) + T\beta(z+1)}{2(z-1) + T\beta(z+1)}$$

$$= \frac{z2\alpha(\theta)(1-z^{-1}) + zT\beta(1+z^{-1})}{z2(1-z^{-1}) + zT\beta(1+z^{-1})}$$

$$= \frac{2\alpha(\theta)(1-z^{-1}) + T\beta(1+z^{-1})}{2(1-z^{-1}) + T\beta(1+z^{-1})}$$

$$= \frac{2\alpha(\theta) - 2\alpha(\theta)z^{-1} + T\beta + T\beta z^{-1}}{2 - 2z^{-1} + T\beta + T\beta z^{-1}}$$

$$= \frac{(2\alpha(\theta) + T\beta) - (2\alpha(\theta) + T\beta)z^{-1}}{(2 + T\beta) - (2 + T\beta)z^{-1}}$$

$$= \frac{(2\alpha(\theta) + T\beta) + (-2\alpha(\theta) + T\beta)z^{-1}}{(2 + T\beta) + (-2 + T\beta)z^{-1}}$$

APPENDIX B
DERIVATIONS

$$H(z,\theta) = \frac{Y(z)}{X(z)} = \frac{a_0 + a_1 z^{-1}}{b_0 + b_1 z^{-1}}$$

$$\iff Y(z)(b_0 + b_1 z^{-1}) = X(z)(a_0 + a_1 z^{-1})$$

$$\iff Y(z)b_0 + b_1 Y(z)z^{-1} = a_0 X(z) + a_1 X(z)z^{-1}$$

$$\iff Y(z) = \frac{a_0 X(z) + a_1 X(z)z^{-1} - b_1 Y(z)z^{-1}}{b_0}$$

$$\rightarrow Y[n] = \frac{a_0 X[n] + a_1 X[n-1] - b_1 Y[n-1]}{b_0}$$