3D AUDIO FOR VR

Andrea Silvestro, Marc Illa

DSAP course project, 2021-22, Masters MET&MATT
Barcelona School of Telecom Engineering (ETSETB), UPC

ABSTRACT

This project aims to create a system that improves the experience when using VR for different scopes. The VR-related technologies have made great strides in the past few years and by just wearing simple cardboard it's possible to experience VR affordably. The efforts in developing this technology are often more focused on the visual aspects than the hearing clues. With the proposed system the user will be able to aurally locate the sound sources so that there will be no inconsistencies between the visual and audio clues, to offer, ideally, a better VR experience. There are different approaches to implement a solution based on the different existing sound formats. We have used Ambisonics (the most widely used audio format for 360 applications). To simulate a "head-tracker", we use the azimuth information through the compass of a smartphone that is retrieved in real-time thanks to the Open Sound Control (OSC) transmission protocol via UDP/IP.

1. INTRODUCTION

1.1. Ambisonics

1.1.1. Description

The roots of Ambisonics date back to the 1970s, when Michael Gerzon introduced First-Order Ambisonic [1]. Ambisonics is a full-sphere surround sound format which means that in addition to the horizontal plane, it covers sound sources above and below the listener. Differently from other multichannel surround formats, it does not carry speaker signals on the transmission channels that contain a speaker-independent representation of a sound field. This sound field is called B-format and it is then decoded to the listener's speaker setup. Thanks to this gimmick the think in terms of source directions rather than loudspeaker positions, and offers to the listener the freedom to choose the layout and the number of speaker used for playback. The B-format component channels are labelled **W** for the sound pressure, **X** for the front-minus-back sound pressure gradient, **Y** for left-minus-right and **Z** for up-minus-down. The **W** signal corresponds to an omnidirectional microphone, whereas **XYZ** are the components that would be picked up by figure-of-eight capsules oriented along the three spatial axes.

1.1.2. First-order Ambisonics: encoding

The simpler Ambisonic encoder has three inputs: a source signal **S** the horizontal angle $\boldsymbol{\theta}$ and the elevation angle $\boldsymbol{\phi}$. The source is placed at the desired angle by distributing the signal over the Ambisonic components with four channels called W (omnidirectional information), X (x-directional), Y (y-directional) and Z (z-directional)[2]:

$$\boldsymbol{W} = \boldsymbol{S} \cdot \tfrac{\mathbf{1}}{\sqrt{\mathbf{2}}}$$
$$\boldsymbol{X} = \boldsymbol{S} \cdot \mathbf{cos}\,\boldsymbol{\theta}\,\mathbf{cos}\,\boldsymbol{\phi}$$
$$\boldsymbol{Y} = \boldsymbol{S} \cdot \mathbf{sin}\,\boldsymbol{\theta}\,\mathbf{cos}\,\boldsymbol{\phi}$$
$$\boldsymbol{Z} = \boldsymbol{S} \cdot \mathbf{sin}\,\boldsymbol{\phi}$$

Being omnidirectional, the channel always gets the same constant input signal, regardless of the angles.

1.1.3. First-order Ambisonics: decoding

As mentioned at the introduction, Ambisonic-encoded signals are not feeding speakers themselves, but carry the directional information of an entire soundfield. An Ambisonic decoder must, therefore, be always designed for a specific speaker layout, and an Ambisonic-encoded soundfield can be reproduced on any Ambisonic decoding system. Ambisonic does not imply a certain number of loudspeakers used for reproduction. The only limit on the number of speakers is that the minimum number of loudspeakers L should be at least equal to the number of Ambisonic channels N. However, it is fine and even desirable to use more speakers than the number of Ambisonic channels since this increases the overall quality of sound localization. Also, the layout should be as regular as possible in order to simplify the decoding stage[2].

To do the decoding some different techniques and approaches can be used, we based our system on a physical decoding (aka basic decoding): assumption of coherent sum of all speaker signals at the centre (sweet spot). Then we can recreate the original pressure and velocity. Psychoacoustically, this decoding reproduces the impression of the original sound for frequencies below 500Hz.

1.1.4. Higher order Ambisonics

In the 1990s, it was shown that Ambisonic approach could be extended to higher orders, so that the resolution increases and

the sweet spot is enlarged by adding groups of more selective directional components to the B-format. The price to be paid is that more Ambisonic channels are introduced (2L+1 new channels per order) where L is the order: 5 more channels for second order (9 in total), 7 more for third etc.

1.1.5. Ambisonics to binaural

Ambisonics to binaural method is presented by Noistering et al in A 3D AMBISONIC BASED BINAURAL SOUND REPRODUCTION SYSTEM. The main idea is processing the audio in Ambisonics and computing the decoding for a given set of loudspeakers. With the HRTFs measurements from each loudspeaker to the listener (for left and right ear), we can convolve the audio with them and get binaural audio. This method is also called "virtual speakers"as there is no need to have speakers (only headphones), the speaker positions and HRTF measurements from these positions. As the number of speakers would be significatively lower than a set of HRTFs with a good resolution (1 degree for azimuth and 5-10 degrees of resolution for the vertical plane), this method ends up using much less bandwidth. As the computations are easier (one matrix multiplication for decoding and then some convolutions), in general it is more efficient than other methods such as VBAP with HRTF interpolation, as there is no need to do ray triangle intersection, search over triangles... Moreover, several HRTFs for different loudspeaker setups are available (ambix, sofa).

1.2. OSC

OSC stands for Open Sound Control. Matt Wright and Adrian Freed unveiled it as a result of a continued investigation. In 1994 their investigation leaded to a new protocol named ZIPI but it did not attract significant interest. Aiming to address many of MIDI's shortfalls they continued investigating until they came up with OSC protocol in 1997 [3]. In our application this protocol is used to gather the azimuth orientation from a smartphone. We are using the azimuth of a phone to simulate the orientation of the head in relation to the sound sources.

2. APPLICATION AND TECHNIQUES

2.1. Introduction

When comparing approaches based on HRTF interpolation with Ambisonic to binaural, it is easy to deduce that Ambisonic is not only easier to implement but also less expensive (algorithmically talking) [4]. It would be less expensive as HRTF functions will be time-invariant and independent of the number of sources to encode. So, there will be no need to do interpolation. Furthermore, one of the disadvantages of Ambisonic is that the sweet spot (listening area with good localization performance) is small [5], but as binaural sound reproduction does not suffer from this effect, the result will be optimal. Finally, the head rotation can be taken into account with simple time-variant matrices, which simplifies the process with respect to the HRTF interpolation methods. For these reasons our approach will be based on the Ambisonics technique. In our method, the distance from the sources is not taken into account (only the angle), but it could be implemented in the future (see Future work section).

2.2. Encoding

First of all there are the mono audio signals s_i which come from different angles β_i. Then, each angle β_i will be used to get the coefficient values of the encoding matrix for Ambisonics of order 3 (16 values in total) - although on the experiments we will try with different orders-. To get the encoded signal, blocks of 512 samples from the corresponding input buffer will be used and just like any other Ambisonic's encoder, being multiplied for their corresponding encoding matrix.

2.3. Decoding

Once there are all the encoded signals, each one in a matrix of 16 coefficients by 512 samples each, the following step is decoding the signals for the corresponding layout. Our layout will be a virtual scenario as the audio will in fact be reproduced via headphones. The decoding matrix has been taken from Ambix [6] plugin suite corresponding to measurements done at Institut für Elektronische Musik und Akustik (IEM) in a cube with 24 loudspeakers. So, the only thing it is needed to do is sum all the encoded signals and multiply the decoding matrix by them.

Fig. 1. the Cube at IEM [7]

2.4. Convolving the audio

At this point there is a matrix of 24 channels with 512 samples; each one corresponding to the 24 loudspeakers of our (virtual) layout. In order to reproduce the sound on the headphones it is needed to convolve this signal by the impulse response of each loudspeaker (which has also been taken from the Ambix plugins) as it is a virtual environment and we need to simulate the sound that would arrive to the inner ear from

each loudspeaker. So, in short, our application decodes Ambisonics to virtual loudspeakers and then convolves the virtual loudspeaker signals with the HRTFs corresponding to the loudspeaker position in space, to produce the filtered signals for each ear (left and right headphone signals).

For the convolutions, an algorithm called Overlap-Add will be used, as it takes advantage from the efficiency of the Fast Fourier Transform (FFT) and is more efficient than implementing a simple time-domain circular convolution [8]. Efficiency is needed as it is a real-time application and the lesser delay during the Digital Signal Processing (DSP) there is, the better. The FFT is needed to get the signal representation in the frequency domain, as multiplying on the frequency domain equals point-wise circular convolution in the other domain (time domain) [9], which is what needed to do (convolve the audio signal with the impulse response of its corresponding loudspeaker).

2.5. Output the audio

Once the convolutions (between the signal and the virtual loudspeakers impulse responses) are applied for left and right channels (24 convolutions each), the left signals are summed among them and so are the right ones. Then, the resulting signals are multiplied by the master volume and sent to the output buffer (left output buffer and right output buffer respectively) from which each output channel of the sound card will be reading, and thus outputting the sound.

2.6. Implementation

The code was written in c++14 using JUCE framework [10] for the GUI and audio management, the library "FFTCONVOLVER" [11] for the convolutions [11], the library "OSCPACK" [12] for reading the values of the smartphone's azimuth with the use of a third-party app named oscHook[13] which reads the sensors of a smartphone and send their values over OSC. The version control of the project was done using git and Bitbucket where the project can be found [14].

3. EXPERIMENT AND DISCUSSION

3.1. Software interface

The user interface is simple and intuitive. The program accepts two channels were to receive an input that can be enabled and disabled. For each channel there are its corresponding azimuth and elevation with which panning the source. There is a calibrate HT (head tracking) button that initializes to 0 the value of the head tracker so that the azimuth value is relative to the initial value sent by the compass of the application.

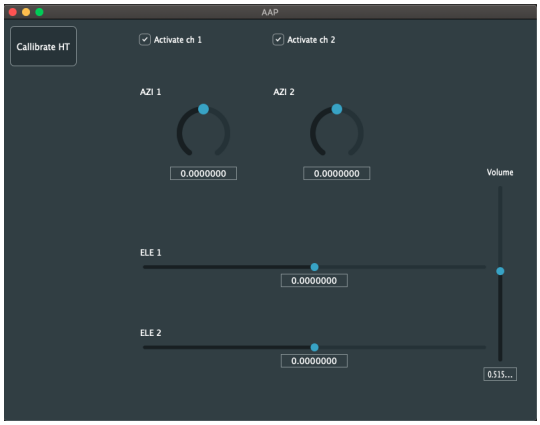

Fig. 2. User interface

3.2. General testing of the app

For testing the app we routed some tracks from a Reaper [15] session to the input of our application with Soundflower [16]. We tested it with several audio tracks, we played with the volume, azimuth and elevation controls and also with the headtracking from the smartphone. From this test, perceptually, we could observe that: 1) The audio did not present clicks due to a sudden change on azimuth. 2) There was no perceptual latency in the system. 3) Without headtracking the system is stable. 4) If using headtracking it works fine but needs to be calibrated and the smartphone needs to be on a flat surface.

3.3. Experiments

The experiments we carried where the ones we present below:

1- Comparison with stereo panning: When comparing the aforementioned application with a simple stereo panning we could see several differences: Firstly, it was easy to notice that while stereo panning did not change the timbre of the sound sources, our application changed it, as well as it added some reverberation of the room. This would be a problem if we wanted to stick to the original timbre of the sound source. On the other hand, our application provided a better space resolution as it could pan the sound sources to any place in a 3D-sphere, while the stereo panning could only place it anywhere on a line between the left and right ear (indeed when panning is at the centre we heard the sound sources coming from inside our head, which of course our application could not). However, although in overall the location of the sound sources is very good, we could see that it was difficult to spot if the position was on the front or on the back when the azimuth angle was exactly 0 or 180. This is probably due to the well-known front-back confusions [17].

2- With head-tracking versus no head-tracking: Another test we carried was comparing our application with and without the use of the head-tracker (the information of the compass of a smartphone being sent via UPD with the OSC pro-

tocol to our application). When using it, we could rotate our head and the sound sources rotated accordingly to maintain their relative position to the head rotation. Indeed, as it is not a real head-tracker (it would be an expensive solution and a development of one would have gone out of the scope of this project), we needed to rotate the smartphone instead of the head, and it needs to be in a flat surface to be precise, so the rotations were not very natural as we had to rotate both at the same time. However, we could appreciate that this would probably be a good improvement, as when switching it off, the experience was poorer. Also we noticed the importance of calibrating it at the beginning of each use, to be consistent with where the angle 0 is. This is why we added a calibrate button on the app, that makes this process very easy. An important improvement we could see with the head-tracking is that it solves the problem of the front-back confusions we mentioned on the stereo panning test, as it with a small rotation is very easy then to notice if the sound source is at the front or at the back.

3- First-order Ambisonics versus High Order Ambisonics: Another experiment has been trying different order of Ambisonics, more concretely we compared first-order Ambisonics (4 channels) versus third-order Ambisonics (it is a high-order Ambisonics) which, as explained, has 16 channels. The result of first-order Ambisonics was that although we could more or less say a zone where the sound source was, the source was blurry and was not clearly located on the space; while with a HOA (3rd order) the resolution was much higher and the overall experience better. Also it is worth mentioning that there was no effect on the latency of the system and the CPU cost was kept low (about 20 percent on a 2018 macbookPro, intel core i7).

4- Different number of loudspeakers (10,12,14,24): This experiment consisted of using a different number of loudspeakers of the room. Doing this with a high-order Ambisonics configuration we could see that with 24 loudspeakers the resolution was much better. However we still got decent results with 10,12 and 14 loudspeakers (better results than 24 loudspeakers with first-order Ambisonics). This means that although the number of speakers potentially increases resolution, if we come from a low resolution encoding it is not very useful as the sound scene is not represented with enough detail. Indeed, we also tried to compare the number of loudspeakers with first-order Ambisonics and actually there was not much difference between 14 and 24 loudspeakers. So, from our experience, it could be said that the number of loudspeakers increases the resolution as long as the encoded signals have enough resolution.

5- Different HRTFs: The last experiment we carried was comparing how the sound scene was perceived with 3 different head-related transfer functions (HRTFs): the first is named h1, the second, h2dummy, recorded with a dummy head microphone, and the third h2dpa, which is the same as the second but recorded with a dpa microphone. They were tested with HOA and 24 loudspeakers (best configuration we found from previous experiments). We found out that both h1 and h2dummy performed very similar, indeed blindly testing it was very difficult to know which was which. However, with the h2dpa the sound scene was much more diffuse. This is probably because the dpa microphone was not capturing the HRTF as well as the microphone inside the ear of the dummy head where these HRTFs were recorded.

3.4. Future work

Some of the ideas of immediate further work that could be done from this application include: 1) Comparison with VBAP panning which might improve our application or confirm that we used a good algorithm, 2) implementation of the distance encoding to make it even more realistic, 3) ability to change of the ambisonic order, HRTF and number of speakers from the app as we have it hardcoded on the code and is not user friendly, 4) selection of the input and output devices (at the moment we use the one the computer is using), 5) develop a version of the application in VST format (by now it is a standalone application) in order to use it from any audio DAW (digital audio workstation).

4. CONCLUSIONS

With this project we have been able to achieve two big goals:

First of all we developed an application for simulating sound sources in a 3D scene. We achieved it and the application does not present latency, clicks or any artifacts so it is suitable for a real-time application which was the main goal. We also have been able to implement a head-tracking which turned out to be an important feature to solve the front-back confusions.

Secondly, we have been able to carry out several experiments where we have seen the importance of several aspects of Ambisonics and the perception of a 3D sound scene: We have seen the differences in behaviour with stereo panning, how our application could represent better a 3D sound scene but also how it added colouring to the timber and how we faced front back confusions. We have been able to solve the front-back confusions with the head-tracker, compared HOA with first-order Ambisonics and showed how the number of virtual speakers affects the resolution but how it also depends on the order of Ambisonics we are using. Finally we have seen that the HRTF are key to represent the sources on the 3D space and an inaccurate recording leads to a diffuse representation of it.

It could be said that in this project we have not only provided with a prototype to render sound sources in the 3D space but also had an insight of which are the trade-offs, techniques available and how important are some of the elements to provide an Ambisonics encoding, decoding and playback application.

5. REFERENCES

[1] M. Gerzon, "With-height sound reproduction," *Journal of the Audio Engineering Society*, 1973.

[2] Florian Hollerweger, "An introduction to higher order ambisonic," *Florian Hollerweger Website*, 2013.

[3] Dave Phillips, "An introduction to osc," *https://www.linuxjournal.com/content/introduction-osc*, 2008.

[4] Alois Sontacchi Markus Noisternig, Thomas Musil and Robert Holdrich, "3d binaural sound reproduction using a virtual ambisonic approach. in virtual environments, human-computer interfaces and measurement systems," *VECIMS'03. 2003 IEEE International Symposium on*, pp. 174–178, 2003.

[5] Audun Solvang, "Spectral impairment of two-dimensional higher order ambisonics," *Journal of the Audio engineering Society*, vol. 56(4), 2008.

[6] "Virtual reality audio format adopted by google. more info about ambix plug-ins on http://www.matthiaskronlachner.com/?p=201," .

[7] "https://github.com/kronihias/ambix/blob/master/ambix_binaural/presets/iem_cube_h2_mb/cube_bw.jpg," 2019.

[8] E. Bryan George and colleagues, "Speech analysis/synthesis and modification using an analysis-by-synthesis/overlap- add sinusoidal model," *IEEE*, 1997.

[9] "https://en.wikipedia.org/wiki/convolution_theorem," 2019.

[10] "https://juce.com/," 2019.

[11] "https://github.com/hifi-lofi/fftconvolver," 2019.

[12] "http://www.rossbencina.com/code/oscpack," 2019.

[13] "https://play.google.com/store/apps/details?id=com.hollyhook.oschookhl=it," 2019.

[14] "https://bitbucket.org/marc_95/aap/src/master/," 2019.

[15] "https://www.reaper.fm/," 2019.

[16] "https://github.com/mattingalls/soundflower," 2019.

[17] "https://ieeexplore.ieee.org/document/4107369," 2019.