Bits: a Permanent Virtual Installation

Gabriel Zalles Ballivian
Department of Music
UC San Diego
San Diego, CA
gzalles@ucsd.edu

***Abstract*—*Bits* is a permanent virtual installation that features a multi-tile display, spatial audio, poetry, video, and geometric primitives. The installation was built using the WebXR platform A-Frame and was inspired by the classic computer music work *Poème électronique* by Edgard Varèse, written in 1958 for the Brussels World Fair. In this work, we explore the concept of digital greed or virtual hoarding, which we define as the propensity for people to accumulate or amass data without regard for the human, ecological, or psychological impact of this practice. This work is also part of a larger exploration of open-source spatial audio tools, which can be leveraged to improve access to elite technologies.**

***Index Terms*—spatial audio, art, media, VBAP, music**

I. Introduction

> The emergence of many new media art genres calls into question qualitative issues in regards to performance in virtual and electronic spaces. What constitutes performance in technological art, and how can we form a critique of new media performance by analyzing these aesthetic spaces? [1]

Bits is a permanent virtual installation created in A-Frame that explores the concept of data hoarding or digital greed. At various points during our academic studies in music departments throughout the US, we had the inclination to create a musical work that featured scale as one of its major defining parameters. Given our background, and financial limitations, it was clear that physical scale would be difficult to accomplish, so instead digital scale was pursued. This meant using a large collection of data types and as many techniques as possible in a single experience.

In order to achieve this work, a small but considerable number of high-quality images were scraped from the internet and sequenced rapidly to create a stochastic video file by using command line tools, MAX/MSP, and Quicktime. The video was also at various points transcoded using Handbrake to reduce the memory demands of the project. The original work had multiple instruments, but ultimately we decided to feature only a single sound-making algorithm - a rapid sampling system that drew from a collection of 10,000 audio files in the FSDKaggle Dataset [2]. The algorithm randomly selects a sample from the set, applies a random pitch-shifting value, and also randomly spatializes the signal using the VBAP external in Puredata [3].

Fig. 1. Varèse

II. Poème électronique

Edgard Varèse composed *Poème électronique* in 1958 for the Brussels World Fair [4]. It is one of the seminal works of computer music, due to its gargantuan scope, made even more impressive due to the time period in which it was created. The composition featured one of the largest speaker arrays ever assembled, in addition to video, lights, and other theatrical elements. The piece is eight hours long in total and was divided into one-hour segments.

This composition is one which I have known about for many years. Much like in the original composition, this project seeks to utilize scale as a key parameter of the musical composition. The number of speakers relative to the original work is substantially smaller this is in part because we want

to keep the bandwidth requirements of the project relatively low. This is one of the key differences between working in the physical and digital world - if the bandwidth requirements are exorbitant, few people will be able to access the oeuvre.

There are a total of eight virtual displays which project the video on a loop but each display begins playback at a different starting point. The goal was to create enough randomness in the number of images and sounds that one feels as though the entire experience is entirely aleatory. At this time the amount of text in the scene is restricted to a very small amount and no additional geometries other than primitives have been imported. Besides using vast amounts of data, we want to eventually create an environment that features many different types of media such GLTFs [5] and animations - the problem is conveying this data elegantly.

III. VBAP

VBAP [6] is a panning algorithm that relies on linear algebra principles to calculate the distance from a point inside a triangle to the three vertices that comprise it. This distance result is then used to calculate the corresponding gains for those speakers. In this project, we make use of the VBAP external which can be found via Deken in Puredata. We created an abstraction that randomly selects a sample from a bank of 10,000 possible choices and plays back the audio in a virtual octophonic configuration, which is then recorded. The sampling of these sound files is done in rapid succession to coincide with the stochastic and abrupt nature of the video file. The files also go through random pitch shifting before being reproduced. In the future, we would like to add an even greater number of effects to create even less predictable material and increase the number of possible combinations or permutations.

In 2D VBAP the strength of each speaker is controlled by a gain factor g_i. The basis is composed of unit-length vectors directed toward the virtual source [7]:

$$\begin{aligned} p &= \begin{bmatrix} p_1 & p_2 \end{bmatrix}^T \\ &= g_1 l_1 + g_2 l_2 \end{aligned} \tag{1}$$

Where T denotes the transposition operator, g are the gain factors and l are the unit vectors.

In matrix form:

$$p^T = g L_{12}$$

Using the right inverse of the matrix L, we can solve for g,

$$g = p^T L_{12}^{-1}$$

Three-dimensional VBAP has a similar solution but requires three loudspeakers to produce a virtual source.

The gain factors are computed using the same technique,

$$g = p^T L_{123}^{-1}$$

Figure 2 shows one of the voices of our [clone] object which randomly selects a sample from our set, applies the pitch shifting, generates a random position for the source, and spatializes the material. The result, due to the quick nature

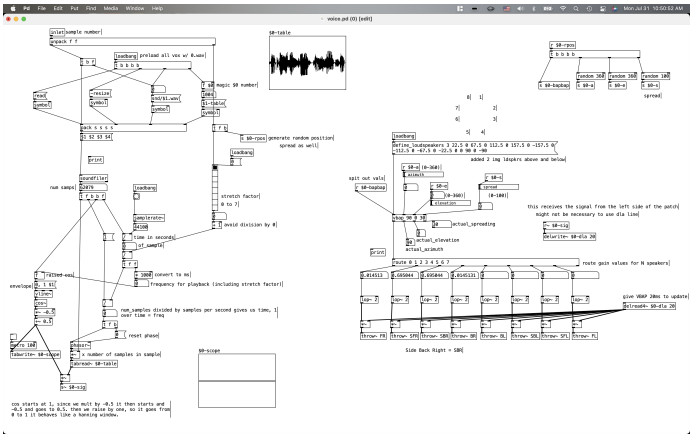

Fig. 2. The heart of our Pd patch.

of the sequencing, feels similar to a granular synthesizer in which the grains originate from a corpus of sources, rather than from a single file. In order to create our virtual auditory display, the VBAP channel feeds are recorded and attached to virtual elements in the A-Frame [8] environment.

IV. Web Audio API Binaural Filters

The Web Audio API (WAA) contains certain functions for spatializing sound and is reasonably comprehensive. The WAA [9], in contrast to external JS libraries, describes the capabilities of any browser using this framework[1]. In contrast to ambisonic libraries, the WAA does not need to be imported into one's project, as it is internally contained with various browser distributions.

There are three main functions for spatialization within the WAA:

1) **Panner Node**: which allows for *equal-power* (e.g. traditional stereo panning), or HRTF panning[2],
2) **Distance Effects:** which calculates gain values based on the position of listener(s) and source(s); sounds are amplified as a listener approaches a source as in real life,
3) **Sound Cones**: determine the directivity of the sound source (e.g. omnidirectional, cardioid, etc.).

Unfortunately, the workflow for creating spatial music using the WAA is slightly more complicated than other tools, since it requires one to program every sound source in JS/HTML, rather than simply importing a sound field created using a more conventional tool - such as a DAW. *A-Frame* and *THREE.js*[3], provide WAA-based positional audio which allows us to map geometries to sound sources. This makes the WAA a very powerful option, especially in circumstances where the number of sound sources is small. It has also been noted by researchers that audio localization is a multi-sensory task, seeing an object

[1]The WAA defines how these methods must be implemented to conform to the standard, it is not a library per se.

[2]Using interpolation methods rather than ambisonics.

[3]Related via the fact the A-Frame is built on THREE.js

and hearing an associated sound can help us discriminate source positions.

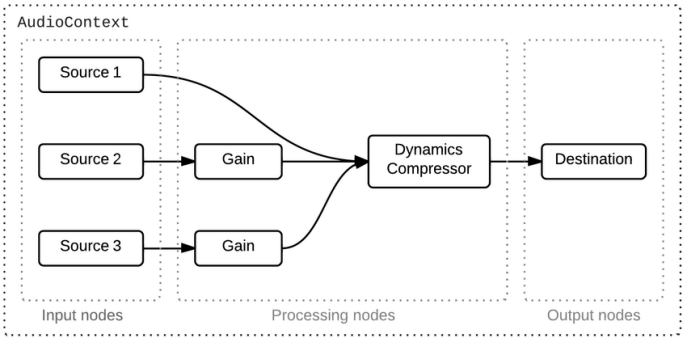

Fig. 3. The WAA graph showing the hierarchy between AudioContext and children.

A. Inside the WAA

Carpentier [10] provides a good summary of the capabilities and limitations of the WAA. According to the author, both Mozilla's Firefox and Google's Chrome browsers are based on the Blink engine [11], which has the following deficits:

1) Filter kernels are not partitioned, which causes extra latency of half the FFT (e.g. *Fast Fourier Transform*) size,
2) The IRCAM Listen HRTF data set [12], is hard-coded into the panner method, and,
3) The documentation does not explain how the HRTF set was selected, or why the *Impulse Responses* (IRs) were truncated to half their size - both of which likely affect the psycho-acoustic character of the system.

Despite its limitations, we believe the WAA is one of the most accessible and complete tools to disseminate spatial audio using FOSS. In contrast to game engines, one does not need to purchase any additional software to start programming for the web, and there are countless sources for free assets online. We believe web applications using spatial audio will proliferate in the next decade, as many systems start to become cloud-integrated and browser-based.

V. Digital Installation

The concept of a virtual installation is not something that is largely discussed or understood. It is a common experience in museums and galleries for artists to create installations, either interactive or passive, that patrons can experience. When we shift to the digital domain certain limitations of this medium are no longer present. Usually, an exhibit might be featured in a museum for several months and it takes up one room in the building. In the case of digital installations, however, the physical and temporal restrictions are naught.

We can create a virtual environment that can be visited at any time, be arbitrarily large, and theoretically exist forever. The reason why I labeled this work a virtual installation as opposed to an XR experience is because in my mind it has no beginning or end. There is a finite amount of audio and video material, but the hope is that it's stochastic enough that one can stay indefinitely if desired. The starting point, ideally, does not change the experience of the user, who can choose to leave at any time - like in the real world.

I believe this is an interesting and meaningful distinction we will see more artists incorporate into their works in the coming decades. The idea is that now we can create experiences where the user can be suspended in time, a bit like an open-world video game, except that in these worlds there really no objectives, points, or winners. The benefit of this genre is also that it enables access to elite spaces, such as galleries, by treating the internet as the world's largest museum.

VI. DEI Statement

In recent years my work process has shifted to include more FOSS as a philosophical impediment of the oeuvres. The idea is that everything from process, product, and purpose, can be replicated by as many people as possible of limited means. Rather than relying on multi-million dollar spaces which are restrictive and protected, we can manufacture similar worlds which feature these components, in a synthesized form. The multi-channel/multi-display experience of *Bits* would like cost thousands of dollars to realize in the physical world. This project is part of my larger initiative to shift my process to focus on accessible software and hardware. I am originally from Bolivia and part of the motivation for this is being able to teach courses in my hometown that are economically viable for the people in them. That means that individuals who don't own HMDs or powerful desktop computers can participate using existing infrastructure.

VII. Future Work

This project, as it currently stands, is just a fraction of what we believe this piece could be. The most appealing direction for us is one where we can integrate a JS library to request samples and images from an online repository in real time. Let this repository be public, and its size of it be very large. The level of indeterminacy, and by proxy, the number of permutations is also quite low in its current state since both the audio and the video are pre-described. It might be possible to compile our Pd code using WebPd which would translate the patches into Javascript. This would then allow us to perform many of the random operations in real time but be able to maintain the virtual scene.

VIII. Conclusion

This position paper has described a multimedia installation living on the internet on a permanent basis. The piece entitled *Bits* features a synthesis algorithm that randomly samples data from a modestly large corpus, both in audio and image form. The composition seeks to address the concept of digital greed: a human disposition to accumulate and consume data, without regard for the physical consequences. Data is the new oil, and we are all producers. Paradoxically, this composition

underscores this issue by using a sizeable database[4]. The scene also features spatial audio and multi-tile display, in an attempt to subvert the expectation that production value equates with the strength of the message.

REFERENCES

[1] P. Lichty, "The cybernetics of performance and new media art," *Leonardo*, vol. 33, no. 5, pp. 351–354, 2000.

[2] 2023. [Online]. Available: https://www.kaggle.com/competitions/freesound-audio-tagging/data

[3] M. S. Puckette *et al.*, "Pure data," in *ICMC*, 1997.

[4] E. Varèse and C. Wen-Chung, "The liberation of sound," *Perspectives of new music*, vol. 5, no. 1, pp. 11–19, 1966.

[5] F. Robinet *et al.*, "gltf: Designing an open-standard runtime asset format fabrice robinet, re mi arnaud, tony parisi, and patrick cozzi," in *GPU Pro 360 guide to 3D engine design*. AK Peters/CRC Press, 2018, pp. 243–260.

[6] V. Pulkki and T. Lokki, "Creating auditory displays with multiple loudspeakers using vbap: A case study with diva project," in *International Conference on Auditory Display*, 1998, pp. 1–5.

[7] 2020. [Online]. Available: https://vocal.com/speech-processing-and-audio/vector-base-amplitude-panning/

[8] 2023. [Online]. Available: https://aframe.io/

[9] "Web audio api," https://webaudio.github.io/web-audio-api/index.html, (Accessed on 04/26/2021).

[10] T. Carpentier, "Binaural synthesis with the web audio api," in *1st Web Audio Conference (WAC)*, 2015.

[11] "Blink (rendering engine) - the chromium projects," https://www.chromium.org/blink, (Accessed on 04/26/2021).

[12] "Listen hrtf database," http://recherche.ircam.fr/equipes/salles/listen/, (Accessed on 04/26/2021).

[4]Compared to large companies, however, this is minuscule.