“Not A Dance”

ITP Thesis Paper

Peter Holzkorn

28 Apr 2011

1. Abstract

Not A Dance is a single-player gestural audio game in the form of an installation. The primate of visuals over sound one finds in most digital games is reversed: The player navigates a sonic landscape by ear as the mapping of virtual and real space allows spatial audio perception via wireless headphones.

2. Introduction

2.1. Motivation

My interest in game design is rooted in the belief that games allow for a spectrum from Hollywood-style mass-entertainment to sportive competition, casual tinkering and authorial, multi-layered works of art. But only recently have both possibilities for small-scale publishing and an increasing acceptance of games as a form of culture led to a wealth of games that can be called experimental or artistic. It is a better time than ever to be designing games.

With the release of the *Wii* (2006), *Playstation Move* (2010) and the *Kinect* (2010), mass-market digital games have been increasingly incorporating physical activity. While the distribution issue with these platforms prevents the kind of independent production that we can find with iPad apps, for example, it is worth exploring their possibilities for more experimental games.

My hypothesis is that since such interfaces are based on gradual physical movement and (often) algorithmic estimation of input, the interaction paradigm is fundamentally different to that afforded by precise controllers, and that its particularities can be harnessed for unique forms of expression and entertainment.

My first experiment with gestural game interfaces was a wiimote-based competitive/ collaborative game prototype I developed in 2010[1]. The release of the Kinect and its technical and financial accessibility motivated my experimentation with "controllerless" input.

All of the mentioned platforms have one thing in common: Increasingly free-form movement, but a visual focus of attention on the screen. The tension between these characteristics motivated me to try and work on a different assumption - by removing the screen as the centre of attention, and reversing the general hierarchy of visual and aural cues.

This led me to a spatial audio game, as described below. My goal was threefold:

1. **Design an experimental game for a new motion-based interface,** and learn about the technical & design-related challenges involved, in particular regarding the control scheme.
2. **Design a navigable soundscape** that is both playable and aesthetically interesting, and learn more about sound generative sound synthesis and signal processing.
3. **Reflect on the performative aspect of gameplay**. Motion-based games put the player in the position of a physical performer. Combined with the sound-focus, the experiences for player and audience might be very different. This aspect is discussed in section 6.

I want to emphasize that I aimed to create an experience that uses the structure of a game, hoping to contribute to the expansion of the boundaries of digital games as a medium.

2.2. References

Three broad directions of interactive experiences provided references for the design process:

1. Audio Games. "Pure" audio games (i.e. those that offer nearly the same experience when played with and without visual perception) can be categorized in various ways, but for my purposes it is most interesting to note whether a game was explicitly designed for vision-impaired people or not.
2. Sound toys and (generative) audiovisual systems. From the *Theremin* to the *Reactable*[2], systems that generate sounds (and sometimes visuals) based on gesture- or touch-based interfaces allow rich experiences without the confines of a game. "Confines" means that I interpret a game as a "system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome" (Salen & Zimmerman 2003). A "sound toy" or even instrument such as the examples mentioned do not pose a "conflict" nor result in a "quantifiable outcome", yet they are *playable*. The expanded field of this category includes spatial audio installations such as *Audio Space*[3] that are specifically concerned with mapping virtual and real sonic space.
3. Gesture-based games. Obviously, *Kinect, Move* and *Wii* titles, while generally targeted at a mainstream casual-gaming audience, are great examples of gesture-based game design.

2.3. Game Description

The player stands in the middle of a 9'x9' area, wearing wireless headphones. The player starts to hear a sound. As the player rotates, they notice that the sound appears to come from a certain direction, as if an invisible sound source was floating in the air somewhere.

The player's mission is to collect the sounds. In the simplest variant of the game mechanics, that is accomplished by closing up to it and "touching" it in virtual space.

The design of the control scheme was a big part of the exploration, both in terms of technology and interaction design. The process and final scheme is explained in section 4.1.

There is also a visual component that shifted its role during the development process, see section 4.4. The primary purpose of the visual component is to complete the experience for the audience who doesn't have the spatial audio information the player has.

[1] http://holzkorn.com/?p=tear

[2] http://www.reactable.com/

[3] http://www.theowatson.com/site_docs/work.php?id=15

2.4. Technical Overview

The technical part of the project consists of four major components.

1. The user's orientation and hand gestures are tracked by a *Kinect* the data output of which is processed in a *C++/Cinder* application that uses the open depth image processing middleware *OpenNI/NITE* and a Cinder-specific wrapper of that. This allows the software to provide realtime updates of the position of the player's limbs and joints.

2. The Cinder application calculates the results of the player's interaction with a virtual world that contains static or moving objects. All data relevant for sound generation is sent to the application *Max/MSP* via *OpenSoundControl* (OSC).

3. The audio engine is written in *Max/MSP* and transforms the sounds so that the listener's perception of their location coincides with their position in virtual space. The sounds are also processed for various purposes, using techniques of filtering and subtractive synthesis. Frequency and waveform information for the currently active sound sources and certain sound-dependent game events are sent back to the Cinder application via OSC.

4. The Cinder application provides graphical output for the audience in the form of a top-down view of the game world. The representation of the sounds includes a visualization of their spectral characteristics.

3. Game Design

The following is a condensation of thoughts and theories that I developed and adopted over time. As they pertain to the specifics of the game, in particular, they represent the state of thought *after* the development of the project. A more chronological account of changes in design decisions, theme etc. is given in section 5.

3.1. What makes a good game?

What makes an engaging game? Broadly speaking, games need to be "fun". But what does this mean? Raph Koster: "Fun in games arises out of mastery. It arises out of comprehension. ... with games, learning is the drug." (Koster 2004, p. 40) In that it poses a challenge and draws a big part of its appeal (or "fun") out of letting the player learn how to rise to that challenge and master it, a digital game is fundamentally different from movies, novels and music, elements of all of which many games incorporate.

Games incorporate all kinds of other elements than the challenge that one has to master. Roger Caillois, in his description of play (Caillois 2001) coined a useful distinction of *agon* (the struggle, or challenge), *alea* (fortune), *mimikry* (make-believe, or fantasy) and *ilinx* (intoxication). These categories go beyond games, but show the wide range of engagement playing can provide. In games, different audiences may place different emphasis on certain elements: Jesse Schell (2008) suggests that men and women are very different in what games they find engaging (especially in respect to mastery and competition) and Jesper Juul (2009) recently devoted a whole book to the rise of casual gaming and its characteristics, as opposed to "hardcore" gaming.

There are also games that subvert Koster's idea completely, and they tend to lie on the border to toys and/or art.

These notions are important because I needed to choose how much the idea of "fun" (in Koster's sense) should inform the experience. The hardest aspect of designing a game mechanic is, arguably, to balance it: In a single player or cooperative multiplayer game, the challenge is to get "just the right" amount of difficulty that rises with the player's progression. In competitive games, there can be no set of moves that always wins or always leads to a draw if the game wants to appeal to an audience beyond young children.

In summary, if one does not want to subvert the notion of a game at its very core, it is almost unavoidable to pose a challenge to the player that they can master.

As we have seen, there are many elements that can make a game experience engaging. In this project, I emphasized 1) the challenge of navigating by ear, 2) a rather direct mapping of real and virtual space by gesture-based control and 3) the aesthetic experience of perceiving sound in an unusual format.

Accordingly, three main design challenges became apparent:

3.2. Spatial Audio and Game Mechanics

The first challenge is that of balancing the difficulty in an audio game that makes spatial navigation the core task is that for navigation we rely most heavily on our visual perception. How do humans locate sounds without visual aids?

> "At the two ears, the sound signals arrive with differences in time and amplitude. Sound from a source located at the side of the head travels a longer time to the contralateral ear and suffers frequency-dependent damping due to diffraction and absorption. [...] In the median-plane [these effects] are small. Therefore the precision of localisation is less than with clear interaural cues. Nevertheless, humans can distinguish between frontal, up, or back direction, due to elevation of the monaural cues. Monaural cues are identical at both ears and represent the spectral differences with reference to a free sound field or with reference to a specific direction, usually the frontal incidence."
>
> (Vorländer 2007. p. 87)

In everyday life, however, we have additional visual information that facilitates, for example, the distinction whether a sound is in front of or behind us, and whether it is above or below us.

Compared to games that rely on both visual and auditory cues to aid navigation, such as perception- and reaction-intensive First Person Shooters, audio-only games have to give the player much more time and fewer objects that have to be perceived and processed simultaneously.

Also, the nature of the sounds is critical. For locating a sound behind us, for example, we rely, as described above, on spectral differences to how it would sound if it was right in front of us - this is obviously easier to judge for sounds we are familiar with. As I am using synthesized sounds, I have to ensure that this is compensated well enough by other clues.

Additionally, relative loudness (in dB) is logarithmic in its relation to physical sound-wave energy, and perceived loudness (in phon) is uneven across the frequency spectrum at constant relative loudness (e.g. a sound of 80 dB at 8.000 Hz will be perceived as much louder than a sound of 80 dB at 500 Hz).[4]

All these considerations impacted the choice of a particular sound spatialization technology and the general soundscape design.

One game, although strongly narrative-oriented, helped in the assessment of what is possible in a pure audio-game: *Papa Sangre*[5], an iPhone game in which the player navigates what is suggested to be a version of the underworld in order to save lost souls. Although slightly tongue-in-cheek, it is an unexpectedly scary game. It demonstrates both the range of challenges possible in a headphone-based game designed around navigating by ear, and the immense immersion achievable by forcing the player to increase the amount of imagination contributed when faced with the deprivation of visual perception

3.3. Gesture-based Navigation

In fall 2010, Microsoft released the *Xbox Kinect*. The Kinect is a peripheral for the Xbox360 Console that allows the game software to track the position and orientation of the players' bodies, arms and legs. It can be regarded as the next step in controller technology after Nintendo's *Wiimote* that allows a remote-like controller to be tracked in terms of its position and orientation, and the *Playstation Move,* essentially a more accurate version of the Wiimote.

It is no accident that these controllers came at the same time as a wave of "casual games" arrived on smartphones, web portals and consoles. One primary allure of these controllers is described by Emily Newton-Dunn of EA Bright Light, a Wii games developer (EDGE 02/11, p. 82): "These days you look at a controller and it's a complicated piece of kit. A lot of the motion controllers have taken that element out of the equation. It's simplified it again - it allows you to focus more on what's going on on the screen than what's going on in your hands." David Braben, a developer of Kinect games, argues that motion controllers' natural mapping flattens the learning curve: In controllers that use more convention-based mapping (e.g. buttons and joysticks), the time investment required is too much for many people (ibid.).

---

[4] see Vorländer 2007, p 87f.

[5] http://www.papasangre.com/

The introduction of these controllers has generated an interesting situation: The majority of motion-based games are sports-, dance- and party games. The core audience for these games are casual players that would hardly be interested in games that require much time investment or a mastery of complex strategies. The "traditional", "hardcore" gaming audience, on the other hand, might look at these games like a dedicated chess player might look at monopoly - fun, but hardly as interesting or deep and experience as "real" games.

At the same time, the "natural" interface doesn't mean it is easier to create control schemes. Quite the opposite: Motion controllers generate much more ambiguous data, making the processing of gestures far harder and less clear than the processing of button and mouse inputs, thereby posing all sorts of challenges for developers trying to establish a widely applicable mapping of controls to gestures (ibid.).

Finally, while traditional controllers can rely on a rich basis of conventions (the start button pauses console games, an on-screen button clicked with the mouse only activates on mouse-button release inside the on-screen button boundaries), there are few such conventions for motion controllers (ibid).[6]

In my project, I emphasize the unusual situation that the player can play the game facing in any direction, which contrasts the typical screen-orientation of most motion-controlled games. I decided to focus on a small set of possible actions to satisfy time constraints and keep the complexity for the player low. Still, while turning affords a perfectly natural mapping, the gesture for walking would have to be conventional, as the player has to stay in a confined space to be tracked. The initial choice fell on a simple extension of one's arms in the desired walking direction, like a person walking in the dark might do to feel for obstacles. As it turned out (see section 5), this control scheme was not well received and had to be revised.

In the final scheme, real and virtual space are mapped as naturally as possible, allowing the player to physically walk up to a sound source. There are three game modes: In the first mode, it is enough to walk up to a sound to collect it. In the second mode, the sounds are flying by above one's head and one has to be close and "catch" them with a raised arm. In the third mode, a sound consists of two differently pitched components, and one can generate similar signals by moving one's arms. To collect the sound, one has to get close and then position one's arms so that the sounds generated by the arms match the object's sounds in pitch. This last mode is much more complex, but hints at the possibility of more music-related variations of the game.

3.4. Aesthetic Engagement

The third challenge is to create a compelling audiovisual experience that, in itself, is a reward for playing the game. In non-interactive "media art", the viewer/listener can receive the content passively, and then form a judgement. Some of the audiovisual art I find most impressive is completely non-interactive - it is so engaging because the author made a series of choices, and the only requirement for the viewer is to consume the experience passively. The same series of events can be guaranteed for all viewers, guaranteeing a high amount of authorial control.

It lies in the nature of a game, however, that without active user/player participation the experience is severely constrained; and the series of events completely depend on the player's actions, allowing for frustration/stagnation as well as beautiful unanticipated situations.

A reference for design questions in this regard are games that appear to value the aesthetic experience over the agonistic element. This is certainly not an objective measure, but it is obvious that competitive multiplayer games clearly fall outside of this category, as do, for less easily explainable reasons, most puzzle games, platformers, strategy games and shooters. Adventure games and role-playing games typically emphasize the aesthetic element (the fantasy, or mimikry), but base it on a strong narrative structure.

[6] For a first attempt to collect universal patterns for touchscreen and motion controllers, see (Saffer 2008)

In contrast, independent publishing has allowed a niche category of abstract aesthetic games to flourish. Most notably, Thatgamecompany's *Flower*[7] and *Flow*[8] are definitely games rather than toys, but the player's reward is not learning how to master a challenge, or advancing in a story they can influence, but moving through a poetic, aesthetically interesting landscape, be it minimalistic (*Flow,* in which the player navigates an aquatic creature trough a deep-sea landscape) or lush and more concrete (*Flower,* in which the player is a breeze carrying petals that activate more and more flowers). There are more experiments, such as *Linger in Shadows*[9], an interactive abstract graphics demo, and most probably not a game anymore.

When is any work of (at least moderately) abstract art aesthetically interesting? There is no simple answer, of course, but I suggest that, in general, it has to stir imagination and strike a balance between novelty and familiarity.

The opportunity in designing an abstract audio game lay in the fact that we are used to different modes of listening (see Chion 2011): We listen for the cause/origin of a sound (in crossing the street, for example), for the semantic content (in speech) or for the quality of the sound itself (in music). We are very used to certain situations that accompany these modes, and a spatial audio game is an excellent environment to play with these conceptions.

Hence, I was aiming for a sonic landscape of dynamic but still sufficiently harmonic sounds that would stir curiosity in the player, and afford new perspectives on auditory perception.

There is a very loose narrative of an association of sounds and memories that was apparent in the first version of the game, and continued to inspire the general mood and sound design. For the final version, however, I opted not to explicitly communicate this theme. I felt that this narrative would need much more work if it was to be central to the experience.

4. Implementation Process

Technically, the four components (tracking, game logic, sound engine, visuals) posed their own challenge each.

4.1. Tracking

The Kinect is a hardware accessory that consists of an infrared light ("IR") projector, an IR camera and a regular camera. The IR projector projects patterns of IR dots onto a volume of about 10x10' in front of it, and the IR camera processes the distortions of this pattern to calculate a depth image, i.e. an image that encodes the distance of objects from the camera in greyscale values.

This automatic calculation of depth information is the key feature of the Kinect, and all further tracking for the Xbox is performed by the console itself.

Following the Kinect's launch, *OpenNI* and *NITE*[10] were release for public use. These libraries are C++-based middleware (produced by *PrimeSense*, co-developers of the Kinect) that provide image processing techniques that work with a variety of sensors, including the Kinect. The most relevant of these techniques are hand tracking and skeleton tracking.

The skeleton tracking algorithm calibrates its parameters for each user entering the scene (unless otherwise instructed) and then continuously estimates the user's "skeleton" - a stick figure consisting of thinned-out limbs and joints - and returns updates of this skeleton at up to 30 frames per second (the standard Kinect update frequency). Fig. 1 shows the reference skeleton tracking application provided by OpenNI.

[7] http://thatgamecompany.com/games/flower

[8] http://thatgamecompany.com/games/flow

[9] http://www.plastic-demo.org

[10] http://openni.org

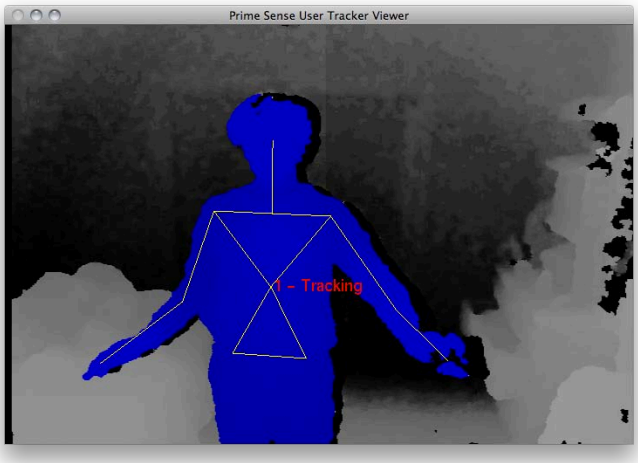

Fig. 1: OpenNI user tracking sample application. Where lines of the stick figure meet, the system provides estimated position and orientation for a joint.

Fig. 2: Combined view of game visuals and debug information. On top (from left to right): depth image, greyscale image and user silhouette. In the middle: the player skeleton (from above). On the right: a sound item with spectral visualization.

The game is written in C++, with the open source multimedia programming framework *Cinder*[11], which provides convenient wrappers around commonly used input-, output and graphics libraries, geometry representations etc.

During the early phases of development, I did some initial experiments with connecting my Cinder application to OpenNI/NITE. In February, a developer contributed a Cinder-specific wrapper for OpenNI/NITE [12] to which I immediately switched, further simplifying the processing of skeleton data as well as providing access to the raw Kinect depth image data.

I calculate the player's orientation using the estimated shoulder-line, averaged with the estimated head rotation. I tested both the estimated head rotation and the estimated shoulder positions for robustness in their values. While head rotation would theoretically yield a more appropriate measurement for spatial audio simulation, the estimated head rotation from OpenNI/NITE turned out to be slightly less reliable than the estimated shoulder positions. Th e average yields a relatively robust measure. Both measures, however, decrease in accuracy when the player is facing in a direction of 135° to 225° (with the Kinect camera being at 0°). This may be due to a higher likelihood of the occlusion of hands and arms held partly in front of the body. (Shotton et al. 2011), describing a skeleton estimation algorithm that can be assumed to be similar to the one used in NITE (which is closed-source), indicate that the accuracy of every single joint estimation is proportional to the number of joints visible/identifiable to the camera.

In general, the accuracy of this estimation from head rotation by looking at the upper body orientation is "good enough". For highly accurate 360° head tracking, a tilt-compensated compass with wireless communication to the system would be preferable, but since a core idea of the project was to explore possibilities of experimental game design that can be achieved by using consumer devices, building a complex wireless communication system with custom parts and microcontrollers would have defeated the purpose.

For the initial control scheme, the player's hands were tracked in relation to their upper body, so that when they extended them in any direction, the distance between their hands and their upper body was directly mapped to movement speed of the

[11] http://libcinder.org

[12] https://github.com/pixelnerve/BlockOpenNI

avatar in the virtual world. There were low and high thresholds for this distance to prevent movement when none is desired and unnaturally fast movement, respectively. In addition, cases of occlusion of arms and hands, in the orientation range described above, had to be handled by preventing movement for estimates that were likely to be highly inaccurate due to missing joint information.

The control scheme as of publication of this paper consists of three game modes, described above. Orientation tracking was kept as implemented for the earlier iteration. The mapping of physical to virtual location was a matter of simply scaling the estimated X and Z coordinates of the player's position to the space in which the game was installed. The Kinect's coordinates are not exactly linear with respect to the real coordinates, but more than good enough for the game if the scaling due to perspective distortion is taken into account. The hand tracking for the additional game modes was also adapted from the hand coordinate calculation of the earlier iteration.

A calibration process of aligning arms in a certain pose is recommended for by each player at the start of the game. On 15 April 2011, PrimeSense released an update to OpenNI that allows it to save and load calibration data from a file, i.e. in theory, the game can be calibrated to any person once, and then every subsequent player will be able to use that calibration data, without having to complete the calibration process (which can be tricky to attain under some circumstances). This may lessen the estimation accuracy, but user testing showed that the advantage of skipping the awkward calibration step was well-received, while no significant decrease in accuracy could be detected.

4.2. Game Logic

The game logic framework was built in C++/Cinder. Object locations, associated sound visualizations and the sequence of events (which object causes which other object to appear upon collection, etc.) are stored in an XML file. As the game is played, a simple custom-built engine calculates distances, collisions etc. All coordinates and events are sent to Max/MSP via the OSC protocol implemented with the OSC-wrapper for Cinder.

The game engine can also output a top-down view of the game scene, to be shown to viewers.

4.3. Sound Engine

The development of the "sound engine" occupied a big part of the technical challenge, not only because the number of pure audio games is too low to have produced a range of commonly accepted standards (as opposed to graphics frameworks), but also because this is the aspect I was least familiar with.

In the following, I will describe a number of techniques I explored in the development process, and the problems and advantages of each. Most of the explorations were made with the software *Max/MSP*[13], a visual programming language for realtime signal processing, that communicates with the Cinder application via a UDP streaming protocol called *OSC* (Open Sound Control)[14].

Max's paradigm is that of connecting "wires" between objects performing algorithms on the signals (streams of numbers) that flow through them. This approach is principally different from procedural programming languages in that it is built around strict timing. It comes with a set of pre-built objects for signal processing stages and makes it easy to connect them to each other. Max/MSP permits an experimental approach to designing a chain of signal processing stages, without the coding overhead required by procedural languages. Some downsides are higher complexity for custom calculations and difficult deployment. The latter does not matter so much as this version of the project was planned to be a singular installation project.

There is a multiplicity of possibilities for simulating spatial sound. Many of the techniques have variants for both sounds recorded for the purpose of spatial reproduction (with multiple microphones) and the simulation of spatial sound from monophonic sources. Since I use synthesized sound, I am only interested in the latter.

The most widely used technique is Stereophonic Sound, the panning of different

[13] http://cycling74.com/products/maxmspjitter

[14] http://opensoundcontrol.org

instruments, voices etc. between two sound channels, the output of which is typically mapped to two speakers or a pair of headphones. Stereo sound produces a basic "illusion of directionality and audible perspective" ("Stereophonic Sound" 2011).

Surround Sound is typically an extension of this principle to multiple speakers positioned in a circle around the listener. My first attempts involved a four-channel surround sound system, with simply panning the sounds in a circular manner between the four speakers, which were placed around the listener. I abandoned this approach, as the accurate localization of sounds from a four-speaker surround system would require a permanent sound-controlled 10x10' space that I didn't have.

I decided to use wireless headphones as the sound output device. Since I had the ability to track orientation, even if only for the upper body rather than for the head, I would be able to solve the problem that the player takes the sound source with them when they turn by just accounting for the orientation offset in the sound engine.

For headphones, the anatomical factors that help us locate sounds (reflection off our body and head, interaural differences etc., see section 3.2) are absent. The sound is directly projected into our ears, so for spatialization different processing has to be applied to each of the two channels, simulating the different modifications that the soundwaves would undergo if they actually travelled from one particular point in space to our ears.

This kind of simulation is called *binaural* processing. Every person has characteristic parameters by which the incoming sounds are modified depending on the angles of incidence. These parameters are called Head Related Transfer Functions (HRTF) and vary depending on the person's anatomy. They can be measured for a specific head:

> [The HRTF] is defined by the sound pressure measured at the eardrum or at the ear canal entrance divided by the sound pressure measured with a microphone at the centre of the head but with the head absent. Accordingly, HRTF is dependent on the direction of sound incidence.
>
> (Vorländer 2007, p 87)

Technically, the HRTF is the Fourier Transform[15] of the Head Related Impulse Response (HRIR), the head-specific function that yields a time-domain output for a given input signal. In binaural processing, the HRIR is convolved[16] with a sound (in digital signal format) to reproduce the individuals' objective perception of the sound.

Since HRTFs differ between individuals, but it is impractical to make individual measurements for every user of a spatial sound system, measurements made on "dummy heads" (puppet heads exhibiting averages of anatomical variation) are commonly used as a good compromise. One standard repository of such data is the MIT KEMAR HRTF database[17].

In the first prototype, I used simple panning for left-right spatialization, and applied basic distortion if the sound source was behind the player. The distortion was a convention rather than a simulation, but it proved the point and served as a basis for further techniques by translating the world coordinates of object distances I sent to Max/MSP into scaled distance/azimuth values.

For the second prototype, I used a third-party Max/MSP object for binaural spatialization[18]. I could not determine which HRTF data this object uses, but it employs a technique of interpolating between a small number of HRTFs to convolute the signal with, as well as a range of additional techniques to enhance the impression and avoid aliasing effects. For me and a small number of people who tested the game, however, the reliability of the object varied greatly between sounds: For a clear recording of human speech, it worked rather well, but for synthesized sounds or sounds with some degree of reverberation, there were azimuth ranges in which the sound appeared to come from exactly the opposite direction than it actually did.

---

[15] for details on this principle see http://mathworld.wolfram.com/FourierTransform.html

[16] for details on this principle see http://mathworld.wolfram.com/Convolution.html

[17] http://sound.media.mit.edu/resources/KEMAR.html

[18] http://eude.nl/maxmsp

The next variant I tried was writing a custom basic implementation of the HRTF processing. The convolution setup was based on another Max/MSP implementation[19] (which lacked the possibility for a full 360° rotation) and a PD patch for the same purpose that was generously provided by a fellow NYU student from the Music Technology department. The original version of this worked well for one or two sound sources, but became unstable and low-performing for multiple sources, presumably because it had to hold a multiplicity of sample data in memory. An improved version, using a different buffer setup, seemed to work fine only to crash after several minutes of execution time.

Frustrated by the problems encountered in Max/MSP, I started working on integrating the C++ game audio library FMOD[20] into the project. The result was a stable spatialization, but with several disadvantages: First, FMOD's 3D audio implementation for headphones does not appear to be based on HRTF processing, but rather on simple panning and volume falloff; alternately it can produce 5.1 surround sound that some headphones may process in their own way. However, all the testing resulted in the realization that HRTF implementations differ so widely that for this project any solution that empirically proves to work reasonably well on a range of different sounds would be acceptable. Hence, even FMOD's lack of a defined front-back differentiation would have been tolerable - especially considering that the performance and deployability would profit from it - had it not been for the fact that after a quick study of the FMOD digital signal processing chain, I realized that for anything that involved sound synthesis or filtering, dozens of lines of code would be necessary for what is accomplished in Max/MSP with a single virtual wire. This would have severely limited the creative freedom and experimental approach in manipulating sounds parametrically depending on time and game-state.

Finally, a hint by a fellow ITP student led me to the COSM framework[21], a system for creating virtual worlds in Max/MSP/Jitter (the latter being Max/MSP's module for graphical output). The audio component, which I am using in my project without COSM's visual module, takes object coordinates and encodes them in *Ambisonics* format, a standard for representing spatial audio information (see "Ambisonics", 2011). The idea behind this standard is that signals are encoded in a format that can then be decoded by a different module, for a variety of speaker setups. COSM comes with such a decoder, but doesn't account for HRTF processing. I tested COSM's audio component and it seemed to be stable in both performance and spatial impression. I also found a Max/MSP object that takes input from an ambisonics decoder outputting a 7-speaker signal (something COSM's decoder can do) and turns it into a binaural Headphone signal via HRTF processing[22]. After long struggles with severe performance issues using this object, I found that the buffer size it uses for the convolution of the audio signal with the HRTFs could be reduced by 75% for an exponential performance gain without significant sacrifice in the quality of the result.

Beyond all this, Max/MSP also allowed me to design a dynamic sound synthesis and filtering system. *All of the sounds are synthesized in realtime*. This allows for enormous flexibility in adjusting the character of the sounds to the game state. I use filters with different attributes, amplitude and frequency envelopes, and the manipulation of these attributes through control signals dependent on game parameters and stochastic variation. Much of this was a discovery process for me, guided by (Cipriani & Giri 2010).

The sound engine also brings variation to the sounds by generating part of the output stochastically. It is notoriously difficult to generate at least minimally pleasant *and* interesting sound by sonifying data that has no meaning in the aural domain, but I had long been interested in this process and decided that I wanted to add surprise and variation to the soundscape by injecting this data.

Finally, the sound engine performs live spectral analysis on the objects' sounds, and sends

[19] http://www.ece.ucdavis.edu/binaural/binaural_tools.html

[20] http://fmod.org

[21] http://www.allosphere.ucsb.edu/cosm/

[22] http://www.friendlyvirus.org/artists/zlb/2008/10/ambi2bin-ambisonics-to-binaural-converter

the results in realtime to the game engine, allowing it to use that data to enhance the visuals.

4.4. Visuals

The visuals shifted significantly in their relevance to the project. I initially considered the abstract visuals part of the feedback for the player, and spent time on creating atmospheric graphics, but I did not find a good way to incorporate that, especially given that it was never clear where the game would be physically installed.

At the time of writing, the visuals are a top-down world view to be shown on a separate screen. The graphics are guided by minimalist aesthetics, and show the player's skeleton along with the sound-emitting objects. Each object's spectrum is presented along with it, in a circular bar-graph visualization. The visuals should enhance the audience experience, and make the in-game actions a little easier to follow.

5. Design Process & Testing

In the earliest stages (February), I considered a variety of spatial audio input. Hence, the first exploration was a basic setup of four speakers, and arbitrary mappings of panning to the skeleton data. Deciding that a speaker setup would be impossible under the space and environment constraints at ITP, I moved to wireless headphones.

The first headphone prototypes were experimentations with different spatialization techniques, and were using low-quality wireless headphones that operated on infrared technology and were highly susceptible to noise. Nevertheless, people could generally detect sound sources, and understood the movement controls after some explanation.

Happy that the basic direction was viable, I proceeded to learning about signal processing and improving the sound engine, which occupied me for several weeks, as described above. I also implemented the first version of the visuals that represent information about the sound.

Once I had a space at ITP that would allow me to test the system in a semi-permanent setup outside of my home, I did that (around late March), using high-quality wireless headphones (Sennheiser *RS 170*) that operate in the 2.4-2.8 GHz range, practically eliminating noise.

Fig. 3: Playtesting of the second game version in late April.

Informal user tests showed that more work needed to be done on the reliability of tracking and audio. In particular, the simultaneous processing of depth images, game logic, game visuals *and* the resource-intensive sound synthesis via Max/MSP went to the border of the performance capabilities of a formidable MacBook Pro (with Core i7 CPU). This resulted in Max/MSP dropping samples (at the lowest reasonable sampling rate of 44100 Hz) and producing distorted sound. Minimizing the background CPU load, hiding Max/MSP GUI elements and slightly reducing the Cinder frame rate managed to keep the performance just below the critical threshold. Eventually, I would be able to largely solve the performance problems by changing buffer sizes, see section 4.3.

Having tuned the performance and the estimation accuracy (by handling some occlusion cases, see 4.1), I continued to ask people to test the game in early April.

The results were highly interesting, although disheartening: Even with a version of the controls that I was happy with in terms of accuracy, audio simulation etc. players felt a strong impulse to physically walk up to the sounds they were hearing, disregarding the arm-gesture controls. After re-explaining they would adjust to it, but this showed that the controls were sending out mixed messages: On the one hand, the orientation tracking and binaural audio afforded a natural interaction; the

gesture controls, on the other hand, were more conventional. Th is disparity was exacerbated by the audio-only feedback: Players would hear their steps when walking and would hear the sounds change volume and pitch, but this reduced feedback (opposed to a visual representation of movement) required more of an imagination of virtual space than I was able to instigate by the auditory means I had at my disposal.

At this point, I realized that I needed to either increase feedback while physically constraining people to a particular spot in order to steer them away from the walking impulse, or I had to cater to the impulse by also mapping physical walking to virtual walking.

Even though it meant greater space requirements, I decided to attempt the second variant. Th is brought my project closer to *Audio Space*, but it was still a game and it still had the player's estimated skeleton. Th e challenge, thus, became a) to map virtual and real space in a way that one could walk through virtual space and collecting the sounds would still be an achievable challenge, and b) to incorporate the player's movements to utilize the fact that the player involves their whole body in the experience, and that we can measure this.

The first part was much easier than expected. In a playtest on 13 April, all three players found all of the sounds, and if anything it felt too easy and the collection of the sounds too "accidental", i.e. without active effort.

I proceeded to implement the two other game modes (described above). Some testing showed that the mode in which the player has to catch moving sounds works rather well. People tended to enjoy this mode mode than the first one, presumably because of the additional movement, and a feeling of more agency in the game.

The third mode, in which players have to match the pitches of sounds, is much more difficult, as it requires two different modes of listening at the same time. At the time of writing, I am not entirely happy with this mode, and I am experimenting with lowering the difficulty by reducing the complexity of the sounds.

In general, I was surprised by the players' actions. I was aiming for a quiet, subtle experience of navigating a soundscape, but the physical quality of the game inspired most players to make rather quick movements, and the awkwardness of chasing sounds that only you can hear in the presence of an audience, combined with the necessary movement, encouraged laughing and the kind of group dynamics that physical games or Wii party games tend to instigate. While we can push the boundaries of games quite far, and inform them by different aesthetics and narratives, a game involving physical aspects seems bound to gravitate towards a particular, light-hearted type of plain old fun.

6. The Performance Aspect

About movement-based game interfaces such as the *Wii*, *Move* and *Kinect*, Michael Nitsche, Game Researcher at Georgia Tech suggests:

> These interfaces are part of a new invasion of the living room, one that not merely suggests more media streaming through more channels, but that engulfs the physical location as part of its interaction design. Th ey transform not only our body and its animations, but turn our living spaces into performance places and remediate the architecture and interior design of our play rooms into parts of the game stage.
>
> (Nitsche 2010)

The (partial) coincidence of play-space and real space makes playing games a performance, now not only in the virtual space (such as the quite literal role-playing performance in World of Warcraft), but also in the physical space.

Game-play as performance, the creation of an audience experience by playing a rule-governed game (with winning and losing conditions) is, of course, common in role playing, competitive sports, competitive games (from Poker to StarCraft), figure skating and many other activities. In many of these, especially in established sports, players are well aware of how their performance will affect the audience in terms of its aesthetics, narrative or creativity. Digital games often don't pay any regard to this factor. With gestural games, it is different: performativity is already central to the play experience in gestural games centred on performance themes (*Dance Dance Revolution*), or games that exploit the silliness of absurd physical activities (*Wario Party*).

But the novelty of gestural game interfaces and the flexibility of the digital content for games played with them warrants an exploration of the possibilities of designing a wider variety of game experiences that have the performative character of the play activity explicitly in mind.

Not A Dance is therefore designed as a simple exploration/collection game for the player, but the resulting soundscape, in combination with the player's physical movement, can be regarded as a performance piece. The spatial audio output results in the unique situation that as a basic consequence of the technology used, the audience can not have a very similar experience to the player, even if they are exposed to the same audio output via the same interface, because input and output are so directly linked to the player's coordinates in real space.

Finally, the project is an experiment in designing a game in which the character of the sound content allows both high playability and an interesting passive experience for the audience.

The setup of the installation includes speakers and a screen for the audience, so the player can perceive the spatial sound information without interference while the audience is able to hear as much as possible of the player's actions in the soundscape, and to follow their actions on screen. All non-directional sounds (except the instructions) are audible as evenly as possible across the whole room, while the spatial audio is significantly attenuated for the audience, so as not to interfere with the player's actions.

7. Conclusion

This project combined several challenging problems that show how technology and game design impose constraints on each other in the development of a game for a new interface:

- Working with new hardware and an unstable computer vision middleware: Kinect development is a very active field at the moment, but the gap between corporate game production and homebrew "hacks" is still very wide. It proved exciting to be involved in the development of applications for such a new (and accessible) technology.
- Constructing a 3D sound engine: When I started the project, I assumed that 3D sound was a solved problem. It is, in a way, but its supportive role has left it in a state that is not comparable to widespread, well-documented and flexible standards in the graphics world, such as OpenGL.
- Designing a game for a new input interface and a highly unusual output mode at the same time proved to be difficult. The measures for complexity and difficulty are very different from games with precise input and visual output. Ability to adapt to the standardized binaural simulation differs among individuals, partly simply because of anatomical differences. Communicating the unusual game mechanics is a challenge in itself. Finally, the "natural" physical interface leads to certain assumptions players have that the designer needs to address in one way or another.

At this point, the game has a very basic progression of three game modes that proved to provide an interesting, unusual experience for both player and audience while posing a manageable challenge to the player. Still, only a few of the possibilities for an abstract gestural 3D audio game have been explored, and every time new players try the game, new ideas for game mechanics emerge from the feedback. I hope to be able to continue with this experiment.

The question that remains is the context in which this game can exist. In the current form, it is an installation that, with some work, could function in a game-focussed art gallery or an adventurous arcade. Such spaces are rare, though, and digital games that are played in a singular event, and in public, are a particular (and rather unusual) experience.

Hence, since *Not A Dance* is designed around consumer devices, it might be possible to target an eventual release as a distributable game for either a Kinect connected to a PC or, of course, for the XBox console. In this scenario, the most important - and most difficult - question is that of handling 3D audio for different hardware, since few people own wireless headphones. In any case, the audio engine would have to be rewritten in a form that allows

distribution. The most promising technologies for this (apart from FMOD with all its downsides) appears to be *libpd*[23], a C library that wraps *puredata*[24], the open source alternative to Max/MSP.

In its current form, *Not A Dance* is an experiment in combining two non-traditional game interfaces to an experience that, I hope, expands the player's idea of what digital games can be. It should remind us that just because games are fun, they can still be conceived and received as art exploring a particular medium, theme or experience. It should also enforce the idea that in game development, like in other fields, "independent" or "art-focussed" need not be at odds with using the newest technology that has been developed by organizations catering to the mainstream market. Director Wim Wenders sums this last notion up as he talks about why he is one of the first "arthouse" directors using 3D (stereoscopic imaging) in a movie:

[...] there is a whole new technology and that is obviously pushed by the industry. And there is also a language. The language has to be used by people who are interested in it; who can extend the realm of expression. And I don't see the studios doing that. They have no interest in that. But that's the history of cinema. It's always been both industry and expression.

(Mason 2011)

Acknowledgements

I would not have found a way through the design challenges without the support and feedback I received from my thesis advisor, Katherine Dillon, and my thesis seminar classmates.

Many thanks to Scott Wayne Indiana, Melissa Clarke, Dalia Othman, Greg Borenstein, Sue Ngo, Don Miller, David Phillips and Gregory Trefry for the help in the design process, and to everyone else who spared some of their time to playtest.

Finally, I am highly grateful for the technical advice regarding 3D audio offered by Matt Ganucheau, Andrew Madden, Amit Pitaru and Agnieszka Roginska, and for the support received in the Cinder and OpenNI developer communities.

Further Information

Full documentation of the project will be made available at http://holzkorn.com/notadance.

My process is already documented in irregular journal entries at http://blog.holzkorn.com/thesis.

The full source code for the C++/Cinder application and the custom max patches is online at https://github.com/pholz/notadance. The following third-party libraries were used:

- Cinder v0.8.2: http://libcinder.org
- My fork of an OpenNI-wrapper for Cinder: https://github.com/pholz/BlockOpenNI
- OpenNI v1.1.0.39: https://github.com/OpenNI/OpenNI)
- NITE v1.3.1.4: http://www.openni.org/downloadfiles/openni-compliant-middleware-binaries/33-latest-unstable)
- SensorKinect v5.0.1.32: https://github.com/avin2/SensorKinect
- COSM for Max, version Feb 2010: http://www.allosphere.ucsb.edu/cosm/
- Ambi2Bin~ for Max: http://www.friendlyvirus.org/artists/zlb/2008/10/ambi2bin-ambisonics-to-binaural-converter/

[23] http://gitorious.org/pdlib

[24] http://puredata.info/

Bibliography

"Ambisonics." In *Wikipedia*. http://en.wikipedia.org/wiki/Ambisonics. Accessed 27 Apr 2011.

Caillois, Roger. 2001. *Man, Play and Games*. University of Illinois Press.

Chion, Michel. *The Three Modes of Listening*. http://helios.hampshire.edu/~hacu123/papers/chion.html. Accessed 17 Apr 2011.

Cipriani, Alessandro, and Maurizio Giri. 2010. *Electronic Music and Sound Design - Theory and Practice with Max/MSP - Volume 1*. Contemponet.

EDGE Magazine. February 2011.

Juul, Jesper. 2009. *A Casual Revolution: Reinventing Video Games and Their Players*. The MIT Press.

Koster, Raph. 2004. *A Theory of Fun for Game Design*. Paraglyph Press.

Mason, Paul. 2011. *Wim Wenders: With 3D "the door is open"*. Interview. http://www.bbc.co.uk/blogs/newsnight/paulmason/2011/04/ww_draft.html. Accessed 27 Apr 2011.

Nitsche, Michael. 2010. "Games as Structures for Mediated Performances." In: *Logic and Structure of the Computer Game*, ed. Stephan Günzel et al., 110-129. Potsdam: University Press.

Saffer, Dan. 2008. *Designing Gestural Interfaces: Touchscreens and Interactive Devices*. O'Reilly Media.

Salen, Katie, and Eric Zimmerman. 2003. *Rules of Play: Game Design Fundamentals*. The MIT Press.

Schell, Jesse. 2008. *The Art of Game Design: A book of lenses*. Morgan Kaufmann.

Shotton, Jamie et al. 2011. *Real-Time Human Pose Recognition in Parts from Single Depth Images*. http://research.microsoft.com/pubs/145347/BodyPartRecognition.pdf.

"Stereophonic Sound". In *Wikipedia*. http://en.wikipedia.org/wiki/Stereophonic_sound Accessed 27 Apr 2011.

Vorländer, Michael. 2007. *Auralization: Fundamentals of Acoustics, Modelling, Simulation, Algorithms and Acoustic Virtual Reality (RWTHedition)*. Springer.