PROFESSIONSBACHELORPROJEKT - MEDIE OG SONOKOMMUNIKATION

# Sound occlusion for virtual 3D environments

## A solution for real-time filtered sound propagation for arbitrary and non-convex geometry

by Janus Lynggaard Thorborg - MS1316, class MS2013

Supervisor: Niels Böttcher

Characters: 47.982

Faglig model: 1

22-12-2016

# TABLE OF CONTENTS

# 1. INTRODUCTION

## 1.1 MOTIVATION AND PROBLEM INVESTIGATION

For real-time virtual 3D environments, most commonly games hereafter, audio has always been a bit behind graphics. The reasons are many, but given the ever-increasing processing power of consumer electronics, we now have new possibilities and degrees of realism at our disposal. Technologies that achieve photorealism for games means we have now reached a state where anything not realistic can break the immersion of the experience, game designers create today.

With virtual reality we are eliminating the barrier between the control interface and effectively feeding input directly into our sensory organs. The point being that we are mapping real movement to virtual movement, thus, theoretically, increasing immersion to newer levels as we are substituting the reality with a virtual one.

This direction then also need to emphasize actual physics, to avoid breaking the illusion - and severely confusing the mind, so people don't get sick[1]. In continuation, audio has had a renaissance as well, mostly through binaural mixing techniques that allows us to spatialise sound in a more convincing matter.

However, there are still many lacking aspects of sound rendering today. The act of accurately rendering audio for a given scene is still far from "photorealistic". Traditionally, sound and geometry have been oblivious about each other, limiting sound designers to synthesize effects approximating the spatial sound of an environment. This is not very realistic, should the geometry interfere with the path of the sound.

To deal with this problem, umbrella-termed sound occlusion, simple techniques have arisen that check whether line of sight exists, and applies attenuation accordingly. This technology is available only in select game engines today[2], for others 3rd party software has to be included. This solution, however, is simplified for more complex scenarios. Consider standing in a room next to another, where a conversation is taking place. Today's technology will correctly muffle the sound as

[1] LaViola, 2000
[2] Crytek, 2016

would be expected, but what if a door to your right is open(ing) into the room? In the real world, the sound escapes through the door. Humans have the ability to localize sounds, and will quickly be able to infer that joining the conversation is possible by walking through the door.

Conversely, if the sound is still as muffled, it would suggest the door doesn't lead into the same room, thus there is potential confusion between senses, breaking immersion. First order solutions (i.e. dealing with one obstruction) exists[3] with various drawbacks, however a general method that handles arbitrary geometry is yet to be seen. Having access to this mechanic would create many new possibilities and dimensions; for instance, map design in games in situations where you can't rely on visual cues for path finding.

This project seeks to reintroduce the actual physics of sound back into the virtual environment, and making it possible to use it for mechanics such as localization, attention and navigation, where localized sources are not heard like direct rays, but are occluded, obstructed and redirected intelligently through geometric reflections. The problem we're investigating here is thus:

**How can you provide real-time spatial localization of sound for arbitrary geometry in virtual 3D environments?**

[3] See 3.

## 1.2 Outline, scope and methodology of project

To provide a realistic solution, we need to solve 3 integral problems:

1. How can we simulate the path of sound through an environment?
2. How can we filter the sound as though it was transmitted through the path?
3. How can we trick the mind into believing that sound is emitted from a specific location?

We will study computer graphics approaches to illuminating scenes and heat radiation physics to understand how radiation reflects, propagates and transmits through materials. Combining our study of the underlying mathematics and algorithms, physics of waves and how humans localise sounds, a solution will be derived that allows to solve the mentioned problems. Additionally, a real-time implementation of the solution is shown in 5.1 with included videos showcasing how the system handles different spatial situations.

Chapter 2. covers the topic of sound occlusion and obstruction, problems as well as related theory that will be the foundation for the presented solution. Chapter 3. briefly visits existing related solutions. Chapter 4. covers the modelled aspects and derivation of the solution, while chapter 5. reviews the final implementation and concludes the project.

Math knowledge, while not required to understand the solution, will be used exclusively for modelling as it allows us to express, derive and prove complex chains of physical filters as collective entities with properties we can manipulate on a higher level, while allowing us to apply and use physics literature directly. Additionally, algorithm complexity analysis will be explored and used for discriminating between multiple possible solutions, from an optimization stand-point.

The presented solution only covers localized sources of sound, a subset of all the sound in a typical game. Given the additional non-linear computational problems involved, it also requires higher processing time compared to simply playing audio files. As will be covered in 2.6, accurately rendering spatialised sound is a task still impossible to do in real time. Thus, the solution presented is a coarse approximation that only models some - important - aspects of reality. However, the accuracy can arbitrarily be increased at the cost of extra processing time, as covered in 4.5.1, 5.2.1 and 5.4.

# 2. SIMULATION OF SOUND OCCLUSION AND -OBSTRUCTION

## 2.1 DEFINITION AND DESCRIPTION

As noted, sound occlusion is often used as an umbrella term for the effect where geometry interferes with the direct path of sound. But specifically, sound occlusions refers to complete interference, where sound obstruction instead is somewhere in between complete attenuation and a mix. To simplify the language used, sound occlusion will from here be referred to as a factor between 0 and 1.

## 2.2 GROUND TRUTH OF SOUND

**ground truth**, n.

**1.** Factual data as ascertainable through direct observation rather than through inference from remote sensing.[4]

It is often preferable to solve a problem by understanding how something behaves. We can use the nature of the ground truth to directly understand the underlying physics, and to differentiate when we make choices based on inference, since both strategies will employed in this report.

Sound are mechanical waves propagating at some speed $w_s$ through some medium $M$. If we define a leaky environment $E$ of mediums, any type of wave will be spread around through the effects of reflection, diffraction and dispersion.[5] Refraction is the bending of waves on travelling through a new medium; dispersion additionally bends the waves depending on frequency. Diffraction, in this context, refers to a wave's ability to move around objects.

Reverberation is typically defined as persistence of sound after the original source has stopped, being the sum of filtered decaying reflections build up from the environment[6]. Following the law of conservation, we know that the energy is constant. Thus if the reverberation decays, we know that the energy is either escaping our environment or being transformed, typically to heat, through friction. We will refer to this effect as absorption.

[4] The Free Dictionary, 2011
[5] Henderson, 2016
[6] Lloyd, 1970

## 2.3 Transmission, emissivity and linear time-invariant systems

To model sound occlusion, we need to understand what happens when a sound wave travels through an object. Whenever some wave's path intersects some other medium, it will be transmitted through the medium. The strength of the transmission is whatever isn't reflected ($\rho$) and isn't absorbed ($A$) by the medium, i.e.:

$$T_{str} = A(1-\rho) \tag{1}$$

If we consider the first medium to be air, and the second to be some black box medium $M$, the resultant transmission strength from air through the other medium and back into air is:

$$y(t) = T_{str}\big(y(t-S) - x(t-S)\big) \tag{2}$$

Where $S = Lw_s$, where $L$ is the length of the medium, and $w_s$ is wave speed in the medium. Notice that we in equation (2) had to define a recurrence relation, that is, some function that relies on previous input. This is because any medium that has a non-zero reflection coefficient also has internal reflections, that continue forever. This effect is also called feedback. The minus between the terms is due to reflection phase inversion.

To completely model the system, we however need to recognize that dispersion- $D_f$ and absorption $A_f$ are frequency dependant, that is, they are functions of the signal:

$$y(t) = A_f\left(D_f\big((1-\rho)y(t-S)\big)\right) - A_f\left(D_f\big((1-\rho)x(t-S)\big)\right) \tag{3}$$

(3) completely describes what gets emitted through a medium $M$ from an original wave (or signal) $x(t)$, ie. the emissivity of the medium. Linear time-invariant (LTI) theory studies the behaviour of LTI systems - linear recurrence relations, like (2) and (3). One important property of LTI systems is, that they are completely characterised by convolution of their impulse response

(IR)[7], where an impulse response $h(t)$ is defined as the reaction of some object or function to a single impulse (a Dirac function $\delta(t)$). This means we can model any system using convolution:

$$y(t) = x(t) * h(t) \overset{\text{def}}{=} \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau \tag{4}$$

where $*$ denotes the convolution operator. This can be proven through the use of Fourier transforms.[8] The special properties of convolution allows us to combine the responses of $N$ filters through associativity, which can be shown like so:

$$h_m(t) = h_0(t) * h_1(t) \dots * h_{N-1}(t) = \mathcal{F}^{-1}\left\{\prod_{k=0}^{N-1} \mathcal{F}\{h_k\}\right\}(t) \tag{5}$$

Where $\mathcal{F}$ and $\mathcal{F}^{-1}$ denotes the Fourier transform operator and the inverse, respectively. Taking some liberty in syntax for brevity, (5) is basically the Cauchy product of $N$ sequences using the identity that convolution in time domain is equivalent to multiplication in the frequency domain[9]. We can use (5) to prove, that any amount of LTI systems can be combined to form a single LTI system, thus we can reduce (3) to:

$$y(t) = x(t) * h_m(t) \tag{6}$$

**In other words,** if we assume the system is linear (it doesn't *distort*) and we know the reflectance of $M$, we can derive the correct emissivity and absorption using (1) and the impulse response or, equivalently, the LTI system of the medium $M$, thus completely simulating the occlusion of $M$ together with its contribution to the environment $E$.

[7] Orfanidis, 1996: p. 106
[8] Orfanidis, 1996: p. 3
[9] Orfanidis, 1996: p. 3

## 2.4 COMPUTATIONAL COMPLEXITY AND ALGORITHM DESIGN

Something to consider for game development is processor usage - graphics or the CPU. This metric is paramount, and is often the limiting factor in the finished quality - as in frames per second, graphics, realism etc. - of the game. A central part of this metric, and something to consider when you're creating technical systems that shall apply to general situations, is *scale*.

We can simplify the execution of games to a set of tasks. Some tasks take the same amount of time no matter how large the input, some take as much time as the amount of input, and some take a higher order of time based on the size of the input. In computer science and algorithmic design, this is often measured in Big-O notation.[10]

For instance, transporting some moving boxes from $A$ to $B$ in a vehicle takes the same amount of time, no matter how many boxes (bounded by the transporting medium capacity of course; ignore that for now). This is a $O(1)$ operation in this contrived example, as it doesn't depend on the input. Now consider moving the boxes from the car into an apartment by hand. If you're a single person, it will take as many trips as there are boxes - i.e. a linear complexity, denoted $O(n)$ where n equals the amount of boxes. If you are two persons, the relationship is still linear, because when you double the amount of boxes, you still double the amount of trips to be taken. Importantly to note from this example, computational complexity only tells us about relationships on scaling, not constants.

This analysis is important, because it describes which tasks can easily be done two more times, for example - and tasks that are problematic in the same context. This question naturally arises quite often, for instance: "*What happens to the processing time if we double the size of the game world*"? Or, perhaps more relevant: "*What happens to the processing time if we double the amount of audio voices*"?

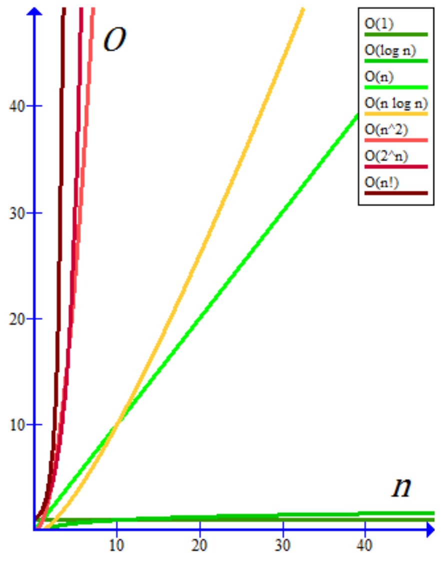

Figure 1. *Different complexity classes as a function of n*

[10] Mohr, 2007

## 2.5 Heat transfer and radiosity

Radiosity, originally developed as a model for heat transfer, was applied to computer graphics in the context of surface emission in 1984[11], and can be used as a means to solve the rendering equation[12]:

$$L(x', \vec{w}') = E(x') + \int_{S} \rho(x') L(x, \vec{w}) G(x, x') V(x, x')\, dx \tag{7}$$

(7) is a formula to calculate the illumination of a scene. We won't go into depth in the rendering equation, however we will note the similarities for the problem we're trying to solve: It calculates the resulting radiance $L(x', \vec{w})$ from the environment on some surface $x'$ using the resulting geometry $G$ and visibility $V$ between surfaces. The last term is especially relevant as it is the equivalent of sound occlusion, the factor of visibility between two objects - this will be analysed further in 4.3.

The radiosity algorithm solves the geometry steps and visibility steps, taking all surfaces combined reflections into account by calculating form factors $F$ between all surfaces $B_N$, and solving the radiation leaving the system iteratively by modelling surfaces as perfect diffuse reflectors. This can be done pretty effectively by discretising the surfaces into patches, thus modelling arbitrary geometry with convergent error, yielding the discrete radiosity equation[13]:

$$B_j = E_j + \rho_j \sum_{i=1}^{N} B_i F_{ij}\,, \qquad for\ j = 1, N \tag{8}$$

Where $\rho$ represents the reflectivity of $B_j$. There are two interesting things to note: We again have an emission term $E_j$ for the surface $B_j$, representing the input to the system, equivalent to the $x(t)$ input signal for LTI systems. Secondly, we're doing $N$ term calculations over all surfaces, $N$ times. Thus, the complexity of this algorithm is $O(n^2)$.

[11] Goral, et al., 1984: p. 1
[12] Lozano-Perez & Popovic, 2016: p. 6
[13] Goral, et al. 1984: p. 4

## 2.6 Core problems in implementing sound occlusion

As we have identified graphics illumination and sound occlusion are largely solving the same problem, we will highlight additional properties that clarify why rendering audio is more difficult, compared to lightning. For these comparisons, we will only consider dynamic lightning; a subset of applied lightning in games today. Most (if not all) global lightning has historically been "baked", a process referring to statically computing the illumination before runtime. [14]

An obvious difference dedicated accelerated hardware for graphics operations. Sound peripherals exists, but consumer grade operations are mostly DAC/ADC conversions - i.e. the rendering of the sound happens in software, on the CPU, meaning sound has to share processing power with everything else, contrary to graphics.

The next, huge difference is how lightning neglects the wave propagation speed, resulting in graphic rendering models like radiosity doesn't model "acoustic" interference and delays, constituting the essential parts of reverberation[15] and sound localization[16]. This means all equivalent recursive LTI systems for lightning can be solved directly, like radiosity does, while sound has to be rendered recursively with multiple delay lines.

For path tracing methods, lightning has 3 orders of magnitude simpler frequency bandwidth, making filter designs for colour bleeding much more feasible and simple. Light is stored as 3 numbers (an RGB channel) in a 3-band frequency-domain format, effectively synthesized on visualization on a display. Typical bandwidth for a 24-bit colour light trace: 1440 bytes/sec (3x channels x 8 bytes x 60 Hz). Conversely, sound is stored in time-domain at a much higher sampling rate and bit depth, typical bandwidth for a sound trace: 1411200 bytes/sec (32 bytes x 44100 Hz).

Radiosity further doesn't model transparency and translucency, which is extremely important for sound. We can hear through doors, but not see through them, for instance. To conclude, we're trying to solve a problem that's much more intensive, without having access to solving radiation models directly, with additional complexities, using the remainder of processing power available not used for anything else.

[14] See also 5.2.1 for sound baking
[15] See 2.2
[16] See 4.4

# 3. Related work

## 3.1 Public or academic

Nicolas Tsingos and Jean-Dominique Gascuel wrote an article[17] on fast rendering for sound occlusion simulating diffraction effects. Using accelerated graphics hardware, properties of wave propagation can be evaluated to high precision (specifically, spherical secondary waves), thus indirect sound paths can be evaluated, diffraction can be simulated and occlusion can therefore be emulated, as diffraction can model a wave's indirect path around an obstacle.

The company Google has released a free, complete VR suite for audio and graphics for many different platforms[18]. The system internally uses a full-sphere mixing space called Ambisonics[19], representing spatial audio in 3D as $N$ channels positioned around a listener using spherical harmonics, down mixed using HRIRs as discussed in 4.4. The system supports only occlusion by tracing direct line of sight, and applying a low-pass filter for each object intersected. Reverberation, while having interesting real-time properties, is confined to cubic rooms with a predefined set of wall materials. Rooms do additionally not contribute to each other. Transitions from occlusion and rooms are done cross-fading over a second to appear seamless.

## 3.2 Commercial or proprietary

The company Two Big Ears is similarly offering a paid, proprietary spatial audio solution using a technology referred to as "3Dception"[20]. As no testable demos, documentation or technical information is available, we will only mention them as their system supposedly supports some interesting features like path finding for occlusion, simulating diffraction. Additionally, geometry-based reflections and environment modelling (like speed of sound) is supposedly coming soon; topics that will also be covered in this report.

As has been indicated, no general solution for arbitrary sound occlusion and reverberation has been found, thus we will investigate what it will take to realise such a system.

[17] Tsingos & Gascuel, 1998
[18] Google: VR SDK, 2016
[19] Gerzon, 1973
[20] Two Big Ears, 2016

# 4. Proposed model

## 4.1 Idea and overview

Sound reaching a listener in an environment where a number of sources emit energy into, is simply the sum of the waves travelling through the room - i.e. the delayed reflections at a point, as defined 2.2. By induction, we can model any environment by subdividing all geometry into lambertian surfaces, that is, mediums that diffusively emit and reflect sound like radiosity, with the key difference that our mediums also support translucent transmission, as covered in 2.3. This enables us to model occlusion and obstruction.

As time is important for humans to be able to localise sounds, as we will investigate in 4.4, we need to propagate waves throughout the environment in the correct order with accumulating delays, so we also need a path finding system. We will trace rays around the environment to model wave propagation, however this will be a coarse approximation that will only approach wave-like behaviour of sound as the division of surfaces approaches infinity - see 4.5.1.

## 4.2 Modelling translucency for sound

Translucency for light defines a material that allows light to pass, but blurs/darkens it to some degree. It is therefore closely related to the terms emissivity and absorption defined earlier, and now we will investigate how to parameterise a modelling of the behaviour. We will define the LTI system for a surface as:

$$y(t) = x(t) * M(v_i, v_o, t) \tag{9}$$

Where $v_i$ is the entry angle, and $v_o$ is the exit angle to a normal on a plane. With this model, we can pass signals as rays to a surface filter $M$ representing a surface and automatically get a diffuse reflection and transmission giving occluded sound, depending on the angle constitution[21]. The relationship here to form $M$ is:

$$y(t) = -V_i(v_i)\cos(v_i)\,x(t) + (1-\rho)\cos(v_i)\cos(v_o)\,x(t) * h_m(t)$$

[21] See 2.5

Where $h_m(t)$ is from (5), and $V_i$ is:

$$V_i(v) = \begin{cases} \cos v\,, & x < \dfrac{\pi}{2} \\ 0, & x \geq \dfrac{\pi}{2} \end{cases}$$

We use $V_i$ to select composition of occluded sound and reflected sound, depending on whether they're in the same hemisphere.

If we look at Figure 2, we can see how a diffuse surface receives all radiation with a strength depending on the input angle, and emits it diffusively with a strength depending on the exit angle to the normal of the surface. This relation is called Lambert's cosine law[22].

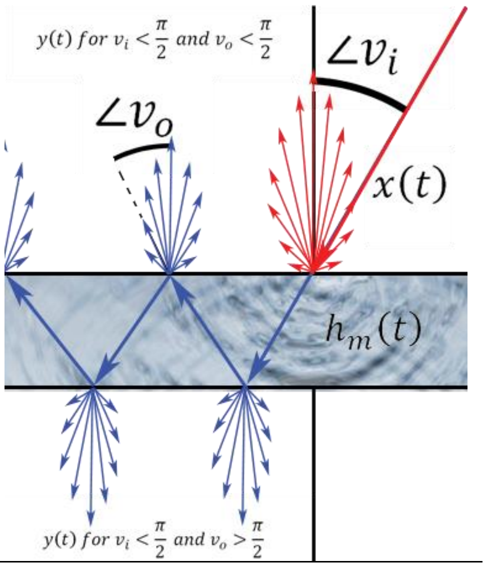
$y(t)\ for\ v_i < \frac{\pi}{2}\ and\ v_o < \frac{\pi}{2}$
$\angle v_i$
$\angle v_o$
$x(t)$
$h_m(t)$
$y(t)\ for\ v_i < \frac{\pi}{2}\ and\ v_o > \frac{\pi}{2}$

Figure 2. *How a diffuse medium receives an input signal $x(t)$ in red, and diffusively reflects it from different input/output angles following Lambert's cosine law*

## 4.2.1 Impulse responses

As we proved in 2.3, an IR together with a reflection parameter is enough to model occlusion and reflection of some medium. We can obtain impulse responses from objects by recording their acoustic response to an impulse, similar to how LTI systems are equivalent to their IR. This can for instance be done in an anechoic chamber, with a speaker in front of an object, and a microphone behind the object.

As the author didn't have access to an anechoic chamber, a different approach was tested out by attaching a contact microphone to a medium and generating acoustic impulses by popping balloons in vicinity, a technique also used for recording reverberant IRs of ambient spaces.[23] The resulting IRs[24] were satisfactory even for quick experiments in the authors opinion, however implementation were too processing intensive.

[22] Lambert, 1760
[23] Abel, et. al., 2010
[24] See annex 6.3.2

If we recall (4), using IRs to filter a signal requires convolution, a process requiring multiplying each signal value with all IR values, thus being similar to an $O(n^2)$ operation. If latency can be accepted, the convolution can be done in the frequency domain by using Fourier transforms. Fourier transforms have a complexity of $O(n^2)$, however for special values of $n$ there exists fast Fourier transforms (FFT), that can obtain complexities of $O(kn \log_2 n)$[25], which is significantly better, but still too much for the author's real-time implementation.

## 4.2.2 N-BAND PARAMETRIC EQUALIZERS

Instead, we can arbitrarily approximate an IR by a combination of $n$ parametric filters, giving a computational complexity of $O(n)$. Using an IR of an occluded door[26], recorded in 4.2.1, we can quite easily match the slope using these 4 filters:

1. Peak @ 70 Hz, 5 Q, 12 dB
1. Peak @ 140 Hz, 2.5 Q, 10.5 dB
2. Peak @ 500 Hz, 2 Q, 10 dB
3. One-pole low pass @ 700 Hz

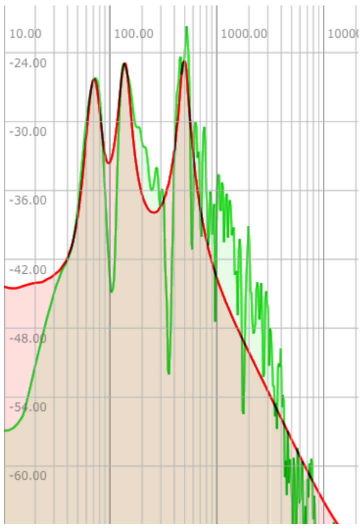

Figure 3. *log-log spectrum of IIR approximation (red), and green IR.*

Together these form an LTI system with an infinite impulse response (IIR). As we see in Figure 3, the slope generally matches but a lot of details are missing. This can be arbitrarily matched with additional filters, at the cost of extra linear processing power.

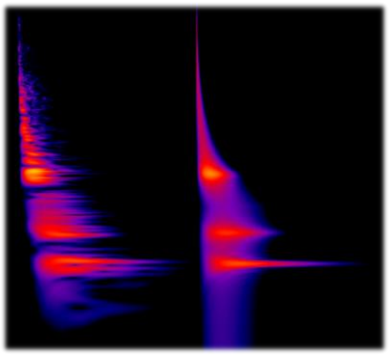

Figure 4. *log spectrogram with original IR to the left, and approximation to the right.*

Looking at Figure 4, we can see the inherit decaying resonance in the parametric filters matches the strongest resonance nodes in the original IR. The result of these filters can be heard in 6.3.2. Using parametric filters has the additional advantage of being modular at runtime, and any IR can be modelled "in the box", thus we have our arbitrary $M(t)$ used in (9).

[25] Orfanidis, 1996: p. 514
[26] IR file is annex 1/2 *"MS2013_Janus_Thorborg_Lydbilag1_Balloon_IR.wav"*, IIR approximation file is annex 2/2 *"MS2013_Janus_Thorborg_Lydbilag2_Model_IIR_IR.wav"*, see annex 6.3.2, p. 35

## 4.3 Sound propagation and occlusion

As has been outlined, emulating sound propagation to the degree that graphics are capable of today is not possible. Algorithms like radiosity "propagates" light throughout the scene iteratively from surface to surface. Now consider $N$ surfaces arranged such that only surface $B_n$ can see surface $B_{n+1}$. The radiosity algorithm takes $N$ passes to pass radiation to the last surface $B_{N-1}$. We will call this concept a completely non-convex map, and radiosity's complexity is $O(n^3)$ in this case.

### 4.3.1 Visibility function

Before we solve the propagation problem, we need a method for determining how well two surfaces see each other. This is the $V$ term in the rendering equation[27], giving a factor between 0 and 1. If our surfaces are the same size as the smallest possible colliding geometry element, a single ray cast between the surfaces suffices can determine if any other object occludes the path. Otherwise multiple ray casts can approximate a factor of occlusion.[28]

### 4.3.2 Handling convex geometry: Propagation algorithm

Doing audio-rendering passes of $O(n^3)$ complexity with no guarantee of $n$ passes being enough to pass radiation through an arbitrary map isn't ideal. Instead, we can move the complexity away to cheaper stage, at the cost of radiation accuracy. We create a map for each emitting source constituting a tree of paths to surfaces, ensuring we visit all possible nodes. The algorithm stages are:

1. Add direct paths for every directly visible surface
2. Use heuristic path finding algorithm for each non-visible surface using the closest visible surface, removing every visited surface from the non-visible collection
3. Reduce all paths to a tree with branches

The last step eliminates all redundant/duplicated paths, thus making the worst-case complexity of the rendering stage $O(n)$ for each source. A pseudo-code implementation of the path-finding algorithm is included in 6.3.3, p. 36. The algorithm works kind of like an $A^*$ algorithm, a

[27] See 2.5
[28] CryEngine does this - see Crytek, 2016.

common heuristic guided best-first path finding algorithm[29], by trying shortest distances, determining weights $\mathcal{W}$ of nodes in the map as penalties:

$$\mathcal{W}(p, B, B') = \frac{D(p, B, B')^2}{V(B, B')} \frac{1}{A(p, B, B')} \tag{10}$$

Where $D(p, B, B')$ is the distance from the position $p$ to the position of the surface $B'$ via $B$.

$$A(p, B, B') = \begin{cases} B_\rho \mathcal{U}(p, B, B'), & (B_p - p) \cdot B_n > 0 = (B_p - B'_p) \cdot B_n > 0 \\ (1 - B_\rho) \mathcal{U}(p, B, B'), & (B_p - p) \cdot B_n > 0 \neq (B_p - B'_p) \cdot B_n > 0 \end{cases}$$

Where $B_p$ is a position vector of the surface $B$, and $B_n$ is the normal to the surface $B$ and $\cdot$ is the dot product of two vectors:

$$a \cdot b = \|a\| \|b\| \cos \theta \tag{11}$$

$\mathcal{U}$ is the diffuse emission attenuation between the entry/exit angle and the surface normal and $\hat{}$ accent denotes the normalized vector, thus modelling Lambert's cosine law for diffuse surfaces:

$$\mathcal{U}(p, B, B') = \left| V_a(B_p - p, B_n) V_a(B_p - B'_p, B_n) \right| \tag{12}$$

$$V_a(a, b) = \hat{a} \cdot \hat{b} \tag{13}$$

Therefore, $A$ is just a complicated way to select amplitude penalty for reflecting off the surface $B$ to reach $B'$, or by travelling through the surface $B$ (thus producing occluded sounds), by checking whether the entry/exit angles are placed on the same hemisphere, relative to the surface normal for $B$.

The included algorithm will short-circuit if a direct path is possible from any node to the destination node ($V(B, B') > 0$ while not intersecting the surface), even though it isn't the most optimal path. The complexity of the presented algorithm is worst-case $O(n!)$, compared to $A^*$ that is $O(2^n)$. The short-circuit and distance sorting gives $O(1)$ complexity for convex maps (open spaces) and $O(n)$ for completely non-convex maps. Both of these scenarios are common for sound paths, and the algorithm is designed for these cases. Only visited dead ends increases complexity.

[29] Hart, et al., 1968

### 4.3.3 Form factors

A form factor between two surfaces $B$ and $B'$, introduced in 2.5, defines the strength of the radiation leaving one surface to the other. A solution for calculating these are given by Goral, et al (1984). Although it currently isn't modelled in the path finding algorithm, it would replace the $\mathcal{U}$ term used in (12) and (9) would have to be redesigned.

### 4.3.4 Modelling emissivity

If we consider the radiation $E_n$ reaching a surface $B_n$, we can form an LTI system for a path from the path finding algorithm. Let $P$ be the number of entries in some path $\mathcal{P}_z$ for a source signal $x_z(t)$, with $B_n$ being the last. $v_{i_p}$ is then the entry angle to a surface $B_p$ with a translucent filter $M_p$, and $v_{o_p}$ is the exit angle towards the next surface. $a_p$ is the delay between the previous and the current surface, and let $F_p$ be the geometry attenuation. Then a LTI system exists for the path:

$$\mathcal{P}_z(v_o, t) = F_{P-1}M_{P-1}\left(v_{i_{P-1}}, v_o, t - \alpha_{P-1}\right) * F_p M_p\left(v_{i_p}, v_{o_p}, t - \alpha_p\right) * \dots F_0 M_o\left(v_{i_0}, v_{o_0}, t - \alpha_0\right)$$

Where unqualified $v_o$ represents the output emission angle. Let $Z$ be the number of sources, then:

$$E_n(v_o, t) = \sum_{z=0}^{Z-1} \frac{x_z(t)}{F_z} * \mathcal{P}_z(v_o, t) \tag{14}$$

The total emission $E_n$ from the surface $B_n$ with an output angle of $v_o$ is then the summed LTI response from all the sources through their paths, distance attenuated for each link.

## 4.4 Modelling spatialisation

We refer to spatialisation as transforming some signal with an imaginary position into another form, where humans are able to identify the position (sound localization). The transformation is based upon a set of unique features for a human including, but not limited to, head size, ear geometry and shoulders etc. Collectively, it is called a head-related transfer function (HRTF) which tries to generalize the parameters to a model, that suits most humans. HRTF models are LTI systems[30], thus we can manipulate them as in 2.3, meaning we can model HRTFs by impulse responses. A non-scaled HRTF $H$ is characterised by azimuth- (horizontal plane) and elevation angles (vertical/median plane), $\theta$ and $\varphi$ respectively:

$$y(t) = x(t) * H_{\theta\varphi}(t) \tag{15}$$

### 4.4.1 Procedural HRTF-model

There are lots of readily available impulse responses representing HRTFs at different angles on a sphere[31], such that you can linearly reconstruct a HRTF for every angle with some quantization error. However, for our purposes we will instead design a procedural LTI system due to problems described in 4.4.2.

Humans have two ears, meaning a signal arrives differently at each ear. There are two effects important effects here, interaural time difference (ITD) $\alpha$ and interaural level difference (ILD) $\beta$ - that is, the brain can deduce a sound is coming from the left, if the left ear picks up a larger signal, and the signal arrives earlier at the left ear. These are functions of the azimuth, and their role in sound localization is referred to as the duplex theory, formalized by Lord Rayleigh.[32] Additionally, there exists a filter $\gamma$ for the elevation.[33]

$$H_{\theta\varphi}(p,t) = \frac{\beta_\theta(t-\alpha) * \gamma_\varphi(t-\alpha)}{1+|p-p_e|} \tag{16}$$

Our modelled HRTF thus is a function of $\beta$ and $\gamma$ filters, delayed by the ITD, scaled by the distance between the centre of the head position $p$ and the ear $p_e$. As we have determined that human ears use time delays for spatial information, we therefore also have to account for wave propagation delays in all paths without losing generality, contrary to illumination.[34]

The ILD is dependent on frequency, as the head itself acts as a filter for sound waves whose wavelength are smaller than the head obstruction angle, thus the wave is unable to diffract around the head. Experiments shows this approximates a first order low-pass filter whose centre frequency

[30] Xiaoqi, 1996.
[31] For instance, Gardner & Martin, 1994. See also 3.
[32] Carlile, 1996: p. 28
[33] Carlile, 1996: p. 66
[34] See 2.6

is around $700 - 1000$ Hz for a worst case angle of $90 - 135°$, depending on where the ears are placed on the head[35].

$$y(n) = b_0 x(n) - a_1 y(n-1) \tag{17}$$

An example filter design in discrete time for worst case angle 90° can look like this:

$$b_0 = e^{-2\pi\frac{\omega_\theta}{f_s}}, \qquad a_0 = 1 - b_0$$

$$\psi(v) = \frac{1 - \cos v}{2}, \qquad \omega_\theta = 700\psi(2\theta) + \frac{f_s \psi(\pi - 2\theta)}{2}$$

Where $f_s$ is the sampling rate. By LTI equivalence, (17) is thus our $\beta_\theta$ filter. For the elevation filter, the author found no simple correlation, so we will have an empirical look at a set of head-related impulse responses on a KEMAR[36] model (Figure 5), noting our method no longer represents the ground truth from 2.2. Interestingly, we can see some deep ridges translated by angle.

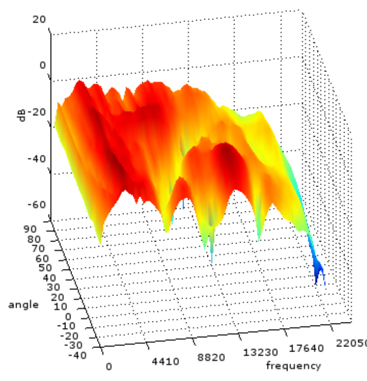

Figure 5. *HRIR plots for varying elevation angles. HRIR source: Gardner, B. & Martin K. (1994).*

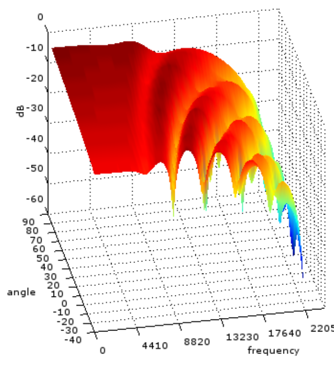

Figure 6. *Bandpassed comb filter.*

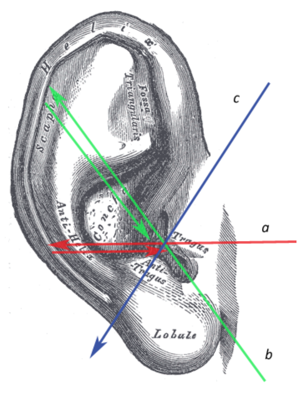

Figure 7. *Pinna anatomy. Original image source: Gray, H.(1918)*

We can emulate the notches using a delay mixed with the original, creating a comb filter. Additionally, the strength seems to depend on the angle, being more prominent for negative angles. Figure 6 is a plot of the designed filter, bandpassed to match the shape of Figure 5 for illustration

[35] Carlile 1996: p. 36
[36] A head and torso simulator for acoustic research

purposes. Figure 7 might explain this effect by nature of the pinna's helix reflections[37]; we can see that ray $a$ representing the angle $\angle 0$ travels a smaller distance than the ray $b$ with a negative angle. Additionally, the helix structure becomes less prominent for positive angles, illustrated by $c$ that doesn't collide with the helix and thus doesn't reflect back. Our elevation filter $\gamma_\varphi$ is thus:

$$y(t) = x(t) + A_\varphi x(t - \sigma_\varphi) \tag{18}$$

Extrapolating the values used for figure 2 yields the following filter design:

$$A_\varphi = 0.35\psi\left(\pi - \varphi + \frac{\pi}{2}\right) + 0.1\psi\left(\varphi + \frac{\pi}{2}\right)$$

$$\sigma_\varphi = 3.5 \times 10^{-4}\psi\left(\pi - \varphi + \frac{\pi}{2}\right) + 7.5 \times 10^{-5}\psi\left(\varphi + \frac{\pi}{2}\right)$$

From these two components (17) and (18), we can spatialise sound for a virtual human listener.

### 4.4.2 View size obstruction

One last thing to consider is that HRTF models and commonly available HRIRs *only* model point-sources, that is, theoretically infinitively small sources of sound with a perfect spherical radiation pattern. Most objects however, radiate sound over a larger surface. For a simple solution, we can model objects we listen to as spheres. Selecting a radius $r$ as the modelled size, we can model how much of our view the object fills:

$$v = \tan^{-1}\frac{r}{|p - p_o|}$$

Where $p_o$ is position of the object we listen to. We can thus define a diffusing function:

$$d_f(v) = v\psi\left(\pi - \tan^{-1}\frac{r}{|p - p_o|}\right) \tag{19}$$

We can then diffuse our filters by scaling their angles with (19), emulating objects of $r^2$ size while keeping other properties like correct ITD and distance attenuation based on ear positions. Note this also can be approximated using HRIRs, by averaging multiple HRIRs in a disk pattern around the selected angle on a sphere, but it becomes expensive.

[37] Google VR audio (Google; spatial audio, 2016) uses similar illustrations and refers to this effect as "spectral filtering"

## 4.5 Modelling reverberation

### 4.5.1 Reflection passes and infinitesimal approximation

To accurately simulate reverberation, each surface has to reflect into any other surface infinitively many times. Nothing in our model prevents a surface's emission factor $E$ to be influenced by other surfaces, using an algorithm similar to (8), the radiosity equation. However, each pass will additionally be $O(n^2)$ complexity, compared to our original complexity of $O(n)$.

Since the reflections form a possibly instantaneous feedback system with respect to the sampling rate of the discrete system, either the feedback system has to be solved analytically[38], or care has to be taken that the delay in the reflection is larger than the processing latency of the algorithm, as audio is often rendered in blocks.

Additionally, each surface has to technically be subdivided in infinitesimally small patches to accurately model sound waves and not diffuse rays. These problems - except the feedback as lightning today doesn't model time[39] - also has to be solved for lightning algorithms like radiosity, where typical solutions are some fixed number of reflection passes (usually between 0 and 10), and adaptive patch division using heuristics to determine where it is needed for accuracy. [40]

### 4.5.2 Realistic approaches using DSP effects

To keep complexity down, we can instead note that direct signals and our first-order reflection generated by our surfaces contains primary information about localization and sound intensity. Assuming this is acceptable, we can synthesize reverb using the output of our system as "impulses" with spatial timbre and feeding it into a parametric algorithmic reverb, that generates the rest of the mostly diffuse room sound. In this case, our system will generate the part normally known as pre-delay, modelling the early reflection parameter.

This effect stage can be done anywhere in the system as long as the reverb is an LTI system (every reverb IR is, by definition). Indeed, the surface filters modelled in 4.2 can implement reverberant effects giving correctly spatialised reverb.

[38] Filters can be solved for zero-delay feedback, for instance - see Zavalishin, 2015
[39] See 2.6
[40] Lozano-Perez & Popovic, 2016: p. 23

### 4.5.3 AUDIO RENDERING EQUATION

If we define $Q$ outputs at positions $p_q$ for the system (commonly 2 for stereo speakers and headphones), our complete audio rendering equation including localisation, occlusion and spatialisation is then:

$$y_q(t) = \sum_{z=0}^{Z-1} \frac{V\left(p_q, x_{z_p}\right) H_{\theta_z \varphi_z}(p_q, t) * x_z(t - \alpha_z)}{\left(\left|p_q - x_{z_p}\right|\right)^2} + \sum_{i=0}^{I-1} \frac{V\left(p_q, B_{i_p}\right) H_{\theta_i \varphi_i}(p_q, t) * E_i\left(\cos^{-1} V_a\left(p_q, B_{i_n}\right), t - a_i\right)}{\left(\left|p_q - B_{i_p}\right|\right)^2} \quad (20)$$

**In other words**, the upper term is the sum of HRTF-"listening" to each source $x_z$ delayed by distance $\alpha_z$, scaled by distance and visibility $V$ from 4.3.1, while the lower term is HRTF-listening to the $a_i$ delayed emission $E_i$ from each surface $B_i$ with the angle between the listener position and the surface normal of the surface $B_i$ from (13), scaled by distance and visibility.

We have thus, albeit in other terms, come to a result that emulates our defined reverberation, and therefore solved the problem posed in 1.1 by solving the problems in 1.2.

# 5. Review and conclusion

## 5.1 Prototype implementation

The given solution in (20) can be realized in software today by representing each sound emitting object (surfaces and sources) as delay lines. Each surface additionally has interfaces for creating an abstract filter state as covered in 4.2, and to filter a signal using the abstract state. With engine support for ray casting, all necessary components to build the system is covered.

A free, open-source system has been implemented by the author in the game engine Unity3D; 6.3.1 covers how to acquire the source code and ready-to-use builds. The project will however be presented here by referring to scenes in a pre-rendered video, found in 6.1 on page 32, demonstrating use cases discussed in this project, as well as all parameters considered so far.

## 5.2 Evaluation and reflection

### 5.2.1 Tests, performance and optimizations not covered

First, let us consider what happens when we scale the system, as that has been our primary concern. We will consider the scene "Non-Convex Maze", and duplicate all geometry in it 5 times vertically displacing it and investigate processing load percentage[41]:

| | 1x | 2x | 3x | 4x | 5x |
|---|---|---|---|---|---|
| Audio% | 9% | 17% | 26% | 35% | 43% |
| Physics% | 2.5% | 6.5% | 13% | 20% | 36% |
| Path finding% | 1.5% | 3.5% | 6.2% | 9.5% | 15% |

As we can see, the audio rendering scales very close to linearly, as the design predicted in 4.3.2. Physics handles pre-calculation of geometry from each surface to the other - an $O(n^2)$ problem, and as the results tell, the scaling is very close to quadratic. The path finding algorithm keeps around $O(n)$ in this case, again per the design.

[41] See 6.1, scene 8, p. 32

Some aspects of the solution do not scale linearly, for instance, the path finding algorithm in 4.3.2, and reflection passes, if implemented, covered in 4.5.1. However, at lot of the potential optimizations reside in distance attenuation. It doesn't take a lot of distance before sound waves and reflections decay to inaudible levels, and as of such you can easily have 100 $N$-sized "rooms" in a level with $O(n^2)$ complexity, that doesn't interfere audibly, and thus only require a complexity of 1%, compared to not using this optimization. This can also be applied to sources, such that adding sources in non-interfering rooms do not increase complexity at all.

Additionally, the presented solution and implementation is completely dynamic and recalculates all graphs, paths and filters on each physics update. If no geometry has changed, nothing actually has to be recalculated. For scenes with static geometry, caching these calculations can therefore reduce the computational complexity to nothing (this is referred to as *baking*).

Notably, each branched reflection path in the presented solution has no dependency on other paths, thus the system is highly parallelisable. Consumer-grade computer processors today have between 2-8 fully parallel cores and processing instructions sets like Intel's AVX allows to execute 16 32-bit algorithms[42] in parallel, giving a potential 128x performance increase depending on the system for rendering sound paths - the major bottleneck of the solution. Also note the presented data is not optimized at all, dramatic difference can occur if fine-tuned (however, the scaling will remain the same).

### 5.2.2 Review of original problem and how it was solved

All stated problems in 1.1 and 1.2 have either been modelled directly, or practical approximations have been presented and discussed. With the presented solution in (20), we are actually modelling the sound as it travels through the environment dynamically, and our model of surface filters in 4.2 proved crucial to model how sound is transmitted through mediums, such that we can model occlusion and obstruction with arbitrary precision. As we model sound delay propagation as well, a list of modelled physical parameters include:

- Accurate dynamic Doppler-effects on all geometry, and all reflection paths (scene 1, 2 and 10)

[42] Intel, 2013

- Accurate distance delays on all paths (scene 2 and 11)
- Distance attenuation following inverse-square laws of spherical radiation (scene 5)
- Complete and partial occlusion (scene 6, 9 and 11)
- Surface filters that models translucency by dispersion, absorption, reflection and refraction effects, room bleeding (scene 9, 11 and 12)
- Diffuse angle attenuations following Lambert's cosine law (scene 7)
- Head-related transfer functions transforms on spatial sounds achieving sound localisation (scene 1, 4 and 5)
- Source size, i.e. generalization of point-source radiation by diffusing HRTF filters (scene 3)
- First-order approximations of sound wave propagations and reflections modelling sound occlusion and obstruction through arbitrary geometry (scene 8, 11, 12 and 13)

Notable problems in the solution however include aspects not covered physically correct like form factors and room reverb, as discussed in 4.5.1. Radiation patterns for sources are exclusively modelled as spherical, meaning sound directivity isn't modelled - although it will be trivial to add. Additionally, the important effects of sound wave diffraction is not modelled at all, except in the HRTF filter model through ILD. As a result, the solution is not a general solution for simulating room reverberation, but can be used for simulating sound wave propagation. Interestingly, as reflections and room sound are now emulating in the runtime geometry instead, the sound-design process historically done in external music software programs should now be carefully accounted for in the level design of virtual 3D environment, instead.

With the presented model of surfaces, we can gradually increase the accuracy of the simulation as more processing power is available, thus making the model future-proof to some extent. Interestingly, as our solution decouples listener input and environment output, we can add arbitrary amounts of listeners in different spatial positions with correct output with no additional complexity for the environment, something Unity3D doesn't support as of this writing.

Lastly, as the modelled HRTF transforms are parametric, we can model any output speaker matrix correctly, like standard external stereo speakers and surround speakers. Classical binaural HRIRs only supports head-phones (ideally in-ear headphones), as the filtering of the pinna is unavoidably integrated, recorded by binaural in-ear microphones.

### 5.2.3 Comparisons with existing systems

Major game engines today support occlusion to varying degrees, but only through direct line of sight. Even state-of-the-art 3rd party audio engines like Wwise only have access to data given from the game engine, and they only model occlusion/obstruction as filtered attenuation.[43] As of such, examples given in the introduction are yet to be solved by these systems.

As of such, most new architectures solving this problem need to integrate tightly with the game geometry, either explicitly like Google's VR, or implicitly using Tsingos & Gascuel's evaluation of the scene on the graphics card. However, techniques like Google's VR still doesn't model anything but direct line of sight with accumulating attenuation. While the room model is interesting, it only works for cubic geometry, and rooms do not interact acoustically, eliminating sound propagation beyond only working in a subset of geometry used in games today.

While Tsingos & Gascuel's performance results would probably be vastly better today, the lowest calculation time reported (100 ms[44]) to calculate sound maps (i.e. not actually rendering audio) combined with system latency from/to the graphics card suggests real-time evaluation is probably still a bit away. It is also not clear how the complexity would evolve for the case of completely non-convex maps, as the diffraction is evaluated through a frustum[45], not an entire scene.

Lastly, we will consider Two Big Ears, with similar concepts to this solution. The inability to assess the product and/or technique leaves us inconclusive, but it is interesting whether the path finding approach chooses to emulate diffraction or not - the not-yet implemented geometry-based reflections suggests so, which leaves out the environment resonance as modelled in 2.3, and additionally results in only one out of many paths to be rendered, contrary to presented system[46]. Whether transmission is supported as a path of the path finding system is also an open question.

[43] AudioKinetic, 2016
[44] Tsingos & Gascuel, 1998: p. 5, table 1
[45] Tsingos & Gascuel, 1998: p. 4
[46] See scene 13 in 6.1, p. 32

## 5.3 Conclusion

A near-complete physical model for evaluating sound occlusion and obstruction for virtual 3D environments has been presented, together with an actual implementation proving the concept works in real-time. To solve the problem, literature from graphics and diffuse physics radiation was applied to model the aspects of radiation on surfaces, while digital signal processing theory allowed us to model how acoustic waves travel through and interact with materials in arbitrary detail. Combining these, we used path finding heuristics to model simple first order reflections completely through the environment, modelling wave propagation and thus solving the problem of sound occlusion and obstruction intrinsically, together with sound localization.

There are still many difficulties in this area, and the presented solution is merely an pragmatic approximation, intended to keep complexities scaling at reasonable levels. Many aspects can be changed and the author invites collaboration as the entire project is open-sourced.

Additionally, the solution provides for new interesting sound design dynamics as the environment can change the sound in arbitrary and creative ways. Hopefully this inspires using, creating and designing sound in new dimensions, perhaps even being relevant to acousmatic music artists.

## 5.4 Future work

Most implementation technology remains either free but closed-source, or proprietarily protected and costly. Hopefully more investigation will be done, as the motivation for this project exclusively was the lack of solutions altogether. For improving this solution, these topics should be looked at (this has also been discussed at relevant areas in this report):

- Implementation of optimizations not covered, or present a model that proves this
- Modelling view factors for general surfaces
- Empirical comparisons with common existing path finding algorithms to assert the importance of calculating the shortest path in sound propagation
- Better visibility functions, perhaps through diffraction
- Sound diffraction, directivity of sources, radiation patterns
- Cross-fading significantly different path trees, as they can create discontinuities

### 5.4.1 Alternate approaches

An interesting implementation approach could be to use a simplified model for surface filters like a short FIR filter, and trace rays around collecting only amplitudes, delays and filter responses. From this, a single impulse response can be derived that completely models an environment path, instead of iteratively mixing $N$ surfaces in $N$ stages.

Note however, that the final down mixing stage in such a solution still includes $N$ delays, suggesting a huge cumulative impulse response. However, using convolution by FFTs scales much better as $O(n \log n)$ compared to $O(n^2)$ reflection passes, as covered in 4.2.1.

# 6. APPENDIX

## 6.1 AUDIO/VISUAL PRESENTATION OF PROTOTYPE IMPLEMENTATION

*Lydprodukt 1/1: Video "MS2013_Janus_Thorborg_ Lydprodukt.mp4", length: 10 minutes 21 seconds*

A video demonstrating and showcasing all discussed features with annotation and analysis has been produced for this report. It mainly consists of 13 scenes, designed to show features and properties of the presented solution in isolation, to allow for better dissertation. Following is a list of scenes, together with time stamps, and lastly their relevant chapters in this document, for cross-references.

1. 00:13 - shows Doppler effects and HRTF functions analytically (4.4.1)
2. 00:45 - shows Doppler effects and speed of sound delays (4.4)
3. 01:08 - shows view size obstruction (4.4.2)
4. 01:33 - shows horizontal and vertical HRTF functions (4.4.1)
5. 01:53 - shows multiple sources (4.4 and 4.5.3)
6. 02:13 - shows sound occlusion through visibility function (4.3.1)
7. 02:27 - shows surfaces (2.3), diffuse laws and transmission (4.2), emission (4.3.4), occlusion and path finding using the concept of wave guides (4.3.2)
8. 03:34 - shows path finding through arbitrary geometry (specifically "completely non-convex" geometry), both occluded and not (4.3)
9. 05:53 - shows adaptive path finding (4.3.2), partial occlusion by transmission (4.2) and complete occlusion (4.3.1)
10. 06:24 - shows creative use of surface filters (4.2.2), spatialisation (4.4) and occlusion (4.3.1), and LTI system equivalence (2.3)
11. 07:05 - shows occlusion and presented solution's ability to bleed sounds natively through rooms, also through occlusion paths (4.3.2) using surface filters (4.2.2), also mentioned in 5.2.3
12. 08:43 - shows game mechanics that can be constructed from this system using audio cues - this case was specifically discussed in 1.1
13. 09:25 - shows a situation that can only be solved using audio cues, i.e. enabling navigation through audio, discussed briefly in 1.1.

A copy of the video should have been delivered together with this document. If not, the video is available here on YouTube: https://youtu.be/cRWPs-Ns1Pc

The video can also be downloaded here: www.jthorborg.com/occlusion-research/bvideo.mp4

## 6.2 BIBLIOGRAPHY

### 6.2.1 LITERATURE

1st Surname, F. [; nth Surname, F.] (year). [“Chapter”, p. x-y;]. *Title*. [City:] Publisher [, revision].

- Carlile, S. (1996). "The Physical and Psychophysical Basis of Sound Localization", p. 27-28; p. 35-36; p. 66-68. *Virtual Auditory Space: Generation and Applications*. Chapman & Hall.
- Gerzon, M. A. (1973). *Periphony: With-Height Sound Reproduction*. Journal of the Audio Engineering Society.
- Goral C. M.; Torrance K. E.; Greenberg D. P.; Battaile B. (1984). p. 1; p. 4. *Modeling the Interaction of Light Between Diffuse Surfaces.* Computer Graphics, Volume 18 - nr. 3.
- Hart, P. E.; Nilsson, N. J.; Raphael, B. (1968). *A formal Basis for the Heuristic Determination of Minimum Cost Paths*. IEEE Transactions on Systems Science and Cybernetics.
- Lambert, J. H. (1760). *Photometria, sive De mensura et gradibus luminus, colorum et umbrae.* Leipzig: W. Engelmann.
- LaViola, J. J. Jr. (2000). *A discussion of cybersickness in virtual environments*. ACM SIGCHI Bulletin.
- Lloyd, L. S. (1970). p. 169. *Music and Sound*. Ayer Publishing.
- Orfanidis, S. J. (1996). "Sampling and reconstruction: Review of analog signals", p. 3; "Discrete time systems: Impulse response" p. 106, "DFT/FFT Algorithms", p.514. *Introduction to Signal Processing*. New Jersey: Prentice Hall.
- Tsingos, N; Gascuel, J. (1998). p. 4-5. *Fast Rendering of Sound Occlusion and Diffraction Effects for Virtual Acoustic Environments*. AES Convention 104.
- Xiaoqi, Z. (1996). p. 46-50. *Virtual reality technique*. Telecommunications Science.

### 6.2.2 IMAGES

Surname, F. (year). [“Chapter”, p. x-y;]. *Title*. [City:] Publisher [, revision].

- Gray, H. (1918). *Anatomy of the Human Body.* Lea and Febiger, 20th edition.

### 6.2.3 References

[1st Surname, F. [; nth Surname, F.] | company] (year). [“Chapter”, p. x-y;]. *Title*. Type. [Revision.] Link.

- Abel  J. S., Bryan N. J., Huang P. P., Kolar M. A., Pentcheva B. V. (2010). "RIR Measurement Approaches", p. 4. *Estimating Room Impulse Responses from Recorded Balloon Pop.* PDF/slides. Retrieved 4/12/2016 at: https://ccrma.stanford.edu/~njb/research/AES129_Balloon_Slides.pdf
- The Free Dictionary (2011). "ground truth. (n.d.)". *American Heritage® Dictionary of the English Language, Fifth Edition*. Retrieved 30/11/2016 at: http://www.thefreedictionary.com/ground+truth
- AudioKinetic, (2016). "Obstruction and Occlusion in Environments". *Wwise SDK 2016.2.0*. Online Wwise reference. Retrieved 12/12/2016 at: https://www.audiokinetic.com/library/edge/?source=SDK&id=soundengine__obsocc.html
- Crytek, (2016). "ATL for Designers: Sound Obstruction/Occlusion". *CRYENGINE Manual*. Online reference. Retrieved 2/12/2016 at: http://docs.cryengine.com/pages/viewpage.action?pageId=18384659
- Gardner B., Martin K. (1994). *HRTF Measurements of a KEMAR Dummy-Head Microphone.* Online HRTF library. Retrieved 15/11/2016 at: http://sound.media.mit.edu/resources/KEMAR.html
- Google, (2016).  "How spatial audio works". *Spatial Audio*. Online audio VR SDK documentation. Retrieved 9/12/2016 at: https://developers.google.com/vr/concepts/spatial-audio
- Google, (2016). *Google VR SDK*. Online portal for Google's VR SDKs. Retrieved 9/12/2016 at: https://developers.google.com/vr/
- Henderson, T. (2016). "Sound Waves and Music: Reflection, Refraction and Diffraction". *The Physics Classroom.* Online physics textbook. Retrieved 7/12/2016 at: http://www.physicsclassroom.com/class/sound/Lesson-3/Reflection,-Refraction,-and-Diffraction
- Intel (2013). *Intel® Advanced Vector Extensions 512 (Intel® AVX-512)*. Online outline of the capabilities of the Intel AVX instruction set. Retrieved 9/12/2016 at: https://software.intel.com/en-us/blogs/2013/avx-512-instructions
- Lozano-Perez, T.; Popovic, J. p. 6; p. 23. *Radiosity*. PDF/MIT Lecture slides on graphics. Retrieved 6/12/2016 at: http://groups.csail.mit.edu/graphics/classes/6.837/F01/Lecture20/lecture20_4up.pdf
- Mohr A. (2007). "Introduction", p. 2. *Quantum Computing in Complexity Theory and Theory of Computation*. PDF. Retrieved 2/12/2016 at: http://www.austinmohr.com/work/files/complexity.pdf
- Two Big Ears (2016). *3Dception for games*. Online commercial for their spatial audio solution. Retrieved 9/12/2016 at: http://twobigears.com/3dception.php
- Zavalishin, V. (2015). *The Art of VA Filter Design*. PDF book. Retrieved at 7/12/2016 at: https://www.native-instruments.com/fileadmin/ni_media/downloads/pdf/VAFilterDesign_1.1.1.pdf

## 6.3 Additional annex

### 6.3.1 Source code and builds

The source code is released under the GNU GPL v3 license, meaning it is completely free to use, share and make derivative works from, as long as you provide the source code yourself. Bear in mind the prototype was developed to illustrate and demonstrate the general purpose solution presented here, not in fact itself being usable for production. The repository, containing the source code is available online here:

https://bitbucket.org/Mayae/soundocclusion

Alternatively, it can be cloned directly using git here:

https://bitbucket.org/Mayae/soundocclusion.git

Ready-made builds so you can try the scenes yourself for Windows and OS X are available here:

https://bitbucket.org/Mayae/soundocclusion/downloads

Additional information and further work on this project can be found here:

www.jthorborg.com/index.html?ipage=occlusion

### 6.3.2 Sound occlusion using impulse response research

Referenced files in this project ("lydbilag"), that should also have been included with this report:

- 1/2*:* file *"MS2013_Janus_Thorborg_Lydbilag1_Balloon_IR.wav"*
- 2/2: file *"MS2013_Janus_Thorborg_Lydbilag2_Model_IIR_IR.wav"*

Under chapter 4.2.1, using recorded impulse responses of actual objects to design the surface filters, modelling transmission, emission and occlusion mentioned in 2.3, a bunch of interesting files was created very closely approximating the results of real occlusion. Due to technical reasons this method was not used, but the complete results, including the mentioned impulse response files applied to samples, of this approach are available for listening here:

www.jthorborg.com/occlusion-research/examples.zip

## 6.3.3 Sound propagation algorithm

```
let source = sound playing object
let paths = array of path
define candidate node
---
foreach destination in non-visible surfaces:
    let candidates = visible surfaces, sorted by distance to destination
    foreach target in candidates:
        let path = FindPath from player position using target to destination
        if path exists:
            remove identical elements from non-visible surfaces and path
            add path to paths

join paths to optimized tree graph
source propagates sound through each node stage in the graph
---
procedure FindPath, from player position to destination via target:
    let path = empty array of surfaces
    candidate node = nothing
    add target to path
    let surfaces = all possible surfaces without destination
    if RecursiveTracing from source position to destination via target, surfaces is possible:
        report success
---
procedure Trace, from position to destination via target:
    let node = connective angle between position, target and destination
    calculate penalty of node based upon link distance, diffuse angle and translucency
    if target can see destination directly:
        report success
    else if angle intersects the target plane, and target is translucent:
        candidate node = target, if penalty is smaller
    else
        report failure
---
procedure RecursiveTracing, from position to destination via target, using a list of surfaces:
    if Trace from position to destination surface via target is possible:
        add destination to path
        report success

    let surfaces = surface list, sorted by distance from line segment from target and destination
    foreach candidate surface in surfaces:
        if Trace from position to candidate surface using target is possible:
            let new path = path with candidate surface
            let new surfaces = surfaces without candidate
            if RecursiveTracing from target position to destination via candidate, using new surfaces is possible:
                let best node = select least penalty from new node and candidate node
                add best node to new path
                report success

    if candidate node exists:
        remove candidate from surfaces
        if RecursiveTracing from target position to destination via candidate, using surfaces is possible:
            add candidate node to path
            report success
```

## 6.4 CHARACTER CALCULATION

Characters from chapter 1 to 5 inclusive, including spaces and all titles, excluding top and bottom headers: **47.024 characters**

Figure space:

| **Figure** | **Metrics** | **Area** |
|---|---|---|
| **Figure 1** | 7,1cm x 6,2cm | 44,02cm2 |
| **Figure 2** | 6,7cm x 6,6cm | 44,22cm2 |
| **Figure 3** | 6,7cm x 5,2cm | 34,84cm2 |
| **Figure 4** | 4,6cm x 5,6cm | 26,1cm2 |
| **Figure 5** | 5,7cm x 6,9cm | 39,33cm2 |
| **Figure 6** | 5,8cm x 5,5cm | 31,9cm2 |
| **Figure 7** | 5,75cm x 5cm | 28,75cm2 |
| **Total** | N/A | 249cm2 |

Equivalent text occupied by figures, assuming 2400 characters fit on a standard A4 page with the metrics of 21cm x 29,7cm (area = 623,7 cm2):

2400 * 249 / 623,7 = **958 characters**

Final character metric of this report: **47982 characters**

## 6.5 RESUME

The effects of sound occlusion and sound obstruction are investigated and modelled, to propose a solution that allows this effect in virtual 3D environments (such as games) with arbitrary geometry, together with implementing spatialisation and localisation. The solution is specifically designed to handle completely non-convex geometry, since the emitted audio has encoded localisation cues, modelled after acoustic physics, so the system can be used for spatial navigation and attention.

The proposed solution uses theory from heat radiation physics to reduce geometry to surfaces to calculate paths and emissivity. Audio effects and filters are used to simulate sound moving through spaces with delays and Doppler-effects, as well as the filtering of surfaces themselves, whether the sound moves through or reflects off. Head-related transfer functions (HRTF) that describe how humans are able to localize sounds are derived from sound localisation theory and binaural impulse responses to implement a procedural down mixer, that spatialises mono sounds as though they come from a specific direction.

The solution is measured, discussed and compared to relevant work, as well as implemented in a current 3D games engine today, to present the system using some situations and scenes, showing how it can deal with discussed cases, such as completely non-convex geometry.