ScatAR_WGW: Implementation and Evaluation of the Waveguide Web in an Application for Artificial Reverberation in a Virtual Environment

Master´s Thesis

Jonas Holfelt

Aalborg University Copenhagen
Sound and Music Computing

AALBORG UNIVERSITY
STUDENT REPORT

Sound and Music Computing
Aalborg University Copenhagen
http://www.aau.dk

Title:
ScatAR_WGW: Implementation and Evaluation of the Waveguide Web in an Application for Artificial Reverberation in a Virtual Environment

Theme:
Master Thesis

Project Period:
Spring Semester 2018

Project Group:
Jonas Holfelt

Participant(s):
Jonas Holfelt

Supervisor(s):
Cumhur Erkut

Copies: 1

Page Numbers: 56

Date of Completion:
May 31, 2018

Abstract:

Accurate reverberation algorithms are required for authentic experience in virtual or augmented reality experiences. A key element of immersive sound in virtual environments are the room acoustical modeling. Recent efficient and versatile algorithms has been proposed, namely the Scattering Delay Networks (SDN) and the Waveguide Web (WGW). This work presents the implementation of WGW in a current augmented reality application facilitating SDN reverberation. The application obtains informations about a given environment based on the positions of the sound source and listener. An evaluation is performed in a virtual environment to conclude a significant perceptual difference between WGW and SDN, with a significant preference to SDN over WGW. WGW is deemed too computational for real-time applications, but poses for a wide range of interesting possibilities for future work in the domain of acoustical modeling of outdoor spaces.

AALBORG UNIVERSITY
STUDENT REPORT

Sound and Music Computing
Aalborg Universitet København
http://www.aau.dk

Titel:
ScatAR_WGW: Implementering og Evaluering af Waveguide Web i en Applikation for Kunstig Rumklang i et Virtuelt Miljø

Tema:
Master Thesis

Projektperiode:
Forårs Semester 2018

Projektgruppe:
Jonas Holfelt

Deltager(e):
Jonas Holfelt

Vejleder(e):
Cumhur Erkut

Oplagstal: 1

Sidetal: 56

Afleveringsdato:
May 31, 2018

Abstract:

Præcise algoritmer til kunstig rumklang er nødvendige for autentiske oplevelser i Virtual Reality (VR) eller Augmented Reality (AR) oplevelser. Et vigtigt aspekt inden for realistisk lyd i virtuelle miljøer er akustisk modellering af rumklang. Scattering Delay Networks (SDN) og Waveguide Web (WGW) er eksempler på nye og alsidige algoritmer. Denne rapport præsenterer implementering af WGW i en nuværende AR applikation, der benytter SDN genereret rumklang. Applikationen samler information fra et givet miljø baseret på lydkildens og lytterens positioner. En evaluering udføres i et virtuelt miljø, og kan konkludere at der er en betydelig forskel mellem WGW og SDN, med en betydelig præference for SDN. WGW viser sig at være for kompliceret til at kunne køre i real-time applikationer, men opfordrer til nye interessante muligheder inden for akustisk modellering af udendørs områder.

Contents

Preface

This thesis is the final work of my Master education in Sound and Music Computing at Aalborg University in Copenhagen. Through the education I have been in touch with a lot of different fields within Sound and Music Computing, and for my thesis I chose a new challenge. The work in my thesis compliments the skills I have obtained in DSP and object oriented programming. For this last project I have combined skills obtained throughout both my Bacherlor study in Medialogy, and my Master in Sound and Music Computing. The project involves the implementation of a novel model for producing artificial reverberation based on actual room geometries, in a virtual environment. The work is based on previous works by Francis Steven and Alex Baldwin. The project has been done with close supervision by Cumhur Erkut.

Aalborg University, May 31, 2018

[illegible]

Jonas Holfelt
<jholfe13@student.aau.dk>

Chapter 1

Introduction

In research and development of Virtual Reality (VR) and Augmented Reality (AR) the audio modality was somewhat underemphasized in the early phases of development, partly due to the original focus on the visual modality [1]. However, a significant amount investigation in immersive sound for VR has been done. In the recent years, the importance of audio as an immersive modality has been researched in a wide range of studies in VR, AR and virtual environments in general [1]. Spatial sound in AR is found to be an important cue to efficiently locate objects in the environment [2]. Implementing room acoustical modeling to incorporate more authentic acoustical properties into visual models contributed to create a more realistic rendering of a virtual environment [3]. Earlier research also presents solutions for rendering spatialized audio by incorporating convincing acoustic room simulation to be used in immersive virtual environments [4]. A recent journal highlights the importance of efficient sound simulations as essential elements of VR experiences [5], and breaks the sound design pipeline into three major elements: *source modeling, room acoustics modeling* and *listener modeling*. In this work the focus will be on the room acoustics modeling, also referred to as the sound propagation. When sound propagation of of an acoustic space is modeled and applied to a signal it will be referred to as *artificial reverberation*.

A complete review of artificial reverberation methods through several decades are provided in "*More Than 50 Years of Artificial Reverberation*" [6]. The algorithms presented here has been used for musical purposes since the 1970s, and in more recent decades they have found applications in fields such as games, simulations, and very recently VR and AR experiences. The methods are identified as three main classes of reverberation: *delay networks*, *physical room models* and *convolution-based algorithms* [6]. Until recently most of the applications of artificial reverberation has mostly striven for indoor acoustic spaces such as rooms, halls, acoustic bodies and similar configurations [7]. A simple method for an accurate representation of an acoustic space is by capturing the *room impulse response* (RIR) and perform convolution with a dry input signal [8], however this method acks flexibility. For the specific purpose of indoor modeling, there exists numerous specialized modeling techniques, such as ray-tracing [9], image-source [10], [11], digital waveguide networks (DWNs) [12], [13], feedback delay networks (FDNs) [14], [15], and finite-difference time-domain [16],[17]. With most of these methods there is a troublesome trade off between high-fidelity and efficiency, making real-time applications difficult to realize. Recently an algorithm named the Scattering Delay Network (SDN) was proposed to accurately and efficiently model first-order reflections, and approximate higher-order reflections, for given room geometries [18]. This reverberator represents an extension of the digital waveguide methods.

Efficient simulation of outdoor environments is a very limited field of research. Research has been done to model specific outdoor areas, such as urban environments [19], cliffs in an alpine valley [20] and forest structures [21]. Recent work has shown interest in modeling sparsely reflecting outdoor scenes and proposed a solution for simulating reverberation outdoors utilizing a so-called digital Waveguide Web [22]. The design of the Waveguide Web (WGW) algorithm is based on a set of digital waveguides connected by scattering junctions representing the reflection points of the given space. This extends the previously proposed SDN, and also includes the modeling of acoustics formed by trees in a forest, based on another waveguide method named *Treeverb* [21]. This allows WGW to be flexible for modeling both tree structures and structures with plane surfaces such as rooms, or outside courtyards [22], since the scattering junctions can represent multiple types of reflecting surfaces. The Waveguide Web has yet to be implemented in a real-time context. The model is highly flexbile, but computational costs increase exponentially with added scattering junctions. Computational benchmarks derived from evaluating different case studies of the WGW, indicates that even the simplest cases using just 5 scattering junctions would prove too computational to be realized in real-time.

This work will propose a solution for running WGW in a real-time application. The application will be based on previous work implementing SDN in an AR application [23]. The application *scatAR* is implemented on an AR device and allows the user to scan an environment from which a 3D mesh is generated. First-order reflections points are calculated using the image-source method. Source, reflection points and listener are connected by delay waveguides according to SDN [18] and reverberation is calculated. Both the listener and sound source can be moved around in the space, while reflections are updated in real time. The application was designed for indoor use, and preferred a standard room configuration of 6 surfaces. Through a perceptual test on object-presence a marginal difference was found between the ratings of SDN processed sound and unprocessed anechoic sound was found, with just a marginal preference towards SDN [24]. These results highlight a need for further evaluation of SDN in similar conditions, as well as the need for a more perceptually accurate model. This work poses a solution for implementing WGW in the *scatAR* framework to evaluate the two methods together. The implementation of WGW also opens up for the possibility of supporting the simulation of outdoor areas in the application, but current AR technology might not support it yet.

This paper is organized as follows. Section 2 gives a brief overview of important fundamental reverberation methods, followed by an overview of the SDN and WGW methods, which form the theory needed for the implementation. Section 3 describes the implementation of WGW in *scatAR*. Section 4 presents the evaluation through three case studies: a comparison of two impulse responses generated with SDN and WGW respectively , an objective perceptual evaluation of audio quality using PEAQ, and a perceptual evaluation in a virtual environment. Section 5 discusses the implementation and the gathered results. Section 6 concludes the paper.

Chapter 2

Background

2.1 Fundamental Artificial Reverberation Methods

This section will present fundamental methods for artificial reverberation from the categories *delay networks, physical room models* and *convolution-based algorithms*. The former represents the foundation for the implementation of both SDN and WGW, while the latter is a less flexible alternative, which will also be utilized in the proposed solution in this work.

2.1.1 Feedback Delay Network

The most common recursive linear time-invariant reverberator is known as the Feedback Delay Network (FDN). An FDN is build by a number of delay lines in a feedback loop utilizing matrices for attenuation and filtering. Ideally the loop should be lossless, thus energy-preserving [6]. Absorption filters can be designed to obtain a desired reverberation time in various frequency bands [18]. The FDN offers high quality reverberation at an efficient cost, and with the right delay-lines and filter matrices precise models can be made. Techniques for configuring the elements of an FDN have been widely studied [15].

2.1.2 Digital Waveguide Network

A digital waveguide can be viewed as a bidirectional delay line representing a discrete-time counterpart of an electric transmission line, where traveling waves are propagating [6]. *Digital Waveguide Networks* (DWNs), are an example of a *delay network*, consisting of a closed network of digital waveguides interconnected by so-called *scattering junctions*. These junctions are lossless, meaning that the signal power is conserved [6]. The network topology and waveguide impedances can be determined from geometrical analysis of a given environment to be simulated based on a path-tracing algorithm [13].

2.1.3 Digital Waveguide Mesh

By arranging scattering junctions in a grid and connected to adjacent junctions by digital waveguides, a *Digital Waveguide Mesh* (DWM) is formed. DWMs can simulate multi-dimensional wave propagation to model resonant structures such as acoustical enclosures or membranes [13], thereby it falls within the family of *physical room models*. The scattering junctions of a DWM has K-ports. For a 3D mesh structure, each junction would need 6 connections [6]. For accurate modeling, a fine resolution of junctions is required, which introduces very high complexity and computational requirements. These drawbacks has been studied and less computational methods has been proposed, namely *Reduced Digital Waveguide Mesh* [13].

2.1.4 The Image Method

The image method is another example of a *physical room model*. With the image method, physical boundaries are replaced by an infinite lattice of image sources [10]. This approach is equivalent of solving the wave equation, given a rectangular room where the walls are perfectly rigid [10]. Solving the entire impulse response for a given room is rather computational with this approach, however, it is often applied in virtual acoustics systems to calculate the delays and directions of early reflections [6]. The higher-order reflections can then be modeled utilizing other reverberation structures.

2.1.5 Convolutional Reverb

Convolutional reverb is a quite straight forward method, performed by convolving an obtained *room impulse response* (RIR), or impulse response, with a dry input signal. The RIR can be obtained either by a recording in a physical acoustic space, or by synthesis. The result is a very accurate reverberation model, though it is not flexible in any way [6]. Another drawback of the convolution operation is that is very computationally heavy. *Fast convolution techniques* have been proposed, where one solution is to multiply signals after FFT has been performed [6], thus simplifying the computation a lot.

2.2 Scattering Delay Network

The scattering delay network is an digital reverberator extending the traditional digital waveguide network [25] , [18]. A scattering delay network uses a small digital waveguide network to provide a computationally efficient approximation to geometric ray tracing [11]. The method is conceptually similar to feedback delay networks (FDN) [15], which is built by a number of delay lines build in cascade with filters whose outputs enters a mixing matrix unitary scattering matrix [13].

The scattering delay network involves discrete nodes representing the reflection points of a given environment. These nodes are connected by a set of waveguides and represent the first-order reflection points of a given space. The nodes are also referred to as scattering junctions and are interconnected via bidirectional waveguides. Figure 2.1 shows the how the elements are connected for a 2D room of four surfaces, but the same concept applies to 3D rooms.

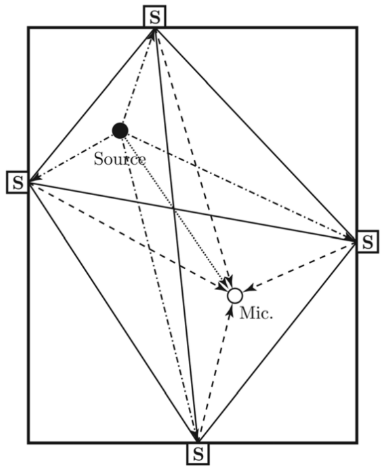


Figure 2.1: The source is connected to the microphone and the scattering nodes with unidirectional delaylines. The scattering nodes are all connected to the microphone, also with unidirectional delaylines. Finally the solid black lines denote the bidirectional connection between each node. The figure is retrieved from [18]

As the scattering junctions are positioned at the points, where the first-order reflections originate, the model can accurately simulate the first-order reflections. The higher order reflections and reverberation tail is progressively approximated but still provides a rich reverberant tail [18]. Figure 2.2 shows how already at the second order reflections the delay lines are wrong, since the system is limited to the N amount of nodes.

2.2.1 Network Structure

This section will provide an overview and the essential elements of the model. Figure 2.3 shows a block diagram providing an overview of the structure of the SDN reverberator, based on [18], where a detailed description of each stage can be found. The input $x(n)$ is fed through directivity matrix γ_s, input delay matrix $\mathbf{D}_s(z)$ and input attenuation matrix $\mathbf{G}_s$. The output from the attenuation matrix is then transmitted to the wall nodes, where the scattering matrix $\bar{\mathbf{S}}$ is applied to scatter the incoming signal between the wall-nodes frequency dependent absorption applied by $\mathbf{H}(z)$.

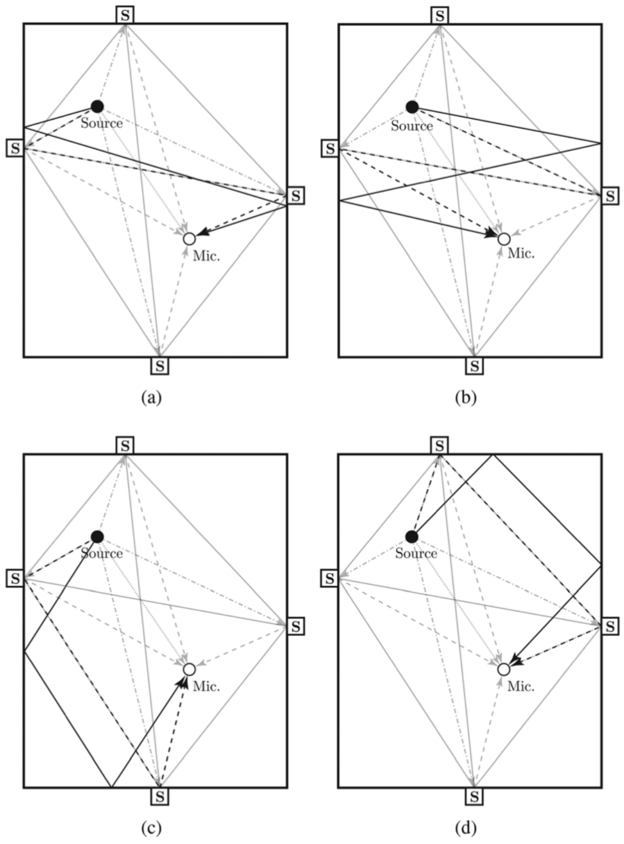


Figure 2.2: Second order reflections modeled with SDN (the dashed lines) versus how they should actually be (the solid lines). Precision will continue to decrease with higher order reflections. The figure is retrieved from [18]

The higher order reflections are modeled through a feedback loop by applying the inter-node delay matrix $\mathbf{D}_f(z)$ in series with the permutation matrix $\mathbf{P}$, and thus calculating the approximated higher order reflectance of the system. The output attenuation matrix $\mathbf{G}_r$ and output delay matrix $\mathbf{D}_r(z)$ is applied to produce the output $y(n)$. It should be noted that all the implementations and simulations in this paper will be modeled as omni-directional so the matrices γ_s and γ_r, which represent the directivity patterns between the SDN nodes, will not be considered any further. Finally, the direct path form source to listener is treated explicitly by applying delay $z^{-D_{sr}}$ and attenuation factor g_{sr} to the split input signal and then added to the output.

2.2.2 Scattering Matrix

The scattering matrix $\bar{\mathbf{S}}$ represents the scattering of the SDN system as a whole. It is a $N(N-1)N \times (N-1)$ matrix comprised of identical smaller scattering matrices

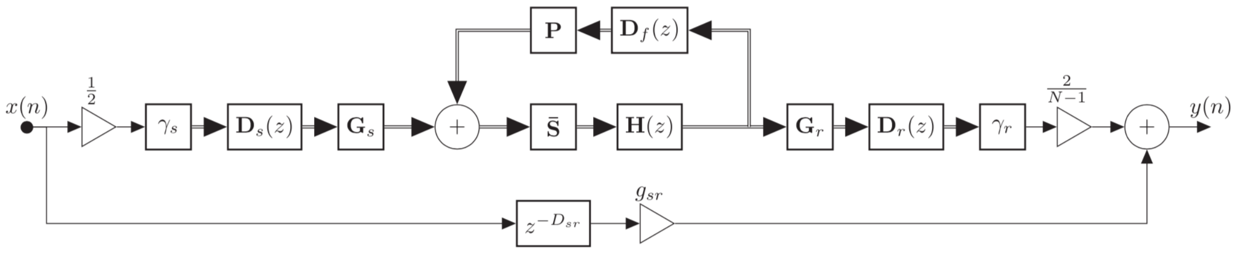


Figure 2.3: Overview of the Scattering Delay Network block diagram. The direct sound is excluded from the network. First-order reflections are calculated through matrices and fed through a feedback network for approximated higher order reflections. Figure retrieved from [22]

S, which represent the scattering at the individual wall-nodes. **S** is defined as:

$$\mathbf{S} = \frac{2}{N-1}\mathbf{1}(N-1)(N-1) - \mathbf{I} \tag{2.1}$$

Here, N is the the number of nodes in total, **1** is a matrix of ones and **I** is an identity matrix. **S** is used to calculate the outgoing wave pressure signal p_{ij}^- from wall-node i to wall-node j by applying **S** to the incoming wave pressure signal p_{ij}^+:

$$p_{ij}^- = \mathbf{S}p_{ij}^+ \tag{2.2}$$

To characterize the whole system this is extended to $\bar{\mathbf{S}}$:

$$\bar{\mathbf{S}} = diag(\underbrace{\mathbf{S}\cdots\mathbf{S}}_{N}) \tag{2.3}$$

Thus $\bar{\mathbf{S}}$ becomes the $N(N-1)N \times (N-1)$ matrix that determine the energy scattered among the bidirectional delay lines between the wall-nodes. The output of the scattering matrix is filtered by $\mathbf{H}(z)$ and then fed back through the feedback loop. Inside the feedback loop, it is required that the result of the scattering matrix is re-ordered, since after the scattering, the outgoing pressure value p_{ij}^- will be equal to the incoming pressure value p_{ij}^+ at the next scattering instance. The permutation matrix **P** is applied to perform this re-ordering, assuring that the input for the next scattering operation is correct. Finally the input is scaled by $\frac{1}{2}$ to provide the intended energy pressure at each node [18]. This energy attenuation is then compensated for the output by scaling it with $\frac{2}{N-1}$. The denominator value of $N-1$ compensates for the amount of nodes in the system, since the energy of the system will scale up according to the number of wall-nodes .

To conclude, the SDN can be deemed an efficient and effective reverberator for acoustic room simulation, providing accurate first-order reflections and rich higher order reflections [18]. For more complicated systems involving objects such as rigid cylinders rather than just planes, it would require the incorporation of directionally dependent filtering as proposed by [22]. In addition it would also be desirable to acquire precise characterization of the attenuation for the second order reflections to possibly improve the perceptual accuracy of a room model, or even a more complicated outdoor scene environment. The importance of precise first-order and second-order reflections in the modelling of outdoor acoustic spaces is indicated in a previous study [26]. The Waveguide Web, which was introduced earlier offers this type of modelling and will be described in the next section.

2.3 The Waveguide Web

This section will present the Waveguide Web (WGW) as proposed by [22] focusing on the concept and theory behind it, rather than the implementation of it. WGW is in its essence very similar to SDN, but it offers some additional accuracy and possibilities. Like SDN the modeled space consists of wall-nodes connected to each other with bidirectional delay lines. The source and receiver are also connected to the wall-nodes by unidirectional delay lines. WGW provides the same approximated accuracy of delay lines for the higher order reflections (including second-order reflections), in that no additional nodes are introduced to the model. The main difference between the two algorithms is the scattering action at each node [22]. SDN only allows for one filtering operation at each node. WGW allows for directionally dependent filtering to be implemented, which means that each node can provide different filtering according to where the incoming pressure wave signal is coming from. This can be done to second-order reflection precision. The nodes can be positioned at any 3D position depending on the required space being modeled, just like SDN.

2.3.1 Node Structure

Figure 2.4 depicts all the connections and operations associated with a single node j in an N-node network. There are three types of connections:

1. Source-to-node connection
2. Inter-node connections
3. Node-to-receiver connection

K represents a vector of $N-1$ elements holding the indices of the nodes in the system apart from node j. If $N=6$ and $j=3$, then $K=\begin{bmatrix}1 & 2 & 4 & 5 & 6\end{bmatrix}$.

As the diagram in figure 2.4 shows, each node contains N^2 filters. One filter for the first-order reflection between the source, node and receiver. Note that the first order reflection is treated explicitly and is not part of the feedback loop. Additionally the node contains $N-1$ filters for the signal incoming from the source and outgoing to all other nodes, $N-1$ filters corresponding to incoming signals from all other nodes and outgoing to the receiver, and finally $(N-1)(N-1)$ filters for the recursive signals incoming from all other nodes, and outgoing to all other nodes [22].

H_{ijk} is a filter at node j processing a signal from node i, going to node k. S and R are used in place of i and k respectively, in the case of source or receiver connections.

There are three steps for performing attenuation in the structure, as seen in figure 2.4:

1. g_{Sj} is the attenuation multiplier between the source-node and wall-node j
2. g_{jR} is the attenuation multiplier between wall-node j and the receiver node.
3. $g_{K_1jR} \cdots g_{K_{N-1}jR}$ is the attenuation multiplier between each node and the receiver, where the incoming signal is from all the other nodes.

The attenuation values for these attenuation multipliers are calculated according to $\frac{1}{r}$, where r is the distance traveled. The following equations are given to calculate the various attenuation values:

$$g_{ST_M} = \frac{1}{\|x_S - x_{T_M}\|} \tag{2.4}$$

$$g_{T_MR} = \frac{1}{1 + \frac{\|x_{T_M} - x_R\|}{\|x_S - x_{T_M}\|}} \tag{2.5}$$

$$g_{T_MT_NT_R} = \frac{1}{1 + \frac{\|x_{T_N} - x_{T_M}\| + \|x_R - x_{T_N}\|}{\|x_S - x_{T_M}\|}} \tag{2.6}$$

Here, T_M and T_N represent different wall-nodes in the system. Equation 2.4 and 2.5 are formulated for the SDN [25], while equation 2.6 is introduced for attenuating the second-order for WGW specifically [22]. Reflections of higher order do not follow the $\frac{1}{r}$ law.

Finally, the scattering operation for node j is also shown in figure 2.4. $\mathbf{S}_{ij}$ is the element in row i and column j of the matrix $\mathbf{S}$ and is applied to the incoming signal from each node. Exclusively for WGW, $N-1$ copies of each incoming signal is made to allow for directionally dependent filtering before the scattering operation is applied.

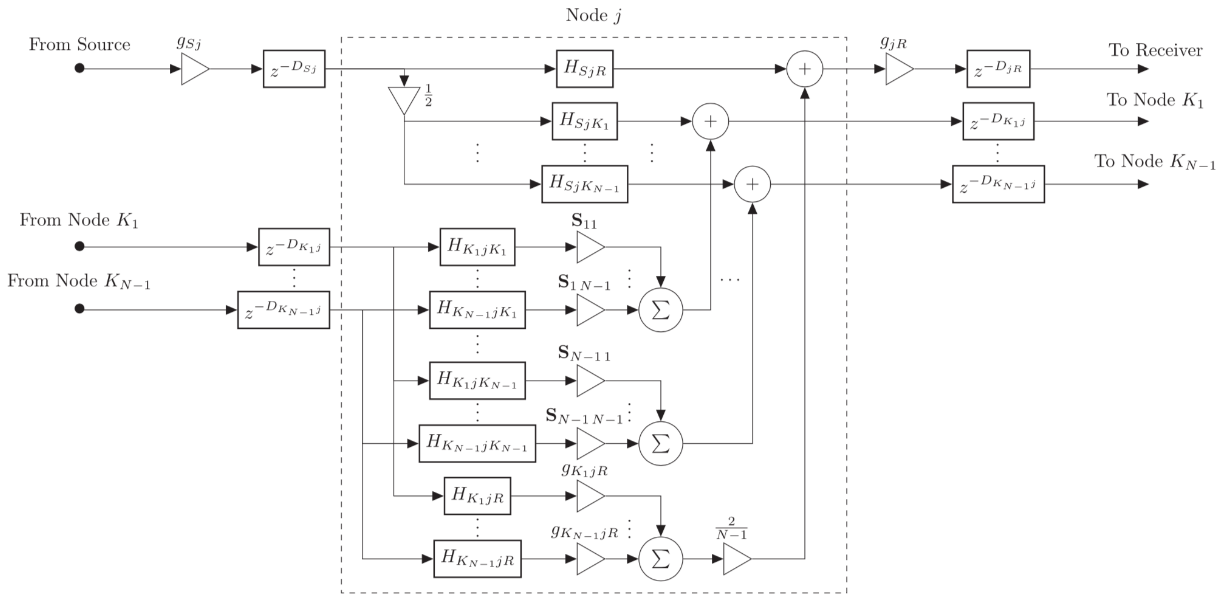


Figure 2.4: Structure of all the connections to a single WGW node. The connections are grouped as source-to-node, inter-node and node-to-receiver connections. The scattering operations and directionally dependent filtering are also depicted here. Figure retrieved from [22]

2.3.2 Network Structure

This section will provide an overview the overall structure of the WGW system according to [22]. Figure 2.5 shows the block diagram of the WGW reverberator. As mentioned earlier, the first-order reflections are calculated separatly along the direct delay path. The main section of the WGW models second-order reflections, with third-order and higher-order reflections are approximated using the feedback loop. The WGW thus offers an additional layer compared to SDN.

2.3.2.1 Feedforward Path

Firstly the first-order feedforward path consists of the source and receiver delay matrices:

$$\mathbf{D}_S(z) = diag(z^{-D_{S1}}, z^{-D_{S2}}, \cdots, z^{-D_{SN}}) \tag{2.7}$$

$$\mathbf{D}_R(z) = diag(z^{-D_{1R}}, z^{-D_{2R}}, \cdots, z^{-D_{NR}}) \tag{2.8}$$

providing the first order delay times.

$$\mathbf{H}_{STR}(z) = diag(H_{S1R}(z), H_{S2R}(z), \cdots, H_{SNR}(z)) \tag{2.9}$$

represents the $N \times N$ matrix containing the filters for the first order reflections.

$$\mathbf{G}_S = diag(g_{S1}, g_{S2}, \cdots, g_{SN}) \tag{2.10}$$

$$\mathbf{G}_R = diag(g_{1R}, g_{2R}, \cdots, g_{NR}) \tag{2.11}$$

Are the source and receiver attenuation matrices for the first-order reflections, and $z^{-D_{SR}}$ and g_{SR} are the direct delay and attenuation, also calculated separate from the main section.

Next, consider the second-order reflection feedforward path in figure 2.5.

$$\mathbf{D}_{ST}(z) = diag(\underbrace{z^{-D_{S1}} \cdots z^{-D_{S1}}}_{N-1}, z^{-D_{S2}} \cdots z^{-D_N}) \tag{2.12}$$

$$\mathbf{D}_{TT}(z) = diag(\underbrace{z^{-D_{12}} \cdots z^{-D_{12}}}_{N-1}, z^{-D_{1N}} \cdots z^{-D_{N(N-1)}}) \tag{2.13}$$

$$\mathbf{D}_{TR}(z) = diag(z^{-D_{2R}} \cdots z^{-D_{NR}}, \cdots, z^{-D_{1R}} \cdots z^{-D_{(N-1)R}}) \tag{2.14}$$

are the delay matrices for source-node, inter-noder and node-retriever, and

$$\mathbf{H}_{STT}(z) = diag(H_{S12}(z), \cdots, H_{S2N}(z), \cdots, H_{SN1}(z), \cdots, H_{SN(N-1)}) \tag{2.15}$$

$$\mathbf{H}_{TTR}(z) = diag(H_{12R}(z), \cdots, H_{1NR}(z), \cdots, H_{N1R}(z), \cdots, H_{N(N-1)R}) \tag{2.16}$$

are the source-node-node and node-node-receiver filter matrices. Finally the second-order source-node and node-receiver attenuation matrices are given by:

$$\mathbf{G}_{ST} = diag(\underbrace{g_{S1} \cdots g_{S1}}_{N-1}, g_{S2} \cdots g_{SN}) \tag{2.17}$$

$$\mathbf{G}_{TTR} = diag(g_{12R}, \cdots, g_{1NR}, \cdots, g_{N1R}, \cdots, g_{N(N-1)R}) \tag{2.18}$$

The permutation matrix $\mathbf{P}$ is formulated identically to the permutation matrix in the SDN, according to [18].

2.3.2.2 Feedback Path

The functionality of the feedback path is very similar to that of the SDN reverberator. The main difference is that the input signals are derived from second-order reflections rather than first-order reflections. $N(N-1)$ channels are used in the calculation of the second-order reflections, so $N-1$ copies of each channel must be made to allow directionally dependent filtering at each node. This action is

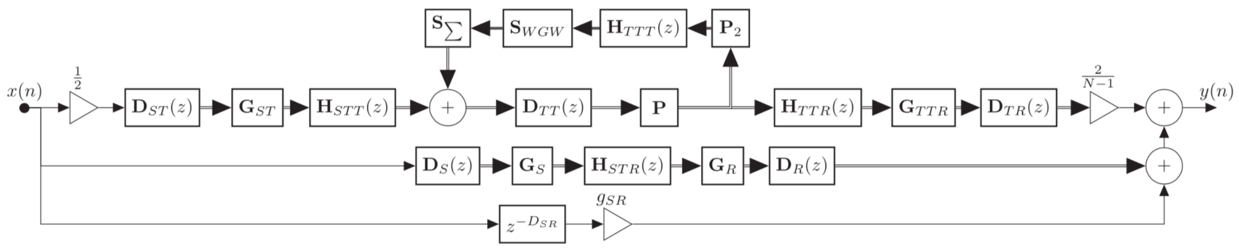


Figure 2.5: The overall structure of the WGW network. Direct delay, and first-order reflections are treated explicitely, while second-order reflections are scattered in the feedback delay loop. Figure retrieved from [22]

performed by the second permutation matrix $\mathbf{P_2}$, creating $N-1$ copies of the output from $\mathbf{P}$. $\mathbf{P_2}$ is an $N(N-1)(N-1) \times N(N-1)$ formed of multiple copies of sub-matrix $\mathbf{P}_{2s}$

$$\mathbf{P}_{2s} = [\underbrace{\mathbf{I}_{N-1} \cdots \mathbf{I}_{N-1}}_{N-1}]^T \tag{2.19}$$

$$\mathbf{P}_2 = \mathbf{I}_N \otimes \mathbf{P}_{2s} \tag{2.20}$$

The output from $\mathbf{P}_2$ is then input to $\mathbf{H}_{TTT}(z)$:

$$\mathbf{H}_{TTT}(z) = diag(H_{212}(z), \cdots, H_{N1N}(z), \cdots, H_{1N1}(z), \cdots, H_{(N-1)N(N-1)}(z)) \tag{2.21}$$

which represents the $N(N-1)(N-1) \times N(N-1)(N-1)$ matrix for the node-node-node filters.

Next, the scattering operation can be applied. $\mathbf{S}_{WGW}$ consists of multiple copies of scattering matrix $\mathbf{S}$:

$$\mathbf{S}_{WGW} = diag(\underbrace{vec(\mathbf{S})^T \cdots vec(\mathbf{S})^T}_{N}) \tag{2.22}$$

and ensures that the correct $\mathbf{S}$ elements are applied to each incoming filter signal. Lastly, before the output signals returns to the feedforward path, they must be recombined to the $N(N-1)$ inter-node wave variables. This is done by summing the output of $\mathbf{S}_{WGW}$ to give the total scattering output at each node, by applying the operator $\mathbf{S}_{\Sigma}$:

$$\mathbf{S}_{\Sigma} = \mathbf{I}_{N(N-1)} \otimes [\underbrace{1 \cdots 1}_{N-1}] \tag{2.23}$$

After this step, the output signals are reinserted in the loop for further propagation.

2.3.2.3 Evaluating the WGW

The implementation of WGW, which is available online [27] was evaluated for some different cases to show the potential, and efficiency.

A comparison between SDN and WGW both modelling the same space of a shoebox room of 9x7x4 meters, were performed to prove that the two algorithms produced identical first-order reflections, and similar reverberation tails, with a noticeable difference due to the different characterization of the higher-order reflections [22].

The potential for modeling sparsely reflecting outdoor scenes was also proved possible by modeling e.g. a forest environment consisting of various numbers of tree in different distributions. The WGW offers this type of modeling, by considering the nodes as rigid cylindrical shapes, representing trees. The structure of the network is identical to modeling a room or any other space, but the filter design and scattering is based on Morse's solution [28] to emulate the acoustic scattering from a rigid cylinder, which was also used to model a forest environment in a previous study [21]. The results are validated by comparing impulse responses and spectrums generated from Morse's solution and the WGW approximations. None of the simulations proposed in this study will not consider rigid cylinders as surfaces, so the this type of filtering will not be considered any further [22].

Another example of an outdoor scene was based on a courtyard. The dimensions of the courtyard being modeled, is based on a real environment whose acoustics were measured in another study [26]. The node positions were determined from a 3D model of this given environment. The impulse response generated from the WGW model was compared with an actual impulse response recorded in the real environment in a T30 plot. By utilizing an all-absorbing sky-node, the two impulse responses was deemed quite similar. This proves the plausibility of accurately modeling open outdoor spaces, however it can not be concluded yet, which node positions would be optimal [22].

Finally an evaluation of the computational requirements was performed. For the simplest case of 5 nodes, the simulation required 4.36 seconds and 5.65 MB of memory to produce 1 second of audio at a sample rate of 48 kHz. For higher numbers of nodes, the computational requirements increased exponentially, where a case of 30 nodes would require 667.23 seconds and 102 404.16 MB of memory. This exponential increase is due to the N^2 at each node, which means an increase of $N(N^2) = N^3$ filters for the entire system [22]. The computational requirements makes it difficult to imagine a real-time solution, but this work will attempt to build a solution implementing WGW producing audio output at real-time.

This section has provided an overview of the WGW structure. Next a review of a current implementation of the digital SDN reverberator in an augmented reality application is presented and the possibility of replacing the SDN network with a WGW network to run real-time in an augmented reality setting is considered.

2.4 ScatAR

2.4.1 Overview of the Design

ScatAR is an Augmented Reality (AR) application that successfully implements the scattering delay network to efficiently generate and organize an artificial reverberator based on room geometry scanned with an AR device [23]. The application was developed for the Google Tango-enabled Lenovo Phab 2 Pro 1, using the now deprecated augmented reality computing platform Tango by Google. The Lenovo device is equipped with a wide-angle camera for optical motion track, an IMU, and an infrared depth sensor for providing geometric information about the surroundings. This allows the device for much more exact measurements of real environments, compared to the usual smart-phones. The sensors can keep track of source positions, user position and of course the 3D environment geometry. Hereby, sufficient amount of data can be extracted with device, to calculate the parameters for the SDN.

Initially the environment is scanned by the user, where after a digital mesh is generated from a point-cloud representation of the room. After scanning the room, an AR object is placed in the environment as seen on figure 2.6. In this example the object was a drone, which acted as the sound source. Based on the position of the user and the position of the source, first-order reflection points are found using the image-source method. New reflections are calculated, when source or user position is updated.

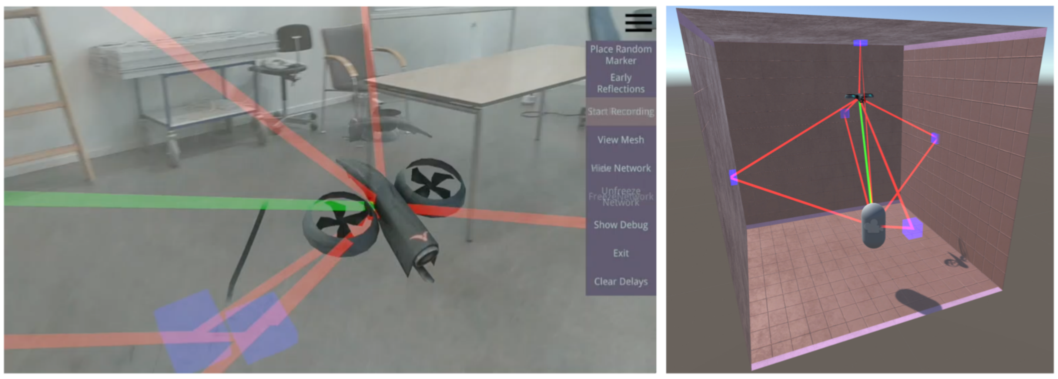


Figure 2.6: Left: The AR object (Drone) placed as the sound source in a real environment. Red rays are the first-order reflections. Green ray is the direct path to the listener. Right: A Unity3D simulation of a room. The blue boxes are the scattering junctions for the first-order reflection points. Figure retrieved from [23]

The rays for the direct source to listener connection, as well as the first-order reflection paths are calculated by using Unity's build in ray-tracing system. This process will be explained in greater detail in the implementation chapter. The paths are saved in a list, and their information is used to create SDN node objects, which store the position of the node in the environment, the reflection path the node is situated upon, a list of connections to other nodes, two delay lines for input and output, an absorptive IIR filter for these delay lines and the scattering matrix for distributing energy to its node connections [23].

2.4.2 Results

ScatAR was evaluated in a real room of a considerable size, focusing on *Spatial Presence* and *Percptual Realism* perspective. The test was performed on two groups. One was experiencing the sounds processed with SDN, while the other group was experiencing unprocessed sounds [24]. The null-hypothesis was that the type of sound processing would not affect the user's rating of object-presence. Detailed results can be seen in [24]. The results indicated only a marginal preference toward the reverb condition. Thereby the results suggest an inconclusive view of the relationship between anechoic, reverb conditions and presence. An important remark from the results is that both cases tended towards the high end of the scale, so the AR visual stimuli alone might have been enough to induce a sense of presence, and the reverberation itself might prove to be an insignificant part of the overall object-presence. Further testing, taking the remarks from this study into account is needed to confirm that high quality artificial reverberation could provide a higher object-presence. Another aspect to consider is of course that the SDN algorithm might not be accurate enough in the case of an AR environment. This highlights the need for an even more accurate model such as the Waveguide Web (WGW), presented earlier. While the WGW is much more computationally demanding, it can provide more accurate second-order reflections, which might be crucial for a realistic experience in AR. By implementing the Waveguide Web into scatAR, a new test can be performed, accounting for both the relationship between anechoic, reverberation and object-presence, as well as a possible difference between SDN and WGW.

2.5 AR Technology

Since Tango has been deprecated, other augmented reality computing platforms must be taken into consideration for future development in this field. As of this date, two main AR platforms can be considered: ARcore and ARkit.

2.5.1 ARCore

ARCore [29] is a platform by Google for building augmented reality experiences. It uses three key components to understand the environment and integrating virtual content in it: *Motion Tracking*, *Light Estimation* and *Environmental Understanding*.

These concepts allow the phone to understand its position in the space, the geometries of the environment (it can detect size and location of all types of surfaces), and approximate lighting settings for a realistic integration of virtual content. Some of the API's for ARCore is available across Android and iOS devices.

2.5.2 ARKit

Like ARCore, ARKit [30] also works with the concepts of scene understanding and light estimation. It can analyze a given scene from camera view to find both horizontal and vertical planes and can track and place objects on smaller feature points as well. By estimating the light in the given scene it can apply a similar light source to the virtual content implemented in the scene. ARKit can also exploit the data derived from the TrueDepth camera on iPhone X, however this is mostly relevant for facial recognition.

A recent article compared the capabilities of ARCore and ARKit [31]. ARKit was found to have some advantages regarding hardware/software integration and more reliable tracking, while ARCore has some advantages regarding more reliable recovery and mapping [31]. However, the technical differences is not considered to be significant for the user's perspective, but mostly for the developer. Both can provide good consumer experiences, so it all boils down to which type of devices and platforms the developer wants to develop for. In the case of this project ARCore would be most suitable, since access to a project created for Tango is available. ARCore builds on the research knowledge gained from Project Tango, and would provide a strong base for a future build providing the intended functionality.

2.5.3 Tango

Various options for implementing augmented reality applications exists and provides great potential for future AR builds for an application like scatAR. Tango, which was presented in the previous section has been deprecated, but the Tango SDK for Unity3D is still accessible, and posses the Tango-enabled Lenovo device. As a first iteration and proof-of-concept this platform will be suitable, since a project set up to scan a room and generate a 3D mesh from the geometries, is readily available. The initial goal will be to implement the Waveguide Web to instantiate the scattering delay network currently implemented in scatAR. The project is publicly available on Github [32].

Chapter 3

Implementation

This section will document how WGW was implemented into the open-source Unity3D project *scatAR* [32]. The proposed project scatAR_WGW is also available as an open-source project on Github [33]. This section will focus on explaining the behaviour of the new scripts and the added functionality. The goal of this project is to implement a solution for having WGW running real-time in an augmented reality application. As a proof-of-concept, the goal is implement it using the now deprecated Tango API, and device. Future development would consider the new tools ARKit or ARCore presented earlier, for the final application.

3.1 Game Setup

ScatAR contains two projects, both facilitating the SDN class structure. One is for building the augmented reality application on android, while the other is a prototype scene where the features can be tested in a virtual room in the Unity editor. For prototyping of WGW, the virtual room project served as the base of the implementation. The scene contains a number of essential game objects, as seen in figure 3.1, which will be referred to throughout this section:

1. The sound source, to which the reverberator scripts, reflection finder scripts and Audio Source component will be attached. In figure 3.1 the sound source is depicted by the sound icon. In scatAR it was a flying drone as seen in figure 2.6.

2. The listener, an object representing the user's point of view. This is given a rigid body and a mesh collider so it can be hit by rays. An Audio Listener component is attached, and the Main Camera is snapped to the position and orientation of the listener object

3. The boundary object, sets the colliders for the surfaces that are to reflect the emitted rays on to the listener. In figure 3.1 the boundaries are simply the

walls of a room. Figure 3.2 shows how the system instantiates a boundary object (the pink cube) to account for holes in a generated mesh from a room scan.

4. Scattering Junctions are the blue boxes shown at the points of first-order reflections. The scattering junctions are instantiated, when the first-order reflection points have been calculated.

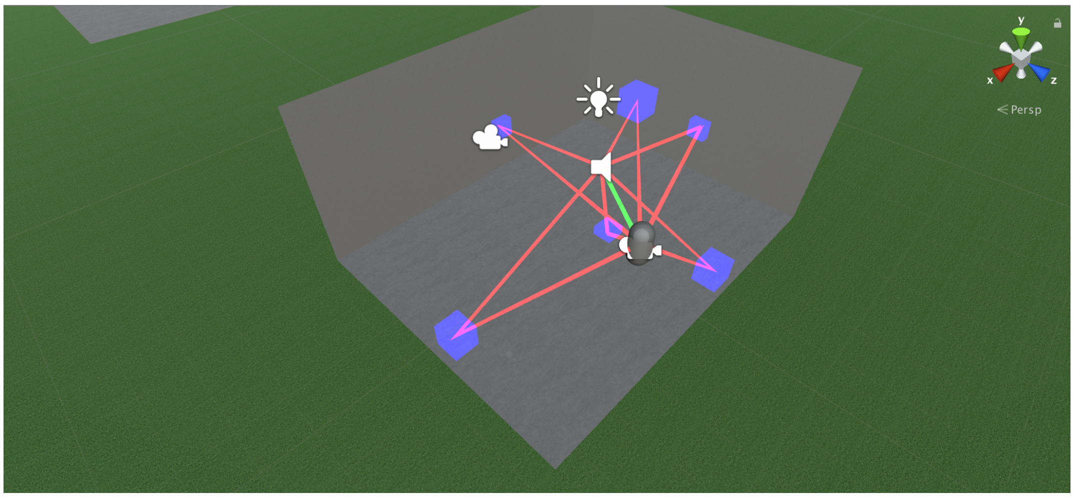


Figure 3.1: The reflection finder in action. An empty game object emits rays, and using the image method, finds the first-order reflection paths to the listener object (the capsule object). Here the boundary is the walls of a simple shoebox room.

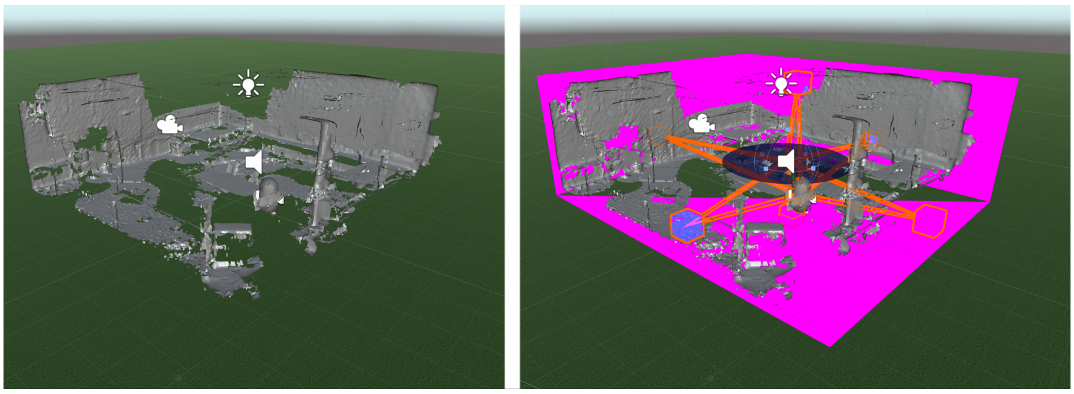


Figure 3.2: How the system compensate for finding reflections in an incomplete mesh environment. The pink cube, is a boundary object created to encapsule the mesh with a collider material. This mesh is generated from a room scanned with the Tango Device.

3.2 Implementation of WGW

The WGW structure is implemented as a new class. First, the essential functions that are shared between the SDN class and WGW class will be described, where after the main differences will be covered.

Figure 3.3 shows a flowchart of the signal chain when the program is executed. There are three main threads running in parallel.

The *Unity Game Thread* is running at frame rate. It is always checking for a new position, and when one is found, the reflection path finding is updated using the image method. The information gathered from the reflection paths are sent to the WGW objects to update the positions of the scattering junctions, and update the network parameters they contain.

The *Audio Streaming Thread* retrieves the input audio buffer to be processed. The size of the input buffer is set in the Unity Audio Settings. The input is sent sample by sample to the *Audio Processing Thread*. This is done by putting the samples in a queue acting as a FIFO Input array. When the samples have been processed they are received in the Audio Streaming Thread again to be dequeued for the output signal.

The *Audio Processing Thread* gets input samples from the Audio Streaming Thread. This input is sent to all the node instances and starts propagating the WGW network. Every time a node process a sample the output is sent back to the Audio Streaming Thread to be outputted to the listener. The exact operation occurring at the nodes will be explained in greater detail.

3.2.1 Reflection Path Finding

The reflection path (the red lines depicted in figures 3.1 and 3.2) is a class which holds: origin position (the audio source position), destination position (the listener position), a *ray* list for every ray segment (for first-order reflections there would be two segments: one from source to wall-node, and one from wall-node to receiver), and a float list containing the lengths of these segments. When instantiated, the path can simply be thought of as a line (with several segments) from one point to another. The line can be drawn, for visualization.

The reflection finder class creates a list of reflection paths. The class is instantiated as a component on the source object. The direct path between the source and the listener is simple to find. Utilizing the built-in ray tracing system in Unity, a ray is fired from the position of the source, in the direction of the listener. When the ray hit's an object with the tag "listener", the ray is deemed valid and the path is saved as a reflection path instance. The component proceeds to attempt to find a pre-specified number of reflections, which is generally 6, for the number of first-order reflections in a rectangular room geometry. To find the reflections for each surface of the room model the spherical Fibonacci point set algorithm [34]

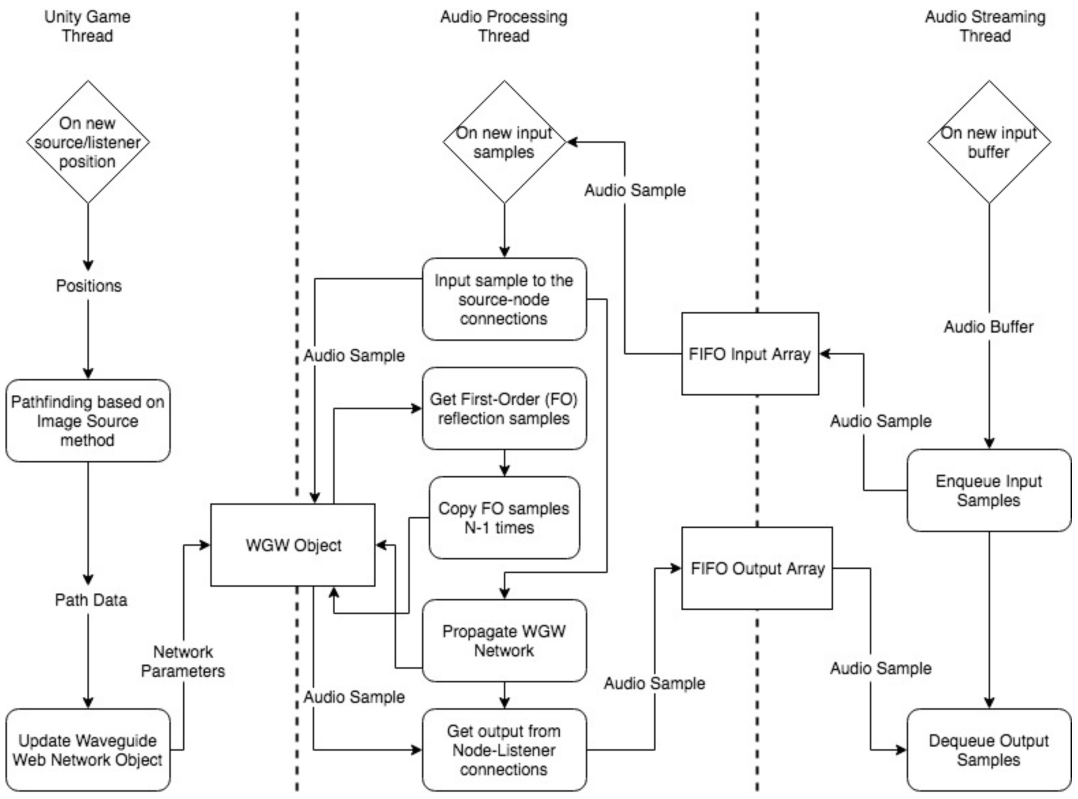


Figure 3.3: This flowchart shows a simplified overview of the code in the Unity project. This is specifically using the WGW.cs script. This is close to the original SDN implementation in scatAR. The difference lies in the scattering operation, where multiple copies of the signals are created to represent the second order reflections.

is utilized. The main WGW component receives the list of paths and instantiates the WGW node objects at the positions of the first-order reflections points. These nodes are depicted by the blue boxes in figure 3.1.

3.2.2 The WGW Class

The main WGW class is responsible for instantiating the WGW nodes upon receiving the list of reflection paths from the reflection finder component.

The WGW node objects are instances of the WGWnode class and holds: the nodes position in the environment, a list of connections to the other nodes, two delay lines for input and output, N^2 absorbtive IIR filters, the reflection path which the node is placed upon, and finally the scattering values which are applied through a for-loop. The WGWnode class is based purely on the node structure proposed by [22] as seen on figure 2.4 in section 2.

In addition, the WGW class holds a class definition for delay lines, a function for propagating through the network, a class definition for the WGW connections

(connections between all the other nodes), a function for performing the scattering operation and the initializing of the audio streaming thread.

When a number of WGW nodes have been instantiated, they are all connected with bidirectional delay lines.

3.2.2.1 Filters and Coefficients for Second-order Reflections

From section 2.3, it is know that the WGW should allow for correct characterization of second-order reflections and directionally dependent filtering. Let us first consider the attenuation multipliers. The input and output attenuation values are simply calculated by the $\frac{1}{r}$ law by using the two segments lengths from the reflection path that the node is placed upon. The first segment length is the length from source to node, and the second segment length is the length from node to listener. To satisfy the correct second-order attenuation the length from the connected nodes must be obtained, and each node should contain $N-1$ different attenuation factors depending on the node sending the second-order sample. A list called *interNodeAttFactor* is therefore filled with $N-1$ values to be used for the attenuation. Figure 3.4 shows how the distances needed to calculate the attenuation factors are derived to be input in this list. The attenuation factors are then calculated according to equation 2.6.

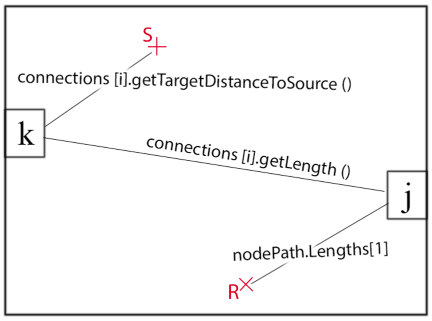

S
connections [i].getTargetDistanceToSource ()
k
connections [i].getLength ()
j
nodePath.Lengths[1]
R

Figure 3.4: An example of a second order reflection from node *k* to node *j*, where S is the source and R is the listener. Consider the current node being *j*. The distances for calculating the second-order attenuation factors are retrieved for node *j* here. NodePath.Lengths[1] is the distance from the node to the receiver. The structure loops through all the connections to get the distances between the other nodes and node *j*, and the distance from all the other nodes (in this case node *k*) and the source.

Next, consider the filters for the directionally dependent filtering. As stated in section 2.3, there needs to be N^2 filters for each node, resulting in N^3 filters for the entire system. The first filter is declared as a *wallFilter* and applied to the first-order sample. Next, a $N-1$ list *secondInFilters* of filters is declared and applied to the samples incoming from the source and outgoing to the other nodes. An identical $N-1$ list *secondOutFilters* is declared and applied to the incoming signals from all other nodes. Finally a (N-1)(N-1) list *interFilters* is declared and applied to the samples incoming from all nodes and outgoing to all other nodes, thus concluding N^2 filters for each node.

3.2.3 Audio Processing and Streaming

The audio streaming thread receives an input buffer, which is sent sample by sample, to the audio processing thread as seen in figure 3.3. When the samples have been processed they are sent back to the audio streaming thread for the output signal. In the audio processing thread, the samples are propagated through the network, where scattering operations and filtering is applied. The direct delay line from source to listener, and the first-order delay line from source to node to listener, are treated explicitly. Only the second-order reflections are being fed through the scattering feedback loop.

3.2.4 Performance

As expected, the WGW was not able to produce output in real-time in the Unity application. The algorithm was verified by producing recorded material, by letting the WGW run for a while. For 6 nodes the system was stressed to spend averagely 100ms each sample, which is far from real-time. Therefore, to be able to make a real-time simulation based on the WGW, a workaround had to be done. Since the system was build in C#, which is not optimal for audio processing, a solution could be to try to develop WGW as a Native Audio Code plugin for Unity. This, however would require an entire restructuring of the code from C# to C++, and introduce the learning curve of implementing Native Audio Plugins in Unity. Therefore another solution was proposed, involving real-time convolution.

3.3 Running WGW in real-time

This section proposes a solution for generating an audio output based on the WGW reverberator, by generating an impulse response with WGW and performing convolution-based reverberation with the audio input and the generated impulse response. This will of course introduce some compromises regarding the update time for updated reflections, but it was deemed the only feasible solution. Figure 3.5 shows how the structure was altered. This flowchart is based on a new script *generateWGWIR.cs* script, which is made specifically for generating an impulse response rather than performing the algorithm directly to the incoming audio input. The important thing to note here is that the input for the WGW network is no longer based on the input sample from the Audio Streaming Thread. Now the WGW network simply receives an impulse signal, upon receiving new reflection paths, and then the reverberation is processed through a convolution-based reverberator.

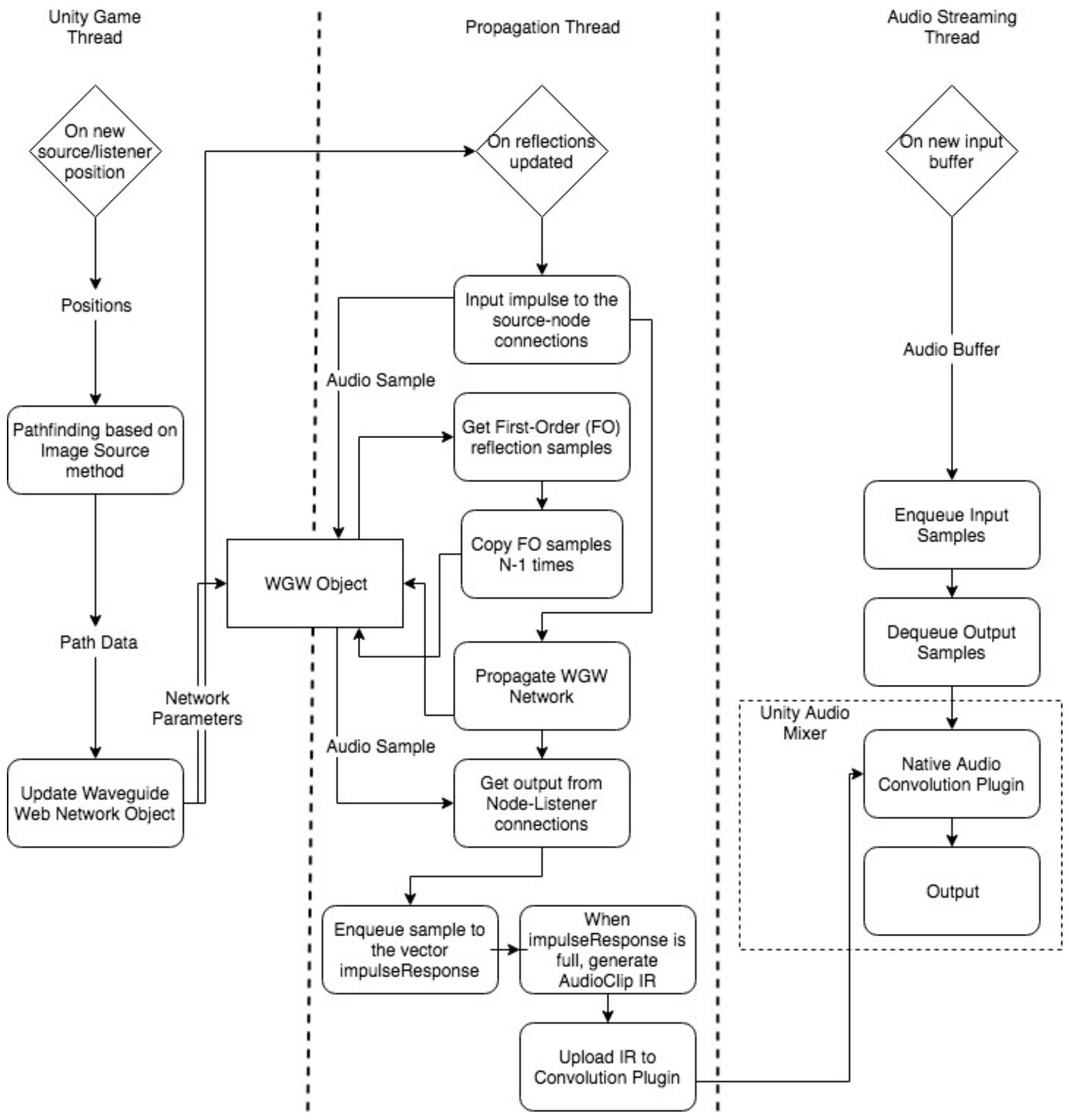

Unity Game Thread
Propagation Thread
Audio Streaming Thread
On new source/listener position
On reflections updated
On new input buffer
Positions
Input impulse to the source-node connections
Audio Buffer
Audio Sample
Pathfinding based on Image Source method
Get First-Order (FO) reflection samples
Copy FO samples N-1 times
Enqueue Input Samples
WGW Object
Path Data
Dequeue Output Samples
Network Parameters
Propagate WGW Network
Unity Audio Mixer
Audio Sample
Native Audio Convolution Plugin
Update Waveguide Web Network Object
Get output from Node-Listener connections
Output
Enqueue sample to the vector impulseResponse
When impulseResponse is full, generate AudioClip IR
Upload IR to Convolution Plugin

Figure 3.5: This flowchart shows a simplified overview of the code in the Unity project. This is specifically using the generateWGWIR.cs script. Here the WGW network and audio streaming input/output is not directly connected. The propagation thread generates a new impulse response of 1 second length, every time the position is updated. This impulse response is uploaded as an AudioClip to a convolution plugin attached to the audio mixer. This is a required workaround to make WGW produce audio in real-time.

3.3.1 Native Audio Convolution Plug-in

To perform the convolution a native plugin was created with Unity's Native Audio Plugin SDK [35]. The convolution plugin is based on an example from the Audio-PluginDemo in the SDK. The demo allows the user to choose a number of audio files to upload before running the application. In run-time the user can then switch between the uploaded audio files to be convolved with an input signal. However, for this case a generated audio file needs to be uploaded in run-time. Some minor adjustments had to be done to the main convolution C++ file, and then the script for uploading the impulse responses had to be rewritten.

3.3.2 Generating and Uploading the Impulse Response

The script *ConvolutionReverbUploadIR* was altered to expect an audio clip, which would be uploaded to the convolution plugin, every time a new impulse response is successfully generated. The audio clip is retrieved using a function *getIR()* in the main generateWGWIR component.

The input for the WGW network in the generateWGWIR component is simply an impulse signal. Upon receiving an updated reflection path, the network will initiate propagation and fill up a buffer with 44100 samples. When the buffer is full an AudioClip *IR* is generated using the buffer as audio data. The Audio Clip is received in the upload script and uploaded to the convolution reverb plugin. The process of generating an impulse response can take up to 4-5 seconds, which fits the results from [22], where the computational benchmark found the WGW to generate 1 second of audio in 4.35 seconds. This means that upon finding a new reflection path (when either source position or listener position is updated), it will take 4-5 seconds before the audio output from the convolution reverb will actually be based on the current position. This is of course not optimal, but it will serve as a temporary solution.

This concludes a compromised solution for running WGW in real-time in a desktop application. The next section will document the attempt to build the application for the Android device, and utilizing the augmented reality framedwork Tango.

3.4 Building for Android

The initial goal was to build the project with the WGW implementation on the Tango enabled Lenovo device. The compromised solution for running WGW in a real-time solution, required the use of a Native Audio Plugin in Unity. Building such a plugin for a desktop application was straight forward, by using a C++ framework, such as Xcode or Visual Studio. However, building a native plugin for Android proved to be troublesome. It was necessary to use the Android NDK to build the plugins, which introduced a steep learning curve to a new toolkit. Some example plugins were build, but the integration of them in the Unity build, resulted in a crashing application. Due to time constraints, building for Android therefore proved to be unsuccessful. Instead, the application will be evaluated as a desktop application, where the reverberation will be based on a virtual room modeled in Unity.

Chapter 4

Evaluation

This section will evaluate the Unity3D implementation of WGW in *scatAR_WGW* through three case studies:

1. A comparison of impulse responses generated with SDN and WGW respectively. The purpose of this comparison is to validate the implementation of WGW, by reviewing the objective differences between the two signals.

2. An objective perceptual evaluation utilizing the Perceptual Evaluation of Audio Quality (PEAQ) [36], which is a standardized algorithm for an objective measure of perceived audio quality.

3. A perceptual experimental test, where participants rates the audio quality for sound samples processed with SDN and WGW, and unprocessed samples, for a virtual room at different positions and different sound sources.

All the case studies will be performed using sound samples processed in the same virtual room in Unity3D. Th geometries of this room is identical to the 9x7x4 meter shoebox room modeled and evaluated in [22]. All sound files used for the evaluation can be found in the Github repository *scatAR_WGW* [33] The results gathered in each case study will be triangulated with regards to understand if the there is a noticeable perceptual difference between the sounds processed by SDN and WGW. A marginal preference for SDN processed sound, over unprocessed sound was found in [23] using a rather complicated measurement of object-presence. In this evaluation the intention is to simplify the measurement, disregarding theorems such as immersion and presence, to specifically aim to find a similar marginal preference for WGW processed sound, over SDN processed sound, as WGW is deemed to provide a more accurate acoustical model. However, model accuracy and perceptual preference may not be correlated.

The overall *research question* is defined as: *Is there a perceptual difference between audio processed with the Scattering Delay Network and the Waveguide Web respectively, and is one version preferred over the other?*

4.1 Signal Comparison for a Shoebox Room

Identical settings for SDN and WGW were used to produce two impulse responses to be compared. The virtual room, which served as the geometries for the simulation can be seen in figure 4.1, measuring 9x7x4 meters.

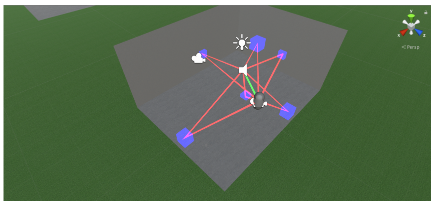


Figure 4.1: A virtual room in unity measuring 9x7x4 meters.

The wall reflectance coefficient was set to 0.97 to ensure a rich reverberant tail, and the filters in both simulations were designed from a flat frequency response, thereby an all pass filter. Figure 4.2 shows the impulse responses plotted on top of each other (left figure), as well as a difference plot containing the remainding signal after the two have been subtracted from each other (right figure). Theoretically the first-order reflections should cancel each other out completely in the difference plot. In this case however, the first-order reflections is actually slightly more amplified in WGW than the first-order reflections in SDN. It is a small, but considerable difference in amplitude.

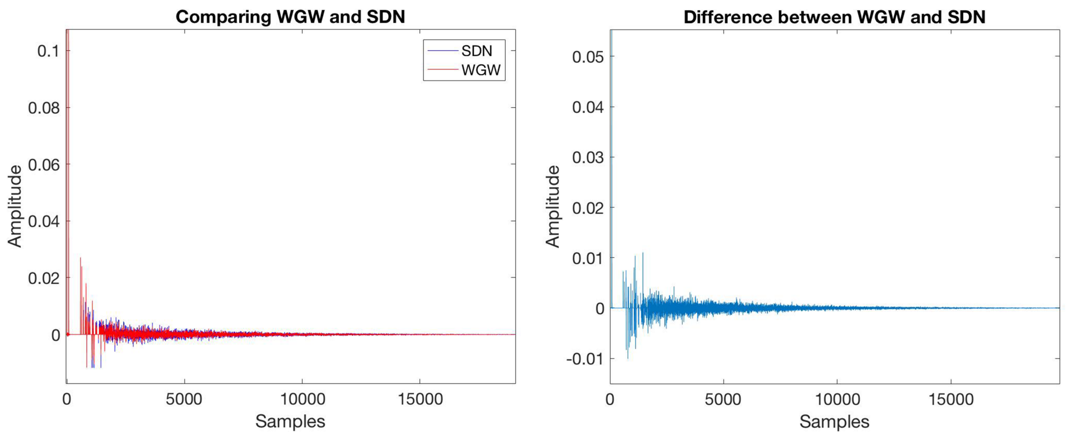


Figure 4.2: Left: Comparison of an impulse response generated with SDN (blue), and WGW (red). Right: The difference of the two signals

Most prominent in this figure is the difference in the higher-order reflections. This validates the correct attenuation of the second-order reflections. In WGW the attenuation factors for the second-order reflections are calculated according to the $\frac{1}{r}$ law, leading to small difference in the reflections above the second-orders.

These results are also validated by the T30 plot in figure 4.3, which shows the reverberation time for octave bands from 125 Hz to 10 kHz, based on the T30 room acoustic parameter (reference Acoustics, Measurement of Room Acoustic Parameters—Part 1: Perfor- mance Spaces). The reverberation times are clearly similar, but differ slightly due to the added design elements in WGW.

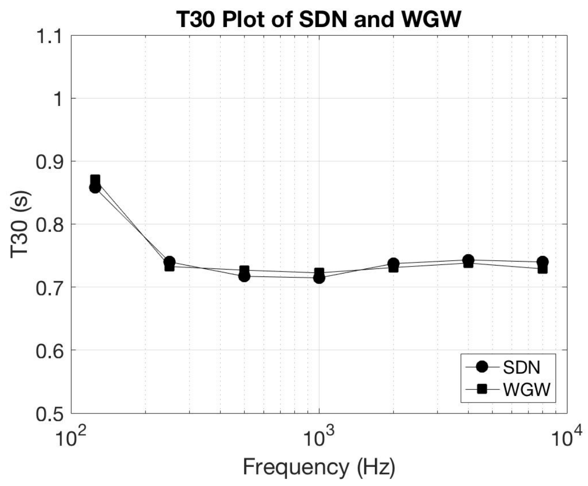


Figure 4.3: T30 plot of impulse responses generated with SDN and WGW.

This comparison was done to firstly, validate the WGW by comparing to SDN, and secondly to validate that the results gathered from the Unity3D simulation corresponded to the results found by [22]. Some issues, such as the difference in first-order reflections have been high-lighted, but all in all the results gathered in this section corresponds to the findings from [22].

4.2 PEAQ Evaluation

PEAQ (Perceptual Evaluation of Audio Quality) is an method for objectively measuring the perceptual quality of audio, and was released as ITU-R Recommendation BS.1387 in 1998 [36]. PEAQ uses a number of psycho-acoustical measures combined to give a measure of the quality difference between a reference and a test signal. Typical applications of PEAQ are performed to test audio quality between various audio compression algorithms or audio equipment [37], [38], but can generally be used to evaluate perceptual differences in audio signals of any relevance.

There are generally two types of the PEAQ algorithm: the basic version, and the advanced version. For this evaluation the basic version is used as implemented by P. Kabal in 2002 [39]. A public repository provides an easy-to-use MATLAB implementation based on the implementation by P. Kabal [40].

PEAQ contains several steps of psycho-acoustical measures, to output a set of acoustically based Model Output Variables (MOVs), which are then mapped to produce a single value representing the basic audio quality, named the Objective Difference Grade (ODG) [41]. The meaning behind the ODG value is provided in table 4.1.

Impairment	ODG
Imperceptible	0.00
Perceptible, but not annoying	-1.00
Slightly annoying	-2.00
Annoying	-3.00
Very annoying	-4.00

Table 4.1: The meaning of the ODG values

The labels in table 4.1 can be interpreted as a range from *identical* to *very different*. For the evaluation in this section, it will be assumed that a value around 0.00, means that the two signals are identical, and a value around -4.00 is completely different.

4.2.1 Generating Test and Reference Samples

48 samples were generated to be evaluated with PEAQ. The samples were generated in the same room as seen in figure 4.1, with different wall absorption coefficients, filters, sounds and reverberators:

- Reverberators: SDN and WGW
- Sounds: Impulse, Ball Impact and Speech
- Filters: All pass and Low pass
- Wall reflectance coefficients: 0.80, 0.90, 0.94, 0.97

This configuration produces $4 \times 3 \times 2 \times 2 = 48$ samples. The samples were produced by recording audio files from Unity3D in run time. This proved not to be a secure way of producing correctly time-aligned signals. It is important that the signals being compared are time-aligned to sample precision, so all the files were truncated and fixed to the same length, ensuring that the only difference in the signals were the type of reverberation. Additionally all the audio files were resampled from 44.1 kHz to 48 kHz, since the utilized MATLAB implementation of PEAQ only allows 48 kHz samples. Since it is in our interest to evaluate the difference between WGW and SDN, 24 comparisons with PEAQ will be performed, where all the WGW samples are used as reference samples, and all the SDN samples are used as testing samples, and thus the output will be 24 ODG values. This method provides a fast way of testing for multiple conditions, as an alternative to solely performing subjective tests with real participants. Based on the previous evaluation section, it is known that the difference between the two algorithms lie in the higher-order reflections. Therefore, the assumption is that the ODG values will increase in correlation with the wall absorption coefficients. Additionally, it is interesting to test if the type of filtering has an impact on the perceptual difference between the two reverberators. The ODG values will be divided by sound source and filter type to form 6 groups, which will be presented in the following section.

4.2.2 Results - ODG Values

A visualization of the derived ODG values can be seen in figure 4.4. Each graph in the figure compares the ODG values gathered using all pass filters, and the ODG values gathered using low pass filters. SDN and WGW applied the same filter design, but of course, WGW applies filtering multiple times at each node, where SDN only applies it once. The ODG values are the result of comparing WGW and SDN processed with the same settings, using the WGW signals as reference, and the SDN signals as test.

The plots in figure 4.4 clearly indicate a perceptual difference between WGW and SDN. The ODG values generally decrease (thereby indicating a higher difference) as the wall reflectance coefficient increases. This is due to the longer and richer reverberation tail introduced by higher reflectance in the room, since the higher-order reflections will take longer time to decay. This compliments the finding from the previous section, where the main difference between WGW and SDN was found to be due to the correct characterization of the second-order reflections.

Another observation to be made, is that the results from the low passed cases tend to yield a slightly higher difference than the all passed cases. This suggests that the different filter structures produce a perceptual difference even when all the filters are identical. The idea behind WGW is that each filter can be specifically designed according to the direction of the incoming wave signal. This was not facilitated in this test, but even with this filter configuration a perceptual difference was found.

The final observation is that the more complex signals generally produce lower ODG values, which is visible in figure 4.4. The speech signal produce lower ODG scores than the ball impact signal, and the simplest signal the impulse response, produce the highest ODG values. There is one oddity with the ODG values for the

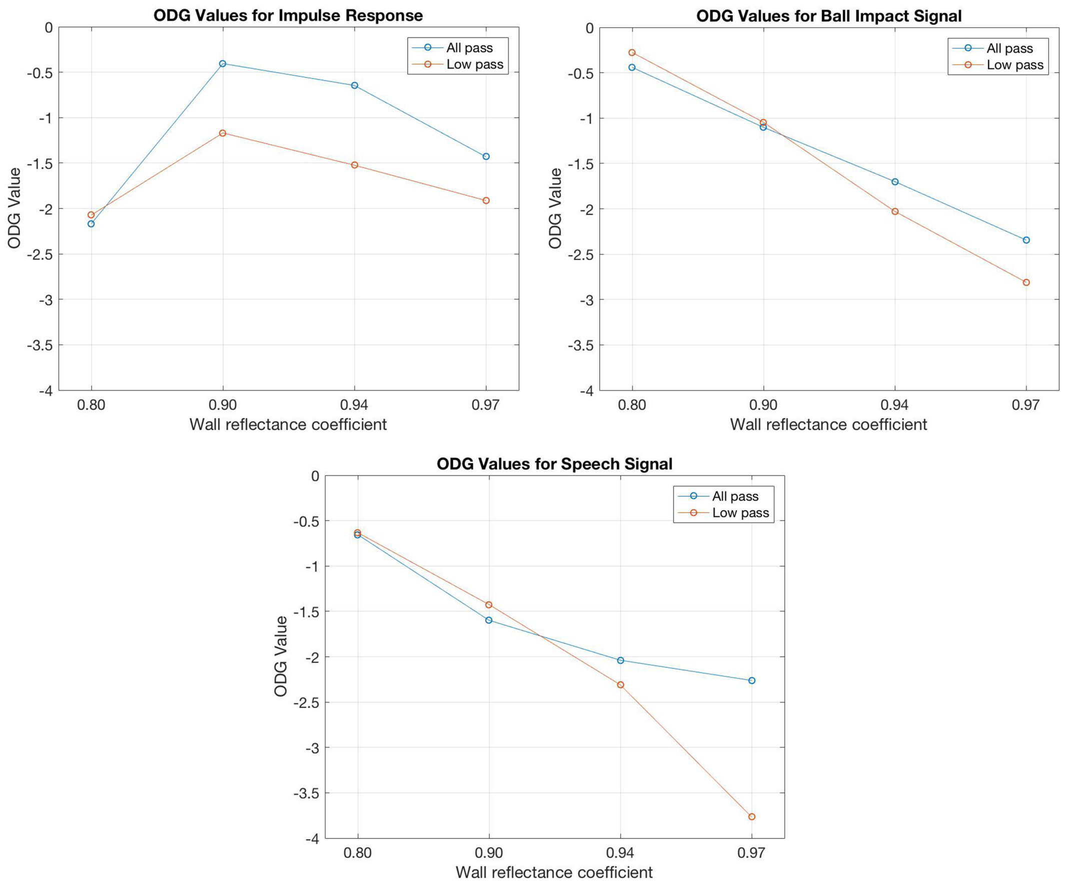


Figure 4.4: ODG Values obtained from the three signals. All the ODG values are derived from comparing a WGW processed signal with an SDN processed signal with the same filters and settings. Top left: Impulse Response. Top Right: Ball Impact Signal. Bottom: Speech Signal.

impulse response, which is the score for the 0.80 reflectance coefficient. According to the pattern, this should be the least different of all the scores. This could be due to a misalignment on the time-axis due to failed truncation of the original signals.

4.3 Perceptual Test

PEAQ has clearly indicated a perceptual difference, especially for the conditions offering a rich reverberant tail. This section will be used to validate the perceptual difference with a subjective measurement by testing actual participants, and highlight if the participants prefer one version version over the other. In addition to the sounds processed with respectively SDN and WGW, an unprocessed anechoic sound will also be used in the test. This will serve as a control condition.

4.3.1 Unity3D Environment

The room was design in Unity, based on geometries identical to the two previous evaluation cases. These geometries produce a room that would be considered quite large in a real-world setting. Since it would be difficult to assess the size of the room without any relative elements, some objects were implemented as seen in figure 4.5. The room needed to remain empty, for the modeled reverberation would not consider diffusion of furniture and other objects. Therefore only a door and windows were added to the room, since they are flat and would not alter the room reflections. The door is scaled relative to the 4 meter tall wall. Buildings, grass and road objects are added to the background, for a paralax effect when looking through the windows, to obtain a greater sense of depth. To make the room more relatable, textures were added to floor, wall and ceiling as seen in figure 4.5.

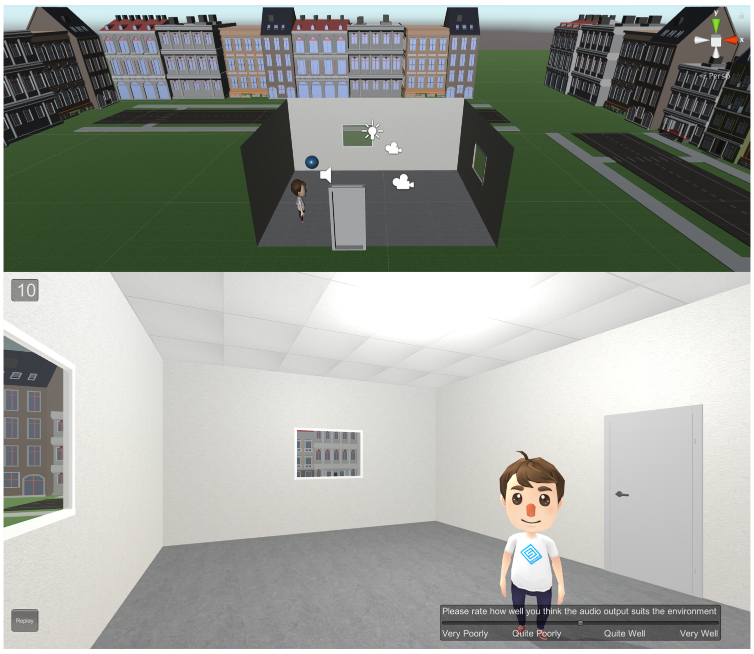

Please rate how well you think the audio output suits the environment

Figure 4.5: A small environment was created, to simulate relative size as well as possible. The room is supposed to represent a large room of 9x7x4 meters

All decisions regarding the size of objects, textures, lighting and materials have been made without precise measurements or references. Neither has the environment been tested for verification of its perceived size. Therefore, there is a risk that the participants will not be able to perceive the actual size of the room. Nevertheless, it should not affect the end results, since the condition will be the same for all cases.

4.3.2 Audio Stimuli

The input sounds for the perceptual test were recorded by the author in an anechoic room at Aalborg University in Copenhagen. The sounds were: the impact of a ball hitting a surface and a small speech excerpt. These two sounds served as the input for the sound source in two different conditions. One condition shows the character speaking (see figure 4.5) while the other condition shows a ball bouncing (see figure 4.6).

The original intention was to process everything in real time for the perceptual test, hereby utilizing the real time convolution reverberation solution proposed in chapter 3. However, with the initial pilot tests periodical glitches occurred both with the SDN and WGW processing. In addition, the impulse response generation was too slow and caused a significant delay in updating the convolution plugin. All in all, there was too much inconsistency for an experimental test.

Therefore, the sounds were pre-recorded for the experimental test. The test was designed to take the participant through three different positions in the same room, as shown in figure 4.6. Therefore a recording for each sound (ball and speech) and reverberation type, was recorded at each of the three positions, producing 12 audio files. A wall reflectance coefficient of 0.94 was chosen for the recordings. This amount of reflectance was deemed suitable for the environment, from a subjective assessment of the author. Both reverberators facilitated identical filters. To keep the test as controlled as possible an all pass filter was used. Similar settings were used for the original evaluation of WGW [22]. The two original anechoic recordings would be used as the control condition.

The recorded sounds would then be played in the corresponding conditions throughout the test. All the sounds were normalized, assuring that there would be no sudden changes in volume between anechoic, SDN- and WGW processed sounds.

4.3.3 Experimental Design

The original evaluation of scatAR [24] specifically focused on object-presence and utilized ratings of *spatial presence* and *perceptual realism*. The aim was to find a difference in these ratings between SDN processed sound and unprocessed anechoic sound. The participants were not directly asked to evaluate the output of the sound, but rather how present they felt in the environment. Both cases had ratings tending towards the higher end of the scale [24], which indicates that the reverberated sound was not enough to induce a higher sense of presence, when the participant already felt immersed in the Augmented Reality environment. Therefore, the aim of this test will be to specifically assess the quality of the audio output in a given context.

The measurement for this test is inspired by an experimental method proposed

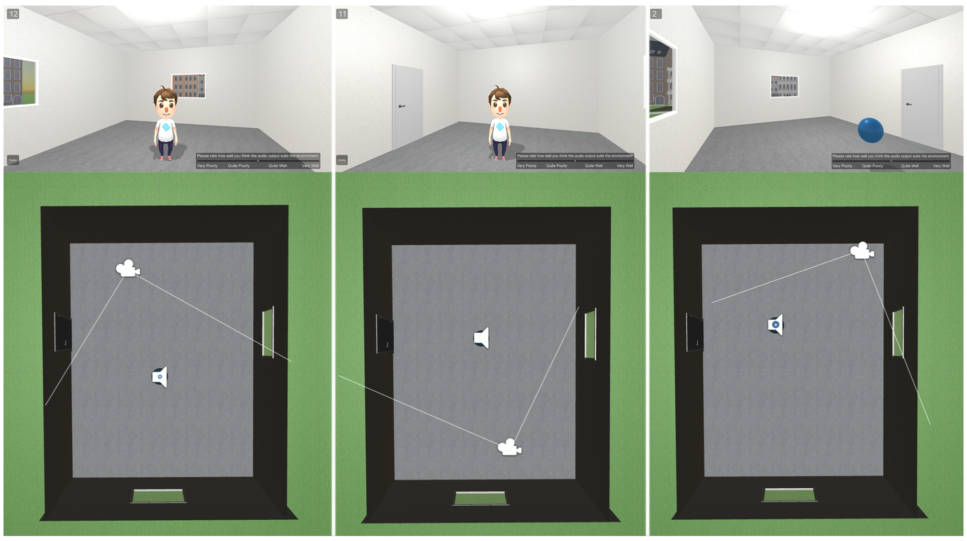


Figure 4.6: The three positions, seen from both the camera perspective and a top view. The sound icon depicts the position of the sound source, while the camera icon depicts the position of the listener. The figure also shows the two game objects (the character and the ball) associated with the two different types of audio stimuli.

by [42] to evaluate the quality of the audio quality of synthesized effects. In this method, the participant is asked to evaluate a number of sound samples on a scale from *very unrealistic* to *very realistic* as seen in figure 4.7.

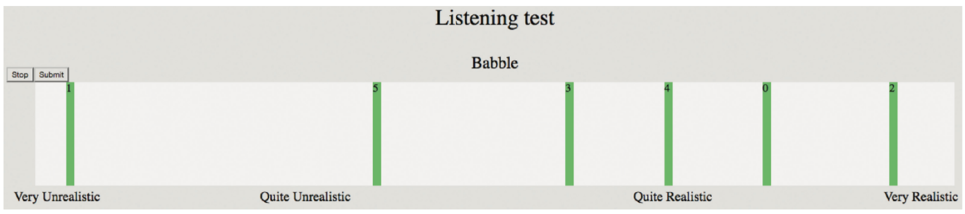


Figure 4.7: The scale used to evaluate the subjective perceptual quality of synthesized effects [42]. Here the participant can play a sound by clicking on of the green bars, and then drag them to place them on the scale.

The motivation for this test is also to evaluate the quality of a number of different sounds in the same manner. However, different adjectives will be used on the scale for the specific context in the virtual room, since *realistic* is too general in this case. The interest is to know how realistic the audio output is in the given context of the room and sound source position, therefore the scale is formulated as in figure 4.8.

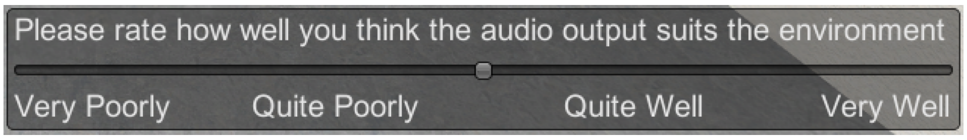

Please rate how well you think the audio output suits the environment
Very Poorly
Quite Poorly
Quite Well
Very Well

Figure 4.8: The scale implemented in the Unity3D test. For each case (combination of position, sound, and reverberation) the participant was asked to place a rating.

4.3.3.1 Test Control

The participant was asked to evaluate a total of 18 cases, consisting of combinations between three different room positions, two different sound sources and three different types of processing (SDN, WGW and anechoic).

The test started with a quick tour of the room, where the camera moved around in a fluent motion in the room to show the surrounding space, ensuring that the participant was well-aware of the room dimensions and their position in the room. Thereafter the source and listener positions are transported to one of the three positions (see figure 4.6) and the test begins. The first half of the test will instantiate one sound source, and the second half will instantiate the other sound source. Half of the participants was first introduced to the speaking character, while the other half was first introduced to the bouncing ball. The order of position and reverberation combinations were completely random.

When the participants had completed the test in Unity, they were asked to answer a short questionnaire, where they were allowed to comment on the environment, sound, and perceived room size. The specific questions can be found in appendix A.1.

4.3.4 Results

Valid data was successfully gathered from 20 participants. After a completed test, a CSV file was saved containing labels for sound, position, and algorithm, and the audio quality rating. The minimum rating corresponding to *very poorly* was 0, and the maximum rating corresponding to *very well* was 10. The mean ratings for WGW, SDN and Anechoic was found respectively to form a 1×3 vector containing the three mean ratings for each participant. These vectors were then combined to form a $N_{participants} \times 3$ matrix which will serve as the dataset for the descriptive statistics and experimental test.

4.3.4.1 Descriptive Statistics

Means and standard deviations were calculated for the ratings and is shown in table 4.2. The left table shows the results derived from the overall ratings. These have been labeled by WGW, SDN and Anechoic, and is not dependent on the sound source or position. These results are also visualized in figure 4.9. Anechoic was a control condition and was expected to have a low mean, which it definitely had compared to the two other ratings. The ratings for SDN were generally slightly higher than WGW. Whether this preference towards SDN is significant or not will be evaluated in the next section.

The two other tables in table 4.2 show the ratings depending on sound source and position respectively. These ratings are independent of algorithm type, and merely represent the total ratings for the given conditions. This is done to evaluate whether some of the conditions were biased.

Algorithm	Mean	STD
WGW	5.76	1.15
SDN	6.32	1.30
Anechoic	3.31	1.76

Sound	Mean	STD
Ball	5.30	1.16
Speech	4.96	0.97

Position	Mean	STD
Pos 1	5.28	1.06
Pos 2	4.79	1.15
Pos 3	5.32	1.17

Table 4.2: Means and Standard Deviations for the ratings. Left: General ratings independent of sound source or position. Middle: Overall ratings (independent from algorithm) for the ball and speech respectively. Right: Overall ratings (independent from algorithm) for the various positions

The ratings for the ball sounds have a slightly higher mean, indicating that the speech signal did not fit the environment as well. The ratings in the different positions are very similar, except for position 2, in which there is a dip in the mean. The standard deviations are generally small for all cases, indicating that the participants have been rating the sounds similarly.

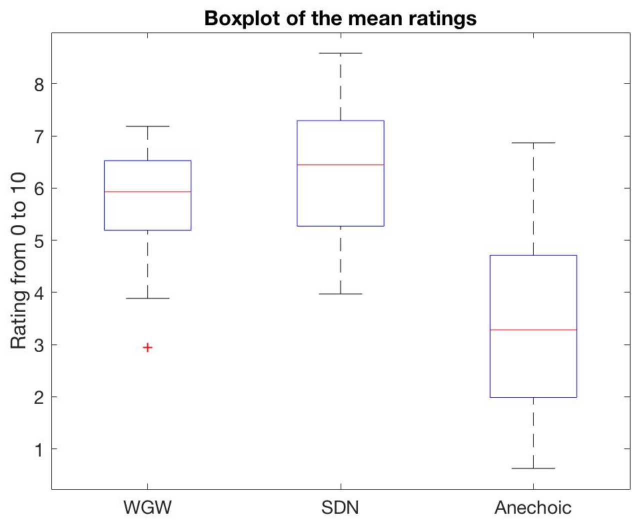


Figure 4.9: Box plot, visualizing the means, standard deviations and maximum/minimum ratings for the three different types of sound processing.

Finally the mean ratings depicted in figure 4.9 were tested for normal distribution using the Anderson Darling test. All three ratings were classified as being normally distributed. The distributions are plotted as histograms and can be found in appendix A.2. Thereby, it can be concluded that the data is parametric, and a parametric statistical test can be used to test for a significant difference.

4.3.4.2 Experimental Test of Significance

Based on the research question posed earlier, a null-hypothesis is formed:

There is no difference between the ratings of audio quality in sound processed with respectively SDN and WGW

A t-test was performed for three conditions as shown in table 4.3.

test	h	p
WGW vs SDN	1	0.0396
WGW vs Anechoic	1	5.19e-05
SDN vs Anechoic	1	1.96e-05

Table 4.3: H and P values for the t-tests

The t-test successfully rejects the null-hypothesis with a probability value of 0.0396 (0.4 percent probability of a type 1 error). Thereby it can be concluded that there is a significant difference between the ratings of SDN and WGW, with a preference towards SDN. The two conditions were also tested with the anechoic control condition to confirm that this difference was indeed also significant.

Chapter 5

Discussion

5.1 Overall Triangulation

The results from the three evaluations are clearly related. A difference is found by plotting signals generated with WGW and SDN. This is complimented by a perceptual different found in the PEAQ evaluation. Both evaluations indicate the difference lies in the higher-order reflection due to the correct characterization of the second order reflections. A perceptual test is performed with more limited conditions, than tested in PEAQ. Here significant differences are found between SDN, WGW and anechoic ratings, also indicating a preference for the SDN processed sound.

5.2 Perception of the Room

The average ratings for the SDN and WGW were only tending towards the positive side of the scale. The means of 5.76 and 6.32 for WGW and SDN respectively, indicates that the sounds did not fit environment as well as assumed. A value around 7.00 would have indicated that the sound fit the environment *quite well*. The results are below this mark, meaning that the participants were not convinced.

In the post questionnaire the participants were asked to rate the room size, by choosing one of the following items:

- Very small (like a close)
- Small (like a storage room)
- Medium (like an average bedroom)
- Large (like an average class room)
- Very Large (like a hall)

These items are ambiguous, as they can be interpreted differently by each participant, but can still provide an idea of how big the room was perceived by each participant. Almost all participants rated the room as being medium sized. But the actual room measurements were 9x7x4 meters, which would be a substantially large room of 63 m^2 with 4 meter tall walls. This indicates a critical mismatch between the room modeled by the reverberators and the room size perceived by the participants. Some participants had specific comments regarding the room size:

"It was hard to get a sense of the actual size of the room"

"The room seemed to be small. The tails of the reverb was a bit too long compared to what I perceived"

There were also comments about the materials and contents of the room:

"It was very difficult to imagine what kind of reverberation to expect from the room"

"The ceiling looks like it has acoustic panels over a large area, which should reduce reverberation in the room quite a lot, even though the walls and floor are untreated. This makes the sound examples either sound too reverberant or too dampened"

"There were no objects in the room, meaning the room should produce echo"

These comments indicate confusion regarding both the dimensions and materials of the room, which could have had an effect on the overall ratings of audio quality, however it should not have affected the relationship between the ratings, i.e. we assume that the interval between WGW, SDN and anechoic ratings would remain similar if the total ratings increased/decreased.

5.3 Spatial Audio

A lot of the participants asked if they should consider spatial panning and binaural properties for their rating. The spatial panning was rendered using Resonance Audio for Unity, and was not thought to affect the sound output, since the sound emitting objects were generally placed towards the center of the view. However, a considerable amount of participants noted that the panning was a bit off in one of the positions:

"There was one position, where the sound was realistic but it sounded like it came too much from the left, even if the character was in front of me"

This issue might explain the dip in the rating at position 2, as seen in table 4.2. The ratings for position 2 was 4.79, where the other positions had a mean close to 5.30. The issue with the panning probably caused the participants to generally rate the sounds worse at this specific position.

5.4 Game Object Believability

As presented in table 4.2 the ratings for the bouncing ball (M = 5.30) was generally higher than the ratings for the talking character (M = 4.96). One reason could simply be that the character was not as credible as a sound source as the ball. With the ball there is a clear congruent visual cue, where the sound should play: when the ball hits the floor and bounces off it. The character has been given an idle animation without any cues of producing audio. The participants were told beforehand, that they should consider the sound to come from the character despite the missing animations, but this info might not have affected the outcome.

Another reason could be that the reverberation effect was too intense for the speech signal. The reverberation of an acoustic space depends on the energy of the sound source. The speech signal was a calm voiced monologue, which would probably be expected to be read at a sound level corresponding to normal speaking. A good amount of participants had specific comments regarding the reverberation amount for the speech signal:

"Some of the voice had unusually high reverberation for such a small room."

"It was different to imagine what can of reverberation the two different signals would stimulate in the room. The voice seemed less accurate"

"No reverb was the worst. The ball test was fairly good. In the speech test the reverb was too much."

This indicates, that based on the expectations of the participants, the wall reflectance values should have been different for the speech signal and the ball impact signal respectively. This would however have compromised the credibility of the test, since the conditions would not be similar for the two cases. A comment regarding the sound of an empty room is considered:

"It is hard to imagine how an empty room should sound, especially when we don't know the material of the room surfaces"

This indicates that, since people are not experiencing sound in completely empty rooms regularly, it is difficult for them to know how an empty room should sound.

So, even though they assess the reverberation to be *too much* for the speech signal, it could be that they just do not have a real world reference. Perhaps, this issue could be solved by providing the participants with a number of reference sounds as a training phase before the actual test.

5.5 Improving The Evaluation of Room Audio

A recent study proposes strategies and methods for listening testing in VR [43]. Even though this test was not actually VR, the principles can still apply. According to the study, tests in VR should involve further consideration compared to traditional audio quality assessment tests, since in VR the sound image is usually attempting to deliver a listening experience closer natural listening [43]. Highlighted, is a study for identifying the listening position in a VR environment, where participants are trained to distinguish acoustics for different listening positions and to detect mismatches between the visual and acoustical representations [44]. The study finds that half of the participants were able to increase their accuracy, after having been trained. This suggest that a training session could improve the results of our perceptual test as well, and will be considered for future work.

Another study evaluates the equalization profiles for signals processed with an FDN reverberator, for different room sizes, and investigates the participants' ability to identify different room sizes based on the auditory information given [45]. The participants evaluate sound generated from 7 instrument performances, modeled for 5 different room sizes, and finds that the participants are able to organize sounds by room sizes. This test allowed the listeners to review sounds from different room sizes, which gave them some different profiles to relate to. This could also be considered for future work in our field, to see if different results would occur for different room sizes.

5.6 Direct to reverberant ratio

The direct to reverberant ratio cues are important to determine the distance to an audio source as highlighted by various studies [46]. As a control condition, the direct distance between source and listener was identical for all room positions, with a distance of approximately 4 meters. However, choosing the direct to reverberant ratio was not exactly calibrated. Utilizing the settings for Resonance Audio in Unity, the audio signal was set to decay over distance, but this decay was chosen in a subjective matter.

As highlighted in the study [46], listeners will consider the direct to reverberant ratio, when determining the localization of an audio source. Some of the participants might have taken this into, consideration, while doing the test. If they perceived the sound source to be close, then they would expect the direct sound to be more prominent than the reverberant sound. If this was the case, then the prominent reverberant tails would cause for a lower rating.

Chapter 6

Conclusion

This paper has presented the implementation of the Waveguide Web in an existing application *scatAR*, with the intention of replacing or accommodating the originally implemented Scattering Delay Network. Following the related theory, and presentation of the object oriented design structure, three case studies were presented to evaluate the perceptual differences between the two advanced artificial reverberators. A direct comparison of impulse responses generated with WGW and SDN respectively proved that there was indeed a difference in the modeling of higher-order reflections, due to the correct attenuation of the second-order reflections. These results corresponded to original findings by the author and creator of WGW, Francis Stevens. The results were complimented by the findings of an objective measurement of perceived audio quality utilizing PEAQ (Perceptual Evaluation of Audio Quality), where an increasing difference between WGW and SDN was found, depending on the length of the reverberant tail, which was controlled by a wall reflectance coefficient. Finally an elaborate perceptual test was performed, where participants were asked to rate how well they though the audio output fit into a given environment. The ratings were generally lower than expected, but they still concluded a significant difference between the reverberators, with a preference of SDN over WGW and anechoic sound. Thereby, the implementation of WGW has been validated, however it did not prove to be more attractive perceptually in this evaluation. The discussion chapter highlights a number of factors that could have affected the ratings for the different cases. These factors can be taken in to account for further research and development in the field. To conclude exclusively from the results presented here, there is no reason to implement the WGW in this type of application, while we are still constrained to simple room geometries build by plane surfaces, for two main reasons. Firstly, the algorithm is highly computational, with an exponential increase in computational requirements, depending on the amount of wall-nodes in the system. Secondly, the SDN proved to be more attractive to the participants. Further research and development of WGW would be interesting for more complicated environments, where SDN would not be sufficient. Such an environment could be an indoor space containing cylindrical shapes or other complex forms. Other environments would be previously evaluated outdoor environments, such as forest structures or open spaces in urban environments. This work posed a compromised solution for running WGW in real-time, by generating an impulse response and performing convolution with the input sound. This limits the system to only update the position of the source and listener at the rate of generating the impulse response. This means, that for WGW to run truly real-time some compromises would have to be made to the complexity, and thereby accuracy of the structure. The Waveguide Web still offers a wide variety of interesting studies for non-real-time purposes, and by modifying the filter calculation in the directionally dependent filtering structure, new types of scattering could be modeled to provide accurate acoustical models of complex spaces and structures. Very recent work presents a novel algorithm for generating virtual acoustic effects from AR, with an extra dimension where it scans both the geometry and the materials of the objects of a room, which are used to determine absorption coefficients and filter responses [47]. This algorithm is not real-time either, but in the journal it is concluded that further development would consider outdoor areas as well, thus highlightnin a need for an algorithm such as the Waveguide Web.

Acknowledgements

The author would like to thank Alex Baldwin for granting access to scatAR, and being available for any questions regarding the framework, Francis Stevens for the work on the Waveguide Web and assistance, resolving doubts about the model, Cumhur Erkut for excellent supervision, and lastly the anonymous participants who took part of the perceptual evaluation.

Bibliography

[1] Michael Cohen, Julián Villegas, and Woodrow Barfield. “Special issue on spatial sound in virtual, augmented, and mixed-reality environments”. In: *Virtual Reality* 19 (Nov. 2015), pp. 147–148.

[2] Dariusz Rumiński. “An experimental study of spatial sound usefulness in searching and navigating through AR environments”. In: *Virtual Reality* 19 (Nov. 2015), pp. 223–233.

[3] Barteld N. J. Postma and Brian F. G. Katz. “Creation and calibration method of acoustical models for historic virtual reality auralizations”. In: *Virtual Reality* 19 (Nov. 2015), pp. 161–180.

[4] Martin Naef, Oliver Staadt, and Markus Gross. “Spatialized Audio Rendering for Immersive Virtual Environments”. In: *02 Proceedings of the ACM symposium on Virtual reality software and technology* (Oct. 2002), pp. 65–72.

[5] Stefania Serafin, Michele Geronazzo, Cumhur Erkut, Niels C Nilsson, and Rolf Nordahl. “Sonic Interactions in Virtual Reality: State of the Art, Current Challenges, and Future Directions”. en. In: *IEEE Computer Graphics and Applications* (Apr. 2018), pp. 31–43.

[6] Vesa Valimaki, Julian D. Parker, Lauri Savioja, Julius O. Smith, and Jonathan S. Abel. “Fifty Years of Artificial Reverberation”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 20.5 (July 2012), pp. 1421–1448.

[7] Lauri Savioja and U. Peter Svensson. “Overview of geometrical room acoustic modeling techniques”. In: *The Journal of the Acoustical Society of America* 138.2 (2015), pp. 708–730.

[8] B. Xie. *Head-Related Transfer Function and Virtual Auditory Display*. 2013, pp. 351–352.

[9] A. Krokstad, S. Strom, and S. Sørsdal. “Calculating the acoustical room response by the use of a ray tracing technique”. In: *Journal of Sound and Vibration* 8.1 (July 1968). (Visited on 05/22/2018).

[10] J. B. Allen and D. A. Berkley. “Image Method for Efficiently Simulating Small-room Acoustics”. In: *The Journal of the Acoustical Society of America* 95.943 (1979).

[11] Julius O. Smith. “Scattering Delay Networks”. In: *Physical Audio Signal Processing*. online book, 2010 edition. `http://ccrma.stanford.edu/~jos/pasp/`, 2010.

[12] Julius O Smith. “A New Approach to Digital Reverberation using Closed Waveguide Networks”. In: *Proc. Int. Comput. Music Conf.* (1985), pp. 47–53.

[13] Matti Karjalainen. “Digital Waveguide Networks for Room Modeling and Auralization”. In: *Proceedings of Forum Acousticum* (2005).

[14] Sebastian J. Schlecht and Emanuel A. P. Habets. “Feedback Delay Networks: Echo Density and Mixing Time”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 25 (Feb. 2017), pp. 374–383.

[15] Davide Rocchesso and Julius O Smith. “Circulant and Elliptic Feedback Delay Networks for Articial Reverberation”. In: *IEEE Transactions on Speech and Audio Processing* 5.1 (1997), pp. 51–63.

[16] Stefan Bilbao, Brian Hamilton, Jonathan Botts, and Lauri Savioja. “Finite Volume Time Domain Room Acoustics Simulation under General Impedance Boundary Conditions”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24.1 (Jan. 2016), pp. 161–173. ISSN: 2329-9290, 2329-9304. DOI: `10.1109/TASLP.2015.2500018`. URL: `http://ieeexplore.ieee.org/document/7327143/` (visited on 05/22/2018).

[17] Konrad Kowalczyk and Maarten van Walstijn. “Room Acoustics Simulation Using 3-D Compact Explicit FDTD Schemes”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.1 (Jan. 2011), pp. 34–46.

[18] Enzo De Sena, Huseyin Haciihabiboglu, Zoran Cvetkovic, and Julius O. Smith. “Efficient Synthesis of Room Acoustics via Scattering Delay Networks”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23.9 (Sept. 2015), pp. 1478–1492.

[19] Jian Kang. “Numerical modeling of the sound fields in urban squares”. In: *The Journal of the Acoustical Society of America* 117.6 (2005), pp. 3695–3706.

[20] Reto Pieren and Jean Marc Wunderli. “A Model to Predict Sound Reflections from Cliffs”. In: *Acta Acustica united with Acustica* 97.2 (2011), pp. 243–253.

[21] K. Spratt and J.S Abel. “A digital reverberator modeled after the scattering of acoustic waves by trees in a forest”. In: *Audio Engineering Society Convention* 125 (Oct. 2008).

[22] Francis Stevens, Damian T Murphy, Lauri Savioja, and Vesa Valimaki. "Modeling Sparsely Reflecting Outdoor Acoustic Scenes Using the Waveguide Web". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 25.8 (Aug. 2017), pp. 1566–1578.

[23] Alex Baldwin, Stefania Serafin, and Cumhur Erkut. "ScatAR: a mobile augmented reality application that uses scattering delay networks for room acoustic synthesis". In: *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology* (2017).

[24] Alex Baldwin, Cumhur Erkut, and Stefania Serafin. "Towards the Design and Evaluation of Delay-based Modeling of AcousticScenes in Mobile Augmented Reality". In: *Presented in SIVE Workshop within IEEE VR Conference* (2018).

[25] Enzo De Sena and Huseyin Hacıhabibog. "Scattering Delay Network: an Interactive Reverberator for Computer Games". In: *41st International Conference: Audio for Games* (Feb. 2011).

[26] Damian Thomas Murphy and Francis Kit Murfin Stevens. "Spatial impulse response measurement in an urban environment". In: *55th International Conference: Spatial Audio* (2014).

[27] Francis Stevens. *Waveguide Web Example Audio*. `http://www.openairlib.net/auralizationdb/content/waveguide-web-example-audio`. Accessed: 2018-05-31. 2017.

[28] P. Morse. *Vibration and Sound*. 1948.

[29] Google. *ARCore*. `https://developers.google.com/ar/discover/`. Accessed: 2018-05-31. 2018.

[30] Apple. *ARKit*. `https://developer.apple.com/arkit/`. Accessed: 2018-05-31. 2018.

[31] Google. *How is ARCore better than ARKit?* `https://medium.com/6d-ai/how-is-arcore-better-than-arkit-5223e6b3e79d`. Accessed: 2018-05-31. 2018.

[32] Alex Baldwing. *scatAR*. `https://github.com/rampartisan/scatAR`. Accessed: 2018-05-31. 2017.

[33] J. Holfelt. *scatAR_WGW*. `https://github.com/jholfelt/scatAR`. Accessed: 2018-05-31. 2018.

[34] R. Marques, C. Bouville, M. Ribardière, L. P. Santos, and K. Bouatouch. "Spherical Fibonacci Point Sets for Illumination Integrals: Spherical Fibonacci Point Sets for Illumination Integrals". In: *Computer Graphics Forum* 32.8 (2013), pp. 134–143.

[35] Unity. *Unity Native Audio Plugin SDK*. https://docs.unity3d.com/500/Documentation/Manual/AudioMixerNativeAudioPlugin.html. Accessed: 2018-05-31. 2018.

[36] "RECOMMENDATION ITU-R BS.1387-1 - Method for objective measurements of perceived audio quality". In: *R BS.* (2002).

[37] Dinu Câmpeanu and Andrei Câmpeanu. "PEAQ — an objective method to assess the perceptual quality of audio compressed files". In: *Proceedings of the International Symposium on System Theory* (2005), pp. 487–492.

[38] M. Salovarda, I. Bolkovac, and H. Domitrovic. "Estimating Perceptual Audio System Quality Using PEAQ Algorithm". In: *18th International Conference on Applied Electromagnetics and Communications* (2005).

[39] P. Kabal. "An Examination and Interpretation of ITU-R BS.1387: Perceptual Evaluation of Audio Quality". In: *Lab Technical Report, Dept. Electrical & Computer Engineering, McGill University* (2002).

[40] N. Andersson. *PEAQ*. https://github.com/NikolajAndersson/PEAQ. Accessed: 2018-05-31. 2018.

[41] Thilo Thiede, William C. Treurniet, Christian Bitto Roland nad Schmidmer, Thomas Sporer, John G. Beerends, Catherine Colomes, Michael Keyhl, Gerhard Stoll, Karlheinz Brandenburg, and Bernhard Feiten. "PEAQ-The ITU Standard for Objective Measurement of Perceived Audio Quality". In: *Journal of the Audio Engineering Society* 48 (2000), pp. 3–29.

[42] David Moffat and Joshua D. Reiss. "Perceptual Evaluation of Synthesized Sound Effects". In: *ACM Transactions on Applied Perception* 15.2 (Apr. 2018).

[43] Francis Rumsey. "Listening strategies, methods, and VR". In: *J. Audio Eng. Soc.* 66.4 (2018).

[44] Florian Klein, Annika Neidhardt, Marius Seipel, and Thomas Sporer. "Training on the Acoustical Identification of the Listening Position in a Virtual Environment". In: *Audio Engineering Society Convention* 143 (2017).

[45] Gabriel Vigliensoni. "Perceptual evaluation of a virtual acoustic room model". In: *The Journal of the Acoustical Society of America* 142 (2017).

[46] Andrew Kolarik, Silvia Cirstea, and Shahina Pardhan. "Discrimination of virtual auditory distance using level and direct-to-reverberant ratio cues". In: *The Journal of the Acoustical Society of America* 134 (2013).

[47] Carl Schissler, Christian Loftin, and Dinesh Manocha. "Acoustic Classification and Optimization for Multi-Modal Rendering of Real-World Scenes". In: *IEEE Transactions on Visualization and Computer Graphics* 24.3 (Mar. 2018), pp. 1246–1259.

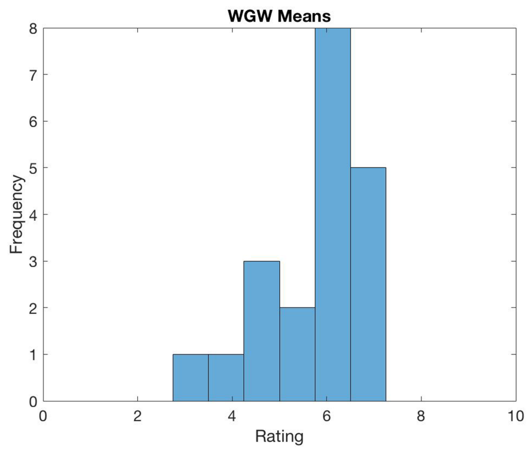
Appendix A

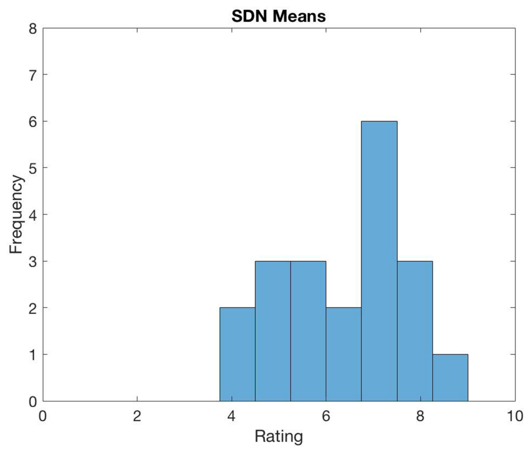
Evaluation

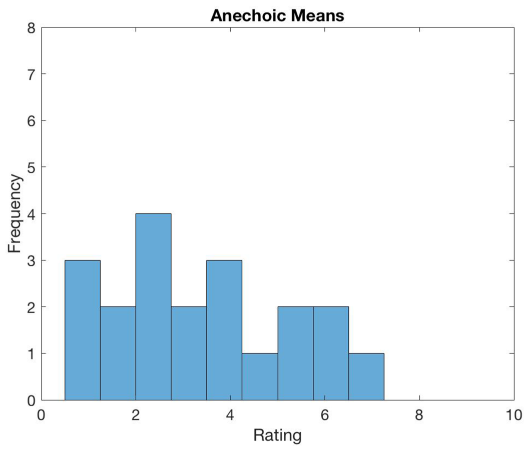
A.1 Post-Test Questionnaire

1. Did the formulation of the rating scale make sense?
2. If no, why not?
3. Did you experience any technical issues? (sound missing or glitching)
4. How would you describe the size of the room? (Choose One)
   - Very small (like a closet)
   - Small (like a storage space)
   - Medium (like an average bedroom)
   - Large (like an average classroom)
   - Very large (like a hall)
5. Please specify if you have any general comments about the environment
6. Please specify if you have any general comments about the sounds
7. Do you have experience recording, mixing or designing audio?

A.2 Histograms from Perceptual Ratings


WGW Means
Frequency
Rating


SDN Means
Frequency
Rating


Anechoic Means
Frequency
Rating