Simulating Room Acoustics in VR

A Feasibility Study Using Off-The-Shelf Equipment

Frederik Sidenius Dam

Acoustics and Audio Technology, Group 977, Fall 2022

3rd Semester Project, MSc

Written in LaTeX and compiled with pdfLaTeX in Overleaf. Body text is written in 10pt Inter Light font. Sketches and diagrams are made using diagrams.net and plots are generated using Matlab 2022b.

Department of Electronic Systems
Aalborg University
https://www.es.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Simulating Room Acoustics in VR

Theme:
Signal Processing and Acoustics

Project Period:
September 2022 - December 2022

Project Group:
977

Participant(s):
Frederik Sidenius Dam

Supervisor(s):
Flemming Christensen

Page Numbers: 54

Date of Completion:
21st December 2022

Abstract:

Recent advances in virtual reality (VR) and the surrounding technologies are accommodated by the digitalisation of the architecture, engineering and construction (AEC) industry, where interactive virtual models could improve the building design process through realistic auralisations.
This project investigates the use of an off-the-shelf VR headset combined with freely available software development kits (SDKs) and application programming interfaces (APIs) to perform room modelling and acoustic simulation. An implementation of a listening room for both Quest 2 and desktop is made using Steam Audio in Unity.
To evaluate the effectiveness of the room acoustic simulation, measurements were taken in both the physical room and a simulation of this. A comparison of the reverberation times as well as a perceptual evaluation of the room impulse responses (RIRs) is made in order to assess the performance. The results, both auralisations and measurements, deviate from the anticipation, however, this is expected due to deficiencies identified in the initial experiments.

Contents

Preface

This project report is written in the 3rd semester of the Master's programme in Signal Processing and Acoustics with specialisation in Acoustics and Audio Technology at Aalborg University.

It is intended to be read chronologically as presented in the table of contents and the appendices are included as supporting information.

Appendices include a description of the platform, configuration details, measurement journals and code snippets but additional documentation is handed in alongside as an external attachment. The attachment folder contains measurement results, scripts, evaluation files and the full implementation.

Citations follow the IEEE referencing style and are listed sequentially in the bibliography based on the order cited. All figures, tables, equations and listings are uniquely numbered and are referenced in the text when applicable.

With the signature below I confirm to have written the full report and provided citations to external material when used.

Aalborg University, 21st December 2022

Frederik Sidenius Dam
fdam21@student.aau.dk
20211693

1 Introduction

A paradigm shift has been seen in the architecture, engineering and construction (AEC) industry with recent digitalisation and dissemination of building information modelling (BIM). Most recently virtual reality (VR) has been adapted with favourable outcomes from research on the application of BIM-based immersive systems. Rapid development has increased computational power and lowered prices of VR products making this powerful technology accessible to the general consumer [1].

Visual and auditory information in the simulation of virtual environments is commonly separated into two tasks. Developing three-dimensional (3D) models for virtual visualisation and auralisation has been a part of building design for many years. However, the main focus in the VR-applications has been on the visual rendering [2].

Sound and room acoustic quality are important aspects of the indoor environmental quality (IEQ) which is made clear by extensive research into human task performance [3] and stricter regulatory requirements and building certifications such as BR18 [4], LEED [5], BREEAM [6] and WELL [7]. This increased awareness indicates the need for evaluation tools to include useful auralisations.

Based on interviews with multiple Danish and Norwegian employees from the case company inquiry in [8], it is known that auralisation is rarely used in practice and without visual supplements, which, to the knowledge of the author, also applies to other acoustic consultancies. This form of perceptual evaluation is mostly used for acoustically challenging spaces such as concert halls or other performance spaces where music and/or speech perception is critical. One reason is the time consumption since for instance objects that are small compared to the wavelength need to be excluded or replaced by simpler geometries in the existing computer aided design (CAD) model [2]. Automating and optimising this process is a whole task in itself.

Effective room acoustic simulation and auralisation software such as ODEON [9] as well as real-time integration into CAD software, as for example in [10], has been available to the industry for many years. Such tools are often expensive and thereby only made available through large corporations. Furthermore, transferring this into a VR domain remains a novel approach with e.g. [8] showing great potential in a VR case study.

Most recently Treble Technologies [11] is developing a state-of-the-art cloud-based room acoustic tool including novel wave-based simulations [12] integrated into VR platforms. Such a tool strengthens the possibility of including sound quality as a design parameter for e.g. educational buildings, public venues and homes, instead of an objective to reduce all sounds and make decisions based solely on calculated values. Furthermore, it is possible to detect and solve problems that otherwise first would be discovered after the building is constructed.

This project sets out to study how inexpensive off-the-shelf hardware combined with freely available software can be used to create realistic acoustic simulations in VR. It is meant as a preliminary investigation into developing a tool for acoustic consultants aiding the acoustic design of new buildings.

2 Analysis

As a consultant, within the field of architectural acoustics, it is essential to communicate the perceptual outcome and potential pitfalls of various geometries, materials, constructions etc. in the process of designing a new building.

The design of the building should ideally be considering proper acoustics from idea to finished building. An aiding tool should allow for user interactions including the possibility to change the material, i.e. acoustical properties, of surfaces and objects. Furthermore, the effect of these variations should be perceptually evaluated using various sound sources to cover everyday use cases, e.g. speech, music, appliances etc.

Besides the fundamental characteristics of e.g. reflection, transmission and absorption, real-life phenomena such as flutter echo and the acoustic coupling of adjacent rooms must be accurately simulated. These conditions and effects can be difficult to explain and the consequences cannot be interpreted solely by looking at calculated or measured material and/or surface parameters. By listening to realistic simulations, it is possible to detect and thereby avoid such unwanted effects. Ideally, such that it is made obvious even for the acoustically inexperienced.

Usually, the design process is a collaboration between architects and acoustic engineers, where collaboration and good communication is essential [8]. Looking into the newest technology and the so-called metaverse [13], a useful feature could be multi-user collaboration, where several people are immersed in the same virtual environment. In addition to communicative benefits, it enables remote collaborations that simply were not practically possible before. Even though similar applications already exist, it is still a relatively new technology and the practical usability still needs to be verified.

This chapter contains a brief analysis starting with a general overview of specific acoustic methods (Sec. 2.1) followed by a description of the development platform (Sec. 2.2).

2.1 Methods

Room acoustic simulation can be split into three phases; sound generation, sound propagation and sound reproduction [14]. Real-world sound generation is modelled using virtual sound sources with different frequency and directivity characteristics. It should be noted that sound sources can be either airborne or structure-born, however, the methods for propagation modelling only consider air propagation. Furthermore, the following descriptions are made assuming a single source, however, multiple sources can be included with their individual contributions added.

The last two phases are explained in more detail in the following sections.

2.1.1 Sound Propagation

Simulation of sound propagation can be done using several different frameworks. Generally speaking, these can be divided into two main approaches, based either on the assumption of geometrical acoustics (GA) or by numerically solving the wave equation (wave-based). Numerical techniques such as the finite-difference method (FDM), finite element method (FEM)

or boundary element method (BEM) are more precise than those based on GA. However, this comes with the cost of being computationally more expensive [15], [16].

Despite recent advances within wave-based virtual acoustics [12] this project focus on GA methods.

Simplifying Assumptions

Geometrical acoustics is often based on several simplifying assumptions with regard to both the propagation and reflection of sound. Usually, this kind of modelling only accounts for the visibility of a contribution and from this calculates the energy and propagation delay. The result from such energy-based methods is a time-energy response (often called a reflectogram) as seen in Fig. 2.1 [15], [17]

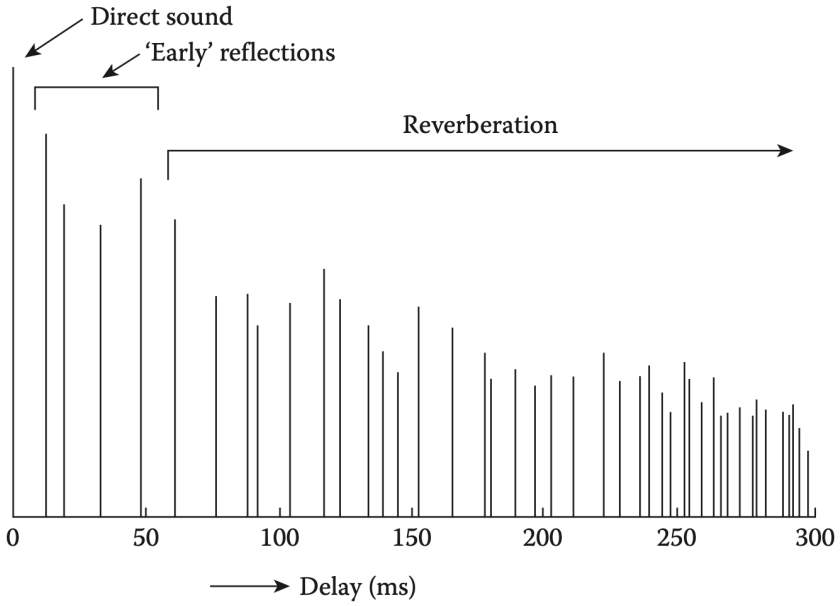


Figure 2.1: Example of a reflectogram [16].

This relative time-energy response, $E(t) = p^2(t)$, comprises the direct sound, early reflections and late reverberation. The transition from early to late reflections is not a fixed point in time and the simulation of it differs depending on the method, implementation and/or room geometry. A pressure-based room impulse response (RIR), $p(t)$, can be synthesised from the time-energy response which can then be used for auralisations [15], [17].

Another simplification is the modelling of reflections, which is roughly divided into two categories; specular or diffuse. A specular reflection has properties of a mirror as seen in Fig. 2.2, where the angle of incidence, θ_i, equals the reflection angle, θ_r, and the sound propagation from the reflection can be illustrated as a mirrored version of the source, i.e. an image. In contrast, a diffuse reflection is all reflections from a rough surface, where $\theta_i \neq \theta_r$, i.e. the reflection can propagate in any direction [15], [16].

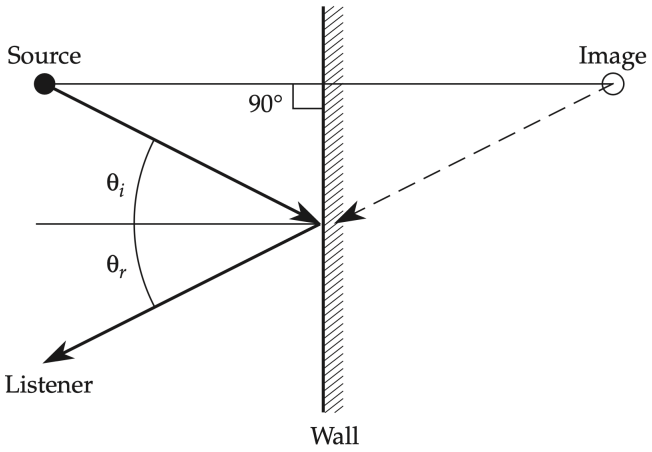


Figure 2.2: Example of a specular reflection and its mirrored image-source [17].

After the early reflections, the sound field in an enclosure typically becomes quite diffuse in practice. This happens for two reasons. First, reflections tend to scatter the sound rather than being completely specular. Second, the density of reflections increases over time meaning that each individual reflection is of less importance [15], [16].

When a plane sound wave hits a wall, not all of its energy is reflected. The incident energy E_i is divided into three parts; reflected energy E_r, absorbed energy E_a and transmitted energy E_t [18]. This is illustrated in Fig. 2.3 and written as the following equation:

$$E_i = E_r + E_a + E_t \tag{2.1}$$

The energy reflected back can then be written as the residual energy when subtracting the energy absorbed by and transmitted through the wall:

$$E_r = E_i - (E_a + E_t) = (1-\alpha)E_i \tag{2.2}$$

This equation introduces the absorption coefficient α defined as:

$$\alpha = \frac{E_i - E_r}{E_i} = \frac{E_a + E_t}{E_i} \tag{2.3}$$

In other words, all portions of the sound not reflected are considered absorbed. The material absorption is actually angle dependent, $\alpha(\theta)$. However, the simplifications often use the random incidence coefficient, denoted α_{rand} [15].

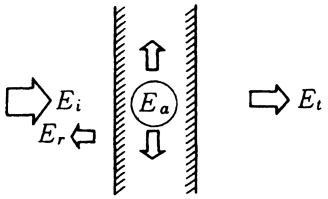


Figure 2.3: Reflection, absorption and transmission of energy [18].

The ratio of transmitted energy to the incident energy can be calculated and is called the transmission coefficient τ:

$$\tau = \frac{E_t}{E_i} \tag{2.4}$$

This is often expressed in decibels and called the transmission loss, TL, or sound reduction index, R:

$$TL = R = 10 \log_{10} \left(\frac{1}{\tau} \right) \tag{2.5}$$

The sound reduction can be used to describe and model sound isolation in building acoustics but is not necessarily a part of geometrical acoustics [18].

General Methods

There are basically two different GA-based sound field simulation methods; image-source (IS) and ray-tracing. The IS method is a deterministic way of finding all reflection paths. In principle, the method is relatively simple where the source is mirrored recursively, i.e. the ISs are also mirrored across each surface as shown in Fig. 2.4. The process is repeated until a stopping criterion is met, e.g. a specific reflection order or length of the response. Lastly, the visibility of the image-sources needs to be checked as for instance A_{21} represents an invalid second-order reflection path [15], [16].

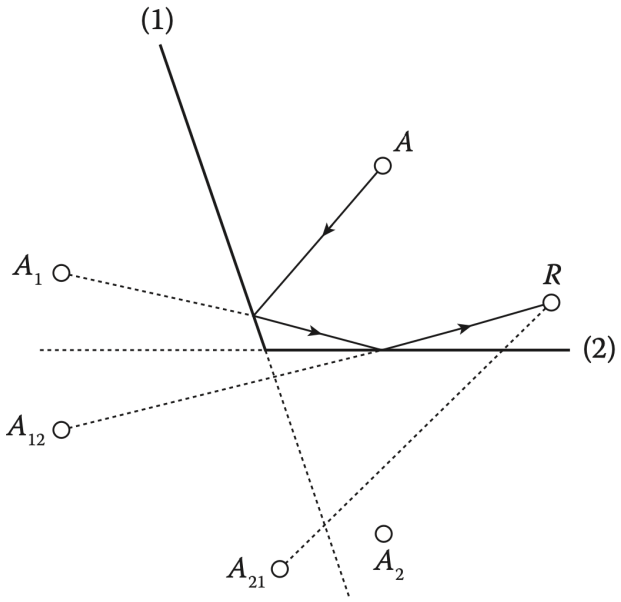


Figure 2.4: Principle of the IS method with valid and invalid (A_{21}) image-sources (A = sources and R = receiver) [16].

In contrast, ray tracing is a stochastic method, based on a random sampling of viable reflection paths. Rays are cast from the source in either a random direction via Monte Carlo or according to a chosen distribution. These rays are then reflected when hitting a surface and are added to the reflectogram if they at some point intersect the pre-defined counting space [16]. An illustration of the principle behind stochastic ray tracing can be seen in Fig. 2.5.

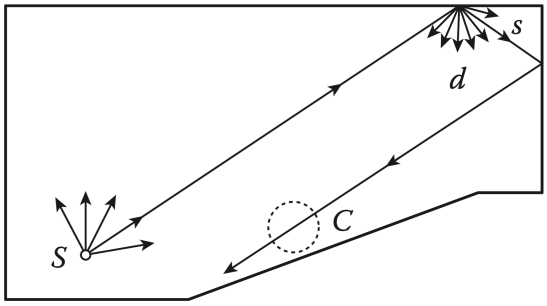


Figure 2.5: Principle of the ray tracing method (S = sound source, C = counting sphere, s = specular reflection, d = diffuse reflection) [16].

Different methods for termination of the rays exist, for example after a certain number of surface hits or when the energy left, after accounting for surface absorption, is below a specified limit. Each reflection can either be purely specular or have some kind of stochastic behaviour applied, i.e. diffusion. Such non-specular reflections can for example be applied by redirecting the wave in a non-specular direction according to some distribution or by splitting the wave into several new waves [15], [16].

There exist other reflection-path-based methods such as beam tracing techniques and surface-based methods like radiosity and acoustic radiance transfer [15]. However, these will not be described in detail here. Advanced implementations, for example in ODEON [9], are hybrid, where the IS method is used for early specular reflections in combination with ray radiosity for late diffuse reflections (scattering).

Deficiencies in GA

Due to the simplifying assumptions in geometrical acoustics, these methods lack accuracy. This is the case, especially when modelling diffraction and room modes. Edge diffraction is the phenomenon where an additional wave is created when a sound wave hits a boundary. This is sometimes described as the sound bending around an object and is most pronounced when the wavelength is large in comparison to the dimensions of the object. This is why the deviation is most significant at low to mid frequencies. According to [16] there is no practical solution to include this in GA algorithms, however, multiple methods to include diffraction exists and are often implemented in current hybrid methods [15].

Simulation of modal behaviour is the other major shortfall of using geometrical acoustics. When calculating energy contributions, the phase of the sound wave is ignored. The constructive and destructive interference is neglected, which is the physics behind these room resonances. Roughly speaking the limiting frequency for where the room modes are statistically overlapping, defined by the Schroeder frequency, represents the region where GA can be successfully applied. With the speed of sound defined as c = 343 m/s, this cutoff frequency can be estimated by Eq. (2.6) [16].

$$f_{sc} \approx 2000\sqrt{\frac{T}{V}} \tag{2.6}$$

Where:

f_{sc}	=	Schroeder frequency	[Hz]
T	=	reverberation time	[s]
V	=	room volume	[m^3]

This lack of accurate modelling in low and mid frequencies needs to be solved using wave-based calculations. For the highest efficiency, the two methods can be combined as done for example by Treble Technologies [11].

2.1.2 Sound Reproduction

The reproduced signal, $g(t)$, can be calculated by convolution of the source signal, $s(t)$, with the calculated time-pressure response of the propagation, $p(t)$:

$$g(t) = s(t) * p(t) \tag{2.7}$$

This is possible under the assumption that the RIR, $p(t)$, represents a linear time-invariant (LTI) system [14], and a block diagram of this is shown in Fig. 2.6.

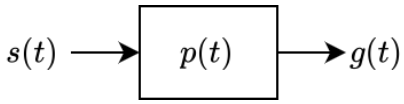


Figure 2.6: Block diagram for monaural auralisation.

The output signal, $g(t)$, can then be used for monaural auralisation. However, as humans have two ears, binaural signals are needed to provide realistic auralisation including spatial information [14].

A binaural signal is created by filtering both the direct sound and the reflective components of the RIR with the directional head-related transfer functions (HRTFs) for each ear [14]. This results in the binaural room impulse responses (BRIRs); $p_L(t)$ and $p_R(t)$, which can be applied to the source signal, $s(t)$, to create the binaural signals; $g_L(t)$ and $g_R(t)$, for the left and right ear respectively. A block diagram illustrating this can be seen in Fig. 2.7 below.

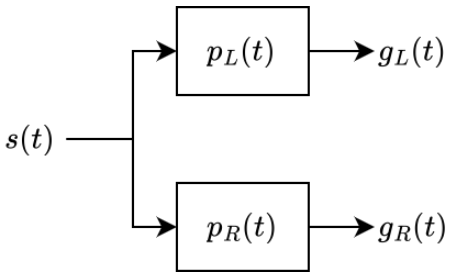


Figure 2.7: Block diagram for binaural auralisation.

Since HRTFs are physically measured at a finite set of discrete positions, a higher spatial resolution may be desired. It is possible to make use of various spatial interpolation schemes, instead of just applying the nearest filter [19].

It should be noted that subjective localisation, based on timing and spectral information, is most affected by direct sound and early reflections. The late reverberation is perceived with much lower temporal resolution and these psychoacoustical facts can advantageously be exploited in the calculations [14]. As mentioned, there is no exact definition of the time when the late reverberation begins. However, attempts of estimating this based on room acoustical parameters or experimental studies, e.g. [20], have been made with results ranging from 40 ms to 150 ms [21].

2.2 Platform

The laboratory at the Department of Electronic Systems at Aalborg University recently acquired a handful of Quest 2 [22] virtual reality (VR) headsets (see Fig. 2.8). Furthermore, the Quest 2 is a self-contained and relatively inexpensive VR device, making it an obvious choice as the development platform. It is a popular hardware platform used for both research and development including, but not limited to, room acoustic simulations.

Figure 2.8: Meta Quest 2 VR headset [22].

Several capable gaming engines exist where Unreal Engine [23] and Unity [24] stand out as the most suitable. With Unity being popular among researchers it is by far the most supported when it comes to software development kits (SDKs) and application programming interfaces (APIs) for spatial audio development and research. However, both engines are compatible with the ones selected. In the end, Unity was chosen due to the use of a general-purpose programming language (C#) and the simple and intuitive user interface.

An external audio plugin is needed to enable advanced spatialisation and room modelling features inside Unity. Many such plugins exist but popular choices are Meta Spatializer [25], Steam Audio [26], Resonance Audio [27] and dearVR [28], where the latter is a commercial product and thereby disregarded. For verification purposes, the idea is to use a virtual measurement microphone to capture the RIR of the simulated environment. It is necessary to use custom HRTFs for this to be feasible, singling out Steam Audio.

A more detailed description of the three aforementioned parts is included in Appx. A.

3 Implementation

Validation of acoustic simulation tools can be done in many ways. Simple scenarios can be analytically solved and compared to the simulation. However more in complex designs, it is more convenient to compare with real-life setups. Round robin comparisons, e.g. [29] are used to evaluate the state-of-the-art and larger benchmark database, e.g. [30], make measured acoustic scenes available for evaluation with thorough descriptions of room geometry, source and receiver characteristics as well as absorption and scattering coefficients.

A different approach is taken in this project since the aim is to evaluate the feasibility of using off-the-shelf equipment rather than tweaking a set of algorithms. Instead, room impulse responses (RIRs) of the standard listening room in the Department of Electronic Systems at Aalborg University is measured both physically (see Appx. C.1) and virtually (see Appx. C.2) for comparisons. Using a location that is physically available enables in-person evaluation, selection of source and receiver positions and modification of the room setup.

The following sections cover the room modelling and acoustic simulation as well as the subjective evaluation. Details regarding the specific implementation and experiments are included in Appx. B, and a compressed version of the Unity implementation is provided in "Attachments/Implementation".

3.1 Room Modelling

The listening room (see Fig. 3.1) is originally designed to conform with the IEC 60268-13 [31] standard, which describes the acoustics of an "average living room". In the standard configuration, the reverberation time is thus approximately 0.4 s. However, movable panels on the wall and ceiling make this adjustable [32].

Figure 3.1: Listening room B4-107 [32].

To simplify the scenario, all removable interior and acoustic wall panels are removed, while the various reflective and absorptive panels are left in the suspended ceiling. A sketch of the room can be seen in Fig. 3.2 where an acoustic partitioning wall is also included. The acoustic partitioning wall is used to modify the acoustic environment and investigate whether the effect is possible to recreate inside the virtual environment.

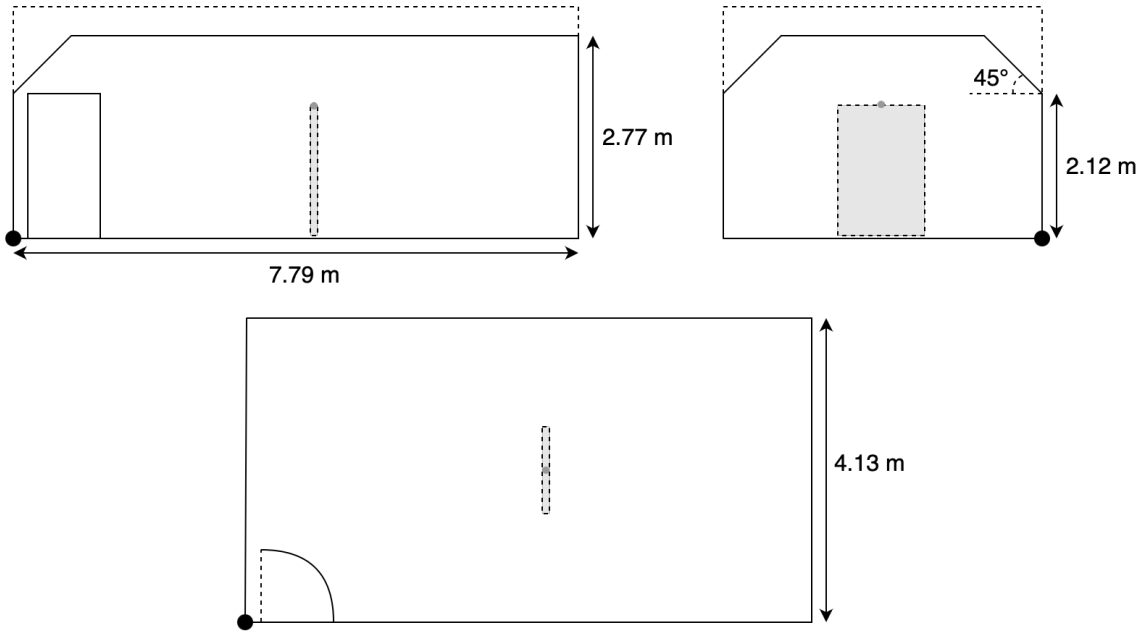
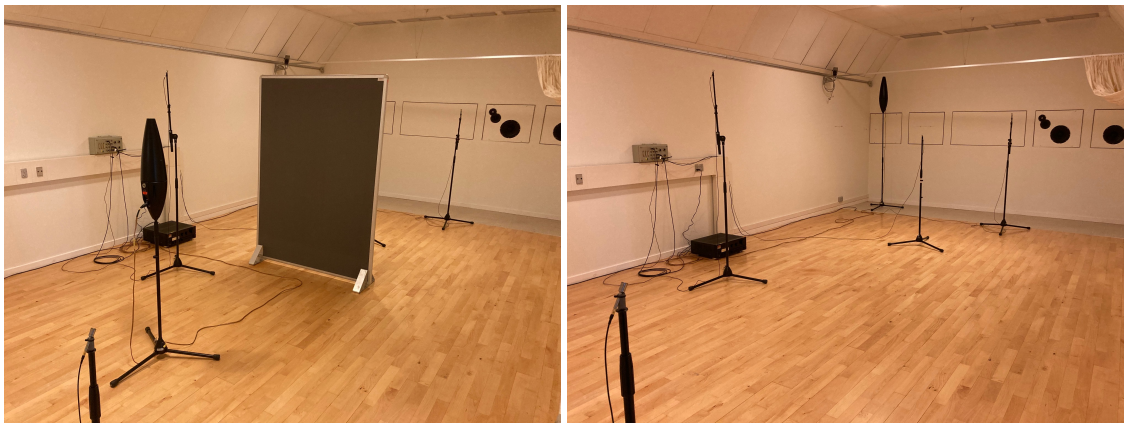


Figure 3.2: Sketch of the physical listening room B4-107 seen from the side, end and top.

Pictures of the emptied listening room can be seen in Fig. 3.3 with and without the acoustic wall and including the equipment used for the measurements.



(a) Source position 1 **(b)** Source position 2

Figure 3.3: Pictures of the physical listening room.

The virtual model is meant to replicate the actual listening room, however, several approximations and assumptions are made to simplify the simulation. First of all the virtual room is constructed as a rectangle using the largest visible dimensions (see Fig. 3.4), i.e. the inclined corners are ignored and the suspended ceiling is regarded as the boundary. Fixed objects such as the door, window, ventilation ducts, curtain, cable boxes etc. are disregarded. Furthermore, there are in-wall loudspeakers which might act as membrane absorbers, however, for the purpose of the simulation, this effect is neglected.

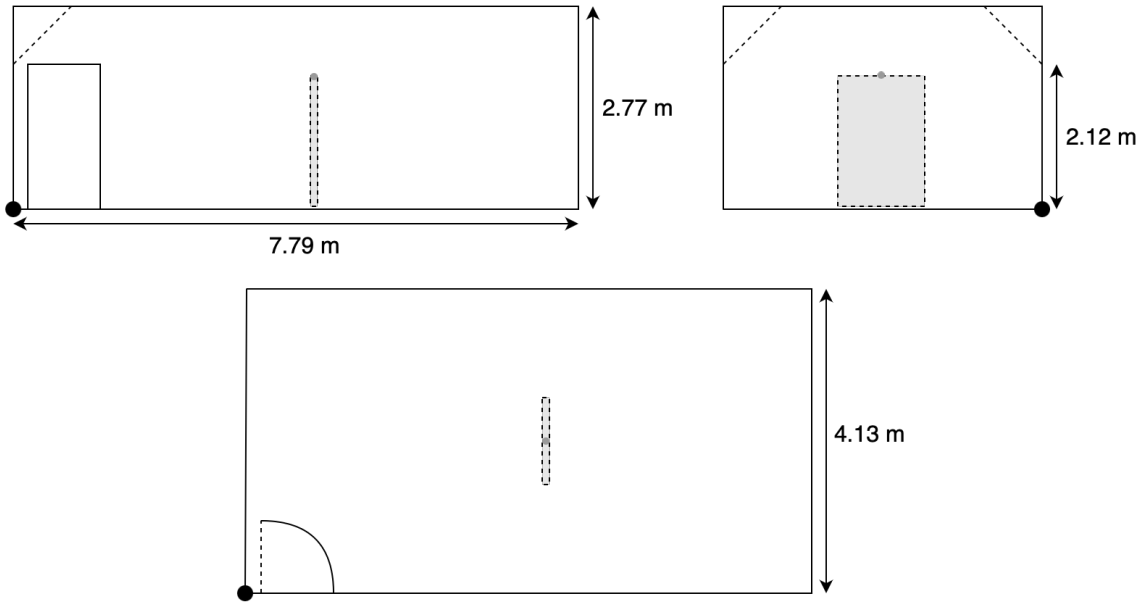


Figure 3.4: Sketch of the simulated listening room B4-107 seen from the side, end and top.

A screenshot of the three-dimensional (3D) model created in Unity is shown in Fig. 3.5 with the acoustic wall included and a single source-receiver combination.

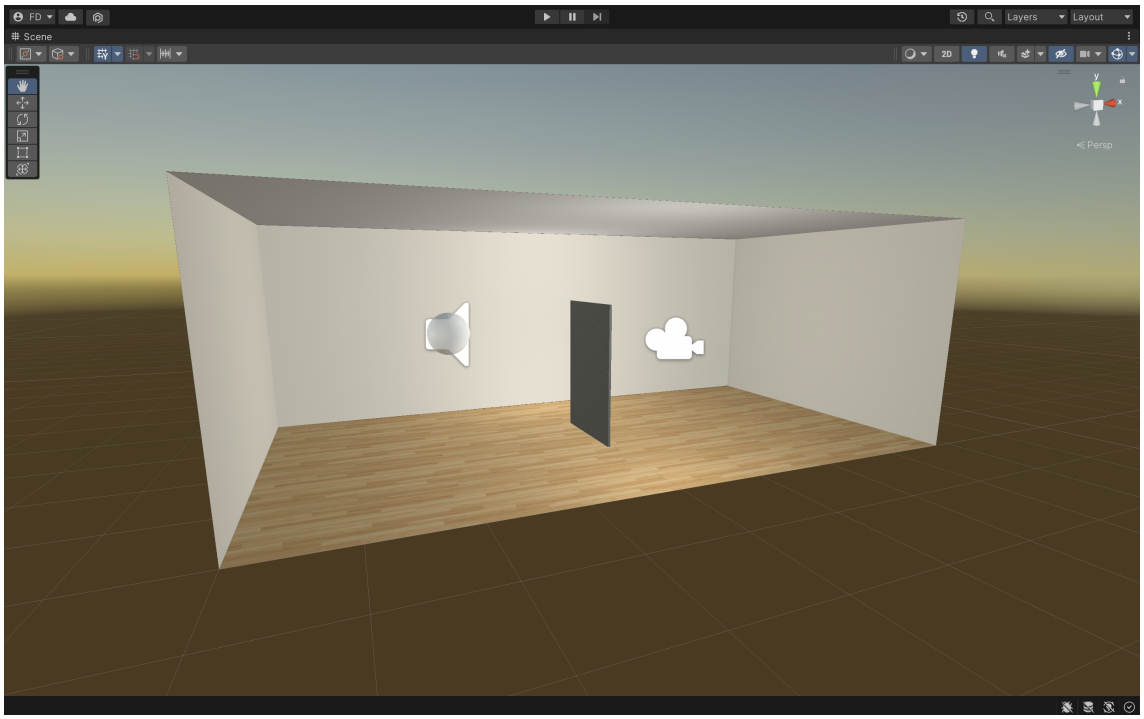


Figure 3.5: Screenshot of the simulated listening room in the Unity Editor.

Each surface is assigned a material with specific acoustic properties. This is also done with some simplifications. Limited information is available regarding the true acoustic properties of the materials, however, a set of measured random-incidence absorption coefficients, α_s, are available for a range of materials in [14]. The selected materials for the walls, ceiling and floor are listed in Tab. 3.1 below.

Frequency [Hz]	125	250	500	1k	2k	4k	8k
Walls, hard surfaces average (brick walls, plaster, hard floors, etc.)	0.02	0.02	0.03	0.03	0.04	0.05	0.05
Plasterboard ceiling on battens with large air-space above	0.20	0.15	0.10	0.08	0.04	0.02	-
16 mm wood on 40 mm studs	0.18	0.12	0.10	0.09	0.08	0.07	0.07

Table 3.1: Random-incidence absorption coefficients for selected materials.

The absorption coefficients are provided in seven octave bands, however, the *Steam Audio Material*, used for the acoustic simulation, is even further limited to three frequency bands. An average value for each band is used and the final absorption coefficients used are listed in Tab. 3.2.

Frequency [Hz]	Low (<800)	Mid (800-8k)	High (>8k)
Walls	0.02	0.04	0.05
Ceiling	0.15	0.05	0.02
Floor	0.13	0.08	0.07

Table 3.2: Absorption settings for selected materials.

Even though the statistical absorption coefficients are measured according to the standard for measurement of sound absorption in a reverberation room (ISO 354 [33]), a systematic difference between such measurements and the true α_{rand} has long been a topic of study [15].

3.2 Acoustic Simulation

The acoustic simulation is made possible with the Steam Audio software development kit (SDK) for Unity. It is cross-platform and supports Android (e.g. Quest 2) and desktop (e.g. Windows and macOS). By default, a new project is configured for building applications to the device running the Unity Editor, which in this case is macOS. Setting the project up for building and deploying to Quest 2, however, is simply done by following the instructions provided in the documentation [34].

The only platform dependency in this project is the user interaction where Meta provides an integration SDK for basic character movement using the Quest 2 headset and its controllers [35]. Two scripts for keyboard and mouse movement (source code is found in Appx. D.2) are made for the desktop version.

Steam Audio provides features for spatialising an audio source as well as simulating occlusion/transmission and reflections using raycasting (see Appx. A.3 for more details). At the time of writing, reflections have not been successfully implemented on the Quest 2 headset. Earlier versions of Steam Audio were limited by this, however, it should no longer be the case according to the first answer in the discussion initiated by the author [36].

Unity includes real-time ray tracing which is used to simulate realistic light behaviour and interactions. While it is possible to use this, the best performance for modelling early reflections and reverberation is done using the Steam Audio built-in ray tracer [26]. The *Steam Audio Settings* allows for changing various parameters including the number of rays and bounces as well as the duration of the impulse responses (IRs). With *Steam Audio Material* it is possible to adjust absorption in the three frequency bands described earlier along with the total scattering in values from 0 to 1.

Similar to the reflections, the occlusion/transmission can be adjusted in the three frequency bands. However, this effect is found to only have an impact on the direct sound. This is

exemplified by the experiments presented in the next section (see Fig. 3.7), where the transmission coefficient has no effect on the overall sound decay and thereby reverberation time. For this reason alone, it is not useful for acoustic simulations of airborne sound transmission.

Other features provided are distance attenuation and air absorption which can be physics-based and frequency-dependent. Furthermore, it is possible to adjust the directivity of the sound source with two parameters controlling the dipole weight and power. At last, it includes baked features, enabling the pre-computation of sound propagation. This has not been experimented with in this project but it can reduce the real-time central processing unit (CPU) usage in complex simulations.

3.3 Experiments

With the room model and acoustic simulation set up, it is possible to preview the real-time simulation inside the Unity Editor. This is used to get a snapshot impression, but verification is done with virtual acoustic measurements. To do this, a virtual microphone is created where the head-related transfer functions (HRTFs) are defined in a single finite Kronecker delta. Steam Audio accepts spatially oriented format for acoustics (SOFA) files [37] and the Matlab [38] code for generating these is included in Appx. D.1. SOFA is a binary file format for storing spatially oriented acoustic data and is defined in the Audio Engineering Society (AES) standard AES69-2020 [39].

Several experiments are made in order to tweak and customise the simulation and explore the effects of the different settings. Normalised impulse responses and Schroeder energy decay plots for all test results are provided in "Attachments/Measurements/Simulation". In practice, the measurements are made by recording the excitation signal played back from Unity with Audacity [40]. The audio signal is routed with zero latency between the two apps using BlackHole [41]. The measurements are made using source position 1 and receiver position 3 with the virtual measurement microphone. A short description of the various experiments is presented in Tab. 3.3.

Name	Description
Ref	Steam Audio Static Mesh disabled, i.e. no geometry and free field conditions
Empty 0	Default Steam Audio Settings
Empty 1	Only Real-Time Rays changed from 4096 to 65536 (maximum)
Empty 2	Only Real-Time Bounces changed from 4 to 64 (maximum)
Empty 3	Only Real-Time Duration changed from 1 to 10 (maximum)
Empty 4	Only Real-Time CPU Cores Percentage changed from 5 to 100 (maximum)
Empty 5	Copying default baked settings with rays = 16384, bounces = 16 and CPU = 50
Empty 6	Same as 5 with Direct Mix Level changed from 1 to 0.5
Empty 7	Same as 6 with Physics-Based Distance Attenuation checked
Wall 0	Absorption = 0 and transmission = 0 (all frequency bands)
Wall 1	Absorption = 0 and transmission = 1 (all frequency bands)
Wall 2	Absorption = 1 and transmission = 0 (all frequency bands)
Wall 3	Absorption = 1 and transmission = 1 (all frequency bands)
Wall 4	Absorption = 1 and transmission = 0 (opposite for low frequencies)

Table 3.3: Modifications applied for different experiments.

The reverberation time, T, is the time it takes for the space-averaged sound energy to drop 60 dB. This is often calculated based on a smaller evaluation range of 20 dB and denoted T_{20} [42]. Here it is calculated for a single position in octave bands and derived from Schroeder energy decay curves. All cases with the empty listening room are shown in Fig. 3.6, where it is clear that some of the measurements stand out.

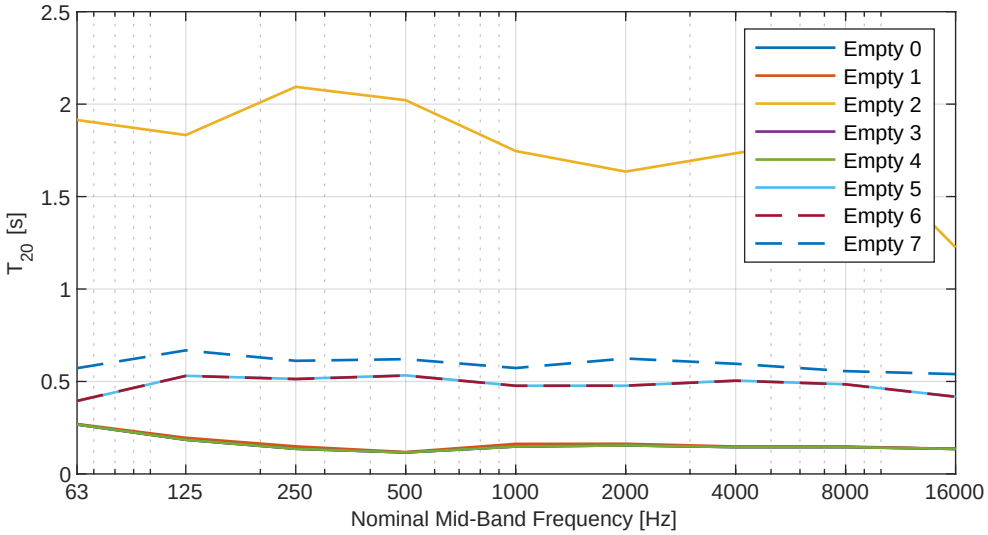


Figure 3.6: T_{20} for the experiments (empty listening room).

Increasing the number of rays (Empty 1) has almost no influence on the reverberation time. This is expected since the room is relatively small and has a simple geometrical construction. An increase in the number of bounces (Empty 2), on the other hand, gives a large increase in reverberation time and uneven distribution over frequency. This is by far the most dominating setting which is not ideal for realistic simulations since the reverberation time should depend solely on the room geometry and the material properties. Applying the default settings set for baked simulation (Empty 5 and 6) increases the reverberation time with the largest values obtained by applying physics-based attenuation (Empty 7).

Figure 3.7 shows the measurement results with the acoustic partitioning wall placed upright in the middle of the room. As already mentioned, the transmission coefficients have no impact on these results, since the effect is only applied to the direct sound. The effect of having an acoustically absorbing wall is obvious, with the reverberation time being decreased. The more realistic case (Wall 4), where only mid and high frequencies are absorbed, behaves as expected. However, these are only examples of extreme cases with no or full absorption. Absorption coefficients for the acoustic partitioning wall are unknown and have not been measured in this project.

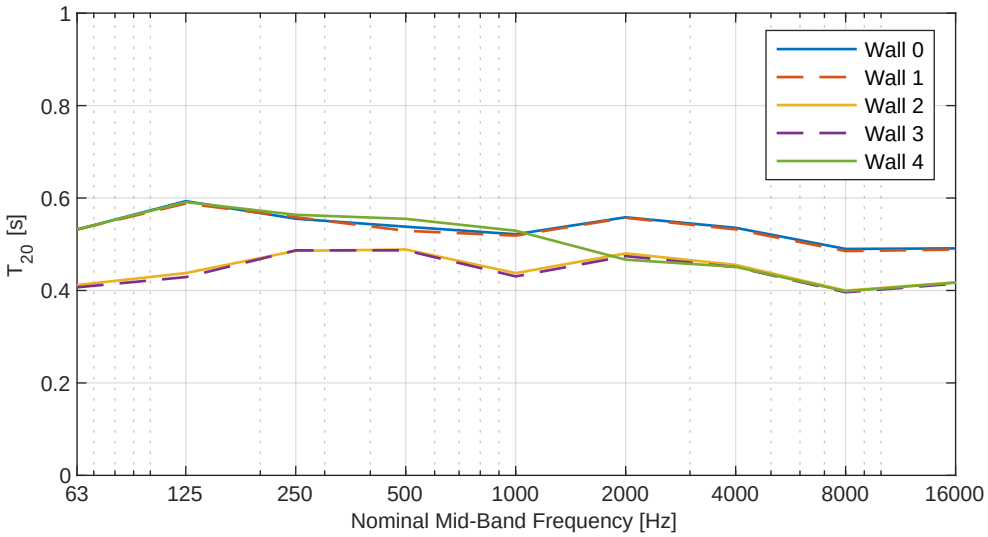


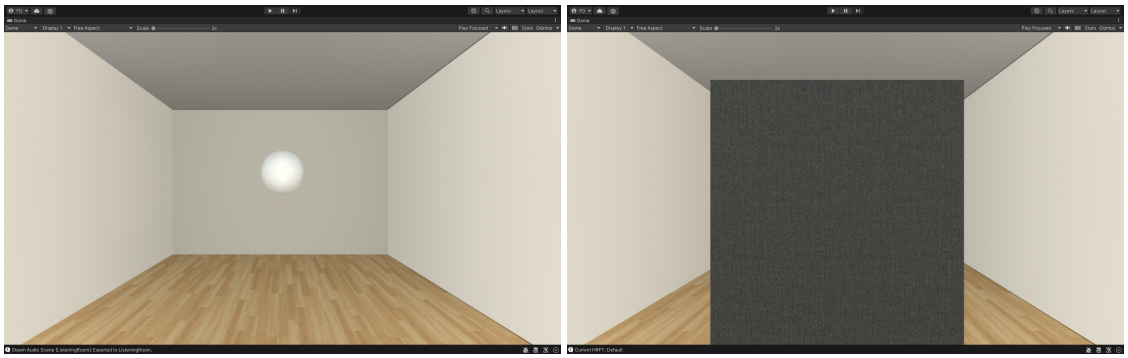
Figure 3.7: T_{20} for the experiments (empty listening room with the acoustic partitioning wall).

The two cases Empty 7 and Wall 3 are used in the final perceptual evaluations and for the complete reverberation time measurements.

3.4 Perceptual Evaluation

The end goal of an acoustic simulation is not to match a measurement numerically. This is both unrealistic in complex scenarios and unnecessary as it is the perceived auralisation quality that is important. Perceptual evaluations are done using the techniques presented regarding sound reproduction.

Monaural impulse responses are listened to directly as well as convolved with an anechoic speech signal from "Music for Archimedes" by Bang & Olufsen [43]. All of the generated audio files are included in "Attachments/Evaluation" along with the raw speech recording. The process is repeated both for the physically measured RIRs and the ones obtained from the simulation. Both IRs are acquired using the same loudspeaker and microphone position (source 1 and receiver 3). Furthermore, real-time auralisations are done with various HRTFs in compiled and built applications for both desktop (macOS) and Quest 2 (Android). Quick auralisations are done using the preview functionality in Unity and an example of the first-person view obtained can be seen in Fig. 3.8.



(a) Empty **(b)** Wall

Figure 3.8: Game View in Unity.

By listening to the monaural IRs alone it is clear that the two differ a lot. The simulated impulse response of the empty room has a metallic and unnatural timbre. This difference is still audible when convolving the IRs with the speech signal. However, the reverberation of the rooms is clearly similar with a difference in the colouration. This can both be due to insufficiency in the algorithms and in the modelling choices. It should also be noted that the sound source used for the physical recording is not a perfect point source as the one in the simulation. The physical loudspeaker (OmniSource Type 4295 [44]) is limited in frequency range and directionality, however, fulfilling the requirements of ISO 3382-1 [45] for an omnidirectional sound source.

Binaural auralisations are carried out in real-time in the same manner as monaural. Besides the default set of HRTFs in Steam Audio, high directional resolution HRTFs measured with the Valdemar head and torso simulator (HATS) [46] are used. The script for generating a SOFA file from these measurements is included in Appx. D.2. It is possible to switch between the different HRTFs in real time using the desktop application and with the preview function. This makes it possible to listen to both the monaural microphone signal and various binaural signals in real time while moving around. The perceptual experience is similar to static monaural listening, however, the sensation is more realistic and now includes the spatial effect of the relative source-listener position. Real-time auralisations using Quest 2 are done as well, but they are only used for initial impressions, since the simulation of reflections unfortunately is not working.

All listening is done using a pair of Sony MDR-7506 [47] closed-back studio-headphones, as well as the built-in loudspeakers of a MacBook Air [48] and Quest 2 headset. It should be

added that both the acoustic response of the headphones/speakers and the HRTFs influence the overall perceived sound quality. With this being said, this is not a scientific experiment but rather the author's momentary impression of the auralisations.

3.5 Results

This section presents results from the reverberation time measurements of the listening room (B4-107) in the laboratory at the Department of Electronic Systems at Aalborg University. Measurements are conducted both in the physical room and in the virtual simulation as described earlier in this chapter.

Two different room configurations are replicated in the virtual environment; the empty room and the empty room including the acoustic partitioning wall. Further documentation of the measurements can be found in Appx. C.1 and C.2. Figure 3.9 shows both the physical and simulated reverberation time results in octave bands for both setups.

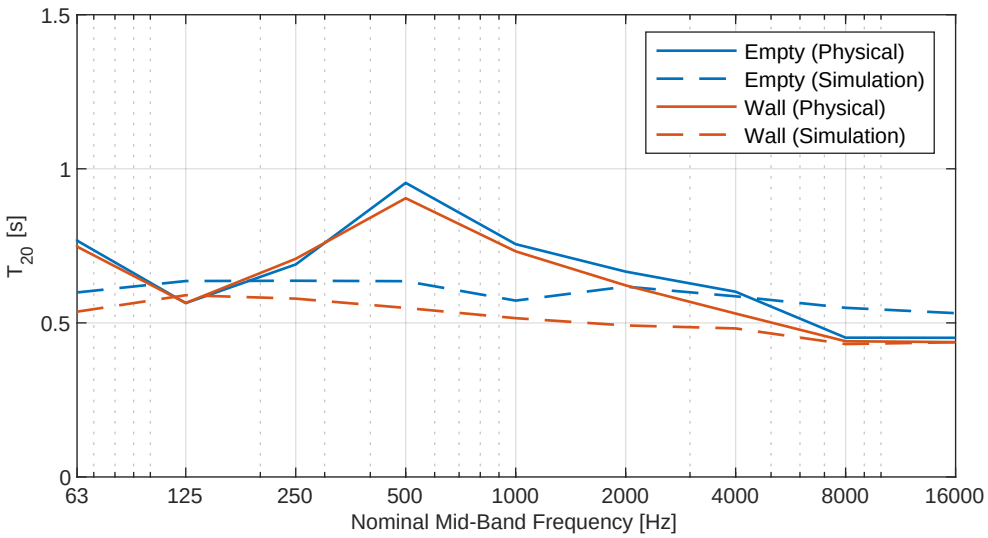


Figure 3.9: Combined T_{20} for the two setups in the listening room.

It is obvious that the difference between the two setups is most explicit in the simulation. This is expected since the acoustic partitioning wall is modelled with total absorption which is not true in the physical world. Furthermore, the reverberation time curve is expected to have a negative slope towards high frequencies due to rough surfaces and air absorption. The simulation curves are almost flat and the physical curves show a dip at the lowest frequencies. These deviations might be due to insufficiency in the simulation and the suspended ceiling providing significant physical absorption at low frequencies.

Numerical results from Fig. 3.9 are listed in Tab. 3.4 and 3.5 below.

Frequency [Hz]	63	125	250	500	1k	2k	4k	8k	16k
Physical [s]	0.77	0.56	0.69	0.95	0.76	0.67	0.60	0.45	0.45
Simulation [s]	0.60	0.64	0.64	0.64	0.57	0.62	0.59	0.55	0.53

Table 3.4: T_{20} for the empty listening room.

Frequency [Hz]	63	125	250	500	1k	2k	4k	8k	16k
Physical [s]	0.75	0.56	0.71	0.90	0.73	0.62	0.53	0.44	0.44
Simulation [s]	0.54	0.59	0.58	0.55	0.52	0.49	0.48	0.43	0.44

Table 3.5: T_{20} for the empty listening room with the acoustic partitioning wall.

As an example, the measurements in the empty listening room including the standard deviation is shown for both the physical (Fig. 3.10) and the simulated (Fig. 3.11) measurements.

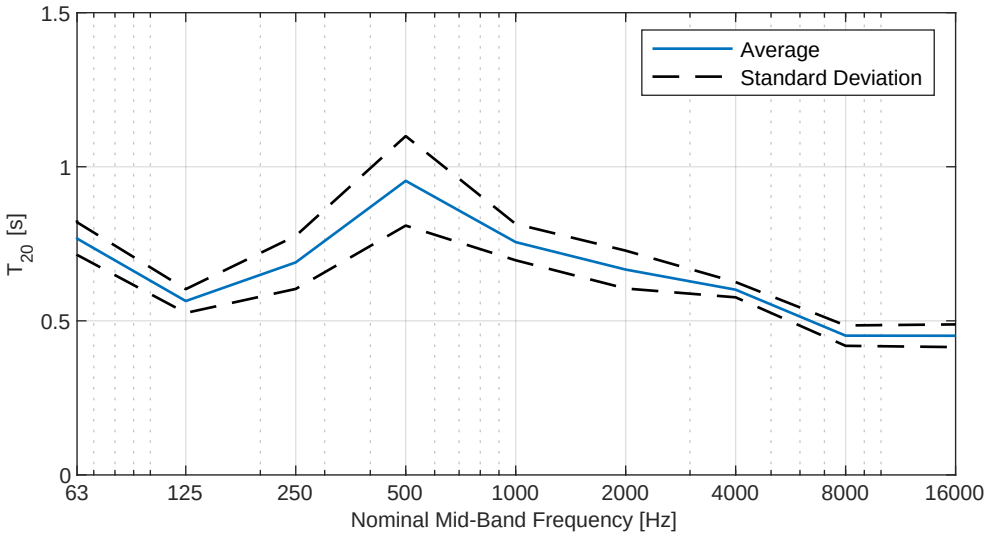


Figure 3.10: T_{20} for the empty listening room.

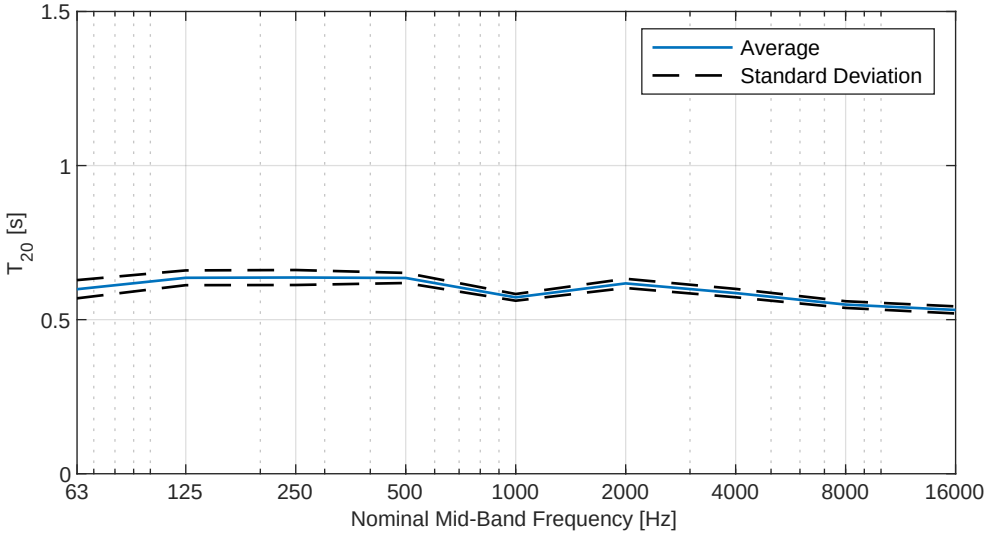


Figure 3.11: T_{20} for the empty listening room.

Larger variations are seen in the physical measurements compared to the simulation, especially in the area above 125 Hz and below 4 kHz. The exact nature of this is not determined, however, several possible sources are identified. Both the loudspeaker and microphones are not perfectly omnidirectional which can result in positional variations and in general the sound field is not completely diffuse at all frequencies, resulting in uneven sound distribution.

With a rough volume estimate of the listening room; $V = 7.79\,\text{m} \times 4.13\,\text{m} \times 2.77\,\text{m} = 89.12\,\text{m}^3$, the Schroeder frequency presented in Eq. (2.6) can be calculated using $T_{20} = 0.69\,\text{s}$ from the full-band IRs:

$$f_{sc} \approx 2000\sqrt{\frac{T}{V}} \approx 2000\sqrt{\frac{0.69}{89.12}} = 175.98\,\text{Hz} \tag{3.1}$$

Even though the density of room modes is relatively high above f_{sc} and as such in the area with a large standard deviation, constructive and destructive interference still have an impact at specific measurement positions. It should also be noted that the Schroeder frequency might be overestimated since a large part of the volume above the suspended ceiling is excluded.

4 Conclusion

The purpose of this study is to investigate the feasibility of using off-the-shelf equipment as an acoustic simulation tool. This project's findings show that the selected combination of hardware and software is not capable of providing sufficiently accurate auralisations for this to be used as a professional tool. However, the Quest 2 virtual reality (VR) headset is well suited for the use case as demonstrated by others.

State-of-the-art room acoustic simulation methods have been identified through comprehensive literature studies and the most suitable for real-time VR implementation are presented in this report. A great understanding of the development platform is obtained through extensive investigations of and experiments with various freely available software development kits (SDKs) and application programming interfaces (APIs).

A virtual model and acoustic simulation of a listening room have successfully been implemented, applying the knowledge gained from the research and initial experiments. The result is a comparison of simulated acoustic measurements with physical ones. Even though the capabilities of the implementation did not match the expectations, it still succeeds as a proof of concept with room for improvement.

While Steam Audio and similar SDKs improve the realism of sound propagation within game development, its purpose is to support the visuals of a virtual environment rather than replicate the real world as closely as possible. With this being said, Steam Audio builds on top of a well-functioning native audio implementation in Unity. It is possible to access and modify the audio stream at any point to implement and insert custom algorithms into the pipeline.

One thing is developing the algorithms but another challenge is the integration into the workflow in the architecture, engineering and construction (AEC) industry. Importing existing architectural computer aided design (CAD) models and inclusion of acoustic properties in the building information modelling (BIM) appears a necessity.

This leaves several new questions to be answered: What modifications are needed to use the CAD model for acoustic simulation? What information is available in the BIM and how to apply this? Is the available data sufficient for proper acoustic simulation or are further details needed and/or wanted? Last but not least: How is all of this integrated into the existing processes and are there actual benefits from including these interactive auralisations?

In conclusion, further work is needed to succeed, but this project demonstrates the potential of using inexpensive hardware and freely available software for simulating room acoustics in VR.

Acronyms

3D three-dimensional 2, 12, 27, 36
6DoF six degrees of freedom 26

AEC architecture, engineering and construction ii, 2, 20
AES Audio Engineering Society 14
API application programming interface ii, 9, 20
AR augmented reality 27

BEM boundary element method 4
BIM building information modelling 2, 20, 27
BRIR binaural room impulse response 8

CAD computer aided design 2, 20
CPU central processing unit 14

FDM finite-difference method 3
FEM finite element method 3

GA geometrical acoustics 3, 4, 6, 7

HATS head and torso simulator 16, 51
HMD head-mounted display 26
HRTF head-related transfer function 8, 9, 14, 16, 17, 30, 32, 33, 51, 52

IEQ indoor environmental quality 2
IR impulse response ii, 4, 8, 10, 13, 14, 16, 19, 21, 38
IS image-source 5–7

LTI linear time-invariant 8

PC personal computer 26

REW Room EQ Wizard 35, 41
RIR room impulse response ii, 4, 8–10, 16, 38, 45

SDK software development kit ii, 9, 13, 20, 26, 27, 32, 33

SNR signal-to-noise ratio 42, 47

SOFA spatially oriented format for acoustics 14, 16, 32, 33, 51

VR virtual reality ii, 2, 9, 20, 26, 27

Bibliography

[1] A. Sidani, F. M. Dinis, L. Sanhudo *et al.*, 'Recent tools and techniques of BIM-based virtual reality: A systematic review,' *Archives of Computational Methods in Engineering*, no. 2, pp. 449–462, Mar. 2021. DOI: 10.1007/s11831-019-09386-0.

[2] M. Vorländer, D. Schröder, S. Pelzer and F. Wefers, 'Virtual reality for architectural acoustics,' *Journal of Building Performance Simulation*, no. 1, pp. 15–25, 2nd Jan. 2015. DOI: 10.1080/19401493.2014.888594.

[3] J. Reinten, P. E. Braat-Eggen, M. Hornikx, H. S. Kort and A. Kohlrausch, 'The indoor sound environment and human task performance: A literature review on the role of room acoustics,' *Building and Environment*, pp. 315–332, Oct. 2017. DOI: 10.1016/j.buildenv.2017.07.005.

[4] Bolig- og Planstyrelsen. 'BR18 - lydforhold,' bygningsreglementet.dk, [Online]. Available: https://bygningsreglementet.dk/Tekniske-bestemmelser/17/Vejledninger/Generel_vejledning/Referenceliste (visited on 23/11/2022).

[5] U.S. Green Building Council. 'LEED - acoustic performance,' usgbc.org, [Online]. Available: https://www.usgbc.org/credits/eq10 (visited on 23/11/2022).

[6] Building Research Establishment Ltd. 'BREEAM - acoustic performance,' bregroup.com, [Online]. Available: https://kb.breeam.com/wp-content/plugins/breeamkb-pdf/pdf/?c=155 (visited on 23/11/2022).

[7] WELL Inc. 'WELL - sound,' v2.wellcertified.com, [Online]. Available: https://v2.wellcertified.com/en/wellv2/sound (visited on 23/11/2022).

[8] S. S. Wyke, K. Svidt, F. Christensen, J. B. Sørensen, T. M. Fadnes and R. L. Jensen, 'Real-time evaluation of room acoustics using IFC-based virtual reality and auralization engines,' presented at the 37th CIB W78 Information Technology for Construction Conference, 2020, pp. 279–290. DOI: 10.46421/2706-6568.37.2020.paper020.

[9] Odeon A/S. 'ODEON 17,' odeon.dk, [Online]. Available: https://odeon.dk (visited on 18/11/2022).

[10] S. Pelzer, L. Aspöck, D. Schröder and M. Vorländer, 'Integrating real-time room acoustics simulation into a CAD modeling software to enhance the architectural design process,' *Buildings*, no. 2, pp. 113–138, 21st Apr. 2014. DOI: 10.3390/buildings4020113.

[11] Treble Inc. 'Better sounding buildings,' treble.tech, [Online]. Available: https://www.treble.tech/buildings (visited on 18/11/2022).

[12] F. Pind, J. F. Einarsson, S. Guðjónsson *et al.*, 'A novel wave-based virtual acoustics and spatial audio framework,' presented at the AES 2022 International Audio for Virtual and Augmented Reality Conference, Aug. 2022.

[13] TechTarget. 'What is the metaverse? an explanation and in-depth guide,' techtarget.com, [Online]. Available: https://www.techtarget.com/whatis/feature/The-metaverse-explained-Everything-you-need-to-know (visited on 28/11/2022).

[14] M. Vorländer, *Auralization*. Springer, 2008. DOI: 10.1007/978-3-540-48830-9.

[15] L. Savioja and U. P. Svensson, 'Overview of geometrical room acoustic modeling techniques,' *The Journal of the Acoustical Society of America*, no. 2, pp. 708–730, 2015. DOI: 10.1121/1.4926438.

[16] H. Kuttruff, *Room Acoustics*, 6th ed. CRC Press, 2016. DOI: 10.1201/9781315372150.

[17] F. A. Everest and K. C. Pohlmann, *Master Handbook of Acoustics*, 6th ed. McGraw-Hill Education, 2015.

[18] Z. Maekawa, J. H. Rindel and P. Lord, *Environmental and Architectural Acoustics*, 2nd ed. Spon Press, 2011.

[19] B. Xie, *Head-Related Transfer Function and Virtual Auditory Display*, 2nd ed. J. Ross Publishing, 2013.
[20] K. Meesawat and D. Hammershøi, 'The time when the reverberation tail in a binaural room impulse response begins,' in *Audio Engineering Society Convention 115*, Oct. 2003.
[21] K. H. Kuttruff, 'Auralization of impulse responses modeled on the basis of ray-tracing results,' in *Audio Engineering Society Convention 91*, Oct. 1991.
[22] Meta. 'Quest 2,' meta.com, [Online]. Available: https://www.meta.com/dk/en/quest/products/quest-2/ (visited on 18/11/2022).
[23] Epic Games, Inc. 'Download unreal engine,' unrealengine.com, [Online]. Available: https://www.unrealengine.com/en-US/download (visited on 25/11/2022).
[24] Unity Technologies. 'Download unity,' unity.com, [Online]. Available: https://unity.com/download (visited on 24/11/2022).
[25] Meta. 'Meta spatializer plugin for unity,' developer.oculus.com, [Online]. Available: https://developer.oculus.com/documentation/unity/audio-osp-unity-req-setup/ (visited on 25/11/2022).
[26] Valve Corporation. 'Steam audio unity integration,' valvesoftware.github.io, [Online]. Available: https://valvesoftware.github.io/steam-audio/doc/unity/index.html (visited on 25/11/2022).
[27] Open Source (Google). 'Resonance audio in unity,' resonance-audio.github.io, [Online]. Available: https://resonance-audio.github.io/resonance-audio/develop/unity/getting-started (visited on 25/11/2022).
[28] Dear Reality. 'dearVR UNITY,' dear-reality.com, [Online]. Available: https://www.dear-reality.com/products/dearvr-unity (visited on 25/11/2022).
[29] F. Brinkmann, L. Aspöck, D. Ackermann, S. Lepa, M. Vorländer and S. Weinzierl, 'A round robin on room acoustical simulation and auralization,' *The Journal of the Acoustical Society of America*, no. 4, pp. 2746–2760, Apr. 2019. DOI: 10.1121/1.5096178.
[30] F. Brinkmann, L. Aspöck, D. Ackermann, R. Opdam, M. Vorländer and S. Weinzierl, 'A benchmark for room acoustical simulation. concept and database,' *Applied Acoustics*, p. 107 867, May 2021. DOI: 10.1016/j.apacoust.2020.107867.
[31] CEI/IEC 60268-13, *Sound system equipment – part 13: Listening tests on loudspeakers*, 1998.
[32] Aalborg University. 'Listening room b4-107,' acoustics.aau.dk, [Online]. Available: http://acoustics.aau.dk/facilities/fac/listening-room/ (visited on 13/12/2022).
[33] DS/EN ISO 354, *Acoustics - measurement of sound absorption in a reverberant room*, 2003.
[34] Meta. 'Oculus app development in unity,' developer.oculus.com, [Online]. Available: https://developer.oculus.com/documentation/unity (visited on 25/11/2022).
[35] Meta. 'Oculus integration SDK,' developer.oculus.com, [Online]. Available: https://developer.oculus.com/downloads/package/unity-integration/ (visited on 25/11/2022).
[36] Valve Corporation. 'Steam audio discussions,' steamcommunity.com, [Online]. Available: https://steamcommunity.com/app/596420/discussions/0/3493131640449268018/ (visited on 25/11/2022).
[37] P. Majdak, Y. Iwaya, T. Carpentier *et al.*, 'Spatially oriented format for acoustics: A data exchange format representing head-related transfer functions,' in *Audio Engineering Society Convention 134*, May 2013.
[38] MathWorks. 'Download matlab,' se.mathworks.com, [Online]. Available: https://se.mathworks.com/downloads/ (visited on 14/12/2022).
[39] AES69-2020, *AES standard for file exchange - spatial acoustic data file format*, 2020.
[40] Audacity. 'Download audacity,' audacityteam.org, [Online]. Available: https://www.audacityteam.org/download/ (visited on 14/12/2022).
[41] Existential Audio. 'Download BlackHole,' existential.audio, [Online]. Available: https://existential.audio/blackhole/ (visited on 14/12/2022).
[42] DS/EN ISO 3382-2, *Acoustics - measurement of room acoustic parameters - part 2: Reverberation time in ordinary rooms*, 2008.
[43] Bang & Olufsen, *Music for archimedes*, 1992.

[44] Brüel & Kjær. 'Sound sources and impact sound source for building acoustics,' bruel.com.pl, [Online]. Available: http://www.bruel.com.pl/wp-content/uploads/2015/10/bp1689.pdf (visited on 17/12/2022).
[45] DS/EN ISO 3382-1, *Acoustics - measurement of room acoustic parameters - part 1: Performance spaces*, 2009.
[46] B. P. Bovbjerg, F. Christensen, P. Minnaar and X. Chen, 'Measuring the head-related transfer functions of an artificial head with a high-directional resolution,' in *Audio Engineering Society Convention 109*, Sep. 2000.
[47] Sony. 'MDR-7506,' pro.sony, [Online]. Available: https://pro.sony/en_DK/products/headphones/mdr-7506 (visited on 27/12/2022).
[48] Apple. 'MacBook air,' apple.com, [Online]. Available: https://www.apple.com/dk/macbook-air-m1/ (visited on 27/12/2022).
[49] Qualcomm Technologies, Inc. 'Oculus quest 2: How snapdragon XR2 powers the next generation of VR,' qualcomm.com, [Online]. Available: https://www.qualcomm.com/news/onq/2020/10/oculus-quest-2-how-snapdragon-xr2-powers-next-generation-vr (visited on 02/12/2022).
[50] Meta. 'Meta quest developer hub,' developer.oculus.com, [Online]. Available: https://developer.oculus.com/documentation/unity/ts-odh/ (visited on 25/11/2022).
[51] Unity Technologies. 'Unity reflect,' unity.com, [Online]. Available: https://unity.com/products/unity-reflect (visited on 12/11/2022).
[52] Unity Technologies. 'Unity manual,' docs.unity3d.com, [Online]. Available: https://docs.unity3d.com/Manual/index.html (visited on 24/11/2022).
[53] John Mulcahy. 'Download REW,' roomeqwizard.com, [Online]. Available: https://www.roomeqwizard.com (visited on 14/12/2022).
[54] Brackeys. 'FIRST PERSON MOVEMENT in unity - FPS controller,' youtube.com, [Online]. Available: https://www.youtube.com/watch?v=_QajrabyTJc (visited on 08/12/2022).

A Platform Description

A.1 Meta Quest 2

Meta (formerly Oculus) released their second version in the Quest series in October 2020. The Quest 2 is a powerful all-in-one virtual reality (VR) headset that can run standalone or connected to a personal computer (PC). It consists of a head-mounted display (HMD) with close-ear loudspeakers and two touch controllers as seen in Fig. A.1.1. No external sensors are needed since the headset itself tracks the movement of both head and body with six degrees of freedom (6DoF), i.e. three translational and three rotational movements. Additionally, it is possible to connect external headphones through the 3.5-mm audio port or wireless with Bluetooth [22].

Quest 2 is powered by the Snapdragon XR2 chipset, specifically designed VR headsets [49]. Unfortunately documentation for this is not openly available.

(a) VR headset **(b)** Two touch controllers

Figure A.1.1: Meta Quest 2 [22].

Oculus/Meta provides several development tools such as the Meta Quest Development Hub [50] (shown in Fig. A.1.2) and Oculus Integration software development kit (SDK) [35] alongside useful documentation and guides [34]. The development hub enables wireless communication for application deployment, on-device metrics and screen casting to name just a few features.

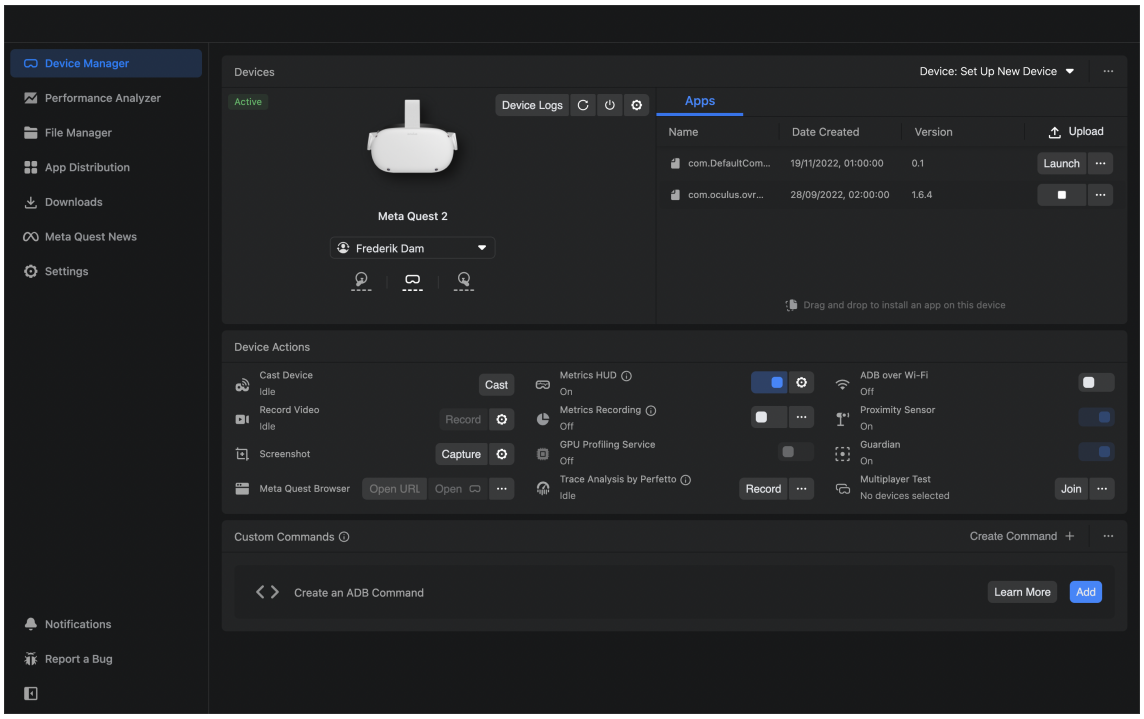


Figure A.1.2: Meta Quest Developer Hub 3.0.4.

Basic tracking with character movement and interactions are included as *Prefab*s in the integration SDK which makes it relatively easy to get started with Quest 2 development in Unity.

A.2 Unity

Unity is a professional real-time three-dimensional (3D) development platform supporting Windows, Mac and Linux. It allows for building and deploying content across all major AR, VR, mobile, desktop, and console platforms. The ecosystem around Unity provides documentation, guides and communities to support developers [24].

A real-time building information modelling (BIM) viewer called Unity Reflect [51] is available, however, it is hidden behind a payment wall and focuses on visuals.

This section describes the most important parts used in this project, but detailed information is available in the online Unity Manual [52].

Hub and Editor

Everything starts at the Unity Hub (see Fig. A.2.1), where projects are managed and Editor versions are installed.

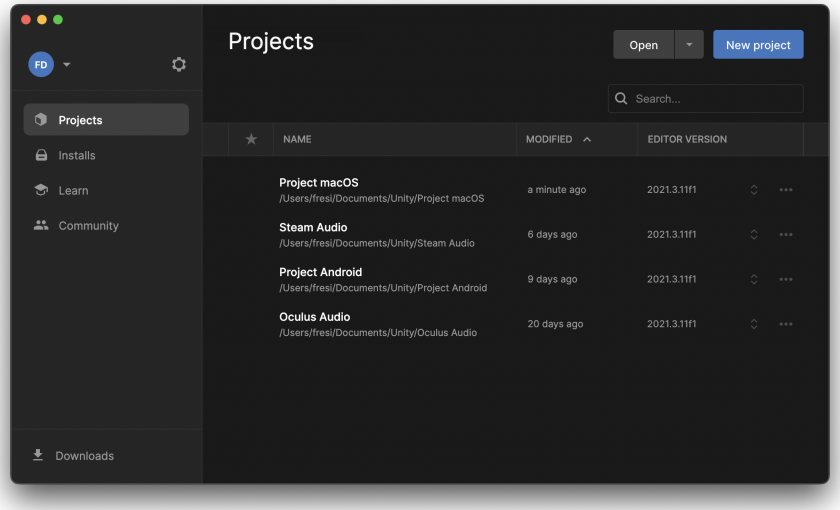


Figure A.2.1: Unity Hub 3.3.0.

New projects are created, imported and launched from the Hub which opens up the Unity Editor (see Fig. A.2.2). A note to the Editor is that the axes in the coordinate system are deviating as y and z are swapped.

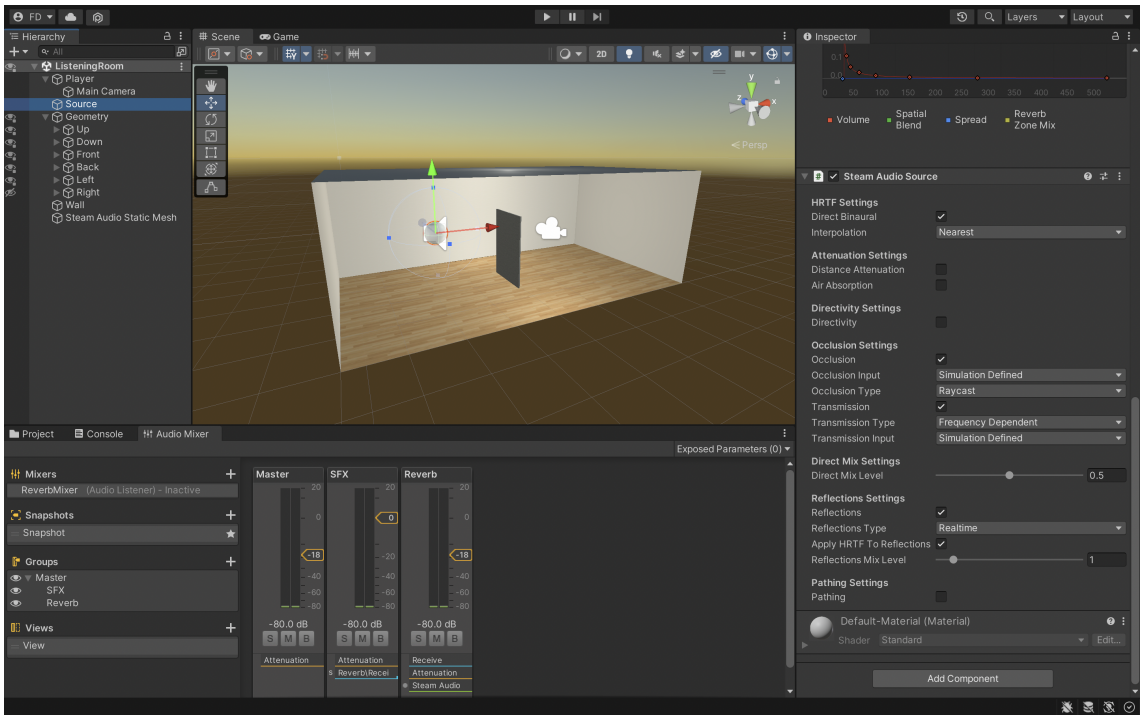


Figure A.2.2: Unity Editor 2021.3.11f1.

The first thing to note is the *Hierarchy* (see Fig. A.2.3) containing the current *Scene* (ListeningRoom) and all the *GameObject*s within.

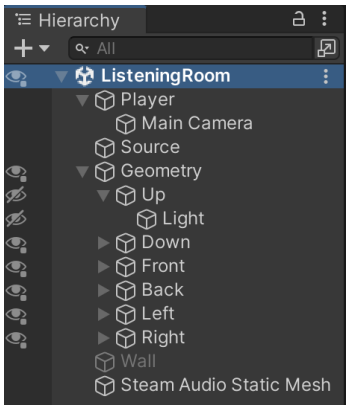


Figure A.2.3: Hierarchy window in Unity.

Highlighting a *GameObject* in the *Hierarchy* enables editing of its properties in the *Inspector* (see Fig. A.2.4). Here it is possible to apply transformation and add components such as custom scripts.

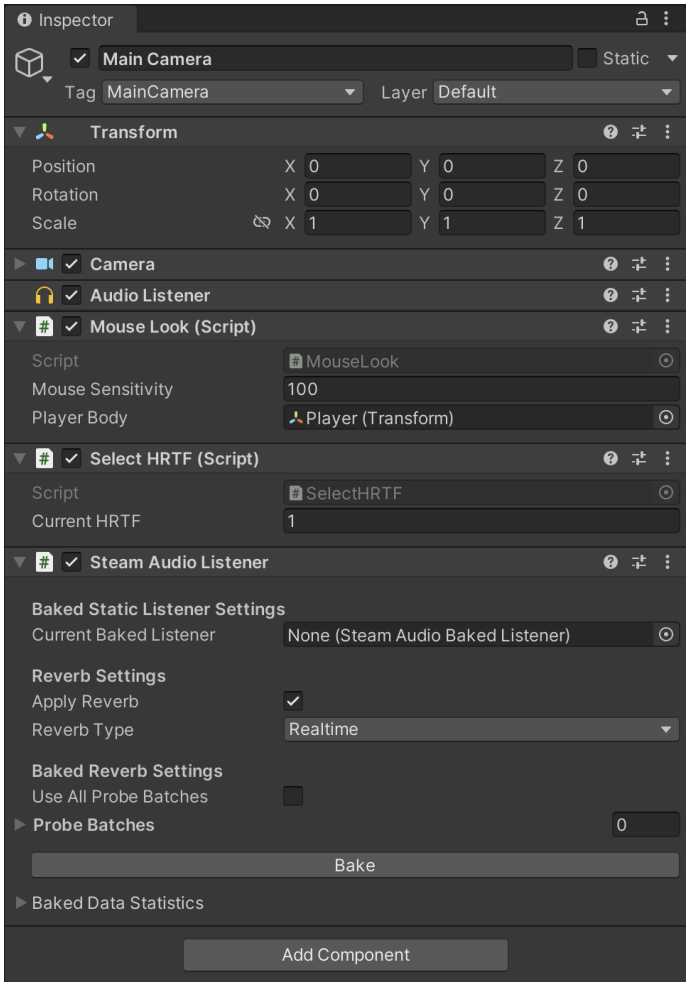


Figure A.2.4: Inspector window in Unity.

All included *Assets* and *Packages* are available through the *Project* window (see Fig. A.2.5). *Assets* include audio files, materials, scenes, scripts, etc. and can be either custom created or imported from external sources.

Figure A.2.5: Project window in Unity.

The *Console* window can be used to view warnings and errors when building as well as in-game logging information such as the currently active set of head-related transfer functions (HRTFs).

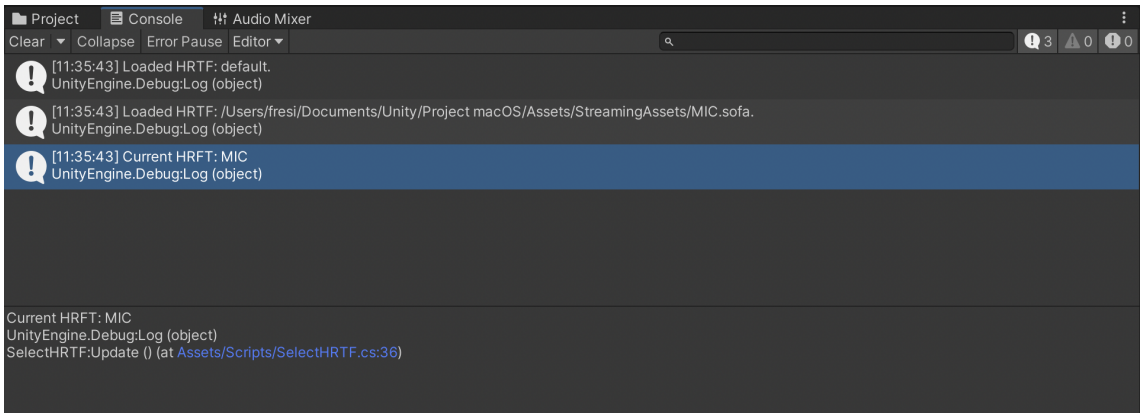


Figure A.2.6: Console window in Unity.

In the *Audio Mixer* window it is possible to mix various *Audio Sources* and apply effects to different mixer groups.

Figure A.2.7: Audio Mixer window in Unity.

Audio

The overall audio signal flow in Unity is illustrated in Fig. A.2.8. It is possible to access and manipulate this using native audio plugins (C++) and/or custom scripts (C#).

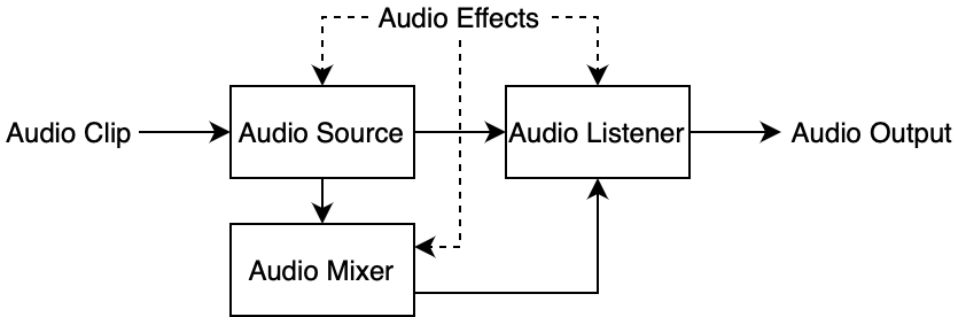


Figure A.2.8: General audio signal flow in Unity.

The *Audio Clip* contains the audio data to be played back by the *Audio Source* (see Fig. A.2.9). *Audio Effects* can be attached directly to either an *Audio Source* (pre-effects) or an *Audio Listener* (post-effects) as general components or custom scripts. The output of an *Audio Source* can either be routed directly to the *Audio Listener* or through an *Audio Mixer* with additional effects and plug-ins.

Multiple instances of *Audio Clips*, *Audio Sources* and *Audio Mixers* can be handled. However, each *Scene* is limited to a single *Audio Listener*. From the *Audio Listener*, often attached to the *Main Camera*, the *Audio Output* is played back through the sound card and connected speakers or headphones.

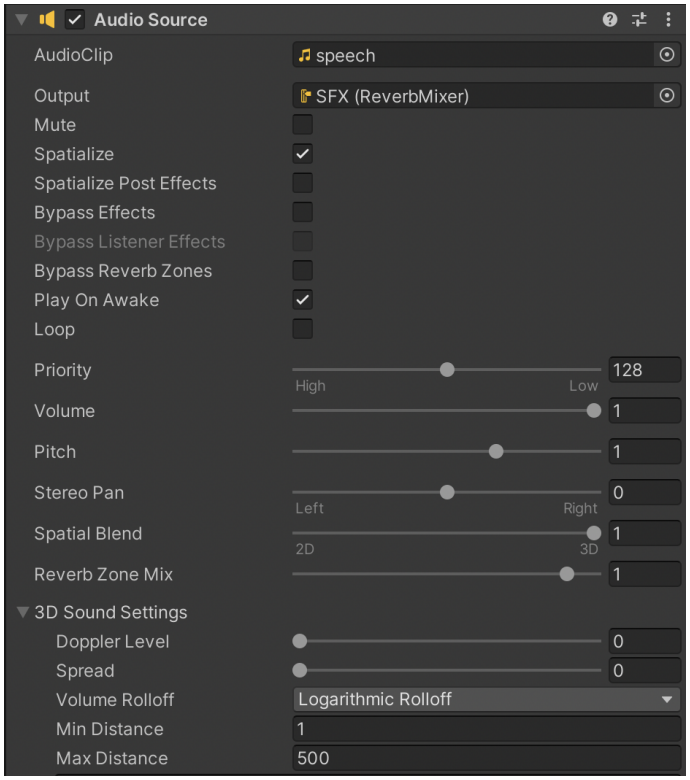


Figure A.2.9: Audio Source component in Unity.

A.3 Steam Audio SDK

The Steam Audio Unity Integration is an engine-specific implementation of Valve's cross-platform spatial audio SDK. It offers a wide variety of features for immersive positional and environmental audio effects [26].

This section will create an overview of the specific features used in this project.

Settings

Steam Audio Settings (see Fig. A.3.1) is the main *Asset* for modifying the audio plugin. It can be accessed through the Steam Audio toolbar together with other project-wide features. Many settings are available but in this project, only the HRTF and real-time reflection settings are modified. This is where the spatially oriented format for acoustics (SOFA) files are included and ray-casting settings are modified.

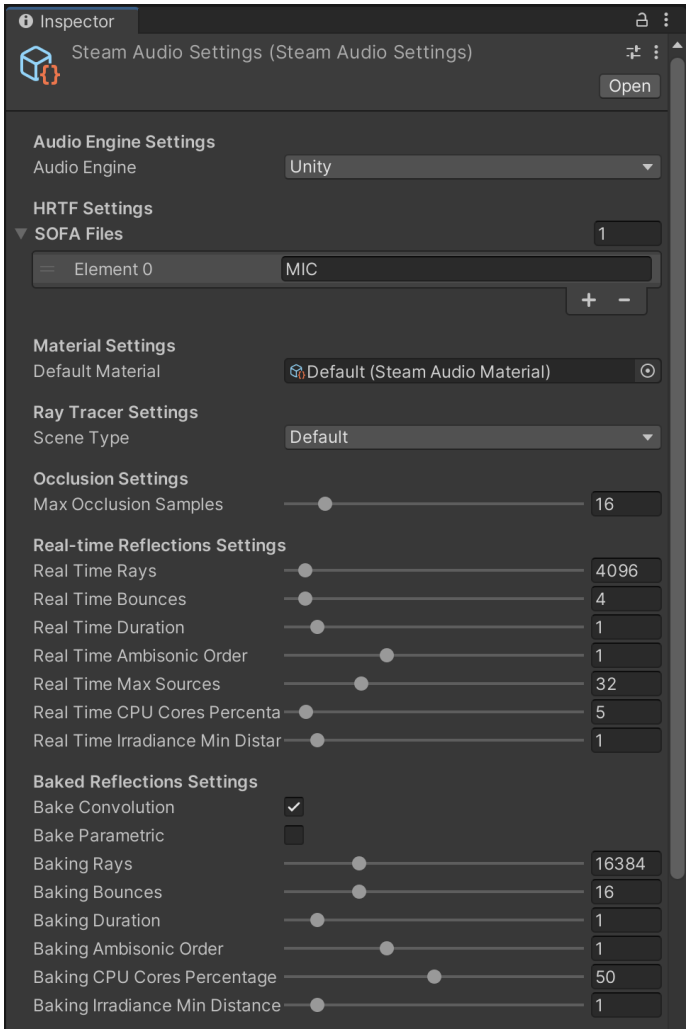


Figure A.3.1: Steam Audio Settings component in Unity.

Source and Listener

The *Steam Audio Source* component (see Fig. A.3.2) can be attached as an addition to the native *Audio Source*. It overrides some of the settings and allows for further modifications as well as some additional features. Mostly attenuation, occlusion and reflections have been used in this project.

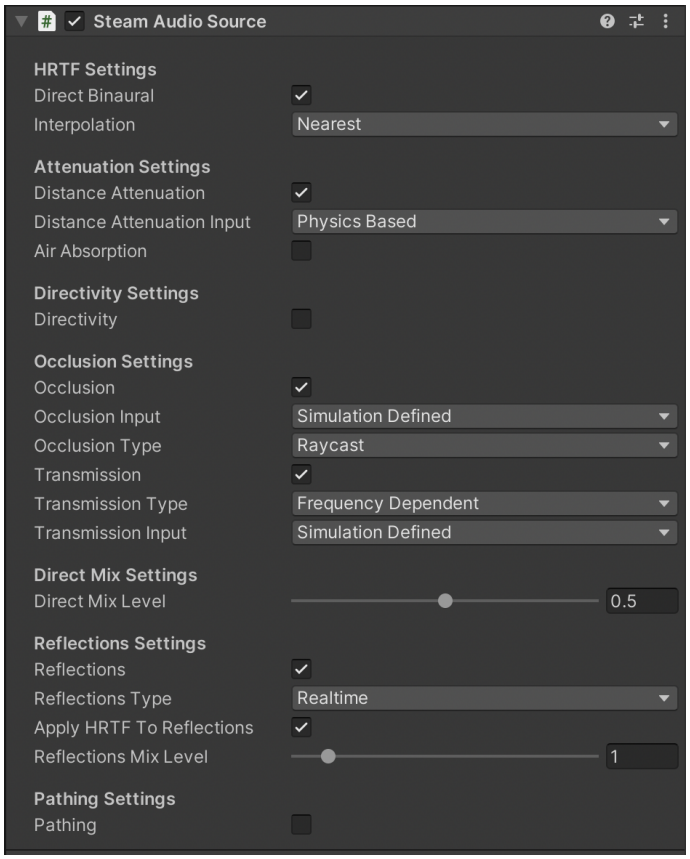


Figure A.3.2: Steam Audio Source component in Unity.

Similarly, the *Steam Audio Listener* is an addition to the generic *Audio Listener* where it is possible to add physics-based reverb at the listener's position.

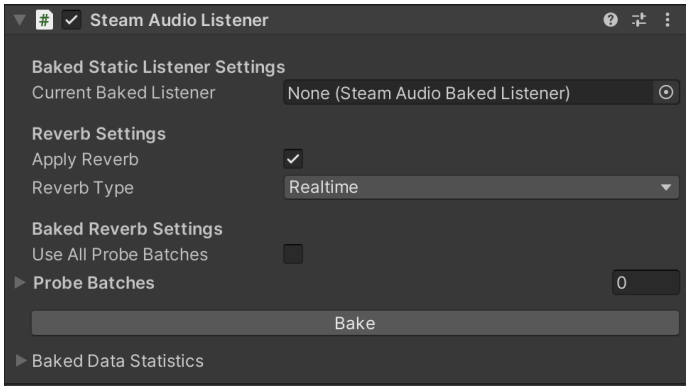


Figure A.3.3: Steam Audio Listener component in Unity.

Custom HRTFs

A very important feature of the Steam Audio SDK is the ability to use custom HRTFs and the support for SOFA files. This allows both for exchanging HRTFs in auralisation and bypassing these filters for virtual measurements.

In order to change the active SOFA file, a custom script (see Appx. D.2) has been made and included as a component called *Select HRTF (Script)* (see Fig. A.3.4). This allows for selecting the HRTF before compiling but also during run-time.

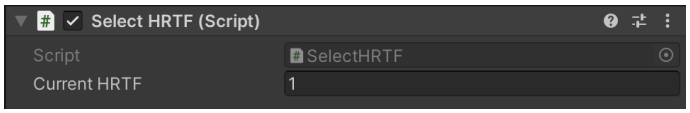


Figure A.3.4: Select HRTF (Script) component in Unity.

Room Simulation

Another important aspect of this project is the Steam Audio implementation of geometry-based room simulation. This is used by adding a *Steam Audio Geometry* component (see Fig. A.3.5) to a geometry object. It is then possible to select a specific material for the given object.

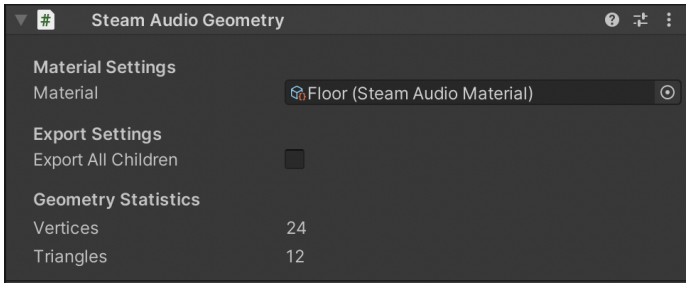


Figure A.3.5: Steam Audio Geometry component in Unity.

Afterwards, it is necessary to Export Active Scene from the Steam Audio toolbar which creates a static mesh and a *Steam Audio Static Mesh* object with the component and asset applied (see Fig. A.3.6).

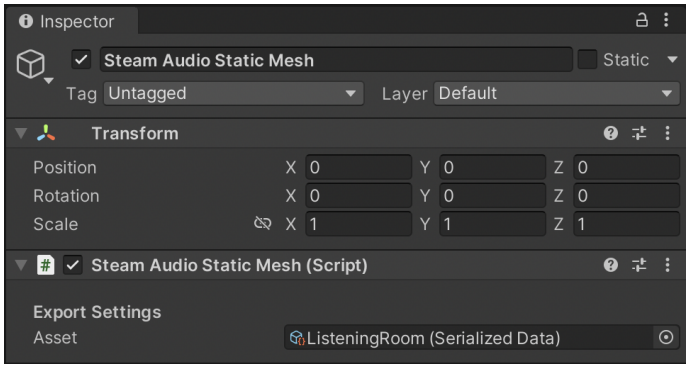


Figure A.3.6: Steam Audio Static Mesh (Script) component in Unity.

Steam Audio provides a selection of *Steam Audio Material*s, however, it is also possible to create custom materials (see Fig. A.3.7). Specific coefficient values from 0 to 1 can be set for absorption, scattering and transmission.

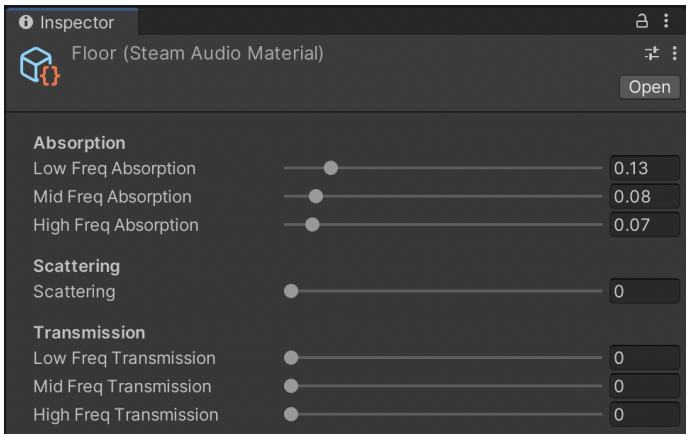


Figure A.3.7: Steam Audio Material component in Unity.

Both the absorption and transmission coefficients are divided into three frequency bands allowing for frequency-dependent simulation. The three frequency bands are defined in the documentation as listed:

- Low < 800 Hz
- Mid = 800 Hz - 8 kHz
- High > 8 kHz

These filter bands have been verified using white periodic noise in Unity routed through BlackHole into a spectrum analyser in Room EQ Wizard (REW) [53].

B Implementation Details

B.1 Project Configuration

This section presents a step-by-step description of the general setup used for the implementation of room simulation in this project. The following procedure provides a brief description of each step:

1. Create empty 3D project in Unity.
2. Import the *Steam Audio Unity Plugin 4.1.1* asset as a custom package.
3. Select *Steam Audio Spatializer* as *Spatializer Plugin* in the project settings.
4. Add the MIC.sofa in the *Steam Audio Settings*.
5. Build geometry using the 3D objects called *Cube*s.
6. Attach a *Steam Audio Geometry* as a *Component* to all geometry objects.
7. Create custom *Steam Audio Material*s and apply these to the corresponding geometries.
8. Create a *Sphere* object and add an *Audio Source* as a *Component*. The following settings are used:
   - AudioClip = "sweep"
   - Spatialize is checked
   - Output = "Master (ReverbMixer)"
   - Spatial Blend = 1
   - Doppler Level = 0
9. Attach a *Steam Audio Source* as a *Component* to the audio source. The following settings deviate from the defaults:
   - HRTF Settings
     - Direct Binaural is checked
     - Interpolation = Nearest
   - Occlusion Settings
     - Occlusion is checked
     - Occlusion Input = Simulation Defined
     - Occlusion Type = Raycast
     - Transmission is checked
     - Transmission Type = Frequency Dependent
     - Transmission Input = Simulation Defined
   - Reflection Settings

- Reflections is checked
- Reflections Type = Realtime
- Apply HRTF To Reflections is checked
- Reflections Mix Level = 1

10. Attach the custom *Select HRTF (Script)* to the *Main Camera* with the *Audio Listener* component.
11. Select *Export Active Scene* from the *Steam Audio* toolbar.

C Measurements

C.1 Physical Listening Room

The purpose of these measurements is to collect room impulse responses (RIRs) of three different setups in the listening room (B4-107). Formalities are presented in Tab. C.1.1 below.

Date	2022-11-21
Location	Listening room (B4-107)
Address	Fredrik Bajers Vej 7B, 9220 Aalborg Ø
Participant(s)	Frederik Sidenius Dam

Table C.1.1: Formalities.

Equipment

The relevant equipment used is listed below in Tab. C.1.2 below.

Brand	Model	Description	AAU/Serial No.
Apple	MacBook Air M1	Laptop	C02GHVURQ6L4
MathWorks	MATLAB R2022b	Software	-
MathWorks	Audio Toolbox 3.3	Software	-
John Mulcahy	REW V.5.20.13	Software	-
RME	Fireface USB Settings	Software	-
RME	TotalMix FX	Software	-
RME	Fireface UFX II	Audio interface	AAU108227
Pioneer	A-656	Power amplifier	AAU08700
Brüel & Kjær	OmniSource 4295	Loudspeaker	AAU86831
G.R.A.S.	26CC+40AZ	Microphone	AAU75565
G.R.A.S.	26CC+40AZ	Microphone	AAU75566
G.R.A.S.	26CC+40AZ	Microphone	AAU75567
G.R.A.S.	26CC+40AZ	Microphone	AAU75568
MONACOR	SM-4	SPL meter	AAU02126-03
Leica	Disto D2	Distance meter	AAU02157-64

Table C.1.2: Equipment used.

Setup

The measurement setup consists of an omnidirectional loudspeaker (the source) and four microphones (the receivers). These are all connected to the laptop through an audio interface and a power amplifier is used with the loudspeaker. An illustration of this can be seen in Fig. C.1.1 below.

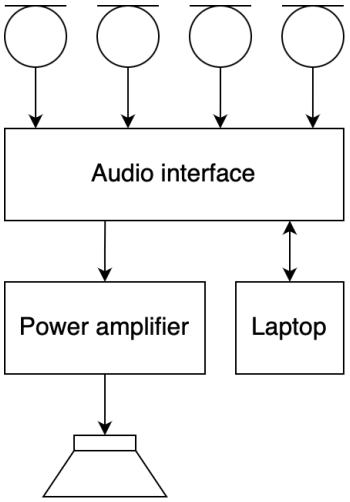


Figure C.1.1: Measurement setup.

A sketch of the listening room including dimensions as well as source and receiver positions is provided in Fig. C.1.2. Furthermore, an acoustic partitioning wall is placed upright in the centre of the room which is also included in the sketch. The wall is elevated 0.025 m above the ground and the outer dimensions are 1.22x1.75x0.03 m. The dashed line above the ceiling indicates the actual roof, whereas the solid lines represent the suspended ceiling.

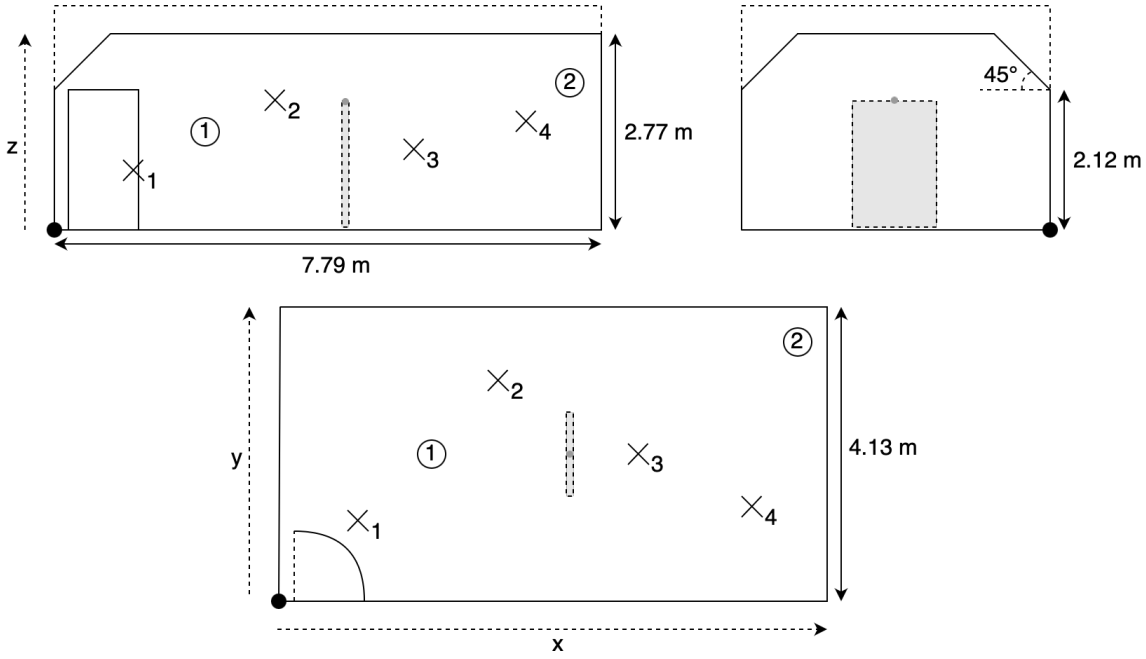


Figure C.1.2: Listening room B4-107 seen from the side, end and top. Origo and wall reference point is marked as well as source (◯) and receiver (X) positions.

The measured positions for the source, receivers and wall are listed in Tab. C.1.3 below.

Name	x [m]	y [m]	z [m]
Source 1	2.10	2.11	1.44
Source 2	7.27	3.59	2.13
Receiver 1	0.97	1.03	1.01
Receiver 2	3.17	3.11	1.78
Receiver 3	5.22	2.11	1.25
Receiver 4	6.56	1.46	1.50
Wall	3.90	2.07	1.78

Table C.1.3: Source, receiver and wall positions.

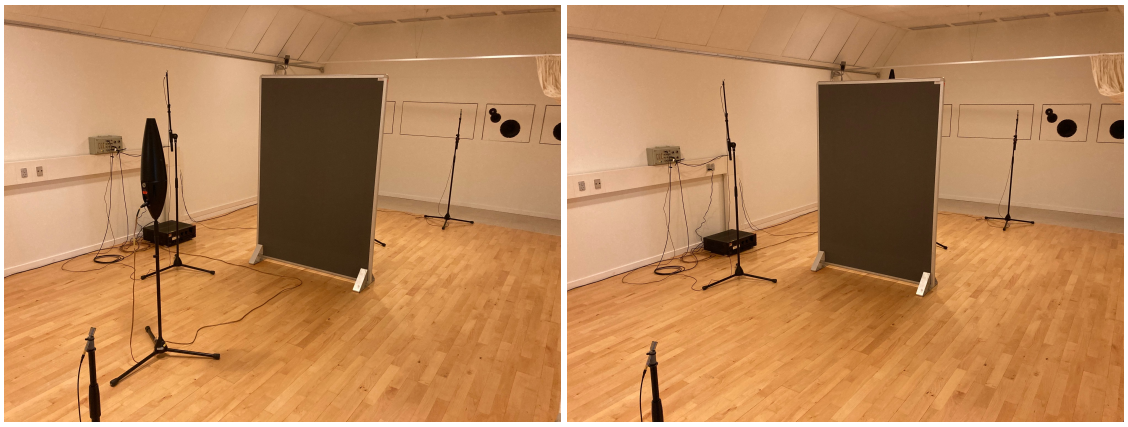
Pictures of the empty listening room can be seen in Fig. C.1.3 and with the wall in Fig. C.1.4 while the standard configured listening room (without carpet) is shown in Fig. C.1.5.

(a) Panorama

(b) Source position 1

(c) Source position 2

Figure C.1.3: Setup in the empty listening room.



(a) Source position 1 **(b)** Source position 2

Figure C.1.4: Setup in the empty listening room with the acoustic partitioning wall.

(a) Panorama

(b) Source position 1 **(c)** Source position 2

Figure C.1.5: Setup in the standard listening room.

Method/Procedure

Test signals are generated using Room EQ Wizard (REW) while Fireface USB settings and TotalMix FX are used for setting up the audio interface. The Impulse Response Measurer from the Audio Toolbox in Matlab is used for measurements and the procedure is listed below:

1. Output level is adjusted to ca. 77 dB(A) re. 20 µPa at 1 m using full-scale band-limited pink noise with a crest factor of 12 dB.
2. Input level is adjusted to min. 3 dB below clipping.

3. Impulse Response Measurer is used with the following settings:
    - Sample rate: 48 kHz
    - Samples per frame: 1024
    - Method: Exponential Swept Sine
    - Number of runs: 4
    - Sweep duration: 6 s
    - End silence duration: 2 s
    - Sweep start frequency: 20 Hz
    - Sweep start frequency: 20 kHz
    - Excitation level: -12 dBFS (peak)
4. Measurements are repeated at two source positions with four microphone positions for all three setups.

Results

The signal-to-noise ratio (SNR), evaluated from the decay of the normalised impulse response before hitting the noise floor, is for all measurements >60 dB.

Example impulse responses and Schroeder energy curves are provided in Fig. C.1.6, C.1.7 and C.1.8 for source position 1 and receiver position 3 in the empty listening room. Similar plots for all source-receiver combinations in all setups can be found in "Attachments/Measurements/Physical".

Octave-band reverberation times (T_{20}) are calculated as described in ISO 3382-2 [42] based on the 8 source-receiver combinations for each setup. This average is provided alongside the standard deviations in Fig. C.1.9, C.1.10 and C.1.11 and a final comparison is given in Tab. C.1.4.

Due to insufficient SNR at 31 Hz this octave band is not included in the calculations.

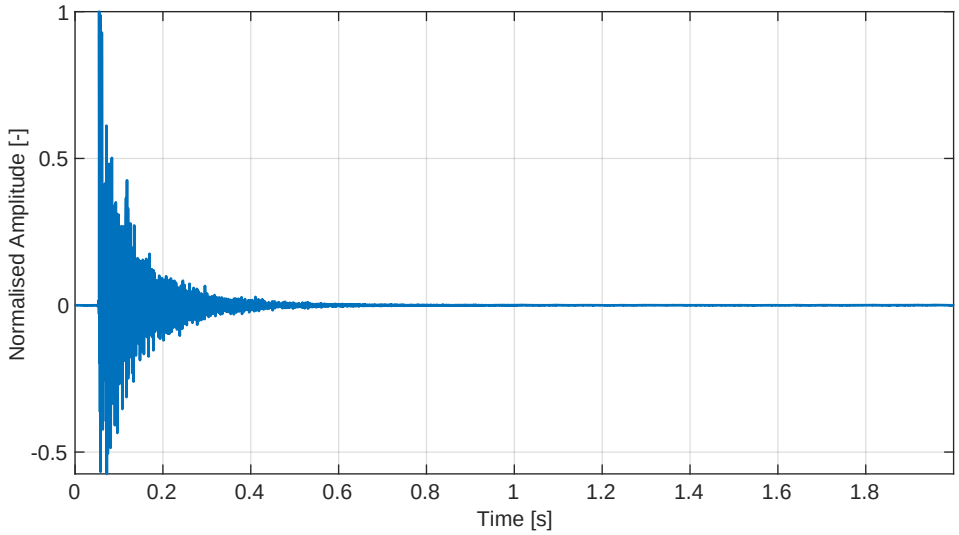


Figure C.1.6: Normalised impulse response.

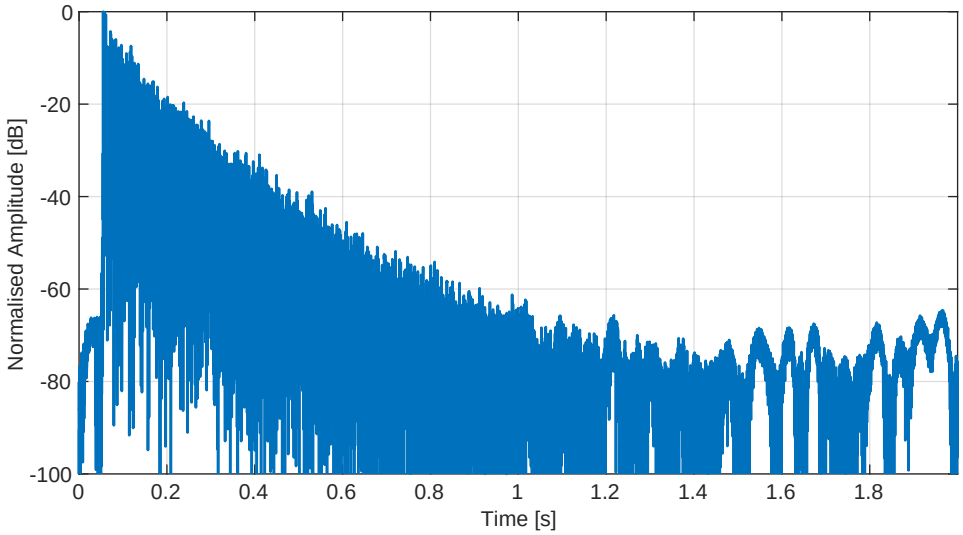


Figure C.1.7: Normalised impulse response in dB.

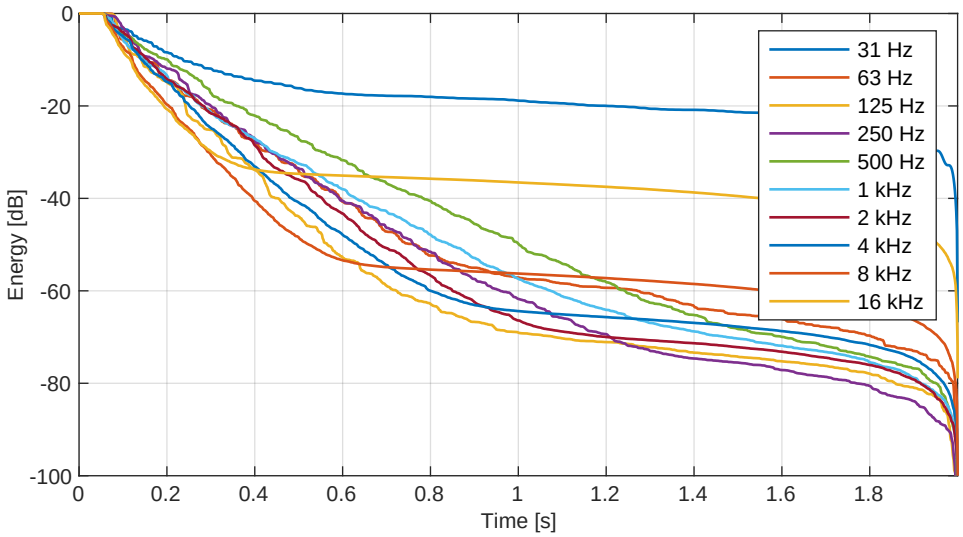


Figure C.1.8: Schroeder energy curves.

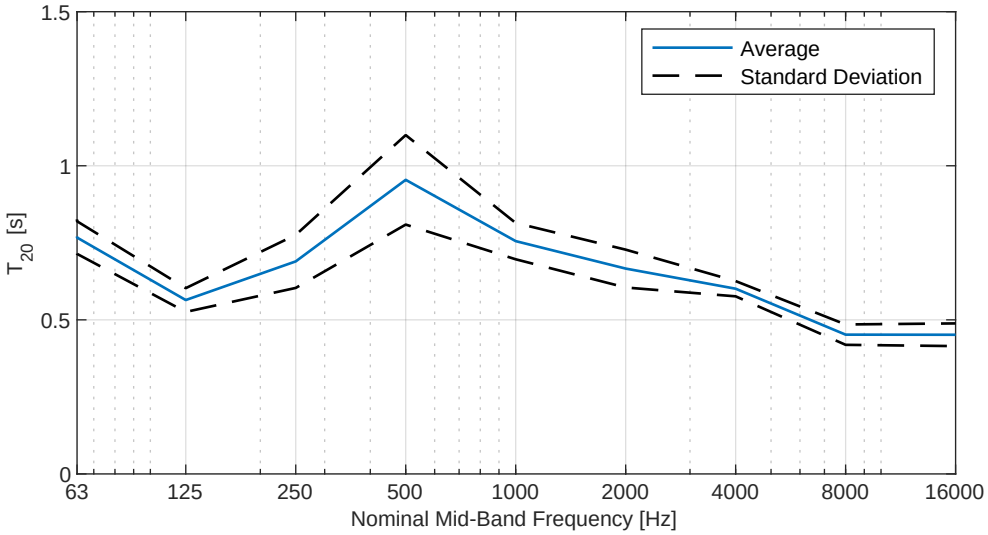


Figure C.1.9: T_{20} for the empty listening room.

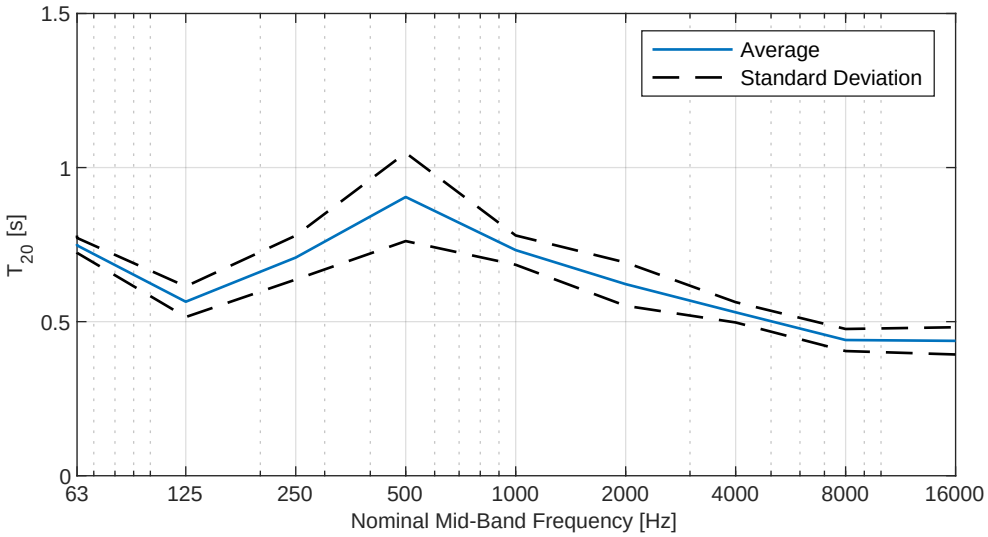


Figure C.1.10: T_{20} for the empty listening room with the acoustic partitioning wall.

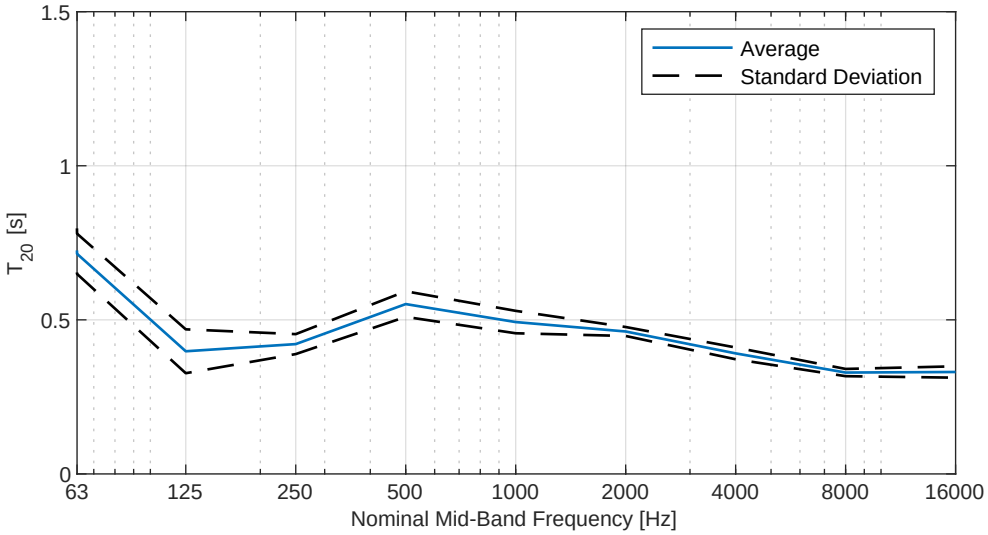


Figure C.1.11: T_{20} for the standard listening room.

Frequency [Hz]	63	125	250	500	1k	2k	4k	8k	16k
Empty [s]	0.77	0.56	0.69	0.95	0.76	0.67	0.60	0.45	0.45
Wall [s]	0.75	0.56	0.71	0.90	0.73	0.62	0.53	0.44	0.44
Standard [s]	0.71	0.40	0.42	0.55	0.49	0.46	0.39	0.33	0.33

Table C.1.4: T_{20} for all three setups.

Sources of Uncertainty

- *Background noise*: Measurements are affected by acoustic background noise which is reduced by increasing excitation level and sweep duration as well as using multiple repeated sweeps.
- *Placement of equipment*: Equipment is placed manually and the positions are measured with a laser distance meter.
- *Environmental variations*: E.g. temperature, humidity and/or pressure changes of the surroundings.
- *Equipment variations*: E.g. temperature changes of equipment when in use, including both fluctuations and steady-state.
- *Equipment tolerances*: Combined tolerances of all equipment, e.g. as stated in the individual data sheets.

C.2 Virtual Listening Room

The purpose of these measurements is to replicate the RIR measurements of two of the setups in the listening room (B4-107) from Appx. C.1 in a simulated virtual environment. Formalities are presented in Tab. C.2.1 below.

Date	2022-12-05
Participant(s)	Frederik Sidenius Dam

Table C.2.1: Formalities.

Equipment

The relevant equipment used is listed below in Tab. C.2.2 below.

Brand	Model	Description	AAU/Serial No.
Apple	MacBook Air M1	Laptop	C02GHVURQ6L4
Unity Technologies	Editor 2021.3.11f1	Software	-
Existential Audio	BlackHole 2ch	Software	-
Audacity	Audacity 3.2.1	Software	-

Table C.2.2: Equipment used.

Setup

The simulated measurement setup in Unity consists of an *Audio Source* (the source) and an *Audio Listener* (the receiver). An excitation signal is attached to the *Audio Source* and the output from the *Audio Listener* is routed to the audio recorder (Audacity) through the virtual audio driver (BlackHole) as shown in Fig. C.2.1 below.

Figure C.2.1: Virtual measurement setup.

A sketch of the simulated listening room including dimensions as well as source and receiver positions is provided in Fig. C.2.2. Furthermore, an acoustic partitioning wall is placed upright in the centre of the room which is also included in the sketch. The wall is elevated 0.025 m above the ground and the outer dimensions are 1.22x1.75x0.03 m. The dashed line above the ceiling indicates the actual roof, whereas the solid lines represent the suspended ceiling.

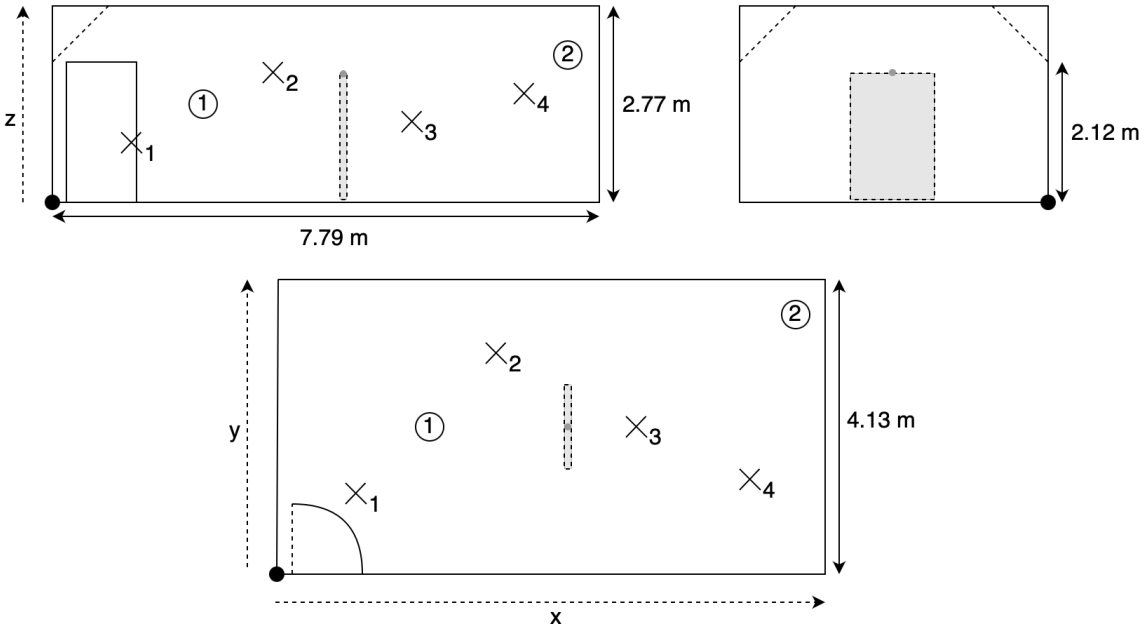


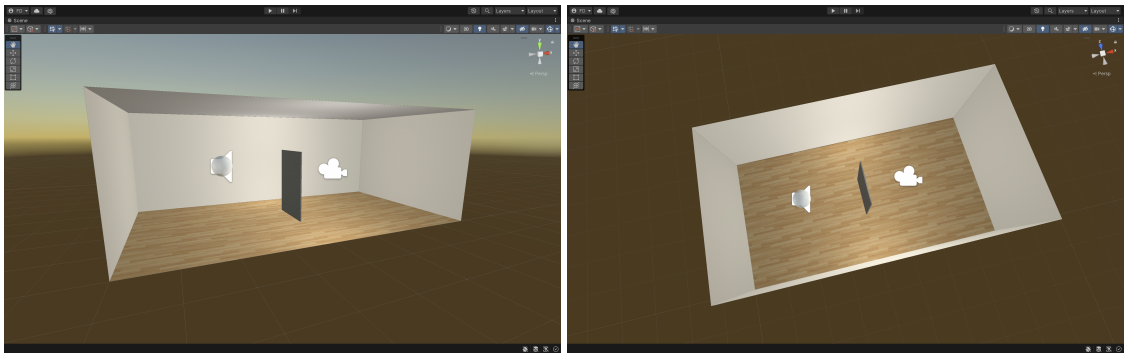
Figure C.2.2: Simulated listening room B4-107 seen from the side, end and top. Origo and wall reference point is marked as well as source (◯) and receiver (X) positions.

The measured positions for the source, receivers and wall are listed in Tab. C.2.3 below.

Name	x [m]	y [m]	z [m]
Source 1	2.10	2.11	1.44
Source 2	7.27	3.59	2.13
Receiver 1	0.97	1.03	1.01
Receiver 2	3.17	3.11	1.78
Receiver 3	5.22	2.11	1.25
Receiver 4	6.56	1.46	1.50
Wall	3.90	2.07	1.78

Table C.2.3: Source, receiver and wall positions.

Screenshots of the listening room simulation can be seen in Fig. C.2.3. The empty setup is identical but with the wall object excluded.



(a) Side view **(b)** Top view

Figure C.2.3: Screenshot of the listening room simulation.

Method/Procedure

The excitation signal is identical to the sweep used in the physical measurements (see Appx. C.1) and is exported from the measurement data. The procedure is listed below:

1. Output level is adjusted in Unity to ensure min. 3 dBFS headroom.
2. The excitation signal is played back and recorded using a warm-up sweep to fill the audio buffer.
3. Measurements are repeated at two source positions and four microphone positions for both setups.

Results

The SNR, evaluated from the decay of the normalised impulse response before hitting the noise floor, is for all measurements >70 dB.

Example impulse responses and Schroeder energy curves are provided in Fig. C.2.4, C.2.5 and C.2.6 for source position 1 and receiver position 3 in the empty listening room. Similar plots for all source-receiver combinations in all setups can be found in "Attachments/Measurements/Simulation".

Octave-band reverberation times (T_{20}) are calculated as described in ISO 3382-2 [42] based on the 8 source-receiver combinations for each setup. This average is provided alongside the standard deviations in Fig. C.2.7 and C.2.8 and a final comparison is given in Tab. C.2.4.

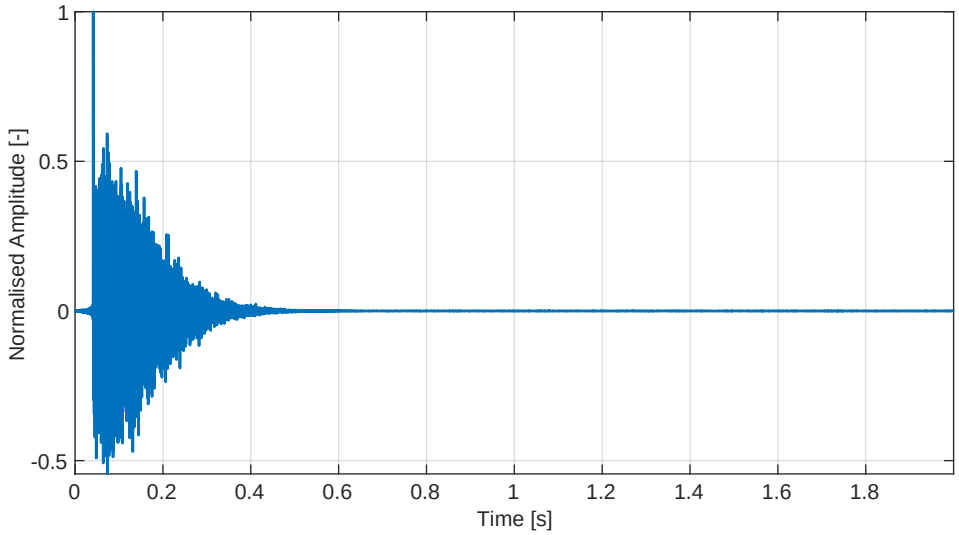


Figure C.2.4: Normalised impulse response.

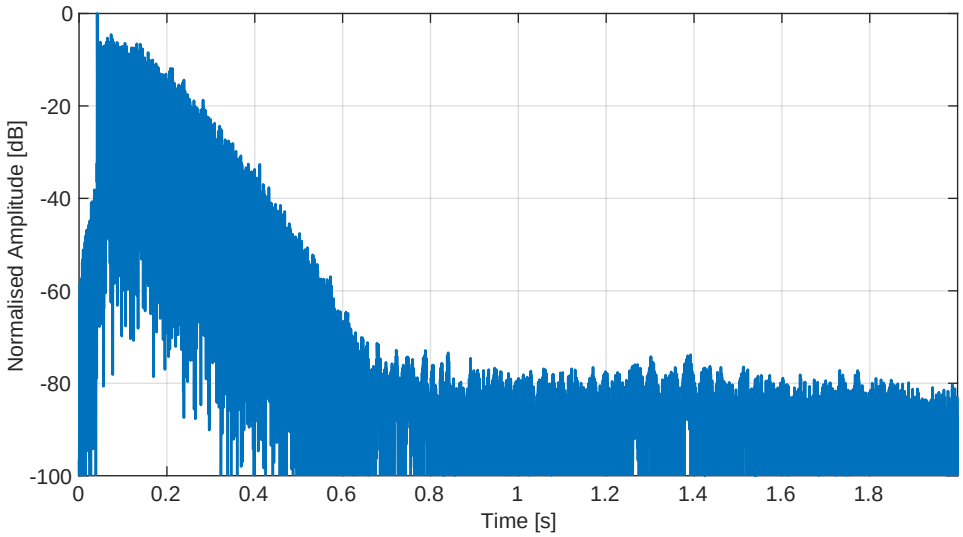


Figure C.2.5: Normalised impulse response in dB.

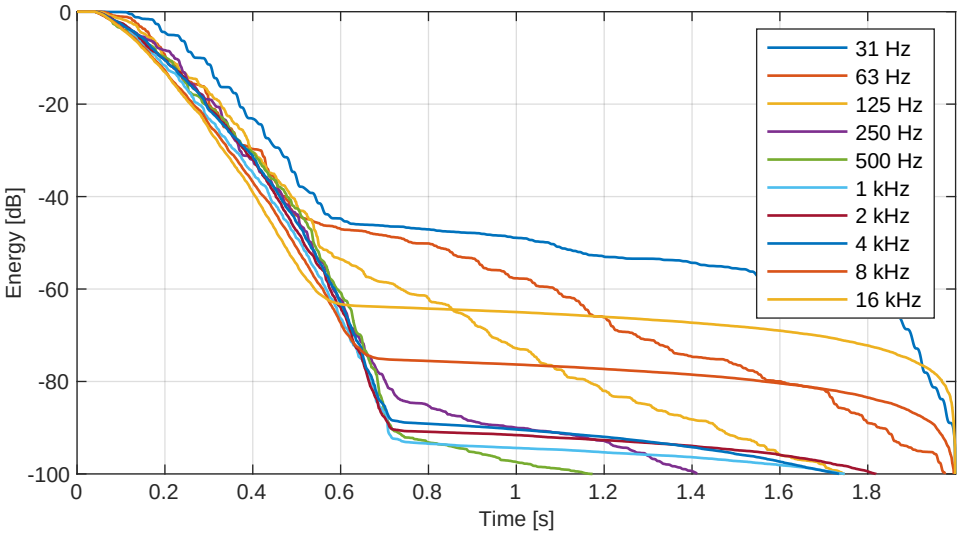


Figure C.2.6: Schroeder energy curves.

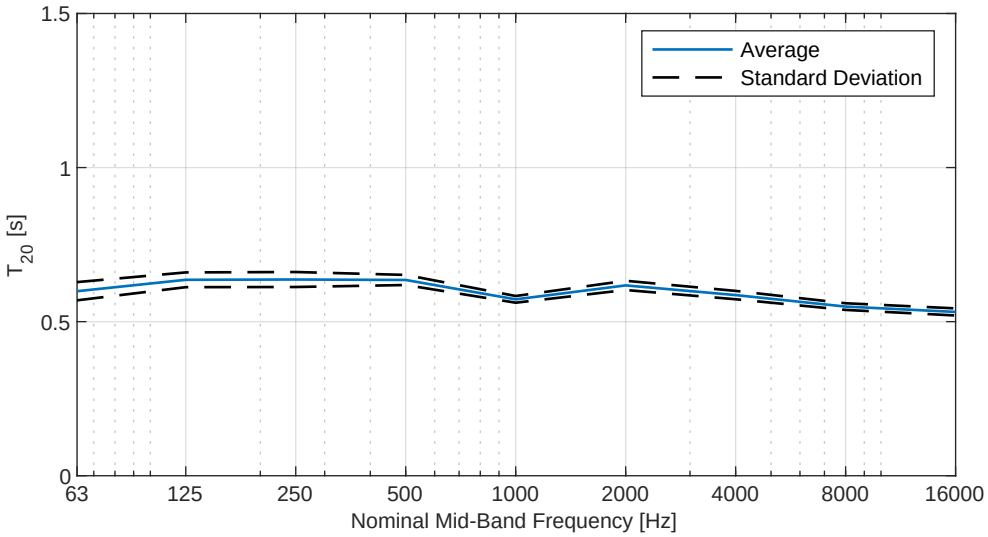


Figure C.2.7: T_{20} for the empty listening room.

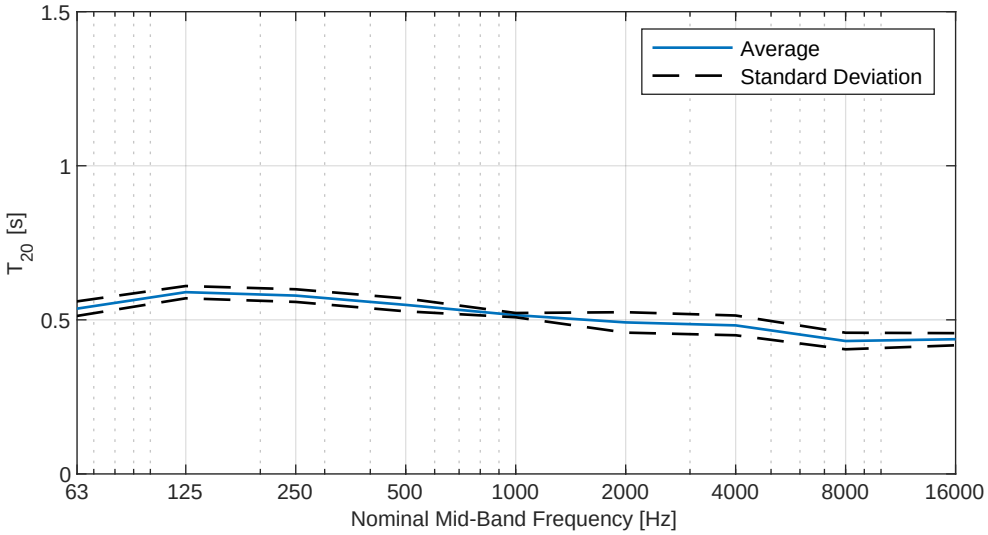


Figure C.2.8: T_{20} for the empty listening room with the acoustic partitioning wall.

Frequency [Hz]	31	63	125	250	500	1k	2k	4k	8k	16k
Empty [s]	0.57	0.60	0.64	0.64	0.64	0.57	0.62	0.59	0.55	0.53
Wall [s]	0.55	0.54	0.59	0.58	0.55	0.52	0.49	0.48	0.43	0.44

Table C.2.4: T_{20} for all three setups.

D Source Code

D.1 Generation of SOFA files

This section presents two Matlab scripts for generating binary spatially oriented format for acoustics (SOFA) files. One for creating an omnidirectional measurement microphone (Lst. D.1.1) and one using the head-related transfer function (HRTF) measurements of the head and torso simulator (HATS) Valdemar (Lst. D.1.2).

It should be noted that the files are generated using Matlab R2015b 32-bit since files created with newer versions did not load on Quest 2. However, files created in newer versions, e.g. R2022b, work fine with the preview in Unity.

```
% SOFAmic
% Author: Frederik Sidenius Dam
% Date: 2022/10/14
% Dependencies: SOFAtoolbox
% Matlab version: R2022b (R2015b 32-bit)
% Functionality: Creates an omnidirectional microphone in SOFA format

clear, clc, close all

% Start SOFA
SOFAstart

% Data compression (0..uncompressed, 9..most compressed)
compression = 1; % results in a nice compression within a reasonable processing time

% Get an empy conventions structure
disp('Creating SOFA file with SimpleFreeFieldHRIR conventions...');
Obj = SOFAgetConventions('SimpleFreeFieldHRIR');

% Create the impulse response
N = 256;
IR = [1; zeros(N-1,1)];

% Fill data with data
Obj.Data.IR = NaN(2,2,N); % must be [M R N]
Obj.Data.IR(1,1,:) = IR*db2mag(0);
Obj.Data.IR(1,2,:) = IR*db2mag(0);
Obj.SourcePosition(1,:) = [0 0 1]; % spherical coordinates [azi ele 1]

% Dummy data (must have M>1 to work with Steam Audio)
Obj.Data.IR(2,1,:) = zeros(N,1);
Obj.Data.IR(2,2,:) = zeros(N,1);
Obj.SourcePosition(2,:) = [0 0 1];

% Update dimensions
Obj = SOFAupdateDimensions(Obj);

% Fill with attributes
Obj.GLOBAL_Title = 'Omnidirectional Microphone';
Obj.GLOBAL_ListenerShortName = 'MIC';
Obj.GLOBAL_ApplicationName = 'SOFA Microphone';
Obj.GLOBAL_ApplicationVersion = SOFAgetVersion('API');
Obj.GLOBAL_DatabaseName = 'Local';
Obj.GLOBAL_Organization = 'Aalborg University';
Obj.GLOBAL_AuthorContact = 'fdam21@student.aau.dk';
```

```

% Save the SOFA file
SOFAfn = 'SOFA/MIC.sofa';
disp(['Saving:  ' SOFAfn]);
Obj = SOFAsave(SOFAfn, Obj, compression);
```

Listing D.1.1: SOFAmic.m

```
clear, clc, close all
addpath("hrtf_val_2deg/")

% Start SOFA
SOFAstart

% Data compression (0..uncompressed, 9..most compressed)
compression = 1; % results in a nice compression within a reasonable processing time

% Get an empy conventions structure
disp('Creating SOFA file with SimpleFreeFieldHRIR conventions...');
Obj = SOFAgetConventions('SimpleFreeFieldHRIR');

% Fill data with data
M = 2*(180*length(0:2:48) + 120*length(50:2:66) + 72*length(68:2:70) + 60*length
    (72:2:76) + 40*length(78) + 36*length(80:2:82) + 20*length(84:2:86) + 8*length(88) +
     1*length(90)) - 180;
N = 256;
Obj.Data.IR = NaN(M,2,N); % data.IR must be [M R N]

m = 0;
for ele=-90:2:90
    [hrtfl,hrtfr]=valload(ele);
    num_meas = size(hrtfl,2);
    res = 360/num_meas;
    for i=1:num_meas
        m = m + 1;
        azi = (i-1)*res;
        Obj.Data.IR(m,1,:) = hrtfl(:,i);
        Obj.Data.IR(m,2,:) = hrtfr(:,i);
        Obj.SourcePosition(m,:)=[azi ele 1];
    end
end

% Update dimensions
Obj = SOFAupdateDimensions(Obj);

% Fill with attributes
Obj.GLOBAL_Title = '2 Degree Resolution HRTFs';
Obj.GLOBAL_ListenerShortName = 'VALDEMAR';
Obj.GLOBAL_ApplicationName = 'SOFA Valdemar';
Obj.GLOBAL_ApplicationVersion = SOFAgetVersion('API');
Obj.GLOBAL_DatabaseName = 'Local';
Obj.GLOBAL_Organization = 'Aalborg University';
Obj.GLOBAL_AuthorContact = 'fdam21@student.aau.dk';

% Save the SOFA file
SOFAfn = 'SOFA/VALDEMAR.sofa';
disp(['Saving:  ' SOFAfn]);
Obj = SOFAsave(SOFAfn, Obj, compression);
```

Listing D.1.2: SOFAvaldemar.m

D.2 Unity Scripts

This section presents three C# scripts for Unity, where the first (Lst. D.2.1) is used to swap between different sets of HRTFs. The last two are used to enable character movement using a keyboard (Lst. D.2.2) and mouse (Lst. D.2.3) following the video tutorial in [54].

The first script is hard-coded for the use of three different sets of HRTFs, where the zero-indexed represents the default.

```
using System.Collections;
using System.Collections.Generic;
using SteamAudio;
using UnityEngine;

public class SelectHRTF : MonoBehaviour
{
    public int currentHRTF = 0;
    bool changedHRTF = true;

    // Update is called once per frame
    void Update()
    {
        if (Input.GetKeyDown("1"))
        {
            currentHRTF = 0;
            changedHRTF = true;

        }
        if (Input.GetKeyDown("2"))
        {
            currentHRTF = 1;
            changedHRTF = true;
        }
        if (Input.GetKeyDown("3"))
        {
            currentHRTF = 2;
            changedHRTF = true;
        }

        currentHRTF = Mathf.Clamp(currentHRTF, 0, 2);

        if (changedHRTF)
        {
            SteamAudioManager.Singleton.currentHRTF = currentHRTF;
            Debug.Log("Current HRFT: " + SteamAudioManager.Singleton.hrtfNames[
    currentHRTF]);
            changedHRTF = false;
        }
    }
}
```

Listing D.2.1: SelectHRTF.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerMovement : MonoBehaviour
{
    public CharacterController controller;

    public float speed = 10f;

    // Update is called once per frame
    void Update()
    {
        float x = Input.GetAxis("Horizontal");
        float y = Input.GetAxis("Elevation");
        float z = Input.GetAxis("Vertical");

        Vector3 move = transform.right * x + transform.up * y + transform.forward * z;

        controller.Move(speed * Time.deltaTime * move);
    }
}
```

Listing D.2.2: PlayerMovement.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

```
4
5  public class MouseLook : MonoBehaviour
6  {
7      public float mouseSensitivity = 100f;
8
9      public Transform playerBody;
10
11     float xRotation = 0f;
12
13     // Start is called before the first frame update
14     void Start()
15     {
16         Cursor.lockState = CursorLockMode.Locked;
17     }
18
19     // Update is called once per frame
20     void Update()
21     {
22         float mouseX = Input.GetAxis("Mouse X") * mouseSensitivity * Time.fixedDeltaTime
       ;
23         float mouseY = Input.GetAxis("Mouse Y") * mouseSensitivity * Time.fixedDeltaTime
       ;
24
25         xRotation -= mouseY;
26         xRotation = Mathf.Clamp(xRotation, -90f, 90f);
27
28         transform.localRotation = Quaternion.Euler(xRotation, 0f, 0f);
29         playerBody.Rotate(Vector3.up * mouseX);
30     }
31 }
```

Listing D.2.3: MouseLook.cs