FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Evaluation of a Hybrid Sound Model for 3D Audio Games with Real Walking

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Medieninformatik

eingereicht von

Michael Urbanek
Matrikelnummer 1328186

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Privatdoz. Mag.rer.nat. Dr.techn. Hannes Kaufmann
Mitwirkung: Projektass. Iana Podkosova, BSc MSc

Wien, 25.08.2015

(Unterschrift Verfasser)

(Unterschrift Betreuung)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Evaluation of a Hybrid Sound Model for 3D Audio Games with Real Walking

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Media Informatics

by

Michael Urbanek
Registration Number 1328186

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Privatdoz. Mag.rer.nat. Dr.techn. Hannes Kaufmann
Assistance: Projektass. Iana Podkosova, BSc MSc

Vienna, 25.08.2015

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Michael Urbanek
Neilreichgasse 85-89/2/2, 1100 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasser)

Acknowledgements

I would like to thank everyone who participated in the user study voluntarily and my advisors for supporting me throughout the whole thesis (and even before).

Abstract

Audio games are a subcategory of computer games that have spatialized audio as the only output that their players receive. In order to provide a compelling impression of the environment, sound in audio games has to be of superior quality in terms of immersion and realism. For improving the immersion and realism in audio games, complex sound models can be used to generate realistic sound effects, including reflections and reverberation. An implementation of a hybrid sound model similar to the ODEON approach is introduced and adapted for real-time sound calculations. This model is evaluated and compared to a baseline model usually used in audio games in a user study in a virtual reality environment. The results show that the implemented hybrid model allows players to adjust to the game faster and provides them more support in avoiding virtual obstacles in simple room geometries than the baseline model. Complex sound models are beneficial in audio games, provided that there is enough computational resources available to perform calculations in real time.

Kurzfassung

Audiogames sind eine Unterkategorie von Computerspielen, welche ausschließlich auditiven Output erzeugen und somit gänzlich auf visuellen verzichten. Da Sound der einzige Output ist, den der Spieler oder die Spielerin erhält, muss er von überragender Qualität hinsichtlich Immersion und Realismus sein. Um diese Qualität zu erreichen, können komplexe Soundmodelle herangezogen werden, die Reflexionen und Echo erzeugen. In dieser Thesis wird ein hybrides Soundmodell ähnlich dem Modell von ODEON vorgestellt, das für Echtzeitberechnungen in der verwendeten Softwareumgebung adaptiert wurde. Das implementierte Modell wird mit einem üblichen, in Audiogames verwendeten Modell (baseline model) innerhalb einer erzeugten virtuellen Realität mit freiwilligen Teilnehmern und Teilnehmerinnen getestet. Die Ergebnisse zeigen, dass das implementierte hybride Soundmodell, im Vergleich zum existierenden Ansatz, Spieler und Spielerinnen unterstützt und es ihnen erlaubt, in simplen, virtuellen Räumen Hindernissen auszuweichen. Audiogames sollten, sofern genügend Rechenleistung vorhanden ist, komplexe Soundmodelle wie beispielsweise das in dieser Thesis vorgestellte, implementieren.

Contents

CHAPTER 1

Introduction

Computer games may be used for entertaining or educational purposes, although the industry has its main focus on the first one. The increasing amount of power and performance of modern computers assist the development of more realistic computer games. Graphics processing units offer the required performance which is needed for realistic rendering and visual output. Most computer games use visual output with an additional support of audio to give the player an immersive and realistic feeling. However, there are people who will not or cannot rely on visuals, e.g. visually impaired people. Audio games can replace common types of computer games for these groups of users.

Audio games are a subcategory of computer games which focus on another sense of a human being: hearing. Since sound is the only output and feedback that players get in audio games, it has to be of superior quality to provide a convincing experience. The sound component in those games serves as replacement for visuals when compared to computer games with visual output. To have a realistic sound in audio games, this sound has to be spatialized in 3D. In this context, spatialization means that the sound that a listener hears is calculated according to the position of the sound source in virtual (3D) space. Without spatialization, it is not possible to locate a sound source.

Sound propagation models that are used for room acoustics are able to produce geometry dependent reverberation effects. Audio games using such models could provide more realistic auditive environments for players. Existing audio games are usually played in a desktop environment. Navigation, e.g. the change of the player's viewport position, is done by using input devices, e.g. mouse, keyboard or other controllers [28]. However, natural sound localization is based on binaural hearing e.g. rotating the head to determine sound direction, which is not possible when using the above mentioned devices. A user's head position and rotation can be tracked in real time with the use of dedicated tracking equipment and used for sound spatialization. If head tracking is available in a large enough physical space, navigation by real walking becomes possible. The possibility to localize the sound source while rotating one's head while walking in a virtual environment just like in the real world provides players with much higher degree of immersion compared to a desktop environment [50].

1.1 Motivation

The motivation of this thesis is to enhance realism and immersion in audio games by using compelling and realistic sound provided by a complex sound simulation model in combination with natural navigation by real walking. There is no research published that investigates such combination. The real-time sound simulation model developed as an outcome of this thesis can be used by audio game designers and developers to enhance the quality of sound in their products.

1.2 Problem statement

The sound models used in existing audio games [28, 29, 45, 55, 58, 62] do not take the structure of the Virtual Environment (VE) into account. They provide spatialized sound, but do not simulate any kind of more complex sound phenomena like reflections or reverberations that can be generated by more sophisticated sound models. The simulation of these effects are included into an audio game in this thesis. In a game environment, sound calculations need to be done in real time depending on a user's position. The spatialization of sound relative to a user's position can be done by using existing technologies, e.g. an audio middleware solution. However, the real-time calculation of a sound model is a difficult task and a proper sound model needs to be chosen to work in an interactive environment in real time.

To summarize, in this thesis it is investigated how complex effects like sound reflections and reverberations can be included into a sound model used in an audio game and calculated in real time together with spatialized sound.

1.3 Aim of the work

The aim of this thesis is to implement a sound propagation model with real-time sound spatialization, reflections and reverberation and to integrate it into an audio game prototype with navigation by real walking.

1.4 Methodological approach

In this thesis, an already existing Virtual Reality (VR) game environment is used. In the used setup, a game is implemented in the Unity3D game engine and runs on a laptop worn by a user. An Oculus Rift DK2 is used for rendering (or blindfolding users in case of an audio game without visuals) and an optical tracking system for the calculation of the user's position.

An audio middleware solution Wwise is chosen as the basis for all sound calculations. The thesis contains a comparison of sound spatialization plug-ins, with the AstoundSound plug-in being chosen for the final game prototype.

An investigation of existing sound models in the domain of room acoustics shows that a hybrid sound model can be adapted and used for real-time application in an audio game. With this sound model, it is possible to simulate sound reflections and reverberation. A baseline

model, which only spatializes sound without any further effects, is also implemented as means of comparison.

The evaluation of the implemented sound model is done in an extensive user study with 37 participants. The implemented hybrid model is compared to the baseline model in a between-subject user test. Subjective measurements (Questionnaires) and objective measurements (Logging) are used for the evaluation.

1.5 Structure of the work

The state of the art in the areas of audio games and sound models is described in Section 2. Section 3 introduces main design principles of the presented work, where the test environment, the requirements, plug-ins and sound models are discussed. In Section 4, the integration of a hybrid sound model into the Unity3D game engine is described. In addition, the integration of the used audio middleware solution Wwise and the used spatialization plug-in AstoundSound is summarized. The evaluation of the implemented hybrid model is presented and discussed in Section 5. There, the proposed hypotheses, a game prototype, the study design and results are presented. Section 6 concludes the thesis. The questionnaires used in the user study can be found in Appendix A.

CHAPTER 2

State of the art

2.1 Sound

This chapter introduces fundamentals of sound formation, sound related phenomena and gives an insight of how humans perceive sound. Sound simulation methods and their appropriateness for audio games are discussed.

2.1.1 Fundamentals

Sound is a disturbance in form of waves in a medium caused by the vibration of a sound source [1]. A medium is necessary for transportation of the sound, like air (usually), water (underwater sound) or earth (e.g. bass vibrations felt on the ground). Vibration in this context means that the size of the sound source oscillates between two sizes, one a bit larger than normal and one a bit smaller [42]. While changing its size, the sound source stimulates the medium (e.g. air) around it which causes compression (when the sound source grows bigger) and rarefy (when the sound source gets smaller) of the medium, thus producing sound waves. In the domain of audio, sound waves are longitudinal, which means that the displacement of the medium is the same as the direction of the energy transport. The rate at which particles of a medium oscillate around a given point in space is defined as *frequency*, which is measured in Hertz (Hz) or cycles per second [42]. The amount of compression and rarefaction of the transport medium defines the amplitude of the wave, which is perceived as *loudness* when it arrives the receiver (e.g. an ear) [42].

The *sound pressure* is the energy of a sound emitting source on its surrounding [42]. The Sound Pressure Level (SPL) is usually measured in Newtons per square meter ($\frac{N}{m^2}$) [42]. However, in practical acoustics a logarithmic scale for SPL is used: *decibel*. 0 dB is near to the hearing threshold and 130 dB near to the threshold of pain [56]. This decibel scale is based on *sound energy* [56]. *Sound intensity* is the product of sound pressure and the sound particle velocity [47]. Sound waves can be described with the wave equation. A sound wave is a space- and time-dependent function of the sound pressure p, which is defined for one dimension as fol-

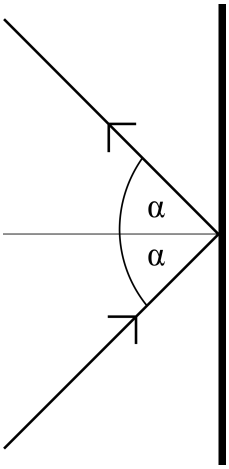


Figure 2.1: The reflection of a sound wave on a smooth plane.

lows [30]. Be p the sound pressure, c the speed of sound (in air), t the time and x the respective position, then the wave equation is defined by

$$\frac{\partial^2 p}{\partial x^2} - \frac{1}{c^2}\frac{\partial^2 p}{\partial t^2} = 0. \tag{2.1}$$

Without proof, the solution p equals

$$p = p_0 sin(\omega t \mp kx) \tag{2.2}$$

where p_0 is the sound pressure amplitude, ω the angular frequency of waves and $k = \frac{2\pi}{\lambda}$ [30]. λ is the wavelength.

Sound pressure can be influenced by obstacles e.g. objects or walls on the path of a wave through reflection, scattering and diffraction [56]. If the surface is completely smooth, the plane sound wave is reflected specularly, according to Snell's law [56]. Figure 2.1 shows this reflection, both angles are equal. Scattering is the reflection of sound waves on rough surfaces [56]. A surface is rough if the irregularities on it are much larger than the wavelength of the sound wave. Diffraction, which is the bending of sound appears at edges or corners of an obstacle [56]. Diffraction waves are produced on the edges of the obstacle. The intensity of this diffraction wave depends on the size of the object and the wavelength.

2.1.2 Perception

Humans use their two ears for sound perception. This is called binaural hearing, which is the basis of sound localization [25].

The human brain is able to locate an audio source depending on intensity as well as time differences between the arrival on both ears [25]. With this information, it is possible to express two variables in the context of hearing: the direction and the distance of an occurring sound [25]. However, if there are more subsequent sound waves (e.g. in a room where sound reflections are

present), the human perceives the location of the sound event where the first sound wave hits the human ear [25].

The human ear consists of several parts that can be separated into the outer ear, the middle ear and the inner ear [42]. The pinna of the outer ear consists of the skin and bone structure that is visible. This area is responsible for focussing the sound waves that go through the ear canal towards the ear drum. This outer area is different for every human being. Three bones that are connected as lever are in the middle ear. They connect the ear drum with another membrane that is located between the middle ear and the inner ear. This membrane then stimulates a fluid in the inner ear. The stimulus of this fluid is finally transmitted by an auditory nerve to the brain that is perceived as hearing. Therefore, sound has to travel through the whole hearing system to get perceived. In addition, the shape of the ear helps to distinguish directions.

There is a difference between static and dynamic sound localization. The authors of [25] describe an example with an approximation of the head through a sphere. A human would not notice the difference between audio sources placed in front or behind, as well as above or beneath when not moving the head because the differences in intensity and time are equal. When the head is not moved, this is called the static localization process. If a person is not sure about which direction a sound comes from, he or she moves the head slightly to determine the location due to changes in the intensity and time domain. This is called dynamic localization. Dynamic localization is the movement of the head for localization purposes and is essential and recommended for an immersive virtual environment of audio games, which emulate spatial sounds [11]. This is possible with continuous tracking of the head's position and movement for adaption in sounds in the virtual environment [14].

The hearing system of humans has another ability which enables listeners to focus on one audio source while suppressing other sources like noise or voices. This is called the cocktail party effect [37]. Therefore, it is possible that several audio sources in audio games can emit sound waves simultaneously, while the player has the ability to focus on the sources he or she finds most interesting.

2.1.3 Room Acoustics

The scientific domain describing the rules of sound propagation in rooms is room acoustics. One major element of sound in rooms is the existence of reverberation. Reverberation is the remainder of sound in a space when the original source has already stopped emitting sound [59]. This effect occurs in rooms. Reverberation in rooms is caused by multiple reflections of the sound at the room's walls that has its origin at the original sound source in space [56]. It can be split into three parts: direct sound, early reflections and late reverberation [56].

Direct sound is the first sound wave that reaches the listener without having been reflected. Therefore it travels the shortest distance between the sound source and the listener compared to early reflections and late reverberation. In an infinitely open space, direct sound is the only part which is perceivable. Direct sound indicates the direction of the sound source. The precedence effect (which is related to direct sound) is a binaural psychoacoustic effect which states that the listener perceives the sound direction according to the first wave recognized by the human hearing system [57]. For example, two sound sources are placed in an open space, both emitting the same sound. The first sound source emits the sound without any restrictions continuously.

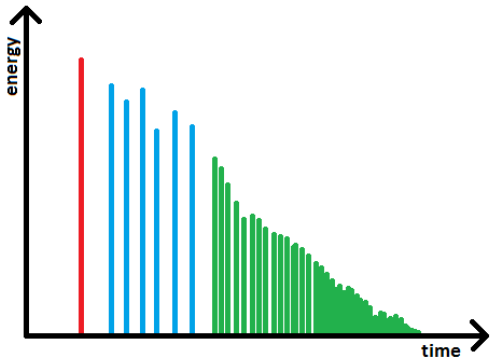


Figure 2.2: An example of the energetic room response. The strongest impulse has direct sound (red), followed by early reflections (blue) and late reverberation (green).

The second sound source emits the same sound delayed but louder. The direction of the first sound source is perceived as more dominant. In a room with several reflections, the precedence effect helps to identify the original sound source position even though reflected sound waves are coming from all directions.

Early reflections are the first subsequent reflections at obstacles (or room boundaries). Like direct sound, early reflections provide information about the position and distance of the sound source. In addition to that, these reflections provide clues about the room geometry. Early reflections are important for the direct sound impression, which Vorländer describes as *"They enhance the loudness, support the intelligibility of speech, the clarity of music and the impression of the auditory source width."* [56].

Late reverberation is contributing to the reverberation mostly and is responsible for the envelopment of the respective listener. They provide information about the room size and reflectivity, however, they do not provide information regarding the direction of the sound anymore.

The combination of direct sound, early reflections and late reverberation forms the overall perceived reverberation in a room. Figure 2.2 shows the energetic room response.

2.1.4 Obstruction

When an obstacle is between a listener and a sound source, the listener perceives the sound as attenuated. The reason for this is that the direct sound which is the shortest distance between the listener and the sound source travels through the obstacle and therefore is perceived as muffled. High-frequency sounds are reflected or absorbed by obstacles, low-frequencies get transmitted [16]. In sound simulation, this effect can be achieved by applying a low-pass filter on the sound source. Figure 2.3 shows the direct sound traveling through two walls. S is the sound source position, R the position of the receiver. When the direct sound wave first hits the wall it gets attenuated. It then travels with less power towards the receiver position and goes through the second wall where it is again attenuated.

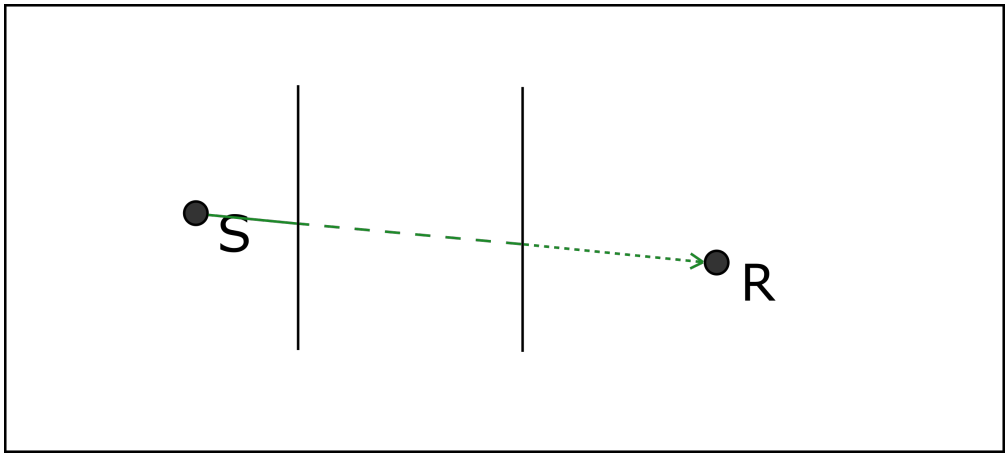


Figure 2.3: An example of how the obstruction effects the sound coming from the sound source. S is the sound source position, R the position of the receiver. The more dots the line has, the stronger the attenuation is.

2.1.5 Sound Simulation

Computer sounds can be pre-recorded or artificially generated and can be modified and played through software. Pre-recorded sounds are stored in sound files. These files can be accessed with sound software at any playback position. In addition to that, effects can be applied that e.g. modify or filter sound frequencies. However, several parameters can be accessed through sound software, e.g. playback position, volume (loudness) or pitch (playback speed).

Sound simulation models can be used to produce reverberation artificially in real time, taking the room geometry into account. For this computations, the listener's position and rotation must be known, otherwise the spatialization and reverberation cannot be calculated. Sound spatialization is *the ability to play a sound as if it is positioned at a specific point in three-dimensional space* [34].

2.1.5.1 Head-Related Transfer Functions

As described in Section 2.1.2, humans perceive sound through their ears. Incoming sound waves are focused at the pinna and transmitted through the ear canal to the ear drum (outer ear). Absorptions at the pinna, the head and the torso are different for every individual and affect the perceived sound. To simulate these absorptions effects, a Head-Related Transfer Function (HRTF) can be used. HRTFs are the response of how the human ear receives a sound in space, which is basically a model of the human hearing system and its auditive interpretation [48]. It describes the effect and absorption of the head, outer ear and torso. Vorländer offers a comprehensive definition of HRTF:

> *It [HRTF] is defined by the sound pressure measured at the eardrum or at the ear canal entrance divided by the sound pressure measured with a microphone at the centre of the head but with the head absent.* [56]

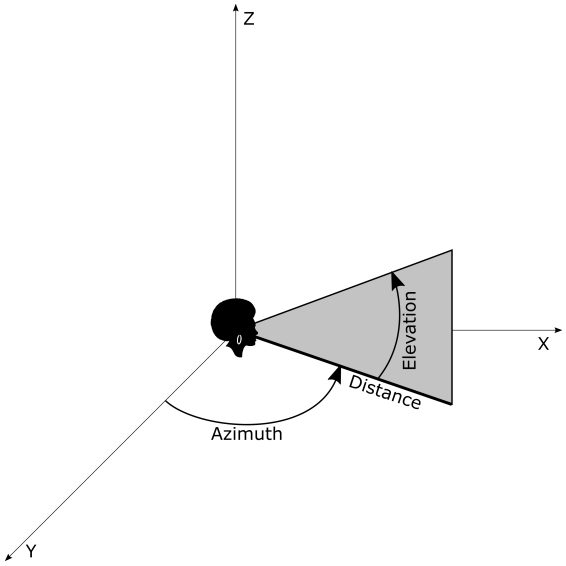


Figure 2.4: The three parameters when measuring the HRTFs. Two angles azimuth and elevation and one distance parameter.

HRTFs can be measured by placing tiny probe microphones inside both ear channels of a person and then playing sound through a loudspeaker at different azimuth, elevation but same fixed distance from the person's head [7]. This way, when the sound source is moved in space and therefore emits sound at different places different HRTFs are measured and stored for later usage e.g. sound spatialization. The three parameters are shown in Figure 2.4. The stored HRTFs can be applied on a sound source signal through convolution or (inverse) Fast Fourier Transform (FFT) to simulate the absorption effects [24]. HRTFs are only required when a listener is wearing headphones.

In sound simulation standard HRTFs are used, e.g. the HRTFs measurements of a KEMAR dummy-head microphone [13].

2.1.5.2 Sound Model Categories

This section covers different types of sound models that may be used to calculate and simulate room acoustics. However, not every method is appropriate in every situation. Existing approaches can be grouped into the following categories:

- wave-based approaches,
- ray-based approaches,
- and statistical approaches.

Waved-based approaches

Solving the wave equation for a room is one method that can be applied in room acoustics to generate realistic but artificial reverberation. Approaches that simulate room acoustics by solving the wave equation are called wave-based approaches. In wave-based approaches, the geometrical structure of the virtual environment must be discretized [56]. This discretization process divides the structure into small elements to allow the creation of a linear system of equations. The spatial discretization is done by creating a discretized grid (mesh) of the surface or points in space [56]. Depending on the size of the wavelengths, the chosen discretization must be sufficient to allow interpolation [56]. At high frequencies (and therefore small wavelengths), the discretization must be large enough.

There are typically two methods that can be used in wave-based approaches, the Boundary Element Method (BEM) and the Finite Element Method (FEM) [56]. In the BEM, only boundaries of the room geometry needs discretization, e.g. the walls, the floor and the ceiling. On the other hand, in FEM the whole room is divided into pieces for which the linear system must be solved.

Wave-based approaches are more accurate compared to ray-based approaches [49]. In wave-based approaches, diffraction can be modelled without applying additional methods [49]. This sound phenomenon is more present at low frequencies only. This makes the sound model more accurate compared to ray-based approaches.

Wave-based approaches are computationally intensive [49]. The computational workload depends on the size of the structure and the amount of frequencies to calculate. Therefore, it is more recommended for a small range of frequencies (e.g. calculation of only low frequencies) [49]. Vorländer gives an calculation duration example for BEM in [56]. With a computer that can solve 8000 nodes in 60 seconds with a desired frequency resolution of 1 kHz, the calculation time equals about $\frac{f}{60}$ hours with f being the frequencies. In FEM with mesh sizes in a range of 100.000 nodes, calculation times are in order of magnitude of 5 minutes per frequency [56].

Due to its high computational costs, wave-based approaches can hardly be used for audio games and games in general. Sound models must do their calculations in real time when applied in interactive software. Wave-based approaches might get wider use with further improvements of hardware processor power. However, recently published literature shows that it is possible to use a wave-based approach in games [27].

Ray-based approaches

Ray-based approaches treat the sound waves as rays similar to the rays of light. Rays can be reflected, refracted or diffracted [56]. There are two possible ways to construct rays in such approaches [56]: One way is to use forward geometric construction which follows the ray from the sound source to the respective listener. The other way is to backtrace the ray from the listener to the source. Both are equivalent to each other due to the law of reciprocity [56]. Rays in these approaches carry sound energy that can get reduced by reflections or absorption effects, however, when a sound ray hits a volume or the listener in a scene, the energy at this point is known.

Two methods are introduced briefly in this section. The image source method and stochastic ray tracing. The image source method is a ray-based approach that is discussed extensively later in this thesis. In this method, the sound source is mirrored at the boundaries of the room, generating image sources. These image sources are mirrored again to create image sources of higher order. Further explanations can be found in Section 2.1.6.1. In stochastic ray tracing, sound is simulated as a bunch of particles that follow a ray's path [56]. These particles get reflected in the same way rays do and lose energy due to these reflections or absorptions. At the receiver's position, the incoming particles are counted over time to provide the listener with reverberation.

Ray-based approaches are not as accurate as wave-based approaches, however, the calculation time for more complex structures is smaller in ray-based approaches compared to wave-based ones [49]. The image source method e.g. can find all specular reflection paths. However, at higher order reflections this computations can be very costly [49].

The computation time of the image source method depends on the amount of boundaries at which the sound source is mirrored and the maximum image source order. Be n the amount of room boundaries (walls + floor + ceiling) and x the maximum image source order, then the amount of generated image sources i can be calculated by

$$i = n^x. \tag{2.3}$$

The amount of generated image sources at higher order must be handled in an interactive application in real time which is not possible. For this purpose, hybrid models (see Section 2.1.6) are introduced that approximate the results at lower computational costs. This is only valid for the image source method, however. The computation costs are smaller for stochastic ray tracing.

Ray-based approaches are appropriate for audio games due to the better computational performance compared to wave-based approaches.

Statistical approaches

The Statistical Energy Analysis (SEA) is a framework for the analysis of systems, where the energy is of interest [23]. It can be used for the prediction of vibration and therefore sound of acoustic systems, e.g. sound transmission through a wall. Statistical approaches therefore can be used in the domain of coupled systems for the prediction of noise levels, where the transmission of structures plays a major role [30]. However, these applications are not of relevance in audio games and therefore such approaches are not used.

2.1.6 Hybrid Model

A hybrid sound propagation model combines two methods to negate the disadvantage of one method with the advantage of another. In this thesis, the hybrid model combines the image source method (ray-based approach) with secondary sources. The combination of these methods allows a realistic result [38].

In this thesis, an approach similar to the model that ODEON uses is used and discussed in the next sections. ODEON is a professional room acoustics software [33]. It uses secondary sources in combination with stochastic scattering to calculate late reverberation.

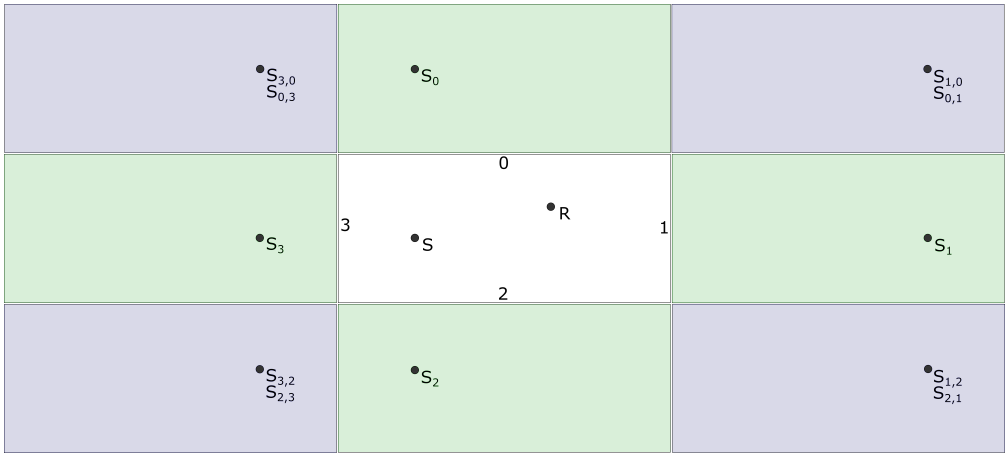


Figure 2.5: Calculation of image sources. S_x: image sources of first order; $S_{x,y}$: image sources of second order.

2.1.6.1 Image Source Model

The image source model calculates sound reflections as long as it is assumed that the walls of the room are smooth and therefore generate specular-reflected sound waves. The energetic room response can be constructed with the use of this model. Sound calculation in this model works as follows.

For starting the calculations, the position of the sound source and the structure of the room must be known. The position of the sound source is a vector in 3D space (x, y, z). First, the sound source is mirrored at every plane of the walls in the room. This procedure is done recursively, creating image sources of higher order, excluding mirroring at the same wall again (generating all permutations without having the same wall consecutively). Figure 2.5 shows this mirroring process in a very simple room. The room consists of 4 walls (numbered from 0 to 3), while S represents the sound source and R the receiver. In this mirroring process, the mirrored sound source positions are of relevance. Starting clockwise, the sound source is first mirrored at wall 0, then at wall 1, 2 and finally 3. The generated mirrored sound sources are called *image sources*. These are first order image sources, illustrated in the rooms with green background. After the first order calculation, the mirroring process starts again, mirroring image source S_0 at wall 0 (top, not shown), then S_0 at wall 1 which generates image source $S_{1,0}$, S_0 at wall 2 (not allowed) and S_0 at wall 3 that generates $S_{3,0}$. This process may be done until a satisfying image source order is reached, e.g. two [38]. The higher the order, the more reflections are generated that produce more reverberation. However, calculation time increases with every wall and must be limited by a maximum order (see discussion about computational costs in image source method above). The second order reflections are highlighted in the purple areas.

The positions of sound sources of higher order can be calculated using the following formula. Be $\vec{S}$ the original sound source position, $\vec{S}_n$ the position of the (mirrored) image source, $\vec{n}$ the normal vector of the respective wall (surface), $\vec{r}$ the vector between the foot point of the wall

normal and the sound source $\vec{S}$, then

$$\vec{S}_n = \vec{S} - 2d\vec{n} \tag{2.4}$$

with

$$d = \vec{r} \cdot \vec{n} = |\vec{r}| \, cos(\alpha) \tag{2.5}$$

calculates the positions of image sources at higher orders [56]. To calculate e.g. order two, the positions of the first order reflections are taken as $\vec{S}$.

Not all of the image sources calculated by this step are used further. An audibility test is performed to find the sources which are valid. For this calculation, the position of the receiver R is needed. The audibility test is a backward check. The starting point for performing this check is at the receivers position R. The unique mirroring order of an image source indicated by its indices (e.g. $S_{0,1}$) defines the order at which walls the image source was mirrored. These indices are used in the backward check.

An audibility test is performed for every image source. A ray is shot from the receiver position towards an image source. If the image source has been produced by mirroring at the same wall that the ray intersects first, this image source is considered valid, and invalid otherwise. Valid image sources will emit sound and contribute to early reflections.

The audibility test is illustrated in detail in Figure 2.6. Here, R marks the receiver position. S is the original sound source. Two walls of the virtual environment are marked with 0 and 1. S_0 and S_1 are image sources of the first order. $S_{0,1}$ and $S_{1,0}$ are image sources of the second order. For the image sources of the second order, the last index indicates the wall at which the image source was produced. In the given example, the image source $S_{1,0}$ is examined for validity. A ray is shot from the receiver position R towards $S_{1,0}$. This ray first intersect wall 1. $S_{1,0}$ was produced by mirroring against the wall 0, as indicated by its last index. This last index does not coincide with the index of the wall that the ray intersects first. Therefore, the image source $S_{1,0}$ is not valid. In comparison, $S_{0,1}$ is a valid image source. Here, a new ray is started at the point of intersection with wall 1 to examine if S_0 which is the predecessor of $S_{0,1}$ is also valid. The principle is the same as described above and therefore S_0 is also a valid image source. If there is no other wall between the last intersection point and the original sound source, the full path ($R \rightarrow S_{0,1} \rightarrow S_0 \rightarrow S$) is reconstructed and valid (blue line). This check has to be done with every image source. In the example of Figure 2.6, the path $R \rightarrow S_0 \rightarrow S$ is also possible and valid. The path $R \rightarrow S$ corresponds to direct sound.

A mother image source is the predecessor or one of the predecessors of an image source. If such a source is invalid, then all the successors are also invalid and a valid path cannot be reconstructed.

Mechel published a paper in 2002 called *Improved Mirror Source Method in Room Acoustics* which lists 8 interrupt criteria for calculation of image sources [26] that reduces the computation time in this method. Mechel defines the field angle as one criteria that has to be valid for each calculated image source. The field angle spans up an area with the image source position as origin in which the receiver must be in. This area is limited by the borders of the wall at which the image source was mirrored at. If the receiver is not in this area the image source is considered as invalid. Therefore neither it nor its successors (children) need to be calculated. Figure 2.7 shows a room, where the sound source is mirrored at wall 5 and then at wall 0. A cone, which

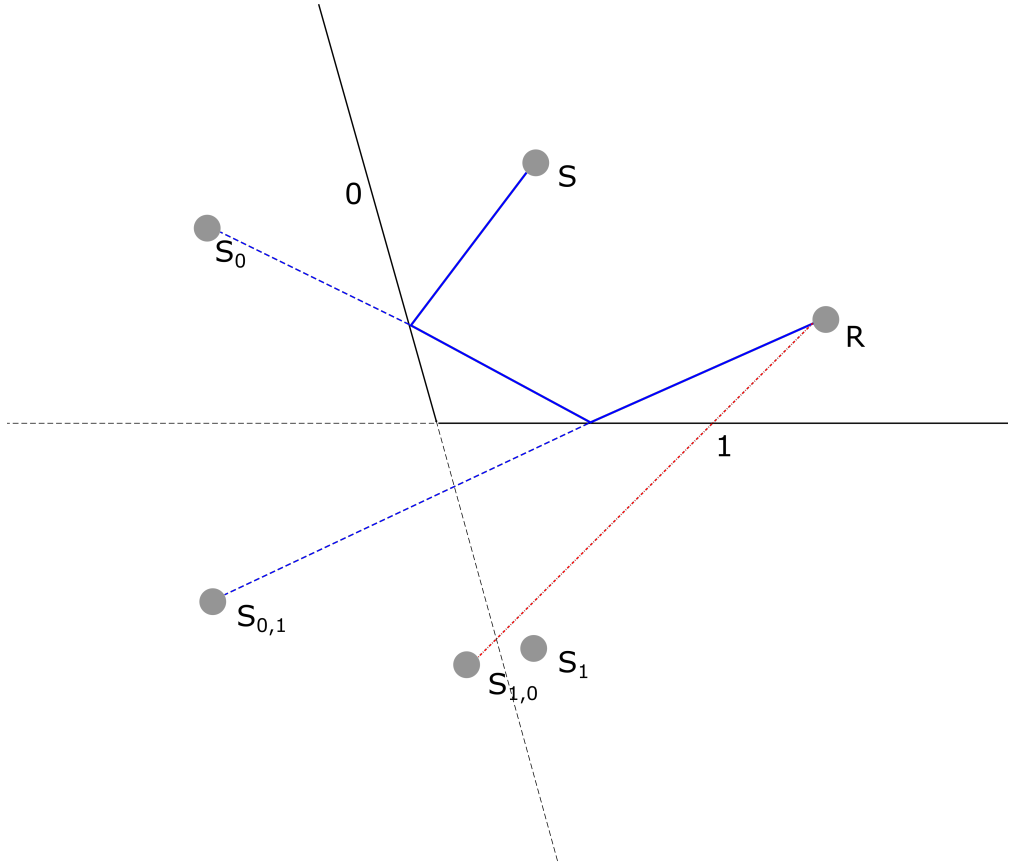


Figure 2.6: An illustration of the audibility test. S is the position of the sound source, R the position of the receiver. The walls are numbered with 0 and 1. First order image sources are S_0 and S_1, second order image sources are $S_{0,1}$ and $S_{1,0}$. $S_{1,0}$ is the only image source that is invalid in this example.

goes through the corners of the last mirrored wall (in case of $S_{5,0}$ wall 0) spans up an area in which the receiver has to be. If this criteria is met, the image source is considered as valid. In this figure, three receiver positions are shown, but only R_1 is valid. R_2 is outside of the cone (gray area) and R_3 is not in line of sight (audibility test) to $S_{5,0}$ and therefore invalid. If this algorithm is applied on earlier order reflections, computational time can be saved.

After the audibility check and interrupt criteria are applied, the valid image sources are identified. At the image source positions, sound sources can be placed that emit sound. Without an appropriate delay, all the sound sources would play the same sound at the same playback position. A sound delay must be added to generate the reverberation effect. The sound delay depends on the distance between the image source and the receivers position multiplied by the speed of sound [56]. If this distance gets bigger, the sound delay also increases.

The above mentioned process of image source creation, audibility check and interrupt criteria are valid for the fixed positions and rotations of the receiver, sound source and the underlying

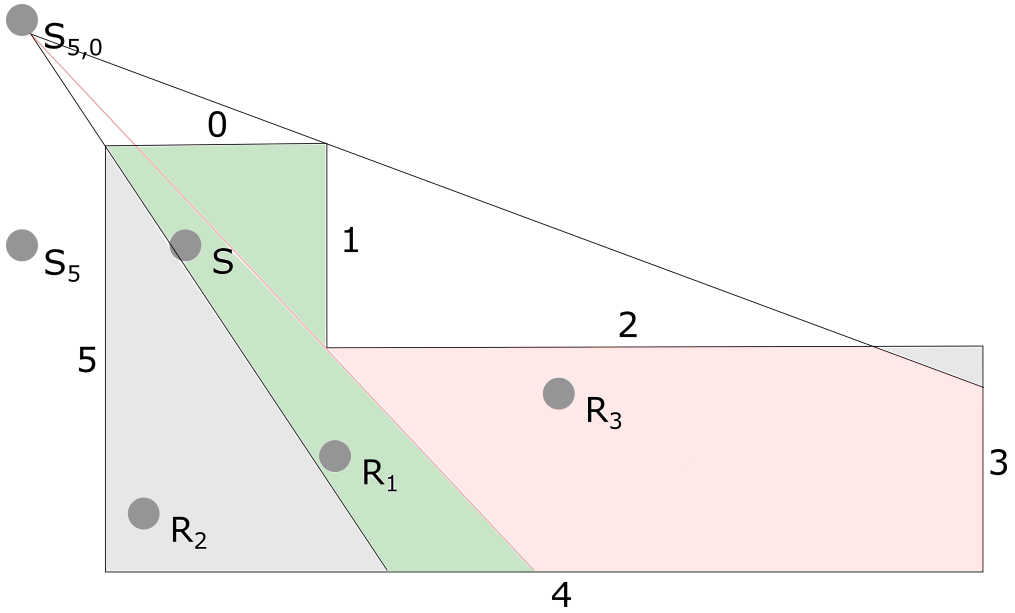


Figure 2.7: The field angle for faster computation. The sound source is mirrored first at wall 5 and then at wall 0. The cone describes the area in which the receiver has to be to be recognized as valid image source. If the receiver is in the green area, the image source passes this test.

virtual structure. However, if the receiver, the sound source or the geometrical structure changes, several actions must be performed [46]. When the room geometry changes or the sound source moves, the whole model needs to be recalculated. This means that the locations of the image sources must be recalculated and the audibility check with the interrupt criteria must be applied again. If the sound source or the listener rotates, only the orientations for sound spatialization of the listener need to be updated. However, if the listener moves, another audibility check must be performed.

2.1.6.2 Secondary Sources

In this hybrid approach, the image sources are calculated like described above. At a predefined point in image source creation order, the secondary source algorithm takes over and starts calculation. This order is called transition order. When the transition order is reached, no further image sources are created. Instead, at the point of intersection when a ray that is shot from last created image source position to the position of the image source next order hits a wall, a secondary source is created. Then, rays are emitted omnidirectional from this secondary source for generating other secondary sources at every new collision (reflection) point. These rays are then reflected at the room boundaries and generate other rays. Every ray is reflected according to Snell's Law with direction $\vec{v}$. In addition to that, a random vector $\vec{r}$ is calculated for simulating surface's roughness and the direction of the reflected ray is modified by $\vec{r}$ [38]. The scattering coefficient defines how much of $\vec{r}$ is added to $\vec{v}$.

If N rays are generated, every ray starts with a N'th part of the energy. The energy at a point can be described by the following formula. Be j the j'th order of the secondary source, E_j the energy at the j'th secondary source, E_s the energy at the original sound source, N the amount of rays and α_i the absorption coefficient of the respective wall, then is

$$E_j = \frac{E_s}{N} \prod_{i=1}^{j} (1 - \alpha_i) \tag{2.6}$$

the amount of energy at the j'th secondary source.

Ideally, these secondary sources should emit a very large number of rays that create new secondary sources at the room's boundaries that again shoot a large number of rays into the room and so on. However, Naylor postulated that the amount of rays will get very large quickly resulting in too much computational load [32]. He states that using one ray that creates secondary sources at reflection points is enough [32].

Figure 2.8 shows the procedure of the secondary source algorithm. S is the position of the sound source, R the position of the receiver. S_0 is the first order image source, $S_{0,2}$ the second order image source. The blue line is the reflection path generated through the image sources. The direction of the end of the blue line is given by the next image source position. However, the transition order is reached and the secondary sources algorithm starts creating secondary sources. Every secondary source is marked as green dot. The directions of the secondary rays depend on the incident angles.

The sound delay at the receiver depends on the total sum of the path length of the ray and the distance between the secondary source and the listener [32]. A secondary source is only valid if the position of this source is visible and not occluded from the position of the listener. Otherwise it does not emit any sound. The intensity at the receivers (listeners) position can be computed, giving the following formula. Be P the power of the original sound source, N the amount of rays, α_i the absorption coefficient of the respective wall, θ the angle between the surface normal and the receiver, r the distance between the secondary source and the receiver, so is

$$I = \frac{P}{N} \prod_{i=1}^{n} (1 - \alpha_i) \frac{2cos(\theta)}{2\pi r^2} \tag{2.7}$$

the intensity at the listener (I) [32].

A hybrid model approach negates the high computational costs at higher order reflections by introducing secondary sources. The created image sources are used for early reflections, the secondary sources are placed on the walls for late reflections. The amount of generated sound sources in the hybrid model is smaller compared to the amount of generated image sources higher order in a plain image source model. Therefore, the hybrid model is faster to evaluate. This advantage makes the hybrid approach appropriate for audio games. The accuracy of hybrid model is less than in a wave-based approach (see discussion above), however, the advantages of faster computation prevail the lack of accuracy.

According to Rindel, 500 to 1000 rays produce a reliable result in an auditorium with set transition order to two or three [38].

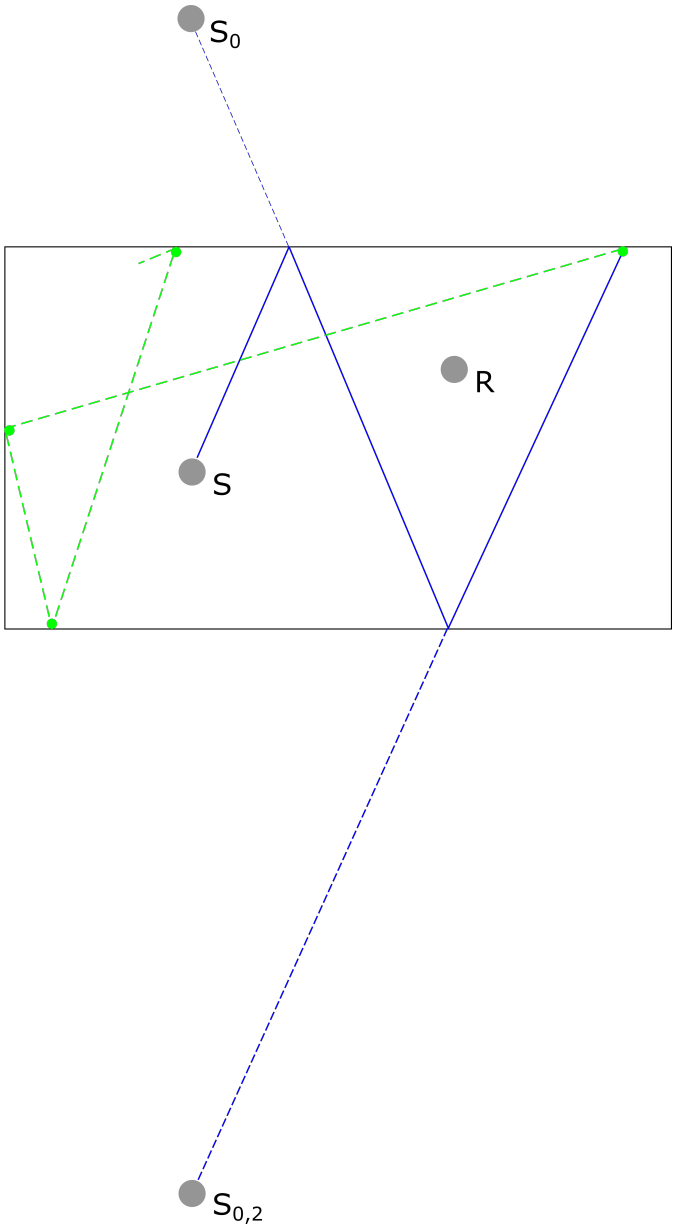


Figure 2.8: An example of how secondary sources are computed. The blue lines are the paths calculated by the image source method. Instead of the creation of the third order image source, a secondary source is created that creates additional secondary sources.

2.1.7 Other Sound Models

In addition to the approach ODEON is using that is described in the last section, other sound models which take the geometry of the environment into account exist. *RAMSETE* [10] is one of them. It uses a pyramid tracer that has the advantage of covering the surface of a spherical source completely, avoiding multiple detection of the same image source and avoiding overlapping cones. The RAMSETE approach is a ray-based approach. *RESound* [52] also is a ray-based approach that uses a combination of discrete ray-tracing and frustum tracing for the calculation of the sound propagation paths. For late reflections, a statistical acoustic model is used. A GPU-based approach named *iSound* [31] uses a modified image source method and a multi-view ray casting algorithm that allows the parallel computation of image sources on GPU. In addition to that, a scheme for reducing audio artifacts was developed. The iSound sound model is a ray-based approach. Wave-based Acoustics for Virtual Environments (WAVE) [27] is a wave-based sound propagation model that uses a combination of precomputation and runtime calculations. In the preprocessing stage, the dynamic transfer operator of the scene is calculated by using the per-object and inter-object transfer functions. The duration of this stage depends on the amount and details of the objects. During runtime, the outputs of the player devices (controller and Head-Mounted Display (HMD)) are used as inputs for the WAVE sound system that auralizes the scene in real time. The calculations are done on CPU and GPU.

All the above mentioned sound models could also be used in games and therefore in audio games. However, in this thesis a hybrid model approach similar to ODEON was chosen for testing sound models in audio games.

2.2 Audio Games

Audio games are described and discussed in this section. First, a definition and then fundamentals of audio games are discussed. Related work in form of a history of audio games closes this section.

2.2.1 Definition

There is no existing scientifically reviewed definition of the term *audio games*. Wikipedia, as the only source that provides a definition, defines audio games as *electronic games played on a device such as a personal computer. It is similar to a video game save that there is audible and tactile feedback but not visual.* [60]. This can be used as basis, since it states that the visual aspects are missing from audio games completely. There are several papers, e.g. [8, 9, 51] on this topic that do not define audio games but describe and use this term in their descriptions as a definition. Literature state that in audio games information is acquired through spatialized and iconic cues [8, 9] and it can be distinguished between two types of audio games; those which use spoken descriptions of situations and those which use non verbal audio cues [51]. To give a definition in a scientific context including the given references, this thesis defines audio games as:

Audio games are a subcategory of computer games played on electronic devices without having visual but auditive or tactile output as feedback for the user. Auditive feedback is given in form of spoken descriptions, non verbal audio cues, spatialized cues or iconic cues.

The described audio games in this thesis do not have their main focus on text-to-speech games, although they are also games with auditive feedback.

2.2.2 Fundamentals

This section describes fundamentals of audio games for further understanding of the creation process and the elements of an audio game. The description of sonification and interaction is followed by a discussion about the imagination part in audio games.

2.2.2.1 Sonification

Due to the lack of visual feedback, interactions or objects in a virtual world of an audio game have to give auditory feedback to the user to signalize their existence. Sonification is a technique that is used to transform non verbal sound [18]. Examples of sonification in daily life are the beep of the stereo rangefinder in a car, the ticking of the Geiger counter or the beep of a fire detector. In the context of audio games, Röber and Masuch identified several information groups and summarized them in three questions [39]:

- Where is something?
- What is this?
- What can I do with it?

These questions do not only relate to things, they also relate to and can be triggered by interactions (see Section 2.2.2.2 for interaction types). The first question determines the place of an object in the auditive VE. Those objects help users with orientation and support them in building an image of the virtual world in their head. They even let them reproduce it in real life [8, 45]. In audio games, the sound component is the only source for orientation purposes. Without any audio source, the player is unable to locate, orientate and navigate himself or herself in a virtual world [39].

The second question (*What is this?*) asks for a description or an intuitive purpose of an object in the virtual environment. Without any response, the player is unable to identify objects that makes them auditive *invisible*. Röber and Masuch describe this phenomenon as *inaudible and not perceivable* [39]. As a result of that statement, they declare that every object in a virtual environment is interactable. They classify interaction groups as: obstacles, portals, and interactables [39].

The third question *What can I do with it?* asks for the relationship between an object and the possible interactions a player can do with it.

Obstacles are insuperable objects in the virtual environment. The authors describe objects as elements which give shape to the environment, e.g. walls. Generally, interaction is not possible with those elements. The only possible interaction is *collision and obstruction* [39]. In an

auditive environment, such elements also have to be detectable by the player. Therefore they must have sonification, like an increase of volume if the player comes closer [39] or even a sound if the player collides with the obstacle [9]. Those sounds could also be used to describe and represent the actual context (e.g. a wall out of stones could have another sound compared to a wall of wood) [41]. As an alternative to the sonification approach, haptic feedback is also a possibility to provide information about the virtual environment [40].

Portals are objects in an audio game that bring the player from point A to point B. This transportation process can or cannot happen instantly. Examples for portals in games are doors, stairs, and teleporters [39]. Feedback while passing through or walking on portals provides the player with information of using them.

Interactables are the third category described by Röber and Masuch. These are objects in the virtual world, someone can interact with, like a button, a phone or a machine. Users know about the functionalities of these examples but the player must receive feedback when they are not active yet. When a game e.g. simulates a living room and the player's next mission goal is to phone Ms. X, the player must know where the phone is located in the room to do his or her call and to proceed in the game. Even when the player knows the position of the phone, he or she needs feedback about whether the interaction with the object was successful or not. If the sound of an interactable is unclear, the authors [39] suggest to describe it at the first interaction verbally. This should help the player recognize the object later. It is also possible to highlight interactables when the player focuses on them (e.g. additional sounds as feedback or lower environmental sounds), to provide him or her with the necessary information of the possibility for interaction.

Figure 2.9 shows the discussed object types of an audio game. Green elements are obstacles that are insuperable and define mostly the border of the scene. The blue elements are portals, e.g. the floor, the door an the stairs. The phone and the paper (red) on the table are interactables that the player can use.

The next section describes different interaction types and their challenges.

2.2.2.2 Interactions

Interaction is the relationship between the input of a player and the subsequent actions in the virtual environment triggered by them [2]. This section describes the different types of interactions in an audio game itself, with some information containing different input hardware devices and their usage. The interaction types in an audio game can be separated into navigation, interaction with objects and communication [39].

The navigation in audio games happens through an input device. This is handled by the player to move an avatar or a cursor to change its position. As stated in Section 2.2.2.1, the player must gain feedback according to the selected interaction type. With that information, the player knows if a set interaction was intended or not. If the audio game is played in front of a computer, Röber and Masuch suggest using navigation with a keyboard or joystick rather than with a mouse [39]. The reason for that is the challenge to place the virtual cursor (when the movement is done with point and click) to the right position, so that the character can move to

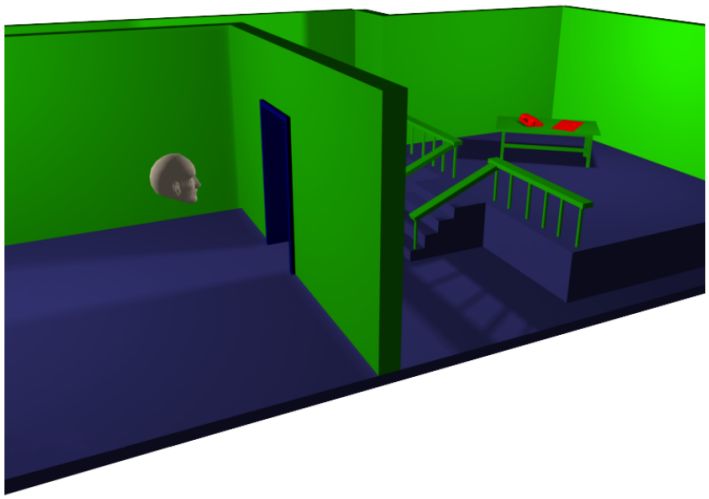


Figure 2.9: Examples of obstacles (green), portals (blue) or interactables (red) in an audio only environment. Illustration taken from [39].

it.[1] Another reason for this sub-optimality is the loss in concentration during the movement of this virtual cursor. In audio games, the player should put his or her whole concentration to the game's audio sources [39]. The authors also recommend using a joystick with haptic feedback. This is easier and better suitable compared to a restrictive keyboard control [39].

The interaction with game objects is the second category. This interaction type depends on the interaction possibilities in the particular game. In AudioDoom f.i., the player is able to interact with a door (which is a portal, see Section 2.2.2.1 for explanation) to open it; therefore the game implements the interaction open a door [45]. One possible interaction in a classical audio only Mastermind game, is the movement and placement of the appropriate stone in the right place that implements the interactions move and place stones [51]. As mentioned in Section 2.1.2, a realistic simulation of the virtual environment with the support of head tracking is a solid basis for a 3D environment. Two interaction techniques are described by the authors Röber and Masuch [39]. One is the auditory cursor that is an object in the 3D environment that describes — with speech, artificial or natural sounds — objects that intersect with it. With this metaphor, interaction is eased. The other technique is called radar that can be described as a cone of a flashlight. Every object that is hit by the light of the cone emits information about their position and functionality.

The third interaction category is communication. This communication is done through speech. An interaction can be initialized by artificial characters or the player itself. In multiplayer games, this interaction type can be used for chatting between players [39]. Bidirectional communication between the player and an artificial character may work on basis of predefined routines.

[1] An example of a point and click adventure is the game Day of the Tentacle. It was released in 1993.

2.2.2.3 Imagination

The last element of audio games that every game has in common, is the visual imagination of the player. Liljedahl et al. introduced a term for this named Scary Shadow Syndrome [20]. They derive this concept from older horror films, where the budget as well as the advances in technology were limited. As example, they list the production of the film Jaws. Instead of filming a shark attack on a female swimmer, the director only filmed the reaction of the girl — due to technical reasons — which triggered the personal imagination [20]. Therefore they advice that a game designer should not tell or show everything to the player but let them construct their own interpretation and imagination of visual and auditive feedback. Another example which supports the advice of Liljedahl et al. is shown in the paper by Velleman et al. [55]. During their research in the development process of Drive, they asked themselves the question, which of the following examples make more fun: *Steering but traveling with 5 mph or no steering but traveling with 700 mph?* [55]. They concluded that it is the latter one because faster speed is more thrilling in the imagination of a player.

2.2.2.4 Game Genres & Sound Types

The website *audiogames.net* shows a high variety of genres and types of audio games. The listed genres include action games, adult games, adventure games, arcade games, card games, compilations, educational games, first person shooters, gamebooks, interactive fiction, multi user dungeons / mmorpgs, puzzle games, racing games, role play games, side scroller, simulations, sports, strategy games, traditional games, trivia games and word games.[2] This range of genres shows that the missing visual output which is typical in audio games gives no restrictions for developers to choose a genre. Another classification of audio games could be done by analyzing the used types of sounds, which are e.g. spoken texts or ambient sounds.

Röber and Masuch did a classification on their so called auditory phenomena concerning audio games, where they decided to group different types of sounds into three categories: speech, music and natural or artificial sounds [39]. Speech is mainly used for uni- or bidirectional communication between elements in audio games in form of spoken words. The audio game Sleuth makes a heavy usage of speech for its interactive narrative [9]. There are different applications for using music as additional sound source in an audio game. Depending on the used type of music, it may influence the user's mood and even the feeling during play [39]. A slow paced music can be used to communicate a chill-out ambiance to the player. It is also possible to use music for sonification of light intensity changes or to highlight someones attitude (e.g. usage of dark sounds for a virtual person in a game to state that this is one of the bad guys) [39]. The third category, natural or artificial sounds, has the biggest variety, postulated by Röber and Masuch [39]. The sound of flying birds or the splashing of water are examples for natural sounds. Artificial sounds therefore are e.g. signals that the player recognizes as a defined sound of something, like an item or an object (see Section 2.2.2.1 for examples of artificial agreed sounds). Friberg and Gärdenfors of Stockholm International Toy Research Center also did a categorization of sounds. This categorization was derived from their Tactile Interactive Multimedia project. They divided

[2]These genre types were taken from the audiogames.net website. This website has the author's recommendation for people who are interested in audio games.

their findings into five categories: avatar sounds, object sounds, character sounds, ornamental sounds and instructions [12].

2.2.2.5 Environments

There are several environments in which audio games can be played. Those are listed as follows:

- without sitting in front of a desktop computer e.g. lying on a couch [28],
- in front of a desktop computer without headphones e.g. [45],
- in front of a desktop computer with headphones e.g. [9],
- and real walking in an open space e.g. [55].

The authors of [28] show that audio games may be played in a relaxed atmosphere, e.g. on a couch and provide an immersive feeling. Playing without headphones is not recommended [9], however, the authors of [45] showed that it is working with speakers though. Real walking in an open space supports the presence [50], therefore, the authors of [55] proofed that audio games are working in an open space environment.

2.2.3 Games in Scientific Context

This section covers selected audio games developed in a scientific context published by members of the research community. However, this section does not cover every game or paper. Only audio games that show a new component compared to other papers are covered in this section. Most of the following games are designed for visual impaired users but are also playable and suitable for users without impairments [51,58]. At the end of the section, papers are listed which show the development of audio games in the scientific context.

Lumbrears and Sánchez published a paper in 1999 called Virtual Environment Interaction through 3D Audio by Blind Children which was the first playable, in a scientific field built audio game in a 3D virtual environment [45]. In this paper, they describe and test their game prototype AudioDoom, which is an audio adaption of the classical Doom game with reduced functionality. In this context, the authors use hyperstories. Therefore, this virtual world has several environments that are represented as node in the hyperstory context. These environments could involve dynamic objects or characters that react to the behavior of the player. However, these environments are connected with each other through predefined links that are usually represented in the virtual world as doors or portals. In addition to that, they use narratives to generate a deeper immersion of the game. The players are able to steer their characters with a joystick while they get auditive feedback through speakers. The authors try to test their hypothesis that those virtual environments (the audio game itself with all associated hyperstory components) can create mental images [45]. To verify that, they let blind people play the game several times. After they have created their mental image, they are told to rebuild the played level with Lego blocks. Each block has a different meaning, f.i. a door in the virtual world is represented by a Lego window in the real world. As conclusion, the participants are able to rebuild the level with Lego blocks successfully. This proofed their proposed hypothesis.

Figure 2.10: GRAB haptic interface for in- and output. Picture taken from [62].

Drewes et al. published a paper in 2000 called Sleuth: An Audio Experience about their identical named game Sleuth [9]. Sleuth is an audio version of the classical game Clue in which the player takes the role of a detective who has to investigate a murderer case to determine the killer, the weapon and the room through asking people. The player can move his detective avatar through the house in different rooms. Every room has its own sound sources. The player receives a detective's notebook, which has names of guests, weapons and rooms in it. Therefore the game provides the player with additional information through a paper but does not provide any visual output. After gathering enough information, the player can make a guess and solve the murderer case. If he or she is right, the game is won, otherwise he or she has to investigate further. The goal of the evaluation part was to *examine the effectiveness of our [their] design decisions in a qualitative manner* [9]. After the participants played the game, the authors asked them several questions about their clues and guesses to evaluate, if their design decisions were effective or not.

Another paper called The Design and Evaluation of a Computer Game for the Blind in the GRAB Haptic Audio Virtual Environment was published in 2003 by Wood et al. [62]. They used — compared to the other presented papers so far — a combination of haptic in- and output as well as auditive output, which they call Haptic Audio Virtual Environment (HAVE). In this environment, the haptic device is called GRAB HAVE which was developed by PERCRO. This device is completely steerable with two forefingers and consists of two arms with three Degrees Of Freedom (DOF) and three DOF in mobility [62]. Figure 2.10 shows the GRAB interface in action, operated by a user. GRAB detects collisions between the player and virtual objects and return them as haptic feedback. The authors decide to use a search and adventure game as test genre for their study. Therefore, they built an virtual world with two rooms, one (locked) door, a key (to unlock it), one attractive trap (which captures and immobilizes one finger), an attractive trap deactivator (which is near the trap itself), two bombs, one bomb deactivator, lives and points. If the player collects all two points, the game is won. If the bomb explodes, the game is lost. To collect items, the player has to tap three times in a row to collect it. By pressing a button or opening the door, the player receives corresponding sounds like a clicking or knocking. The participants in the playtests stated that the game was too simple in term of challenges, but the GRAB environment is suitable for gaming purposes.

Velleman et al. published a compilation and results of three tested audio games in 2004 [55]. These games are called Drive, Curb Game and Demor, while two of them have special properties

which are highlighted in this paragraph. Drive is a racing game without the possibility to steer the car. The authors tried to focus completely on the sense of speed due to their assumption that they should sonify the essence of a game instead of visual feedback. The objective of this game is to collect boosters and use them to gain additional speed, but it gets harder to collect boosters due to higher speeds. After three minutes, the game is over and the score (reached distance and collected boosters) is calculated. Players then have the possibility to upload their scores and compare them to others. Until 2004, over 50.000 people have downloaded the game and as a conclusion of this study, the authors stated that blind users had reached higher scores than non handicapped people. The Curb Game is the second game in this paper which is very similar to Frogger. The third game Demor is a location based 3D audio first person shooter. The player wears a backpack with a laptop (including GPS functionality) and headphones with a head tracking module. In addition to that, the player also has a joystick. This game is supposed to be played outdoors. The GPS and the headphones provide the laptop with the required information. If the player hears an enemy, he or she has to look in the direction of it and pull the trigger of the joystick. Depending on the accuracy, the player gets points. This is the first recorded 3D outdoor audio game.

A case study about Terraformers, an often cited computer game (f.i. [12, 40, 41]), was published in 2004 by Westin [58]. Terraformers itself was released in 2003 and was the first commercial hybrid 3D audio game for sighted as well as for blind people [58]. They use a mixture of 3D sound and voice feedback to substitute missing visual sense. In the game, there is a compass which is represented by spoken feedback, telling the player the rough direction. Another element which supports visually impaired people is a sonar. With this device, the player is able to estimate the distance to objects in the field of view. By pressing a key, the game describes the object. Every object in the game has voice feedback, 3D graphics as well as 3D sound icons. This game offers three different graphic modes. The first one is the standard one, where every object is rendered as intended. Then there is a second mode, called High Contrast Mode, where unimportant objects (like walls) are rendered in black or white while important objects are rendered as in the first game mode. The advantage of this rendering method is that the important objects get additional contrast and are therefore better to recognize. This mode offers low vision gamers the possibility to play the game. The third mode is the No 3D Graphics Mode that completely disables graphic rendering. Even in this mode, the game offers possibilities which allows sighted as well as blind people to play this 3D game.

The paper published by Heller et al. does not demonstrate the possibilities of 3D audio in form of games but rather in form of a historical installation in the Coronation Hall in Aachen called CORONA [17]. The audio installation places several virtual characters into a hall, which are calculated on mobile devices that visitors get. Visitors also get a headphone with a compass sensor for auditory output as well as tracking of the head. People then had the possibility to move through this virtual environment and to interact with the virtual audio characters. The authors did some preliminary tests to identify the best technical solution within the restrictions of the hall. They also tested an optical tracking system as possible technology but found out that this method would need too much fine tuning and it would not fit in this special environment.

The research team Merabet et al. published a paper named Teaching the blind to find their way by playing video games in 2012 [8, 29]. In this paper, the authors transferred the knowledge

of a VE to the real world. The Audio-based Environment Simulator (AbES) allows a simulation of an existing building for navigation and exploration purposes in form of a game metaphor [29]. Therefore, a building was recreated in virtual space and blind people were able to explore this space on a computer to get the environment to know. After some iterations of gaming (there are monsters, jewels and an exit), the blind participants were able to orientate themselves with their generated cognitive map of this building in this real world structure. One application for this technology could be that blind people may get an foreign environment to know, before they visit it in reality for the first time [8].

Since the start of audio games with the introduction of AudioDoom, technical as well as scientific progress advances the possibilities of games without visual feedback. The work of the authors Velleman et al. [55], Heller et al. [17] and Merabet et al. [29] indicate a trend of leaving the computer screen at home to enjoy audio games in the wild. Table 2.1 shows a summarization of audio games related scientific papers, illustrating the authors, years, games and the main characteristics.

2.3 Software Tools

In this thesis, two software tools are used for implementing and testing a hybrid sound model. Both are introduced in this section. Unity3D is a 3D game engine that supports 3D audio. Wwise is an audio middleware solution that can replace an existing audio engine of e.g. a game engine like Unity3D completely.

2.3.1 Unity3D

Unity3D [54] is a runtime- and development environment for interactive real-time applications or computer games. The basic version is free of cost with reduced functionality. However, the professional version requires an annual fee. Unity3D offers state of the art in graphics, animation, sound and programming.

Unity3D uses a scene graph for storing game objects. In this graph, objects, attributes and relationships can be modeled. A *game object* exists in virtual 3D space and has a position, rotation and scale. Game objects can be selected and modified in the *scene view*. Primitive game objects like cubes or spheres can be added through a menu. In Unity3D, *components* can get attached to game objects that are e.g. information about the material of the game object, audio files or scripts. Unity3D supports C#, C++, Boo and UnityScript. The *game view* of Unity3D shows the application from the perspective of the application camera and allows simulation of the application in runtime. Figure 2.11 illustrates an example of the Unity3D layout.

Unity3D uses the FMOD bibliography that allows spatialized 3D sound. FMOD is a sound playback and mixing engine. Several effects, like Doppler- or hall-effects can be applied on sound sources using this integrated bibliography. However, it is not possible to calculate geometry dependent echoes [53]. For sound, Unity3D provides an *audio source* and *audio listener* component. The audio source component allows changes including volume, pitch, maximum distance and effects regarding the Doppler effect and is responsible for the playback of the

Authors	Year	Game	Special
Lumbreras & Rossi [21]	1995	Hypertext story	First 3D audio selectable hypertext story.
Lumbreras et al. [22]	1996	Hypertext story, grab-and-drop technique	Virtual reality glove to grab and drop objects.
Lumbreras & Sánchez [45]	1999	AudioDoom	First 3D audio game, proof with Lego blocks.
Drewes et al. [9]	2000	Sleuth	Combination of audio game with physical paper.
Winberg & Hellström [61]	2000	Towers of Hanoi	Only use earcons as auditory feedback.
Targett & Fernström [51]	2003	Os & Xs, Mastermind	Use of earcons and auditory icons.
Wood et al. [62]	2003	GRAB HAVE	Haptic in- and output and auditive output.
Sánchez et al. [43]	2003	AudioBattleShip	First collaborative audio game.
Velleman et al. [55]	2004	Drive, Curb Game, Demor	Online highscore comparison. GPS, joystick, outdoor game.
Westin [58]	2004	Terraformers	Commercialized and awarded hybrid audio game.
Mendels & Frens [28]	2008	Audio Adventurer	Relaxed context and use of own physical input device.
Heller et al. [17]	2009	CORONA	Interactive audio installation in a coronation hall.
Merabet et al. [29]	2012	AbES	Knowledge of virtual world transferred to real world.
Sánchez et al. [44]	2014	Audiopolis	Simulation of whole city with tactile feedback.

Table 2.1: Summarization of published papers related to audio games in scientific context. 2.2.3.

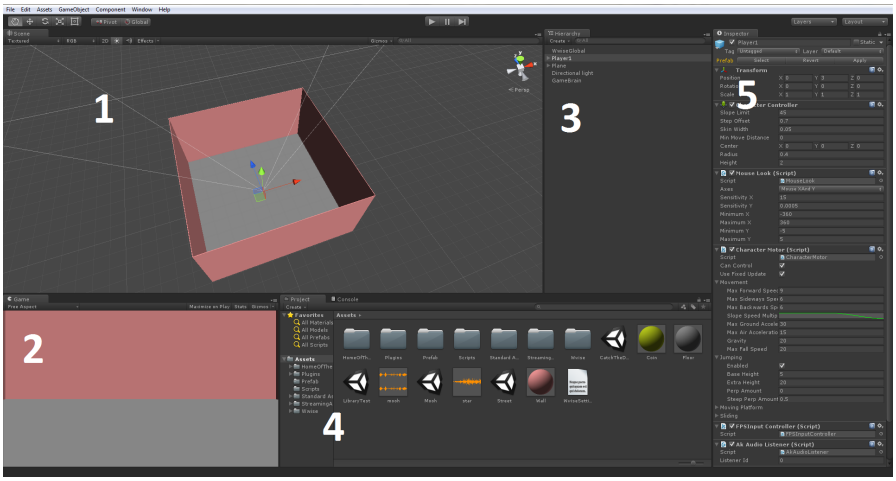


Figure 2.11: The Unity3D layout: (1) scene view, (2) game view, (3) scene graph, (4) project explorer, (5) inspector.

sound source. On the other hand, the audio listener component is attached to the game object where the sound should be received, e.g. the player's avatar.

Another principle that is used in this thesis in Unity3D is ray casting. Ray casting can be used to detect obstacles between two points in 3D virtual space. A ray is shot from a 3D position into a 3D direction. Depending on the cast method, one or more obstacles (game objects) that intersect with this ray are detected and returned. This principle is used in the image source method for sound source mirroring and for the audibility test. In the secondary source method, ray casting is used to calculate the reflection angle as well as the position of the secondary sources.

2.3.2 Wwise

Wwise is an audio middleware solution that provides an authoring application, a sound engine, a game simulator, a plug-in architecture and an interface that allows communication to and from external world builders [4]. Audio middleware solutions can be used as replacement of an already existing audio engine. Wwise is used in modern computer games e.g. *Assassin's Creed Unity* (Ubisoft Montreal), *Borderlands* (Gearbox Software), *Destiny* (Bungie) or *Metal Gear Solid V* (Konami) [3].

Wwise provides access to general sound settings e.g. volume or pitch, allows 2D and 3D sound positioning and Real Time Parameter Control (RTPC). RTPC enables access to sound settings from outside the application during runtime, e.g. the change of the pitch of an already playing sound file. These options can be used for the spatialization of sound and the integration of sound models. In addition, Wwise provides several sound effects e.g. delay, flanger or reverb effects. The provided reverb effects do not take any geometrical information about the environment into account. Wwise uses a bus system to which the sound events are routed. Several sound events can be directed to one bus. In these buses, settings including bus volume or bus

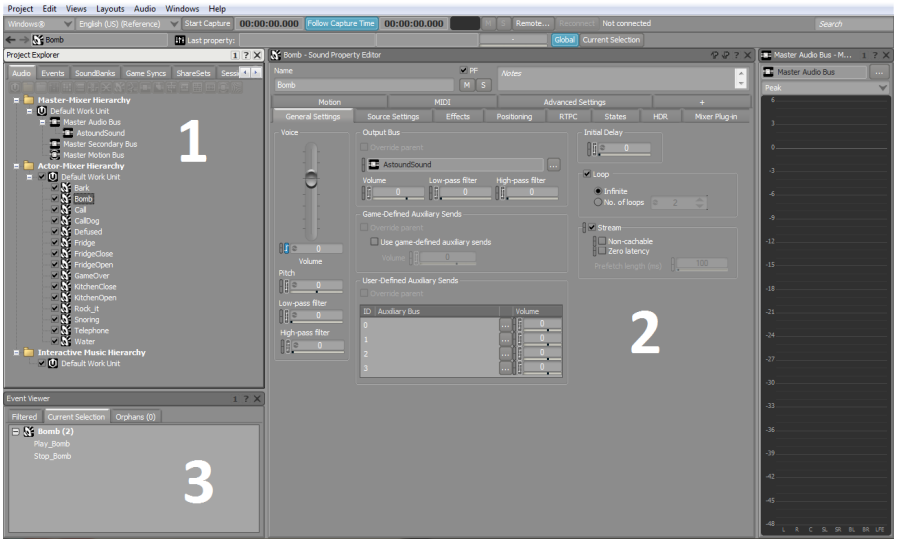


Figure 2.12: The Wwise layout: (1) project explorer, (2) sound properties, (3) event viewer.

effects can be added and are valid for every sound that is routed over the respective bus. When a sound plug-in is used, e.g. GenAudio's AstoundSound, it is attached to an audio bus. Therefore, every sound that is routed through this bus is spatialized in 3D, as the AstoundSound plug-in is responsible for sound spatialization. The elements of Wwise are discussed in the following paragraphs.

The *authoring application* of Wwise is a standalone software in which sound files can be managed, modified, mixed and saved into SoundBanks. SoundBanks are a collection of event and sound data that can be loaded at specific points during a game to increase performance. Events in Wwise can be used to e.g. start or stop a sound file when fired. These events need to be created in the authoring application first, otherwise they are not available. Figure 2.12 shows the layout of the authoring application.

Wwise's *sound engine* manages audio and performs functions. The sound engine is optimized and available for different platforms: Windows, Max, iOS, Android, Linux and Windows Phone.

The *game simulator* consists of a Lua (programming language) script interpreter that allows the reproduction of sound and motion behaviour in a game to monitor specific behaviours and the performance of Wwise before the project is integrated into a game's sound engine.

The *plug-in architecture* of Wwise allows an expansion of available functions. Sound plug-ins can be used to produce artificial sounds or to create sound effects, e.g. reverb. However, it is also possible to use third-party plug-ins, e.g. GenAudio's AstoundSound. A discussion about tested plug-ins for this thesis can be found in Section 3.3.

SoundFrame is an interface that allows communication between Wwise and an external game world builder or 3D application. Through this interface it is possible to access the same functionalities as in the authoring application.

Figure 2.13 shows the production pipeline of Wwise. In the middle of this image, the basic

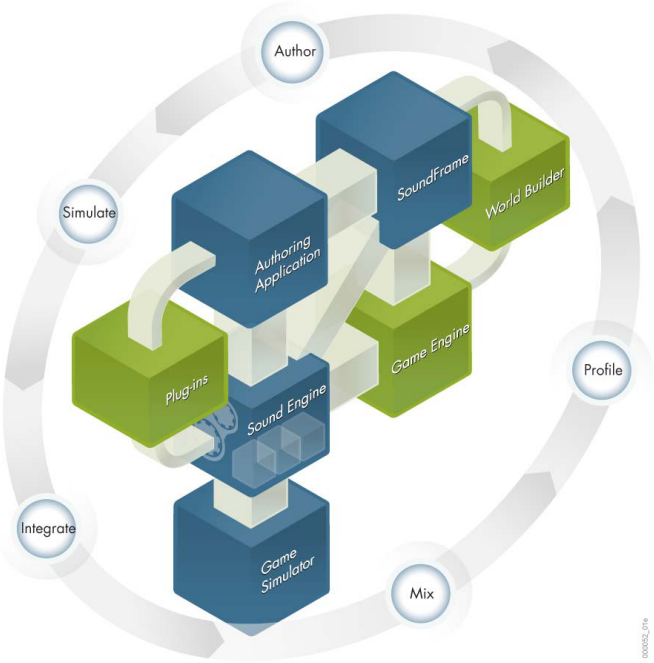


Figure 2.13: The Wwise production pipeline. Author: building of sound and properties, Simulate: first testing and simulation, Integrate: integration without programming, Mix: mix properties in real time and Profile: profiling game during runtime. Image taken from [4].

tools of Wwise (which are discussed above) and their connections are shown. In the outer circle, Author, Simulate, Integrate, Mix and Profile determine the Wwise pipeline. *Author* defines the first step that includes the building of sound and its properties. *Simulate* is the first step of testing and simulation, if everything is working as intended. *Integrate* describes the option that the integration can be done without additional programming. *Mix* stand for the possibility of mixing properties in game in real time. *Profile* is the possibility of profiling the game during runtime.

A full overview of the functionalities of Wwise can be found in [4].

CHAPTER 3

Design

This chapter presents the environment in which the audio game prototype was implemented and discusses the restrictions that the test environment imposed. Afer that, an audio middleware solution and sound spatialization plug-ins available for it are tested and evaluated to find the most appropriate technology for the implementation of the hybrid sound model. Finally, the design of the baseline model and the hybrid model are presented.

3.1 Test Environment

As discussed above, audio games can be played in different environments (see Section 2.2.2.5), e.g. on a couch, in front of a computer or in an open space by using real walking. In this thesis, an audio game in which the player is able to use real walking for the movement of the virtual avatar is used. The player is able to move with full freedom in an open space, therefore tracking is required to locate the player's position in the real environment. The position of the player must be known for the calculation of the sound model and real-time spatialization.

The test environment was already available at the begin of the thesis and consists of a room, a virtual model of it and a tracking solution. The room in which the audio game takes place is modeled virtually in Unity3D. This model is shown in Figure 3.1. The room boundaries, pillars and other obstacles are modeled into the virtual environment. The optical tracking system has been developed in Vienna University of Technology and uses a camera mounted on the Oculus Rift DK2 worn by a tracked user. On the ceiling of the room, markers are placed and detected by the camera to determine the player's position in the room. To calculate the rotation of the player's head, tracking information is combined with the information of the Oculus Rift DK2 sensors. The size of the testing area is 30m x 7m.

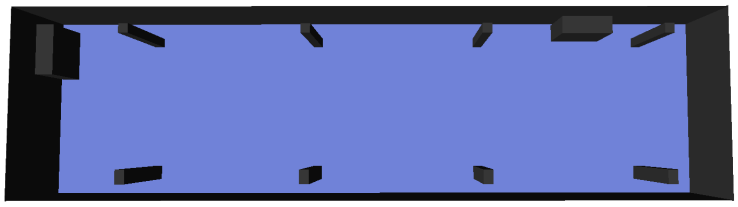


Figure 3.1: The 3D model of the test environment.

3.2 Requirements

Sound processing software and sound propagation models have to be chosen in accordance with the following requirements.

3.2.1 Real-Time

In a game, users have to experience the immediate results of their actions. This is also valid for audio games. Such actions could be the change of the view while moving the viewport. In this test environment, the movement of the viewport is done by real walking. In addition to that, the player must perceive changes in sound when moving or rotating the view otherwise it would not be perceived as realistic and therefore immersive.

In regular computer games, real time calculations of graphics and sound are important to provide a realistic and immersive feeling. However, in audio games there are no graphics at all. Therefore, only the sound component needs to be calculated in real time, including changes in sound when the player moves the body or rotates the head.

A sound model in an interactive scene has to be calculated in real time, otherwise user's will perceive a significant delay. This delay reduces the immersion and realism of the sound model. If real-time calculations during the experience is not possible at all, players might not be able to localize sounds which will make an audio game experience less convincing and less enjoyable.

As conclusion, sound software and particularly sound calculation methods should be able to perform in real time. This also concerns the sound spatialization process that is described in the next section.

3.2.2 Sound Spatialization

In real life, humans are able to distinguish the direction of a sound event (see Section 2.1.2). Computer generated sound needs to be processed in a way that simulates the same cues that allow people to distinguish the directions from which sounds come in real life. For this purpose, HRTFs can be used (see Section 2.1.5.1).

The sound spatialization depends on four parameters. The position and rotation of the listener as well as the position and rotation of the sound source. The rotation of the sound source is only necessary when directed sound is used. When these positions and rotations are known, the most appropriate HRTF can be chosen. Therefore, the azimuth, elevation and distance between

the sound source and the listener needs to be calculated. With these three parameters the most appropriate HRTF is determined. The HRTF is then applied on the sound (convolution or (inverse) FFT) to simulate the absorption of the outer ear, head and torso. This provides the player with spatialized, realistic sound.

In audio games, sound spatialization is crucial. Without spatialization, Sonification (see Section 2.2.2.1) cannot give an answer to the question *Where is something?*.

As conclusion, a software is necessary that calculates the sound spatialization regarding the position and rotation of the listener and the receiver in real time.

3.3 Sound Plug-ins

As stated in Section 2.3.2, Wwise is used as an audio middleware solution for the implementation of the sound effects in the proposed audio game prototype. However, there is additional software available for sound spatialization in real time. This section investigates possible combinations of the audio solution at the game engine (Unity3D), audio middleware solution (Wwise) and additional audio plug-ins. The quality of sound provided by the tested combinations is compared to find the most appropriate software solution. When Wwise is tested, sound plug-ins can extend the software's functionality for sound spatialization. The following software configurations are tested:

- Unity3D audio without any additional software,
- Unity3D with Oculus Spatializer,
- Wwise without any plug-in,
- Wwise with AstoundSound plug-in,
- and Wwise with Auro-3D plug-in.

For testing the built-in sound engine, the audio middleware solution and the available plug-ins, a simple Unity3D project was developed.

3.3.1 Plug-in Test Prototype

A simple prototype scene was created to test the listed sound software. In a test scene, a cube is constantly moving on the horizontal orbit around the player and continuously emitting sound. There are three rotation phases. In the first phase, the cube orbits the player horizontally. After it completes a 360 degrees loop, the second phase starts and the cube orbits the player vertically. After completing this phase, the third phase starts which is the same as the first one but at a different height so that the player can perceive sound coming from above. With these three phases, the player can experience the sound of the cube coming from different directions. By pressing predefined keys, the sound calculation method can be switched between the different sound solutions. A mouse can be used to rotate the character controller and therefore the virtual listener to perform an equivalent of the dynamic sound localization.

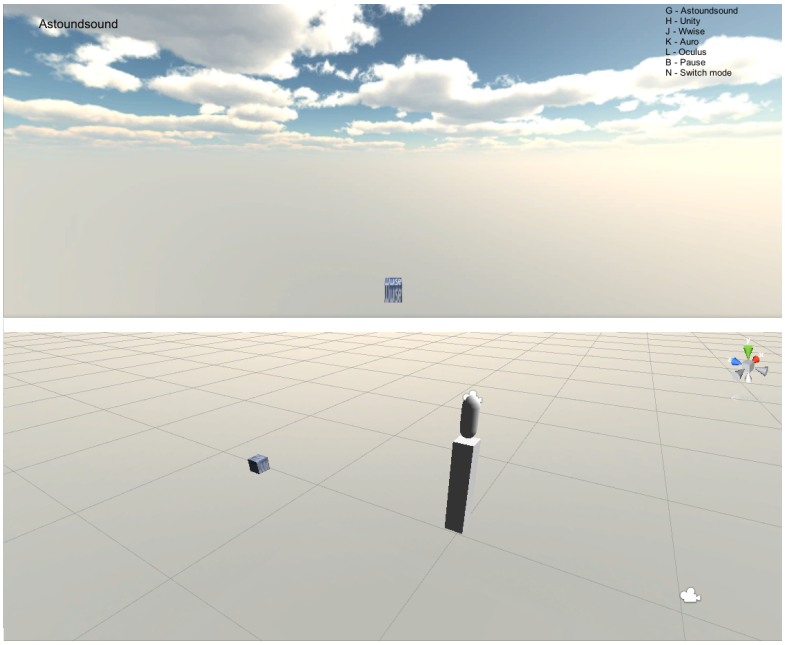


Figure 3.2: The prototype which was used for testing of the different sound plug-ins and the built-in engine. In the image at the top, there is the actual game view, at the bottom the scene view. The cube represents the sound source moving around the player, represented as the capsule.

Figure 3.2 shows this prototype. In the top image, the game view can be seen. The cube in front is the rotating sound source. The active sound solution is displayed in the top left corner, the available sound solutions in the top right one. The bottom image shows the scene, including the player controller, the camera and the rotating cube.

This test was conducted in a silent environment with closed headphones and a mouse. Different sound solutions were tested several times. We were able to make the following conclusions.

3.3.2 Results

Unity3D audio without any additional software
The Unity3D sound engine uses FMOD bibliography. The game engine and therefore the already integrated sound bibliography is freely available. More details about Unity3D sound can be found in Section 2.3.1. The tested game engine and therefore sound engine version was Unity3D 4.6.5f1. In general, Unity3D spatializes sound authentically, however, changes when moving the virtual head up and down were not perceivable. The calculated spatialized sound did not change.

Unity3D with Oculus Spatializer
This plug-in is freely available for sound spatialization. It can be directly integrated into Unity3D without using any audio middleware solution. Oculus Spatializer uses HRTFs for sound spatial-

ization. The tested version was Oculus Audio SDK 0.9.2 Release. The spatialization of a sound source which is in front of the player felt unnatural when moving the head horizontally. The spatialized sound when the sound source was in front of the player felt like standard stereo output that does not seem to be spatialized.

Wwise without any plug-in
The audio middleware solution can spatialize sound without any addition sound plug-ins. The software is available for free as long as no additional plug-ins are used. The positioning of a sound object in Wwise can be set to 2D or 3D. The tested Wwise version was Wwise v2014.1 (64-bit) Build 5158. Without any additional spatialization plug-in, this software produced the worst results of the tested sound solutions. Compared to other software, the sound did not feel realistically spatialized. Horizontal sound spatialization (changes in azimuth) was admissible, however, vertical sound spatialization (changes in elevation) were not perceivable.

Wwise with AstoundSound plug-in
Wwise in combination with AstoundSound spatializes sound around the listener in real time and provides distance cues [15]. The plug-in can spatialize from 0 to 359 degrees of azimuth and -90 to 90 degrees of elevation. These calculations are done with AstoundSound's own transfer functions called Brain-Related Transfer Functions (BRTFs) [15]. These functions model how the brain perceives sound spatially.

The sound was perceived realistically in all the situations. Changes in azimuth as well as in elevation felt realistically spatialized and clearly distinguishable. Compared to the other tested software, this was the most convincing one. The tested version was Mixer plug-in v1.5.

Wwise with Auro-3D plug-in
Auro-3D was originally developed for home theaters (up to 10.1 sound systems) and cinemas (up to 13.1 sound systems). It uses three different layers of sound [6]. The lower layer is responsible for elevation between -15 and 15 degrees. The height layer is responsible for sound spatialization between 15 degrees and 75 degrees of elevation. Spatialized sound at an elevation between 75 and 90 degrees is localized at the top layer. Depending on the vertical coordinates of a sound source, the sound gets assigned to these three layers.

The used version of this plug-in was Plug-in Version 1.0. The output was similar to the spatialized sound produced by Wwise. Changes in the sound source elevation did not make any noticeable difference, however, if the sound source was moved significantly, a reduction in sound volume was perceived.

The results show that the most appropriate technology for this thesis is a combination of Wwise with the AstoundSound plug-in that spatializes sound realistically. To test this software in an audio game, the following two games were implemented and investigated.

3.3.3 Game Prototypes

Two prototypes were implemented to investigate the appropriateness of Wwise in combination with AstoundSound in an audio game environment.

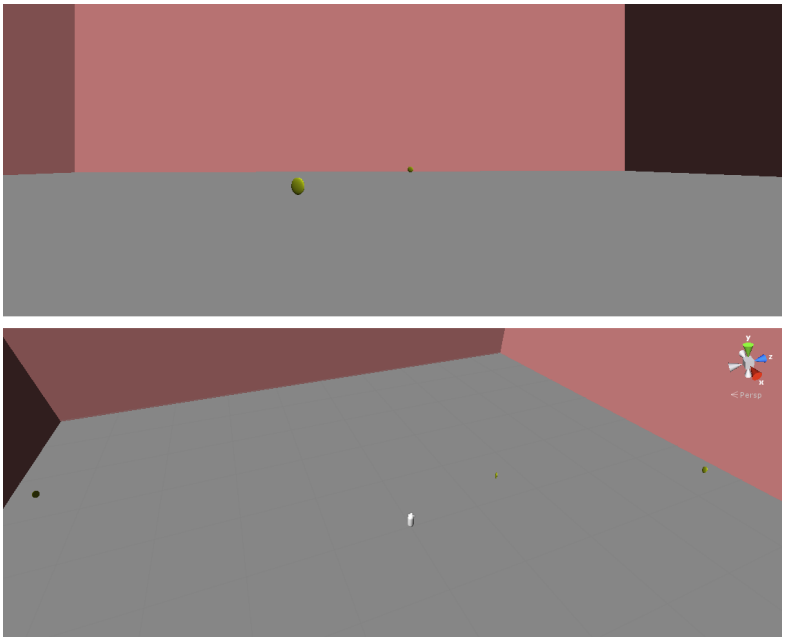


Figure 3.3: Screenshots of Catch the Dot for testing Wwise with AstoundSound in an audio game environment. In the image on the top, there is the actual game view, on the bottom the scene view.

The first game, *Catch the Dot* is a simple audio game where the player is inside a room. There are three coins randomly placed inside this room which continuously emit sound. The main task is to collect coins. After the player has collected the first three coins, another three coins are spawned and rain sound is played additionally. After collecting them, additional sound disturbances start playing and spawn another three coins. After the last coins are collected, the game is completed. The game was played blindfolded in front of a desktop computer with headphones. Figure 3.1 shows a screenshot (debug output) of the scene- and gameview. In the top image, the coin is visible in front of the player. In the image at the bottom, the character controller as well as two coins are visible.

Testing of this prototype showed that localization using this sound technology has worked in this environment. The spatialization of the sound supported the player in finding the coins. Disturbances made the localization process harder, however, the game could be completed.

In Catch the Dot, the sound sources are stationary. However, further investigations were conducted to see if the localization and spatialization work correctly when the sound sources and listener are moving at the same time. For this purpose, another game prototype called *Audio Frogger* was created.

Frogger (Atari, 1981) is a computer game where the player is in the role of a frog who wants to cross a street. On the street, there are several obstacles like moving cars that make crossing harder. If the frog is hit by a car the game is over. The level is successfully completed when the other side of the street is reached.

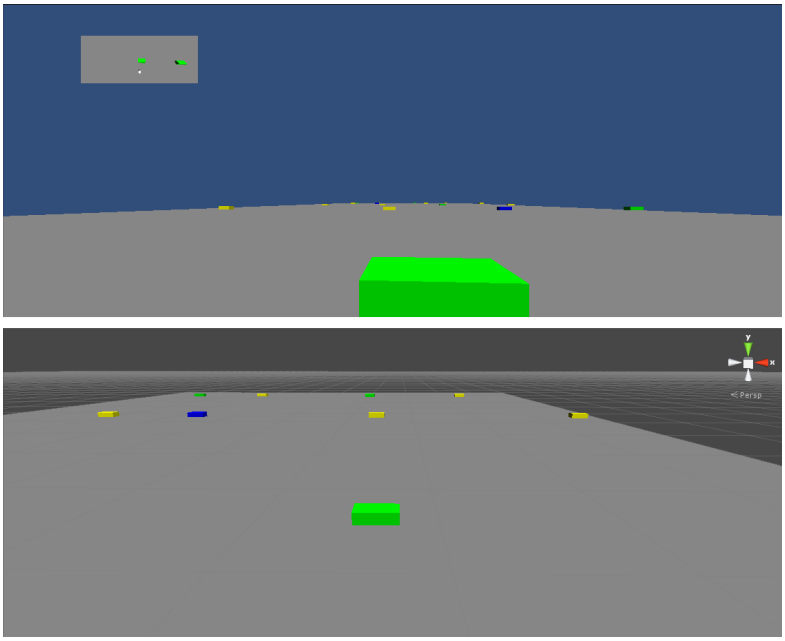


Figure 3.4: Screenshots of Audio Frogger for testing Wwise with AstoundSound in an audio game environment. In the image on the top, there is the actual game view, on the bottom the scene view.

Audio Frogger is similar to the original Frogger game, but it is played in 3D space. Again, the game is played without visuals at all. However, the debug output can be seen in Figure 3.4. There are several streets in regular intervals with cars driving from the right to the left side on the screen. These cars are emitting wheel noise sounds. The pitch is shifted depending on the distance between the object an the listener, the speed of the listener and the sound source to simulate the Doppler effect. Three different sounds are available: one of a regular car, one of a truck and the sound of a regular car paired with the sound of a police siren. With these sounds, the user is able to distinguish the direction of a car and can give an estimation of the distance between the user and the sound source. In addition to the continuous wheel sounds, the cars are honking randomly. Other sound disturbances appear from time to time, e.g. bird sounds or rain. With every passed street, the speeds of the cars are slightly increased. As soon as the last street is passed, the game is completed successfully. However, if a car hits the player, the game is lost.

This prototype proofed that the spatialization and localization is working when the sound sources and the listener are moving in an audio game environment using Wwise with AstoundSound plug-in.

Spatialization with Wwise and AstoundSound supported the player during the game. The player was able to fulfill the tasks only by listening to sound. Therefore it can be concluded that this technology is appropriate for an audio game. Thus, this technology is used for implementing sound models, which are discussed in the next section.

3.4 Sound Models

The tested sound plug-ins provide a possibility to spatialize sound in real time. However, these plug-ins do not provide any geometry-dependent reverberation. As already mentioned above, reverb effects that are available in the audio middleware solution do not take geometry into account. Therefore, an implementation of a sound model is needed that can simulate those effects. The model that is implemented in this thesis is an adaption of the hybrid sound model introduced in 2.1.6.

In this thesis, it is investigated if these additional sound effects matter in audio games (see Section 5.1 for the proposed hypotheses). Therefore, a model that is generally used in audio games is implemented as comparison. This model is hereinafter referred to baseline model and is introduced in the next section.

Both sound models should meet the following requirements for being applicable in this thesis:

- Spatialized sound calculated according to the position and rotation of the listener and sound source.
- Sound attenuation according to the distance between the listener and the respective sound source.
- Obstruction of sound sources through an applied low-pass filter on the respective sound source depending on the listener's position.

3.4.1 Baseline Model

The review at audio games in scientific context shows that all of them use similar sound models [9, 28, 29, 45, 55, 58, 62]. In this section, the common elements of these models are highlighted, discussed and adapted for the use in audio games.

Sound spatialization is the key component in all of the above mentioned papers. Without sound spatialization, an object cannot be localized in virtual space. In [45], the authors created 3D sound by preprocessing the game audio files first by combining different sets of HRTFs. In [28], they use stereo-panning for sound spatialization. However, in this thesis already existing sound technology that is able to spatialize sound in real time is used.

Attenuation over distance is described by one of the above mentioned papers [9]. If the distance between a sound source and the listener becomes bigger, the attenuation on the sound source has to be stronger and therefore the emitted sound is losing power. This feature is essential for the perception of distance and is implemented in the baseline model.

These studies [9, 28, 29, 45, 55, 58, 62] do not take the effect of obstruction into account. Obstruction might be appropriate for audio games where sound sources could be behind walls. This effect can be simulated by applying a low-pass filter on the obstructed sound source [36].

The baseline model mets the above mentioned requirements. However, the next section discusses the hybrid model that introduces another requirement; geometry-dependent reverberation.

3.4.2 Adapted Hybrid Model

As the baseline model, the hybrid model implements spatialized, attenuated sound and takes obstruction into account.

In addition to these three basic principles, it reproduces basic principles of room acoustics. In room acoustics, the reflections and reverberations in a room depend on the structure of the room's geometry, as it is discussed above (see Section 2.1.6). This information needs to be taken into account in a hybrid sound model for audio games. Therefore, the above described requirements are extended by:

- early reflections and late reverberation that are geometry-dependent.

The basic principles of the ODEON approach are discussed in Section 2.1.6. This approach is ray-based and combines the image source method with secondary sources. An implementation of ODEON as it is described in [35] into the given test environment would not be possible. ODEON would generate too many sound sources that cannot be handled by the software simultaneously. Therefore, the ODEON model is adapted for real time in the given environment. This adaptions are discussed in the following paragraphs.

3.4.2.1 General Adaptions

Sound delay
Like discussed above, the sound delay for the image sources is calculated through the distance between the sound source and the listener. This means that every time the player changes position the audio playback position needs to be changed as well. This is not possible in the given software setup, without producing sound artifacts. These artifacts are produced by an output buffer underflow that can be perceived as clicking noise [46].

As replacement for this distance effect, sound is additionally attenuated depending on the distance between the sound source and the listener. This sound attenuation is applied on image sources but also on secondary sources since the delay is calculated in the same way.

Amount of sound sources
In [38], the author states that 500 to 1000 rays are enough to obtain reliable results producing realistic sound effects in an auditorium. If each ray creates 10 sound sources in total, there would be 5000 to 10000 sound sources in the audio game prototype at the same time. This amount cannot be handled by the used software of this thesis.

To limit the amount of produced sound sources, the maximum image source order can be set as recommended in [38] to two or three. However, the amount of secondary sources has to be limited, otherwise real-time calculations would not be possible anymore. The adaptions of the secondary sources are discussed in the next section.

3.4.2.2 Adaptions of the Secondary Sources Algorithm

In general, the secondary sources algorithm starts its calculations when the set transition order is reached. When the transition order is set to two, third order image sources are not produced

anymore. Instead, the first secondary source is placed at the point of intersection at the wall between the image source of second order and the point where an image source of the third order would be. The direction of the secondary ray follows Snell's law for reflection.

However, the secondary sources are only audible for the receiver if the corresponding image source is valid (during the audibility test, see Section 2.1.6.1). This might not always be the case, e.g. if the maximum image source order is too low or there are too many walls. Therefore, an approach similar but not identical to the secondary source approach is implemented, which is referred to as Modified Secondary Sources Approach (MSSA).

Instead of starting the secondary sources algorithm when the maximum image source order is reached, this approach shoots rays into the scene from the original sound source at the beginning. These rays explore the room geometry and therefore produce reverberation even if there are no visible image sources. As long as the maximum length of the secondary ray is not reached, the reflection process is repeated iteratively, placing a secondary source at every point of intersection between a secondary ray and a wall. With every jump, the ray loses energy (volume) and the delay of each secondary source is calculated through the covered distance between the original sound source and the secondary source. The last secondary source is created at the wall where the maximum length was below the threshold. Figure 3.5 shows this process. The green dots represent the created secondary sources.

Formula 2.7 for calculating the energy of the secondary sources cannot be applied in the used setup. The last term of the proposed formula would take the attenuation over distance into account. However, AstoundSound plug-in calculates the attenuation of sound objects depending on the position and rotation of the listener. Therefore, the last term of Forumla Formula 2.7 is dropped.

Sound energy absorptions of obstacles are usually frequency dependent. This is not possible in the used sound software setup. Therefore, the same parameter is used for each frequency. See Section 4.3 for discussion about the chosen value for the sound model implementation.

3.4.2.3 Parameters

The proposed hybrid model approach introduces the following parameters:

- maximum image source order,
- maximum length of secondary rays,
- number of rays emitted from the original sound source,
- scattering coefficient,
- absorption coefficient,
- and obstruction level.

Maximum image source order
In the image source method, sound sources get mirrored at the room boundaries recursively. In a room with four walls, the sound source gets mirrored at all walls and generates four image

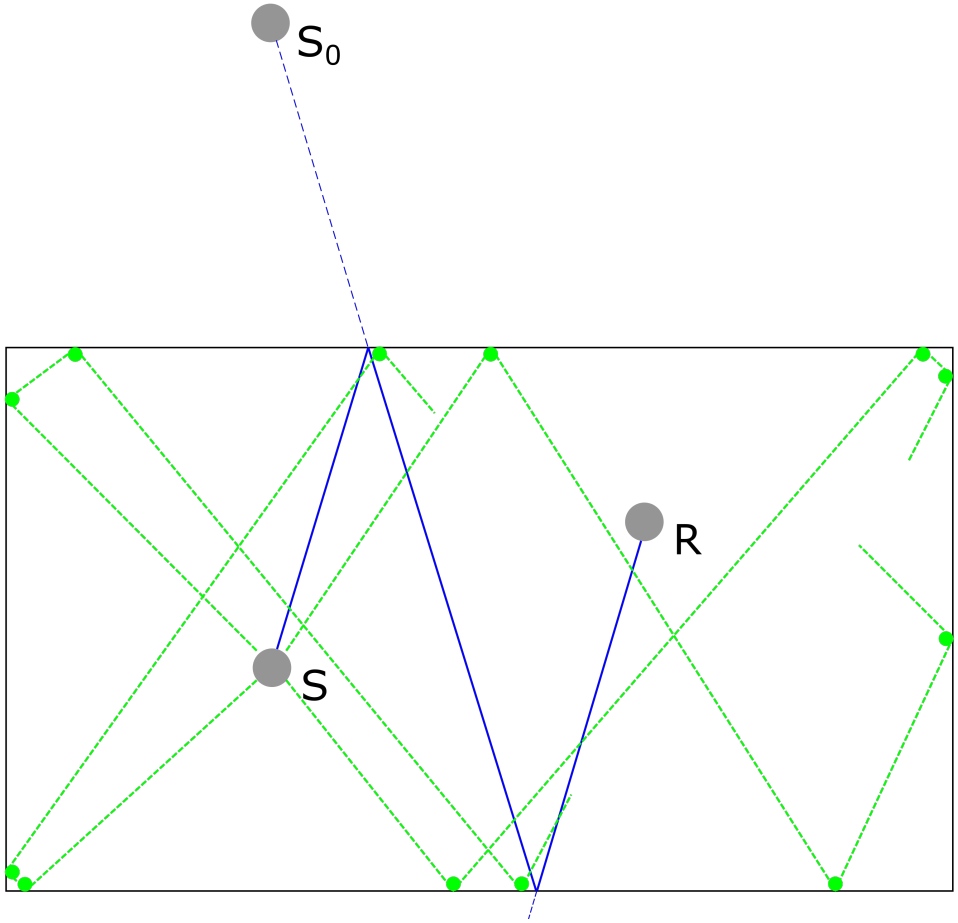


Figure 3.5: Illustration of image sources (S_0) and secondary sources (green dots on the room borders) calculation for the implemented hybrid model.

sources. If the image source algorithm would stop now, the maximum image source order would have been set to one. When the image source, e.g. S_0 again gets reflected on a wall, e.g. wall 1 it generates an image source of second order ($S_{0,1}$).

Rindel [38] recommends a maximum image source order set to two or three.

Maximum length of secondary rays
As discussed above, the amount of sound sources needs to be limited otherwise it would not be manageable by the used software. The maximum length of secondary rays is the total distance a ray has covered since its emitter. In ODEON, a maximum reflection order parameter can be set that limits the amount of secondary sources. It defines how often a ray can get reflected. In ODEON, the maximum value for this parameter can be set to 2000 [35]. However, appropriate parameters in the test environment are discussed in Section 4.4.

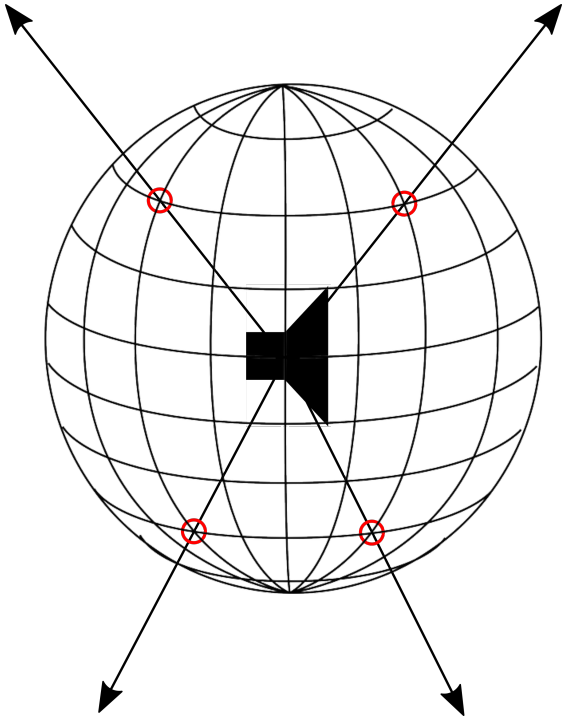


Figure 3.6: The regular split of a sphere for calculating the amount and directions of the rays for MSSA. Rays (lines with black arrows) are shot from the sound source origin through the points of intersections (red circles). The more intersections, the more rays are emitted

Number of Rays emitted from the original sound source
The number of rays emitted is a parameter that is introduced for MSSA. In original secondary sources calculations, the secondary source algorithm starts after the transition order is reached. However, as discussed above, this is not applicable for a room with bigger obstacles in it.

The number of rays emitted are defined as the amount of rays that are emitted from the original sound source into the scene. For the calculations of the directions, a virtual sphere is placed at the sound source position which is split into equal parts on its surface. When the parameter is set to 25, 25 rays are emitted from the origin of the sound source facing the directions of the equally split parts of the sphere surface. Figure 3.6 shows an example of the regularly split of the sphere.

Scattering coefficient
As described in Section 2.1.6.2, the scattering coefficient describes how much of a random vector is added to the vector of the specular reflection of a secondary ray. With this, roughness of a surface can be simulated. The value must be between 0 and 1.

When the parameter is set to 0, no random vector is added to the direction of the specular reflection. Therefore, the surface would be infinitely smooth. When the parameter is set to 1,

the direction of the random vector would be the new direction of the secondary ray.

Ideally, this parameter should be set to 0.1 for large plane surfaces and 0.7 for highly irregular surfaces [38].

Absorption coefficient
The absorption coefficient is needed for the calculation of the sound energy of secondary sources. It defines the amount of sound energy absorbed by a wall. With every reflection a secondary ray loses energy and therefore creates a less powerful secondary source. See Formula 2.6 for the calculation formula.

A list of possible absorption coefficients can be found in [19].

Obstruction level
Obstruction is a sound phenomenon that appears when the direct path between a sound source and the listener is obstructed by an obstacle. A more detailed discussion about obstruction can be found in Section 2.1.4.

The used audio middleware solution Wwise offers an option to set the obstruction level of a sound source. This parameter defines the strength of attenuation caused by a low-pass filter and reduction in volume. The value must be between 0 and 1, where 0 means no obstruction and 1 full obstruction.

3.4.2.4 Testing Prototype

A prototype was implemented for development and testing of the proposed hybrid sound model. Figure 3.7 shows the debug output of this version. The red capsule is the player, while the semi-transparent cube is the position of the sound source. Following the rules of image source creation (see Section 2.1.6.1), the pink lines show the reflections of sound rays on the walls. The higher the order, the more reflection does one ray have. The green cubes are the positions of the secondary sources that are generated like described above.

Several tests with different types of sounds were tested for finding optimal parameters. Tests with German Hip-Hop (Deichkind - So‘ne Musik) have shown that this model is not appropriate for sound sources producing sound continuously. This might be the case due to computational reasons and limitations of the sound model. However, speeches turned out to be very clear and rich in reflections. Sound sources that produce periodically discrete sounds are very suitable.

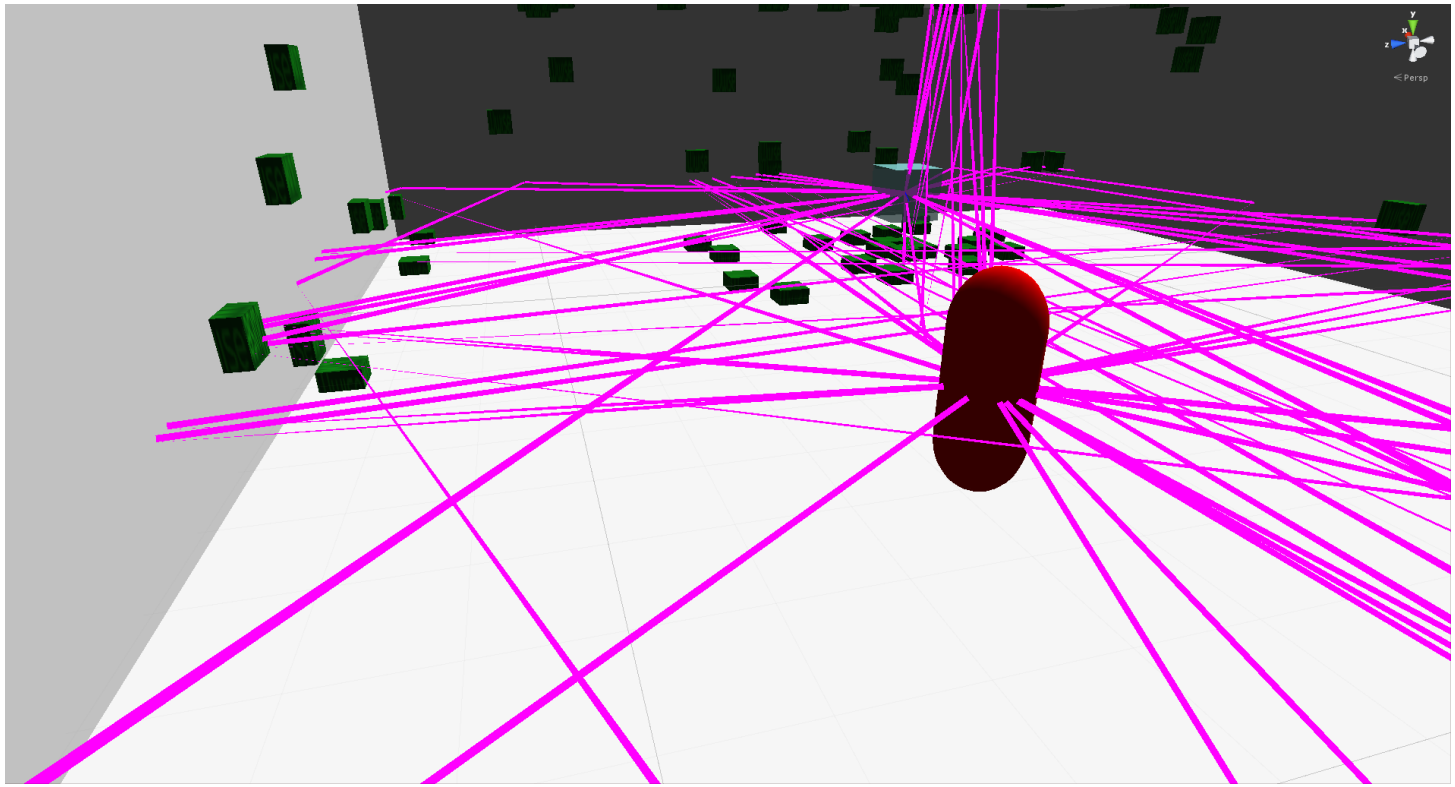


Figure 3.7: A screenshot of the implemented version for testing the hybrid model. The pink lines show the image source model reflections and the green cubes are the positions of the secondary sources.

CHAPTER 4

Implementation

This chapter discusses the implementation of the hybrid sound model in the Unity3D game engine and describes the communication between Unity3D, Wwise and AstoundSound.

4.1 Plug-in Integration

This section describes the necessary configurations to set up the communication between the used software.

4.1.1 Integration of Wwise into Unity 3D

Before Wwise can be integrated into Unity3D, it is advisable to create a new project in Wwise first and select the Unity3D's Assets folder as target (see Figure 4.1). When the new project is created, Unity3D automatically detects the new changes in its folder structure and updates its project explorer's content.

When the new project was created, Wwise Unity3D integration package can be imported into Unity3D. During the import process, the Wwise project should be detected automatically. Wwise is successfully imported into Unity3D then.

4.1.2 Integration of the AstoundSound plug-in into Wwise

As discussed above, AstoundSound plug-in was chosen as the most appropriate technology for this thesis. Before the spatialization functionality of AstoundSound can be used, it must be integrated into Wwise. The plug-in is not free of cost and therefore not enabled by default in Wwise. To active the plug-in, AkSoundEngine.dll needs recompiling.

For this recompiling process, the Unity3D Integration Source must be downloaded from Audiokinetic's website. It can be opened with an appropriate Integrated Development Environment (IDE), e.g. Microsoft Visual Studio. In *SoundEngineStubs.cpp* file, additional plug-ins can be

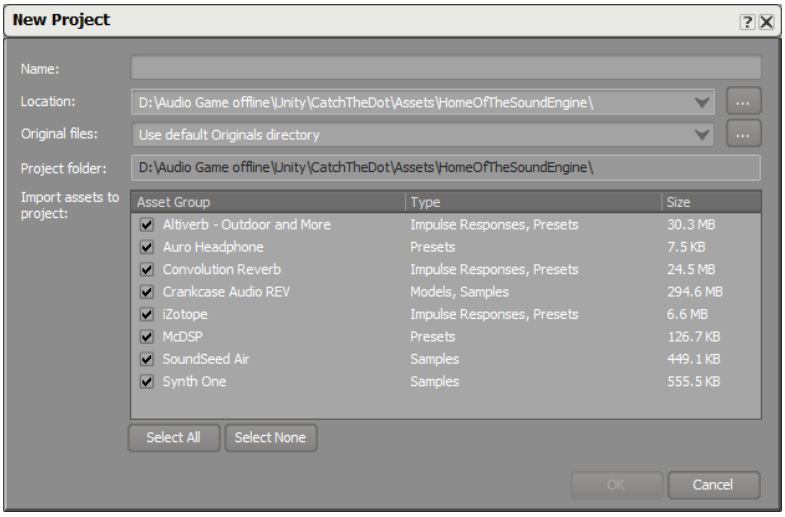


Figure 4.1: This image shows the new project dialog in Wwise. The location of the project should be inside of Unity3D's Assets folder so that Unity3D can automatically update changes done in Wwise.

```
// Register plugins
// If you need to add other plugings (such as paid plugins), uncomment the appropriate line below.
AK::SoundEngine::RegisterAllBuiltInPlugins();

//AK::SoundEngine::RegisterConvolutionReverbPlugin();
//AK::SoundEngine::RegisterSoundSeedPlugins();
//AK::SoundEngine::RegisterMcDSPPlugins();
AK::SoundEngine::RegisterAstoundSoundPlugins();
AK::SoundEngine::RegisterAuroPlugins();
//AK::SoundEngine::RegisteriZotopePlugins();
//AK::SoundEngine::RegisterCrankcaseAudioPlugins();
```

Figure 4.2: The Unity3D Integration Source for Wwise, opened in Visual Studio 2012 IDE. AstoundSound and Auro-3D plug-in are uncommented (activated) and ready for compiling.

activated. Figure 4.2 shows the uncommented lines of code that are necessary to activate plug-ins (in this figure, AstoundSound and Auro are activated). The old AkSoundEngine.dll needs to be replaced with the newly compiled one in order to have the plug-ins working properly.

4.1.3 Calling Wwise Events in Unity3D

After the Wwise sound engine is integrated into Unity3D, Wwise can be used to play sound instead of Unity3D's built-in sound engine. Before a sound file can be played through the audio middleware solution, the sound must be imported into Wwise's authoring application. The sound is then available through Wwise's SoundBank structure. SoundBank is a collection of event and sound data that can be loaded at specific points during a game to increase performance.

Unity3D communicates with Wwise through events. These events can be used to trigger actions, e.g. the start or the stop of an audio file. When a virtual character in an adventure game

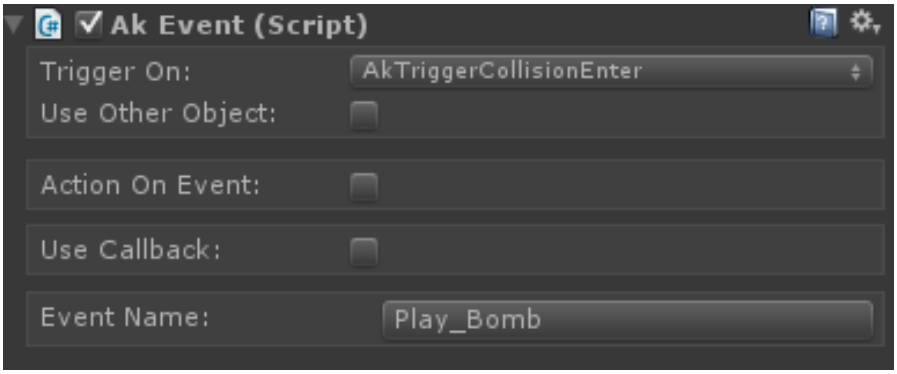


Figure 4.3: The AkEvent script provided by Wwise can be used to fire events without programming. When the object to which is script is attached to enters a collision, the "Play_Bomb" event is fired.

e.g. arrives at a village when she was in the woods before, a sound event (e.g. "stop_twittering") could stop the twittering of birds and another sound event could start village related sounds. An event is in its SoundBank unique. Therefore, an event name can only exist once.

Before an event can be fired, it must be registered in Wwise's authoring application. By right-clicking the imported sound, a new event can be created. Available events are play, break, stop, pause, mute, bus volume, voice volume, voice pitch, voice low-pass filter, voice high-pass filter, state, set switch, bypass effect, seek, trigger, game parameter and release envelope. Events are also stored in Wwise's SoundBank structure.

There are two possible ways to fire sound events in Unity3D for being played through Wwise audio engine. One option is to use the already existing scripts provided by Audiokinetic that load e.g. a Soundbank at start or trigger events when collisions happen automatically. The other option is to fire these events directly to Wwise through code. In this thesis, the second option was preferred over the first one. Regardless of how the event is fired, Wwise knows the sound producing game object (and therefore its position and rotation) and the audio file that needs to be played.

In Unity3D, the AkEvent script provided by Wwise can be used to fire events without programming. The script needs to be attached to the game object at which the sound should be played. Several triggers can be selected that fire the sound event, including CollisionEnter, Disable or MouseEnter. The name of the event which should be fired can be selected in a drop-down box through a Wwise explorer in Unity3D. Figure 4.3 shows the AkEvent component attached to a game object.

Sound events can also be fired through code. For this purpose, the *PostEvent* method of *AkSoundEngine* class in Unity3D needs to be called. The parameters are the event name and the game object at which the event is triggered. In the following example, the call of method

AkSoundEngine.PostEvent("Play_Bomb", gameObject)

is equivalent to the settings in Figure 4.3 without script.

Wwise offers RTPC that allows the changing of parameters of a sound object, e.g. volume or pitch at runtime. They need to be registered in Wwise before they can be used, similar to the registration of events. RTPC parameters are set by calling the *SetRTPCValue* method of *AkSoundEngine* class. In the following example, the call of method

AkSoundEngine.SetRTPCValue("volume", 90, gameObject)

sets the volume of the gameObject to 90.

4.2 Sound Model Implementation

This section describes the implementation of the hybrid sound model. A combination of the image source method with the modified approach of secondary sources was implemented. Every sound source was represented by a game object. The player is represented by a capsule and the sound sources are cubes (see Figure 3.7). A sound source has a defined position in the 3D space.

4.2.1 Code Design

This section describes the structure of the used scripts. The following scripts were implemented:

- AudioMaster.cs,
- CheckObstruction.cs,
- GameElement.cs,
- GameLogic.cs,
- and ReflectionCube.cs.

Audio Master Script
The Audio Master script implements basic methods for loading a sound bank, starting and stopping an audio file and reading or setting a playback position of an audio file. These methods use existing methods of AkSoundEngine. Every other class in the implementation that needs access to audio methods inherits from this class.

Check Obstruction Script
The Check Obstruction script is responsible for setting the right level of obstruction on the sound source. It is attached to the original sound source object because this is the only type of sound source where obstruction is possible. For a detailed description of the implementation, see Section 4.2.5.

Game Element Script
In Game Element script, the hybrid sound model algorithm like it is described in Section 4.2.2

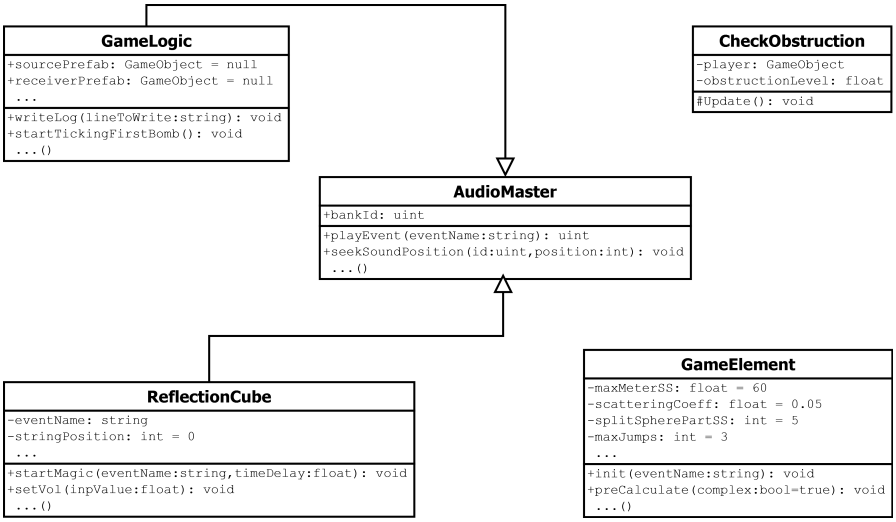


Figure 4.4: The five implemented classes AudioMaster, CheckObstruction, GameElement, GameLogic and ReflectionCube. With these classes, a fully functional audio game with a hybrid model can be implemented.

is implemented. It calculates the image source positions as well as secondary sources and performs the audibility test on both. In addition, it is responsible for enabling and disabling sound sources which are not valid during runtime. Sound model dependent parameters are set in this script. However, it does not inherit from Audio Master since it is not responsible for playing or stopping sound directly. This script is attached to every sound source automatically when they get instantiated.

Game Logic Script
The Game Logic script is the implementation of the context in which this model is used. When it is used for an audio game, this scripts implements the whole logic of the game, e.g. loading of a new level, storing of all sound sources or logging. This script can but does not have to inherit from Audio Master as long as no sound needs to be played. This script takes several prefabs (sound source prefab, secondary source prefab, receiver, etc.) as parameters.

Reflection Cube Script
Reflection Cube script inherits from Audio Master and is attached to every sound source. It implements methods for starting and stopping and volume changing of a sound source and provides the possibility to start an audio file with delay. The possibility of being played with a delay is necessary for the image sources and secondary sources.

Figure 4.4 shows a schematic class diagram as an overview of the implemented classes.

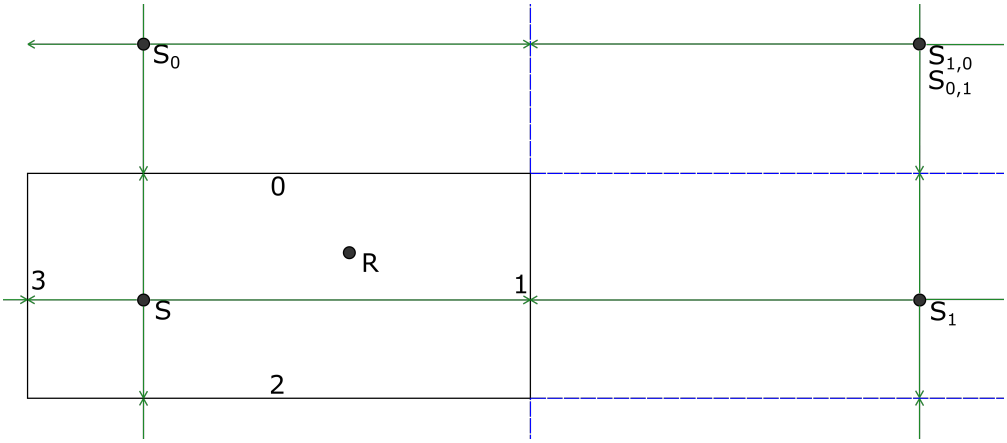


Figure 4.5: This image illustrates the raycast at the creation of image sources in two dimensions.

4.2.2 Image Source Method

In this section, the implementation of the image source method in Unity3D is described. For a general description of the image source method, see Section 2.1.6.1.

Every wall has an additional object called *Mirrorplane* attached that extends the wall on two axes (see the blue lines in Figure 4.5). At the beginning of the calculation, the position of the sound source (S) and the player (R) is saved. Rays are shot from the sound source into six directions, two for each axis. In Figure 4.5, such raycasts are shown by green arrows. When the ray hits the mirrorplane of a wall, an image source is instantiated in form of a new game object (S_1) at the position coresponding to Formula 2.4. S_1 shoots another rays in six directions, while only 5 of them are valid. The one invalid ray is due to the fact that it is not allowed that the image source is mirrored back at the same wall. However, $S_{1,0}$ was mirrored at the virtual extension (mirrorplane) of wall 0. Depending on the maximum image source order (see Section 2.1.6.1), the mirroring stops after this order is reached.

After the image sources have been created, an audibility test needs to be performed when the player moves for enabling or disabling sound cubes that are valid or invalid. For more information about the audibility test itself, see Section 2.1.6.1. This audibility test is illustrated in Figure 4.6. Image source positions are the same as in Figure 4.5. The audibility test starts at the receiver's position. A ray is emitted to an image source of the first order (e.g. S_0). If the hit wall (but not mirrorplane) equals the wall the source was mirrored at, the next step of the audibility test can be performed. A ray then starts at the position of intersection in direction to the predecessor of the actual image source. In the case of S_0 it is the original sound source. Therefore, S_0 is successfully backtraced by the path $R \rightarrow S_0 \rightarrow S$ and the image source remains valid. Also image sources S_1 and $S_{0,1}$ are valid, by taking the paths $R \rightarrow S_1 \rightarrow S$ and $R \rightarrow S_{0,1} \rightarrow S_0 \rightarrow S$. However, $S_{1,0}$ is invalid because the ray first hits wall 1 instead of wall 0 (red line). If the source is invalid, it and its children are disabled because an invalid image source cannot produce valid sources of higher order.

The audibility test is done at every frame when the user moves. As soon as an image source

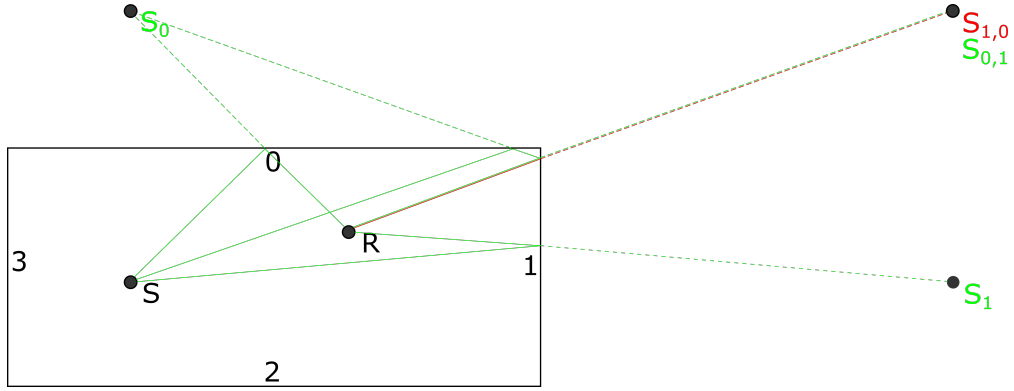


Figure 4.6: The audibility test applied on the given example.

gets invalid, the volume parameter of the respective Wwise game object is set to zero. If the audibility test of an already disabled sound source is valid, the volume is restored.

4.2.3 Modified Secondary Sources Approach

There are two parameters (see Section 3.4.2.3 for an overview of the parameters) affecting the outcome of this algorithm; *number of rays emitted* and *maximum length of secondary rays*. As described in Section 3.4.2, the origin of the emitted rays is the center of the original sound source. The direction as well as the amount of rays depend on a parameter called *steps*. This parameter splits the spheres' surface into equal parts on x- and y-axes, like described in Section 3.4.2.3.

When the parameter is set to six, the algorithm will create 36 rays ($6^2 = 36$).

After the directions are calculated, rays are shot through the sphere parts. When a secondary ray hits a wall, it gets reflected according to Snell's law. However, to simulate the roughness of the surface, a random vector is added to the direction that is calculated through Snell's law. This random vector is generated during runtime and added to every reflected direction. The weight of this random vector can be controlled by the scattering coefficient parameter (see Section 3.4.2.3).

Figure 4.7 shows the vectors used in this approach. On the left, the secondary ray incidents on the surface. Without a random vector, the ray would get reflected according to Snell's Law. The random vector has a different direction. However, when the scattering coefficient is set to 0.5, the secondary direction like it is shown in the figure is the resulting direction of the secondary ray.

The sound energy (volume) of the secondary source is calculated using Formula 2.6. With every reflection, the energy of the next secondary source gets reduced by the factor α_i which is the absorption coefficient. When the sound energy reaches zero, the sound source is not spatialized anymore.

Every time a ray is reflected, the total path it has covered so far is saved. When the total covered path is larger than the maximum length of secondary rays, the algorithm stops creating additional secondary sources.

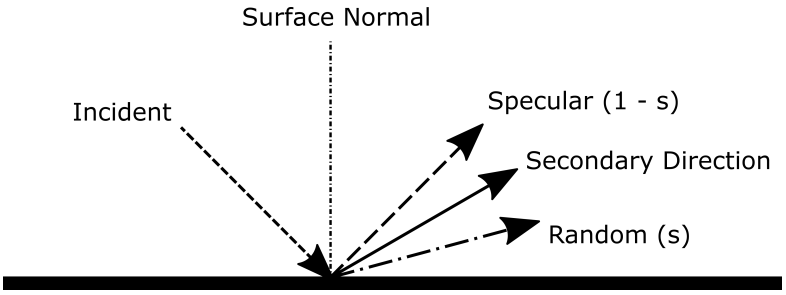


Figure 4.7: The secondary ray would get reflected according to Snell's law (specular) when the scattering coefficient is set to 0. The secondary direction like it is in the figure can be achieved by setting the scattering coefficient to 0.5.

4.2.4 Spatialization

Spatialization is done by Wwise using AstoundSound plug-in. The following parameters are passed to Wwise:

- the player,
- the sound source,
- the image sources,
- and the secondary sources.

The *AkGameObj* script is provided by Wwise and is attached to those objects and transmits the position and rotation to the audio middleware solution.

In Wwise, AstoundSound spatializes the sound of emitting sound sources. The sound signals are routed through an audio bus to which the AstoundSound plug-in is attached to. Every incoming sound signal is spatialized then.

There are several options provided by AstoundSound plug-in that can be selected in Wwise's authoring application. The following features of AstoundSound plug-in are enabled:

- Full 3D - enables 3D panning (azimuth and elevation).
- Spatial interpolation - allows interpolation for smooth transitions between filter points.
- Apply attenuation - calculates sound attenuation depending on the distance between a sound source and the listener.
- Get direction from game object - this setting is important in interactive applications.
- Get distance from game object - this setting is important in interactive applications.

The spatialization is done in real time.

4.2.5 Obstruction

When the sound source is obstructed by an obstacle, a low-pass filter and a reduction in volume is applied on the sound source. Wwise offers the method *SetObjectObstructionAndOcclusion* for this purpose. The first parameter of this method is the game object on which the low-pass filter should be applied. The second one is the number of the listener in the scene. The third parameter can be between zero and one and sets the obstruction level. Zero means no obstruction, one means fully obstructed. This is also possible for occlusion, however, no occluded sound sources are used in this thesis.

Only the original sound source can be obstructed. Image sources cannot be obstructed because obstructed image sources would already fail the audibility test. Secondary sources are only emitting sound, if there is no obstacle in the line of sight between the player and the secondary source.

If there is no obstacle between the source and the receiver (player), no low-pass filter is applied. If there is one wall between them, a low-pass filter is applied. When there are two walls between these two points, the strength of the low-pass filter is squared. The following formula is used to calculate the obstruction parameter for Wwise. Be o the obstruction level, n the amount of walls between the original sound source and the listener so is p with

$$p = 1 - o^n \tag{4.1}$$

the parameter for the *SetObjectObstructionAndOcclusion* method.

4.3 Parameters

The following parameters are adjustable and can be changed:

- maximum image source order,
- number of rays emitted from the original sound source,
- maximum length of secondary rays,
- scattering coefficient,
- absorption coefficient,
- obstruction level,
- maximum audible distance,
- and speed of sound.

For the calculation of the sound delay, the speed of sound is taken (344 m/s). For every meter between the image source and the original sound source and every meter of the secondary ray's length, a delay of $\frac{1}{344} \approx 0.0029$ seconds is applied on the respective sound source. However,

this parameter is not changed because for the tested audio game air is assumed as medium for sound.

The maximum values of the other parameters depend on several factors: computational power, structure of the VE and if the sound sources are able to move or not. Several settings were tested to find an optimal solution for this issue.

The maximum image source order limits the mirroring process during the creation of the image sources (see Section 2.1.6.1 for further explanations). Depending on the chosen value, performance can be increased or decreased. Higher values find more specular sound paths, however, the model calculation time changes significantly when this parameter is modified. In the next section, this parameter is tested to visualize its impact on system performance during runtime. For the audio game, this parameter is set to three.

The amount of emitted rays for the calculation of secondary sources also affects the system performance. The more rays are emitted, the more sound sources need to be managed and spatialized at the same time. For the tested audio game, this parameter is set to 25. However, the next section shows the performance of the system when this parameter is changed.

Related to the amount of emitted rays, the maximum length of secondary rays also affect system performance. The higher this value is, the more sound sources are generated that need to be managed and spatialized at the same time. The optimal value for this parameter depends on the VE structure, however, in the tested audio game this value is set to 60 meters. Different values of this parameter and their effects are shown in the next section.

The absorption coefficient is the amount of energy the secondary ray loses with every reflection. This parameter is set to 0.3. Therefore, the ray loses 30% of its energy with every reflection. This is the absorption coefficient of hardwood that was chosen for this implementation [19].

The scattering coefficient is used to simulate rough surfaces. It is set to 0.05. This means that 95% of the true reflection vector is combined with 5% of the random vector. Other parameters would have been possible too, however, in the simulated audio games the room's surfaces are assumed as very smooth.

The obstruction level is set to 0.5 which sounds natural. Other values tested for this parameter were 0.9, 0.7 and 0.3. However, 0.5 was found to be appropriate for the tested audio game.

The maximum audible distance defines the distance between the listener and a sound source in which the sound source is still perceivable. This distance depends on the VE structure and cannot be generalized but for the tested audio game of this thesis, a maximum audible distance of 20 meters was found to be appropriate. This also served as replacement for the missing delay between the receiver and the image sources as well as between the receiver and the secondary sources.

The maximum image source order, maximum length of a secondary ray, the amount of secondary rays and the maximum audible distance depend on the underlying VE structure and cannot be proposed in general. However, the next section discusses several combinations of the maximum image source order, the maximum length of the secondary rays and the amount of secondary rays in a test scene.

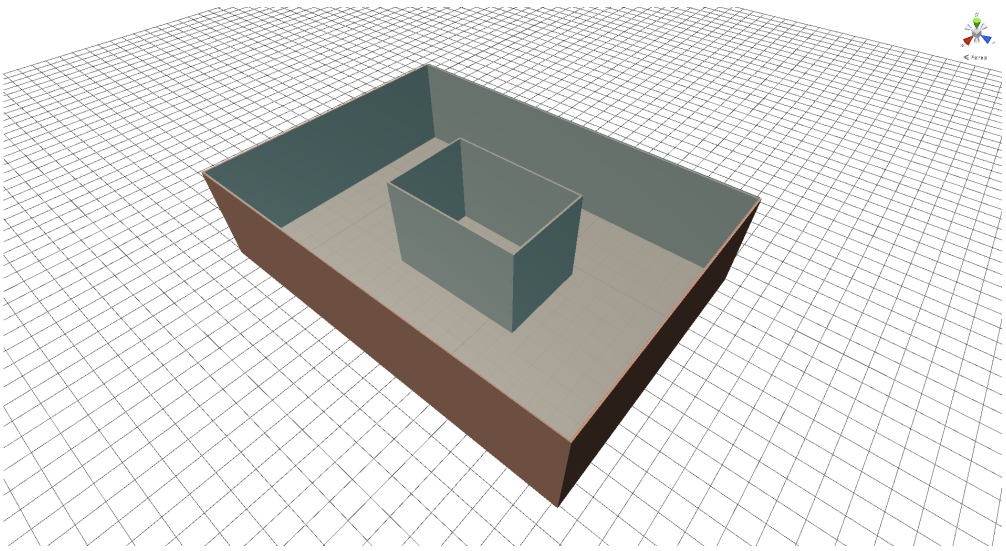


Figure 4.8: The area for testing different parameters to evaluate maximum values.

4.4 Performance

The performance of the implemented hybrid model is discussed in this section. For reasons of comparison, the maximum image source order, the maximum length of the secondary rays and the amount of secondary rays are modified, tested and compared. The test setup consists of an Intel i7-3770 CPU @ 3.4 GHz, 8 GB RAM and Nvidia Geforce GTX 670 running on 64-bit Windows 7.

The virtual test environment consists out of ten reflection surfaces (one ceiling, one floor, eight walls). In the user study described in Section 5.2.1, a similar virtual scene is used. Figure 4.8 shows the testing area. The area is 20m x 15m x 5m large with an inner pillow of the size 7m x 5m x 5m. The ceiling is made transparent but does reflect sound waves and act like a normal wall.

As mentioned above, three parameters are varied for testing system performance: the maximum image source order, the maximum length of the secondary rays and the amount of secondary rays. For the maximum image source order, three values were tested: two, three and four. A fifth image source order would have produced too many sound sources. For the maximum length of secondary sources, the following eight values are tested: 30, 40, 50, 60, 70, 80, 90 and 100. The tested values for the amount of secondary rays are 4, 9, 16, 25, 36, 49, 64 and 81.

Parameter combinations which produced sound starvations are marked with *. Sound starvation happens when the sound engine cannot provide data to the hardware buffer in a timely manner [5]. The reason for this could be an excessive use of the CPU where the audio thread has 100% load.

max. order = 2		Amount of rays emitted							
		4	9	16	25	36	49	64	81
Maximum length of secondary rays (m)	30	169.87	146.29	139.10	94.44	85.64	63.87	56.30	44.59
	40	168.23	134.51	127.72	78.04	72.79	51.68	44.21	30.55*
	50	161.52	125.17	117.26	69.78	60.10	44.37	39.75	25.86*
	60	163.55	117.44	107.46	62.90	50.38	40.28	32.15*	22.53*
	70	151.66	110.43	101.83	57.32	46.70	31.98*	26.44*	20.18*
	80	146.24	97.68	97.54	52.38	41.13	28.32*	22.23*	16.93*
	90	152.11	95.98	87.65	46.08	37.67	24.97*	19.20*	15.32*
	100	141.89	92.93	86.90	42.09	34.54	22.97*	17.82*	13.99*

Table 4.1: The measured performance on a testing desktop. The amount of rays in combination with the maximum length of secondary rays is varied and the output is given in average FPS measured over 30 seconds. The maximum image source order is set to two. Output marked with * produced sound starvation.

max. order = 3		Amount of rays emitted							
		4	9	16	25	36	49	64	81
Maximum length of secondary rays (m)	30	73.59	67.79	67.85	55.42	50.00	41.87	38.97	33.32
	40	73.03	64.95	64.64	50.35	44.39	36.07	31.90	27.04*
	50	71.85	63.05	60.88	46.17	40.96	31.54	27.55	21.84*
	60	70.38	59.02	61.44	43.04	36.33	26.64	23.82*	19.75*
	70	69.22	57.88	55.25	39.22	33.40	25.19*	22.25*	17.06*
	80	69.00	55.34	55.19	37.52	30.54	23.53*	19.44*	15.63*
	90	68.17	55.44	52.05	33.55	27.46	20.48*	18.28*	13.83*
	100	70.32	52.72	50.78	32.40	26.78	18.63*	16.27*	13.21*

Table 4.2: The measured performance on a testing desktop. The amount of rays in combination with the maximum length of secondary rays is varied and the output is given in average FPS measured over 30 seconds. The maximum image source order is set to three. Output marked with * produced sound starvation.

4.4.1 Frames Per Second (FPS)

Table 4.1 shows the effect of varying the parameter of the amount of rays emitted and the maximum length of secondary rays with maximum image source order set to two. Table 4.2 lists the output FPS with maximum image source order set to three, Table 4.3 with parameter set to four. The output is measured in averaged FPS over a duration of 30 seconds in the testing area.

4.4.2 Sound Objects and CPU load

This section shows three tables in which the parameters of the three testing variables, the maximum image source order, the maximum length of the secondary rays and the amount of secondary rays are varied. However, in comparison to Table 4.1, Table 4.2 and Table 4.3, the tables

max. order = 4		Amount of rays emitted							
		4	9	16	25	36	49	64	81
Maximum length of secondary rays (m)	30	36.04	34.49	34.32	29.20	28.84	24.55	23.21	20.91
	40	35.27	33.99	33.33	27.94	26.19	22.76	21.34	17.43*
	50	35.46	32.15	32.61	27.21	24.19	20.54	18.16*	15.03*
	60	35.48	31.79	31.35	25.51	22.43	18.49	16.84*	14.18*
	70	34.86	31.28	31.49	23.34	21.27	17.45*	15.15*	12.57*
	80	35.11	30.85	29.63	22.31	19.26	15.57*	14.05*	11.59*
	90	35.07	29.95	29.83	20.75	18.46	14.48*	12.92*	10.68*
	100	34.39	30.07	29.30	21.63	17.22	13.14*	12.18*	10.13*

Table 4.3: The measured performance on a testing desktop. The amount of rays in combination with the maximum length of secondary rays is varied and the output is given in average FPS measured over 30 seconds. The maximum image source order is set to four. Output marked with * produced sound starvation.

max. order = 2		Amount of rays emitted					
		25			81		
		reg. objects	active objects	cpu load	reg. objects	active objects	cpu load
Max. length sec-ondary rays	30	352	273	26.40%	916	837	94.90%
	60	610	531	51.50%	1831	1752	100.00%
	100	1033	954	78.08%	2843	2764	100.00%

Table 4.4: The registered sound objects in Wwise (in total, including emitting and non-emitting sound sources), the active sound objects (that are actively emitting sound) and the corresponding CPU load in percent. The maximum image source order is set to two.

in this section (Table 4.4, Table 4.5 and Table 4.6) illustrate the *amount of sound objects* registered in Wwise, *actively playing sound objects* and the corresponding *CPU load of the audio thread*. The content of the tables is discussed in the next section.

For a better overview, two values for the parameter amount of secondary rays (25 and 81) and three values for the parameter maximum length of the secondary rays (30, 60 and 100) are chosen. 25 secondary rays with a maximum length of 60 are set for the user study.

4.4.3 Discussion

This section discusses the values of Table 4.1, Table 4.2, Table 4.3, Table 4.4, Table 4.5 and Table 4.6.

Maximum image source order
The maximum image source order has a significant effect on the overall system performance. The higher the maximum image source order is, the higher the computational costs are for model calculation. By comparing the top left values (30m maximum length, 4 rays emitted) of Tables

max. order = 3		Amount of rays emitted					
		25			81		
		reg. objects	active objects	cpu load	reg. objects	active objects	cpu load
Max. length sec-ondary rays	30	1164	320	30.41%	1688	844	90.83%
	60	1375	531	50.34%	2566	1722	100.00%
	100	1724	880	77.88%	3582	2738	100.00%

Table 4.5: The registered sound objects in Wwise (in total, including emitting and non-emitting sound sources), the active sound objects (that are actively emitting sound) and the corresponding CPU load in percent. The maximum image source order is set to three.

max. order = 4		Amount of rays emitted					
		25			81		
		reg. objects	active objects	cpu load	reg. objects	active objects	cpu load
Max. length sec-ondary rays	30	7707	310	35.97%	8269	872	96.41%
	60	7959	562	58.23%	9076	1679	100.00%
	100	8271	874	78.02%	10203	2806	100.00%

Table 4.6: The registered sound objects in Wwise (in total, including emitting and non-emitting sound sources), the active sound objects (that are actively emitting sound) and the corresponding CPU load in percent. The maximum image source order is set to four.

4.1, 4.2 and 4.3, the performance loss can be observed. When the maximum image source order is set to two, the system performance is at 169.87 FPS. When the value is increased by one, the performance drops by more than a half to 73.59 FPS. When the maximum image source order is set to four, the calculated average FPS are 36.04.

When the values of the other two parameters (length and amount of secondary rays) are set to maximal, the effect of performance loss is less significant. When the maximum image source order parameter is set to two, 13.99 FPS are reached. When the parameter is set to three, 13.21 FPS are possible. However, when the parameter is increased to four, 10.13 FPS were measured. The reason for this can be seen in Tables 4.4, 4.5 and 4.6. When the maximum parameters are chosen, the CPU load already is at 100%, therefore the increase of the maximum image source order does not affect the performance anymore.

However, the higher the value of this parameter, the more sound objects are registered in Wwise. In comparison, when both parameters (number of rays and maximum length) are set to max value, 2843 (Table 4.4) sound objects are registered when the maximum image source order is set to two, 3582 (Table 4.5) when the maximum order is set to three and 10203 (Table 4.6) when it is set to four.

Registered objects do not cost additional CPU performance. This can be seen by comparing the registered objects when the maximum length of secondary rays is set to 60 meters, the amount of rays set to 25 and the maximum image source order is set to two and three (Table

4.4 and Table 4.5). The amount of active sound objects is the same, however, the registered objects with maximum image source order set to two are 610 and when the parameter is set to three, there are 1375 registered sound objects. The difference in CPU load is marginal (51.50% compared to 50.34%).

Amount of rays emitted
The amount of rays emitted also affects the system performance. When the parameter is increased, more rays are emitted from the original sound source to be reflected on surfaces. The difference in CPU load can be seen by comparing the CPU load when the parameter maximum length of secondary rays is set to 30 meters and the parameter amount of rays emitted is set to 81. In Table 4.4, the difference in CPU load is 68.5%, in Table 4.5 it is 60.42% and in Table 4.6 it is 60.44%. The reason for this performance difference is the increase of the number of sound sources that actually emit sound in the scene.

Maximum length of the secondary rays
The maximum length of secondary rays also effect overall system performance because the larger amount of generated and especially active sound objects need additional management and spatialization.

The amount of registered sound objects increases with the increase of maximum length of secondary rays. This affects CPU load, as it is shown in Table 4.4. When the amount of rays emitted is set to 25 and the maximum length of secondary rays is increased, the CPU load also increases. With a maximum length of 30 meters, the CPU load is at 26.40%. When the value is increased to 60 meters, 51.50% CPU power is used. However, 78.08% CPU is used when the parameter is set to 100 meters. This shows that the maximum length affects the CPU load. This is shown in Table 4.4.

When a secondary ray has a larger maximum length, it can get reflected more often and therefore it produces a larger amount of secondary sources. When the maximum length is set to 30 meters, 352 registered objects are known in Wwise. If the value is set to 60 meters, the amount of registered sound objects is 610. When the amount of rays emitted is set to 81, 916 objects are registered when the maximum length is set to 30 meters and 1831 when the maximum length is set to 60 meters. These two increased values can be compared to evaluate how the amount of rays emitted effect the amount of objects when the maximum length is varied. When more rays are emitted from the original sound source, the effect of varying the maximum length of secondary sources is stronger due to the fact that more rays can produce more sound objects when the maximum length is increased compared to less rays emitted.

4.4.4 Parameters for User Study

A suitable combination of the parameters maximum image source order, maximum length of secondary rays and amount of rays emitted depend on the chosen hardware (more computational power enables more sound source management and spatialization) and the scene geometry. For the user study, the following values for the parameters were chosen.

The maximum image source order was set to three. This parameter is recommended by [38].y Therefore, the first, second and third order image sources are generated.

The maximum length of the secondary rays was set to 60 meters and the amount of rays emitted was set to 25. The CPU load of 50.34% in the testing scene was found to be a proper value that still leaves calculation reserves for Oculus Rift DK2 and tracking.

CHAPTER 5

Evaluation

The developed hybrid sound model is evaluated by comparing it to a simpler sound model that is usually used in audio games, the baseline model. The design of the models is described in detail in Section 3.4. For the purpose of comparison, an audio game prototype was developed. In this game, participants had to reach several sound source positions without visual but auditive feedback using navigation by real walking in a large room.

The proposed hypotheses, the study procedure and the findings including discussion are described in this chapter.

5.1 Hypotheses

The assumption was that the hybrid sound model reduces the *difficulty* of tasks (H_1 - H_4) and contributes to a higher sense of *realism* (H_5) and *immersion* (H_6) compared to the baseline model. Therefore, the following hypotheses were formulated and investigated:

H_1 : The completion time in the hybrid model is lower than the completion time in the baseline model.

H_2 : The difficulty of finding sound sources is higher with the baseline model than with the hybrid model.

H_3 : Users are likely to better recognize virtual obstacles in a game using the hybrid model than in a game using the baseline model.

H_4 : Users can reconstruct virtual geometry of a game better with the hybrid model than with the baseline model.

H_5 : The hybrid model sounds more realistic than the baseline model.

H_6 : Users playing a game with the hybrid model will experience a higher degree of immersion than the ones playing a game with the baseline model.

5.2 Game Prototype

The game prototype was developed to compare the implemented sound models.

The localization of sound sources in a virtual room is the central element in this game prototype. Therefore, the tasks of this audio game prototype are the reaching of sound source locations in virtual space by moving and listening to sound in real world. When a sound source location is reached, the next sound source starts playing. The game is over when all sound source positions are reached by the player. To avoid any cheating, players have to fulfill the tasks blindfolded.

There are three levels in this audio game prototype that a player has to complete. The first level is an introductory virtual scene where a player can get comfortable with the virtual reality and the sound in it. In this level, the player is able to see the scene. As soon as the player feels comfortable, he or she starts the real experiment by stepping on a button on the floor (see next section for detailed description).

In level two and three, the player has to fulfill the tasks blindfolded. The tasks in each of these levels are the same. In the beginning, the player hears a ringing phone. The task is to reach the position of the ringing phone by moving around in the tracked area and listening to the audio. Every position and rotation change of the player is taken into account, for real-time sound spatialization. When the position of the phone is reached, the player gets further instructions of the situation he or she is in. The voice on the telephone says that he has blindfolded the player and the player has to find three bombs in this room before they explode. After the phone call has ended, the first bomb starts ticking. The task now is to find the location of the bomb through listening to the bomb ticking. Only one bomb emits sound at the same time. When the location of the first bomb is reached, the second bomb starts ticking. As soon as the player has reached all bomb locations, the level is completed.

Although there are virtual walls in the game, the player is able to go through them. There is no game feedback that forbids a player to continue moving through a virtual wall. This solution allows the comparison if the hybrid model supports the player in avoiding obstacles or not.

In the next section, the three levels are described in detail.

5.2.1 Levels

The game prototype that served as basis for the user study consisted of three different levels.

Introductory Scene
The first level (Introductory Scene) is for experimenting and getting comfortable with the virtual reality and sound in it. This is the first room the participants were in, before actually testing the first audio game. In the room, there are three walls (excluding the walls at the border of the room), a water tap and a red button. In this level, participants are able to see the geometry and the sound source.

The game area is 20m x 5.2m in size. The water tap is placed next to the center of the room. It is partially enclosed by two walls so that a player can move around the water tap to perceive sound changes when the sound source is obstructed. A third wall is placed next to it. The red button in the southeast corner starts the experiment.

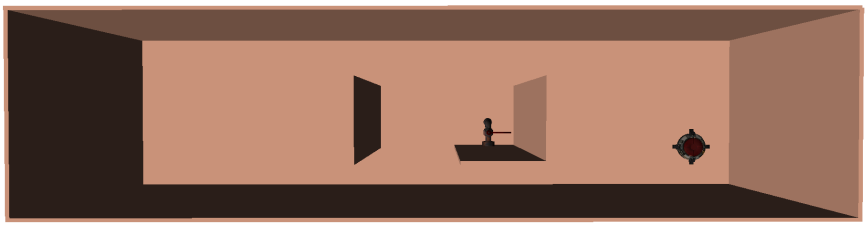


Figure 5.1: A screenshot of the Introductory Scene. The water tap is enclosed by walls, while the red button in the corner starts the audio game.

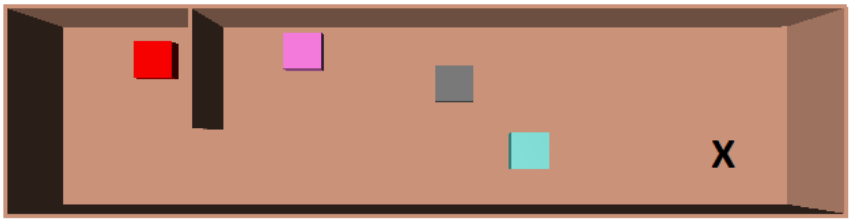


Figure 5.2: A screenshot of the simple room. The turquoise cube marks the position of the phone, the pink cube the position of the first bomb, the gray cube the position of the second and the red cube the position of the third bomb. However, the participant is not able to see them during the test.

Figure 5.1 shows this room. The water tap is periodically emitting sound of dripping water. It is the sound origin of the scene. The red button on the floor triggers the event for starting the experiment.

Simple Room
The second level (shown in Figure 5.2), which is referred to as the *simple room* has one wall and four sound sources in it. The turquoise cube marks the position of the phone, the pink cube the position of the first bomb, the gray cube the position of the second and the red cube the position of the third bomb. The starting position is marked as X. The last bomb is behind a wall and therefore obstructed.

Complex Room
The third level (shown in Figure 5.3), which is referred to as the *complex room* has five walls and four sound sources in it. The order of the sound sources is the same as in the simple room (turquoise, pink, gray, red). The starting position is marked as X.

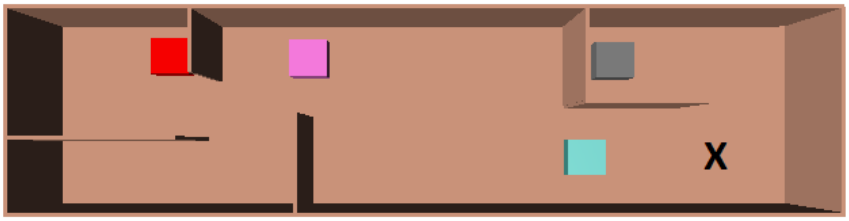


Figure 5.3: A screenshot of the complex room. The turquoise cube marks the position of the phone, the pink cube the position of the first bomb, the gray cube the position of the second and the red cube the position of the third bomb.

5.3 Study Design

The study design is a between-subject experiment. Half of the participants played the game with the baseline model used to simulate sound, half of them had the hybrid model assigned. In this section, the hardware setup, the procedure and the measurements are described. The user study was performed during spring 2015.

5.3.1 Hardware Setup

The game was tested in a 30x7m tracked open space. The participants wore an Oculus Rift HMD (DK2) and 7.1 surround sound headphones (Logitech G35). Players' positions were tracked with the use of a wide area optical tracking system. The VR system had been developed in-house and has not been published yet. The prototype ran on a laptop with Intel Core i7 processor and Nvidia GeForce GTX 980M graphics card. The laptop was attached to a backpack frame and carried by participants on their backs.

5.3.2 Participants and Procedure

Participants were invited through a social media channel to attend at this study. 37 people accepted this invitation. 20 of them were male and 17 female. The social media channel was a public Facebook group named "Foreigners in Vienna". Therefore, the culture, nationalities and backgrounds were very diverse. Participants received some sweets for time compensation at the end of the study. The following paragraphs describe the procedure every participant went through.

After the greeting, the participants had to fill in two forms (one Non-Disclosure Agreement and one Health-Condition Agreement). They received instructions for the test. First, they had to fill in a Pre-Test-Questionnaire. After that, audio games as well as the hardware equipment were described. Then, they put on the hardware and started the game in the Introductory Scene, where they were able to see and hear a dripping water tap for getting comfortable with the sound in the

Figure 5.4: A photo taken during the experiment. The participant wears the backpack, Oculus VR and headphones.

VE, depending on the model they have got assigned. As soon as they felt comfortable with the sound and virtual reality, they were told to go to the red button for starting the first experiment. When they hit the button, the screen turned black and a virtual phone started to ring.

In the simple room, the participants had to fulfill the tasks like described above (see Section 5.2). After all bombs got defused, the participants were able to see the first room they were in at the beginning and were told to remove the hardware and to go back to the test coordinator. Then they had to fill in the Each-Test-Questionnaire, where they had to draw the virtual scene they were in (see Section 5.3.3.1 for an explanation of the drawing task).

After they had filled in this questionnaire, they put the hardware on again and started looking for the red button. This triggered the start of the third audio game level (complex room). The complex room model was similar to the simple room model but the room geometry and the sound source positions were different. As soon as they defused all bombs in this level, they were able to see everything again and to go back to the test coordinator so that the hardware could be removed.

The participants then filled in the second Each-Test-Questionnaire, followed by the Post-Test-Questionnaire. After the completion of the Post-Test-Questionnaire, the participants had the option to talk about their experience and the complex room with visible sound sources was

shown to them.

5.3.3 Measurements

Subjective data in form of questionnaires and objective data obtained from direct measurements were used to test the formulated hypotheses (see Section 5.1). These types of measurements are discussed in this section.

5.3.3.1 Subjective Measurements: Questionnaires

Four different questionnaires were used for the user test. They can be divided into three types. All the questionnaires can be found in the appendix of this thesis (see Section A). Answers to most of the questions were given on a Likert scale in the range from 1 to 7.

> *Pre-Test-Questionnaire*: The participants received a questionnaire before the user test. Questions regarding age, gender and experience with computers, VR and audio games were asked.
>
> *Each-Test-Questionnaire*: The participants received a questionnaire immediately after each audio game (two in total). The task here was to draw the room and to assign an order to the sound sources correctly, including positions of the walls. They were asked to rate the difficulty of the tasks and how sure they think their solution was.
>
> *Post-Test-Questionnaire*: It consisted of a simulator sickness questionnaire (SSQ), questions regarding the locations of sound sources and about immersion as well as realism.

The *Pre-Test-Questionnaire* was intended to obtain answers about the participants' prior study experience. The gender and age was used to find possible differences in terms of e.g. completion time. The questions concerning the experience with computers, VE, computer games, audio games and if they play an instrument or make music was used to get information about the level of experience that participants had. Especially the question concerning the experience with music was used for finding differences between the groups that answers with "*I have experience*" or "*I have no experience.*"

The *Each-Test-Questionnaire* consisted of two parts. First, the participants had to draw the virtual room they were in into a map of the real environment, including the positions of the bombs and the phone. After that, they had to fill in how difficult it was to draw that room as well as how confident they were that their solution was correct. In the second part, the border of the room was already drawn into the map, including the positions of the sound sources. The participants had to bring them into correct order and drew the walls. Afterwards, the questions regarding difficulty and correctness were asked again, with an additional question about the difficulty of drawing the walls. This questionnaire existed in two versions, one for the simple room and one for the complex room. Different questionnaires were necessary because the sound sources were placed at different positions. The drawings as well as the questions were asked for the investigation of H_4.

The *Post-Test-Questionnaire* started with the Simulator Sickness Questionnaire (SSQ). Normally, participants of a study with SSQ have to fill in one questionnaire at the beginning and one at the end of the study. However, in this thesis, only post-study SSQ was used. SSQs are normally used to test interactive software that generates visual output. Visual disturbances normally contribute to the simulator sickness. It was not expected to find elevated levels of sickness symptoms due to the fact that audio games do not have any visuals at all. The SSQ in this user study was used out of curiosity and to possibly have a comparison for other studies. After the participants filled in the SSQ, they were asked if they found the Introductory Scene useful.

The following questions were asked twice in the Post-Test-Questionnaire, once for each room. The first six questions were related to *immersion* (H_6). Participants were asked how aware they were of events occurring around them and how realistic their sense of moving in the virtual world was. The latter question was asked to evaluate effects of real-time spatialization on immersion. The participants were also asked if they felt confused or disorientated during or at the end of the session. This could have happen when the participant was not used to walk around blindfolded which could decrease the degree of immersion. The last two questions were about how involved they were and how quickly they adjusted to the VE experience.

After the questions about immersion were answered, four questions about the *difficulty* (H_2) of finding sound sources in the 3D environment were asked. The difficulty of finding the ringing phone, the first, second and third bomb were asked. These questions were used to assess the difficulty of finding sound sources when different stated sound models were used. After that, participants were asked how much they liked the sound in general and how realistic it seemed to them. These questions were used to assess the correctness of hypothesis (H_5) concerning *realism*. The last questions concerned perceived sound reflections, their realism and helpfulness. These two questions were asked to evaluate if the participants were able to recognize sound reflections correctly and how helpful they found them for task fulfillment.

5.3.3.2 Objective Measurements: Logging

During the experiment, *events* and the *positions* of the player were logged in Unity3D.

An event is a situation in which the player reaches a game state or hits an object. A game state is reached when the audio game begins (Game start) and when the audio game is completed (Game over). A logged event consists of a timestamp and the corresponding event text. When the player reached the position of a sound source (phone, first, second and third bomb) or hit a wall it is also logged as an event. When collisions between the player and walls are logged it is possible to calculate the hit-wall metric that defines how often a participant went through virtual walls. An example of an event log is shown below.

eventLog.txt

```
0.000 - ============ Game start ============
0.004 - Starting game of player with ID: 7259
0.005 - Introscene - no logging. COMPLEX Sound Model.
0.000 - ============ Game start ============
0.000 - Starting game of player with ID: 7259
0.001 - SIMPLE Room Model with selected COMPLEX Sound Model.
```

```
15.582 - Reached status: Phone
46.568 - Reached status: Bomb1
64.876 - Reached status: Bomb2
97.746 - Reached status: Bomb3
97.746 - ============ Game over ============
0.000 - ============ Game start ============
0.000 - Starting game of player with ID: 7259
0.001 - COMPLEX Room Model with selected COMPLEX Sound Model.
15.621 - Reached status: Phone
49.099 - Reached status: Bomb1
86.317 - Hit Wall WallInRoomBottomTopPhys
87.571 - Reached status: Bomb2
101.191 - Hit Wall WallInRoomBottomTopPhys
113.117 - Hit Wall WallInRoomMiddlePhys
121.359 - Reached status: Bomb3
121.359 - ============ Game over ============
```

The position of the player was logged periodically each 500ms as a 3D vector. With this information it is possible to calculate the covered path and distance traveled by a player. Below, there is an example of the position log.

positionLog.txt

```
0.001 - SIMPLE Room Model with selected COMPLEX Sound Model.
-2.9, 1.5, 6.0
-3.1, 1.5, 5.9
-3.1, 1.7, 5.8
-3.0, 1.8, 5.8
-3.1, 1.8, 5.8
...
0.001 - COMPLEX Room Model with selected COMPLEX Sound Model.
-3.3, 1.8, 5.5
-2.8, 1.6, 5.7
...
```

It is possible to calculate the completion time and the time participants need between the tasks with the logged event information (time stamp and event text). This information was used to assess the correctness of H_1. The hit-wall rate was used to investigate H_3. The more participants went through virtual walls the less the assigned sound model supported them in avoiding obstacles.

5.4 Results

The results of objective and subjective measurements of 34 participants were analyzed and are discussed in this section. Statistical software SPSS was used for the statistical analysis.

5.4.1 Difficulty

The participants with the assigned hybrid model completed the game faster in the simple and the complex room. The mean values of completion times and standard deviations are listed in Table 5.1. However, the difference between mean completion times in two rooms has not been found to be statistically significant. Therefore, there is not enough evidence to support H_1.

		Completion Time (s)	
		Mean	σ
Simple Room	Baseline	202.60	88.38
	Hybrid	180.00	72.02
Complex Room	Baseline	204.46	160.37
	Hybrid	178.26	53.09

Table 5.1: Mean values and standard deviations of the completion time.

Participants were asked to rate how quickly they adjusted to the VE experience on the scale from 1 to 7, where 1 corresponded to "*very slowly*" and 7 to "*very quickly.*" According to the resulting scores, the participants adjusted to the hybrid sound model quicker than to the baseline model. The results are listed in Table 5.2. The difference is statistically significant ($p = 0.03$ in Student's T-test, α = 0.05) for the simple room. The null hypothesis is valid with the probability close to marginal for the complex room ($p = 0.08$ in Student's T-test, α = 0.05). Higher subjective adjustment scores are in accordance with shorter game completion times for the hybrid model.

		Adjustment to VE Experience	
		Mean	σ
Simple Room	Baseline	4.97	1.40
	Hybrid	5.94	1.10
Complex Room	Baseline	5.77	1.16
	Hybrid	6.41	0.89

Table 5.2: Mean values and standard deviations of the scores obtained for questions about the adjustment to the VE experience. 1 corresponds to "*adjusted very slowly*" and 7 to "*adjusted very quickly.*"

The participants were asked to rate the difficulty of locating the sound sources in the VE. The results are illustrated in Table 5.3. There is little difference in the mean scores for each sound source if compared between two sound models. H_2 is therefore not supported.

The hit-wall rate measures how often a participant went through a virtual wall. The mean values, standard deviations, minima and maxima are listed in Table 5.4. The mean hit-wall rate with the baseline model was 1.06 and only 0.29 in the hybrid model in the simple room. This difference is statistically significant ($p = 0.00$ in Kruskal-Wallis test, α = 0.05) and supports H_3 in the simple room. In fact, 13 of out 17 participants found all sound sources in the simple room with the hybrid model without hitting a wall, while only one out of 17 participants with the

		Sound Source							
		P	σ	B_1	σ	B_2	σ	B_3	σ
Simple Room	Baseline	3.24	1.65	3.71	1.92	3.85	1.59	3.50	1.80
	Hybrid	3.35	1.51	3.71	1.71	3.68	1.65	4.24	1.78
Complex Room	Baseline	2.77	1.72	2.59	1.63	3.44	1.85	4.12	1.79
	Hybrid	2.82	1.92	2.41	1.43	3.18	1.46	4.29	1.85

Table 5.3: Mean values and standard deviations of the resulting scores for the questions about difficulty of finding the phone (P) and the three bombs (B_n). Answers were given on a scale from 1 to 7, where 1 meant "*very easy*" and 7 "*very difficult.*"

assigned baseline model did not hit the wall. In the complex room, the result was not statistically significant and therefore does not support H_3 in this room. However, the mean hit-wall rate in the hybrid model is lower. The maximum hit-wall rate of the baseline model in the complex room is almost twice as large as of the hybrid model (hit-wall rate baseline model: 11; hybrid model 6).

		Hit-Wall Rate			
		Mean	σ	Min	Max
Simple Room	Baseline	1.06	0.42	0	2
	Hybrid	0.29	0.46	0	1
Complex Room	Baseline	4.59	2.47	3	11
	Hybrid	3.65	1.53	2	6

Table 5.4: Mean values, standard deviations, minima and maxima of the measured hit-wall rate (in times a participant went through a virtual wall during an experiment).

None of the participants were able to reconstruct the virtual room correctly without given borders. An example of a participant's drawing with an overlay of the correct solution is shown in Figure 5.5. The start position was at the bottom right corner. The numbers indicate the position of the sound sources, marked from 1 (phone) to 4 (third bomb). Participants were not able to mark the position of the last bomb correctly, although they removed the head-mounted display and the headphones at this position.

Only four participants reconstructed and ordered the sound sources of the simple room correctly. Eight could reconstruct parts of the complex room (no one was able to recreate it entirely), while only six assigned the order of the sound sources correctly. Two of them assigned the order of the sound sources correctly in both rooms. The corresponding numbers are shown in Table 5.5. The remaining 28 participants could not reconstruct the order in which they discovered the sound sources. H_4 is not supported.

The results show that the reconstruction of a VE which is only audible but not visible is generally a difficult task. During the brief talks after the session, several participants reported that they had experienced a sudden change of loudness when they went through a wall but were not able to identify it as a wall during the test. This change was caused due to a different outcome of the sound model in front of and behind the wall. In the baseline model, this is caused by the

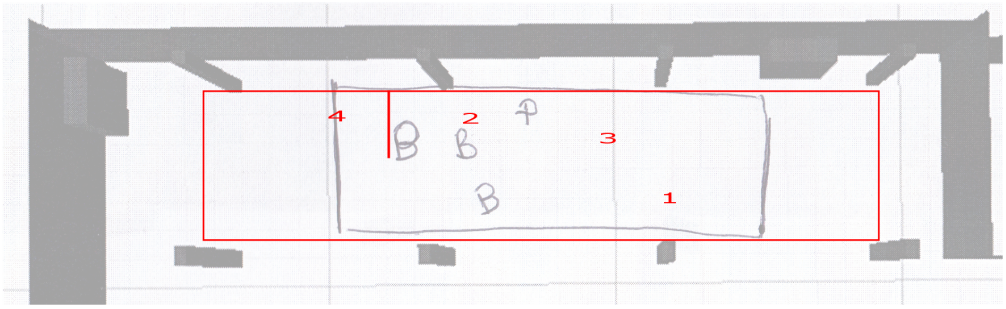


Figure 5.5: An exemplary drawing of a participant, who had to draw the virtual room without given borders. The red overlay shows the correct solution.

		Correctly ordered Sound Sources			
		0	**1**	**2**	**4**
Simple Room	Baseline	2	7	7	1
	Hybrid	4	7	3	3
Complex Room	Baseline	2	10	3	2
	Hybrid	1	5	7	4

Table 5.5: Number of participants who ordered n (0, 1, 2 or 4) sound sources correctly in the Each-Test-Questionnaires.

sound difference in obstruction calculation. However, in the hybrid model, the differences in obstruction and additionally the different calculations of sound reverberation in front of and behind a wall are responsible for perceiving this effect.

This shows that the participants were not familiar with sound behavior and did not think about possible obstacles in the VE. However, the hybrid model provides more precise guidance to the sound source in case of a simple VE with obstacles (simple room) as demonstrated by the partial confirmation of H_3.

5.4.2 Realism and Immersion

This section describes the subjective results obtained by the evaluation of the questionnaires regarding realism and immersion for the confirmation or rejection of H_5 and H_6.

Subjective scores for sound realism were high for both models and are shown in Table 5.6. This result indicates that the spatialization and obstruction in the baseline model and spatialization, obstruction and reverberation of the hybrid model provide realistic sound output but do not differ in perceived realism when compared. This could have two reasons. On the one hand, this could indicate that participants are not much aware of sound phenomena. On the other hand, this could indicate that the reverberation in the proposed hybrid sound model was not convincing enough. However, the baseline model does not have any reverberation. It therefore should be a less realistic sound model compared to a sound model with reverberation. The mean values of the baseline model for both rooms are equal (5.41 out of 7) and high. The mean values of the

hybrid model are lower (5.06 in the simple room, 4.68 in the complex room) compared to the baseline model. The marginal difference could be caused by the unfamiliarity of computer generated sound. The subjective scores do not differ significantly between the two sound models. Therefore, H_5 is not supported.

		Realism of Sound	
		Mean	σ
Simple Room	Baseline	5.41	1.49
	Hybrid	5.06	1.46
Complex Room	Baseline	5.41	1.47
	Hybrid	4.68	1.54

Table 5.6: Mean values and standard deviations of the resulting scores for the questions about sound realism. Answers were given on a scale from 1 to 7, where 1 meant "*not realistic at all*" and 7 "*very realistic.*"

The participants were asked whether they had experienced sound reflections or not. 59% of the users who had the baseline model (without reflections) assigned stated that they had heard reflections. 24% had not heard any reflections, 17% were unsure. Corresponding numbers for the hybrid model (with modeled reflections) are 35%, 29% and 36%. The values are listed in Table 5.7. The confusion about sound reflections might be attributed to the fact that users are not trained in listening to specific sound effects. The simulated reverberation might also be not perceived as a real one although the scores for the question regarding realism were high (see Table 5.6). A study with within-subject design might show the difference in the perceived sound realism between models. In such studies, participants could be asked which sound model they find more realistic or which one they would prefer.

		Reflection heard		
		No	Not Sure	Yes
Assigned Model	Baseline	24%	17%	59%
	Hybrid	29%	36%	35%

Table 5.7: Sound reflection heard by the participants playing the game with the assigned baseline and hybrid model.

The participants were asked, how aware they were of events occurring in the real world during the test and how involved they were in the VE experience. These questions were used to assess the users' immersion. The answers indicate low awareness of real events (see Table 5.8) and high involvement in the VE experience (see Table 5.9).

Real events occurring around a player during the experiment could have been foot steps or the presence of the test coordinator who always was next to the participants. However, as the scores indicate, participants were not aware of real events around them. All scores are smaller than three (see Table 5.8).

The tested prototype provides a highly immersive audio game experience independent of the used sound model. The mean values (as shown in Table 5.9) of both sound models in both

rooms are larger than five, indicating the high degree of immersion. The mean value of the hybrid sound model in the complex room is larger than six.

However, the mean scores regarding the awareness and involvement are not different for both sound models. Therefore, H_6 is not supported.

		Awareness of Real Events	
		Mean	σ
Simple Room	Baseline	2.88	1.68
	Hybrid	2.29	1.35
Complex Room	Baseline	2.94	1.65
	Hybrid	2.18	1.20

Table 5.8: Mean values and standard deviations of the resulting scores for the questions about awareness of occurring events during the test session. Answers were given on a scale from 1 to 7, where 1 meant "*not aware at all*" and 7 "*very aware.*"

		Involvement	
		Mean	σ
Simple Room	Baseline	5.94	1.04
	Hybrid	5.47	1.56
Complex Room	Baseline	5.94	1.10
	Hybrid	6.06	1.13

Table 5.9: Mean values and standard deviations of the resulting scores for the questions about involvement during the test session. Answers were given on a scale from 1 to 7, where 1 meant "*not involved at all*" and 7 "*very much involved.*"

The results show that the completion time in the hybrid sound model was shorter compared to the baseline model, however, the result was not found to be significant. Participants adjusted to the hybrid sound model faster than to the baseline sound model in the simple room. However, there was no difference in difficulty of finding sound sources found when comparing the subjective scores of the participants. The complex sound model supported the players in avoiding obstacles in simple room structures. The reconstruction of a virtual room without seeing but hearing is a difficult task. The sound in both sound models was rated to be very realistic. However, participants were not able to identify the missing reflections in the baseline model and the reflections in the hybrid model correctly. The degree of immersion was found to be high in both tested sound models.

5.4.3 Observations

This section describes observations done by the test coordinator during the user study.

Participant's strategy

The optimal strategy for fulfilling the tasks in the proposed audio game would be that a participant first localizes the sound source direction by moving the head (dynamic sound localization, see Section 2.1.2) and then walks towards the sound source position. The nearer the participant get to the sound source position, the louder the volume is. Participants had two strategies in finding the sound sources that are described in the next paragraphs.

One strategy was the localization of the sound source position by trying to walk in different directions. Participants who followed this strategy completed the game fast, however, they had problems in finding the exact position that was necessary to fulfill a task.

Another strategy was a more cautious one. Participants who followed this strategy moved slowly and stopped when they were not sure about the sound source direction. This strategy was a slower one, however, participants did not have problems in finding the exact sound source location compared to the last strategy.

In both mentioned strategies, the participants usually moved forward. However, two participants also moved backward. There are two possible reasons for this. On the one hand, they could have just passed a sound source position and wanted to go back to determine if they have passed it. On the other hand, they could have went through a wall and experienced the significant sound differences again by moving back and forth.

Reactions

Participants reacted differently when they reached a sound source position. Some of them said "There it is" or "Oh, here." Others just moved to the next task without any further reaction.

When the participants received the Each-Test-Questionnaire the first time after they completed the first audio game, they smiled or laughed. The task in this questionnaire was to draw the virtual room they were in. The reason for this reaction was that they did not expect this task at all. The task of drawing a virtual but only audible room is a difficult task, as shown in Section 5.4.1.

Participants also reacted differently on the virtual instructions. The phone voice was modulated so that it sound like a kidnapper requesting money. This was intended to produce a more intensive and thrilling feeling of this audio game. When the phone call had started, participants also smiled or were a bit afraid.

At the end of the session, the test coordinator showed the complex room model to the participants. The majority was surprised that the room had such a complex structure.

Two participants showed indication of fear. One of them folded her arms and was apparently relieved when she completed the game, the other talked to herself during the experiment to calm herself down. She also repeated "Oh God, oh God, oh God." A reason for this could be the scary shadow syndrome described above so that participants imagine a horrible situation they are currently in that makes them afraid [20]. However, no participant was harmed during the experiment.

CHAPTER 6

Conclusion

Audio games that do not have any visual output at all provide players with an immersive game experience only by simulating sounds. These sounds have to be compelling, otherwise the desired degree of immersion in the game will not be reachable.

In this thesis, a real-time implementation of a hybrid sound model that calculates VE geometry-dependent reverberation in an audio game is presented. The hybrid model is implemented in Unity3D game engine using Wwise with AstoundSound plug-in for real-time sound spatialization. AstoundSound was compared to other sound spatialization tools and was found to be the most appropriate one. The implemented hybrid sound model is compared to the baseline sound propagation model typically used in audio games. This comparison is conducted in a user study where the participants had to play a 3D audio game while walking blindfolded in a large tracked space.

The results of the study indicate that both tested models provide a highly immersive game experience. The proposed hybrid approach provides players with more information about the VE. Although the differences in completion time were not significant, participants who had the hybrid sound model assigned, completed the game tasks faster. In the implemented audio game, the participants with the hybrid sound model adjusted to the VE faster than the participants with the assigned baseline sound model.

Participants had to reconstruct the room and the position of the perceived sound sources. As the results show, the reconstruction of a virtual, non-visual scene is a difficult task. Aspects influencing this difficulty will be investigated in further studies with sound models and audio games.

It can be concluded that a complex sound model supports the player in avoiding obstacles in simple room geometries. It is advisable for game designers and developer to integrate a complex sound model in their games, as long as the computational power is available and the player has to avoid obstacles.

In future work, the used sound model may be implemented by using GPU-based methods. This model can then be compared to more accurate sound models. GPU-based calculations offer higher performances, however, a native implementation instead of using an existing audio

middleware solution might be a more performant option, especially when real time calculation of a large amount of sound sources is necessary.

Authors' comment:

When I was starting an investigation in the domain of audio games, I was really surprised that only few scientists are researching in this scientific field. I hope that the scientific community will get more attention to audio games, because in my humble opinion, non visual impaired people rely too much on their visual sense while they forget that they still have others.

Bibliography

[1] *Fundamentals of Telephone Communication Systems*. Western Electric Company, 1969.

[2] Ernest Adams. *Fundamentals of Game Design*. New Riders Publishing, Thousand Oaks, CA, USA, 2nd edition, 2009.

[3] Audiokinetic, Inc. Customers. `https://www.audiokinetic.com/community/customers/`, 2015. [Online; accessed 25-August-2015].

[4] Audiokinetic, Inc. Wwise fundamentals. `https://www.audiokinetic.com/download/documents/Wwise_Fundamentals.pdf`, 2015. [Online; accessed 25-August-2015].

[5] Audiokinetic, Inc. Wwise help 2014.1 build 5158 - capturing data from the sound engine, 2015. [Online; accessed 25-August-2015].

[6] Auro Technologies. Technology. `http://www.auro-3d.com/consumer/technology/`, 2015. [Online; accessed 25-August-2015].

[7] Corey I Cheng and Gregory H Wakefield. Introduction to head-related transfer functions (hrtfs): Representations of hrtfs in time, frequency, and space. In *Audio Engineering Society Convention 107*. Audio Engineering Society, 1999.

[8] Erin C Connors. Audio-based virtual environments: Teaching spatial navigation skills to the blind.

[9] T Drewes, E Mynatt, Maribeth Gandy, et al. Sleuth: An audio experience. In *Proceedings of The International Conference on Auditory Display*, 2000.

[10] Angelo Farina. Ramsete-a new pyramid tracer for medium and large scale acoustic problems. In *Proc. of Euro-Noise*, volume 95, 1995.

[11] Maria Fellner and Robert Höldrich. *Physiologische und Psychoakustische Grundlagen des räumlichen Hörens*. IEM-Report 03/98, 1998.

[12] Johnny Friberg and Dan Gärdenfors. Audio games: new perspectives on game audio. In *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 148–154. ACM, 2004.

[13] Bill Gardner, Keith Martin, et al. Hrft measurements of a kemar dummy-head microphone. 1994.

[14] Lalya Gaye. A flexible 3d sound system for interactive applications. In *CHI'02 Extended Abstracts on Human Factors in Computing Systems*, pages 840–841. ACM, 2002.

[15] GenAudio, Inc. Our products. `http://www.astoundholdings.com/products.php`, 2015. [Online; accessed 25-August-2015].

[16] W.M. Hartmann. *Principles of Musical Acoustics*. Undergraduate Lecture Notes in Physics. Springer, 2013.

[17] Florian Heller, Thomas Knott, Malte Weiss, and Jan Borchers. Multi-user interaction in virtual audio spaces. In *CHI'09 Extended Abstracts on Human Factors in Computing Systems*, pages 4489–4494. ACM, 2009.

[18] Thomas Hermann, Andy Hunt, and John G. Neuhoff. Introduction. In Thomas Hermann, Andy Hunt, and John G. Neuhoff, editors, *The Sonification Handbook*, chapter 1, pages 1–6. Logos Publishing House, Berlin, Germany, 2011.

[19] R.C. Jaiswal. *Audio-Video Engineering*. Nirali Prakashan.

[20] Mats Liljedahl, Nigel Papworth, and Stefan Lindberg. Beowulf: an audio mostly game. In *Proceedings of the international conference on Advances in computer entertainment technology*, pages 200–203. ACM, 2007.

[21] Mauricio Lumbreras and Gustavo Rossi. A metaphor for the visually impaired: browsing information in a 3d auditory environment. In *Conference companion on Human factors in computing systems*, pages 216–217. ACM, 1995.

[22] Mauricio Lumbreras, J Sánchez, and M Barcia. A 3d sound hypermedial system for the blind. In *Proceedings of the First European Conference on Disability, Virtual Reality and Associated Technologies*, pages 187–191, 1996.

[23] Richard H Lyon. *Theory and application of statistical energy analysis*. Elsevier, 2014.

[24] Matija Marolt. A new approach to hrtf audio spatialization. In *Proceedings of the international computer music conference*, pages 365–367. Citeseer, 1996.

[25] Christian Maschke and André Jakob. Psychoakustische messtechnik. In Michael Mörser, editor, *Messtechnik der Akustik*, chapter 11, pages 599–642. Springer-Verlag Berlin Heidelberg, Berlin, Germany, 2010.

[26] FP Mechel. Improved mirror source method in roomacoustics. *Journal of Sound and vibration*, 256(5):873–940, 2002.

[27] Ravish Mehra, Atul Rungta, Abhinav Golas, Ming Lin, and Dinesh Manocha. Wave: Interactive wave-based sound propagation for virtual environments. *Visualization and Computer Graphics, IEEE Transactions on*, 21(4):434–442, 2015.

[28] Philip Mendels and Joep Frens. The audio adventurer: Design of a portable audio adventure game. In *Fun and Games*, pages 46–58. Springer, 2008.

[29] Lotfi B Merabet, Erin C Connors, Mark A Halko, and Jaime Sánchez. Teaching the blind to find their way by playing video games. *PloS one*, 7(9):e44958, 2012.

[30] Matjaz Mihelj, Domen Novak, and Samo Begus. *Virtual Reality Technology and Applications*. Springer Publishing Company, Incorporated, 2013.

[31] M Taylor A Chandak Q Mo and C Lauterbach C Schissler D Manocha. isound: Interactive gpu-based sound auralization in dynamic scenes.

[32] GM Naylor. Treatment of early and late reflections in a hybrid computer model for room acoustics. In *124th ASA Meeting*, 1992.

[33] Graham M Naylor. Odeon – another hybrid room acoustical model. *Applied Acoustics*, 38(2):131–143, 1993.

[34] Oculus VR, LLC. 3d audio spatialization. `https://developer.oculus.com/documentation/audiosdk/latest/concepts/audio-intro-spatialization/`, 2015. [Online; accessed 25-August-2015].

[35] Odeon A/S. Odeon user manual version 13. `http://www.odeon.dk/download/Version13/ODEONManual.pdf`, 2015. [Online; accessed 25-August-2015].

[36] Vojin G. Oklobdzija. *The Computer Engineering Handbook: Electrical Engineering Handbook*. CRC Press, Inc., Boca Raton, FL, USA, 2001.

[37] Irwin Pollack and JM Pickett. Cocktail party effect. *The Journal of the Acoustical Society of America*, 29(11):1262–1262, 1957.

[38] Jens Holger Rindel. The use of computer modeling in room acoustics. *Journal of Vibroengineering*, 3(4):41–72, 2000.

[39] Niklas Röber and Maic Masuch. Interacting with sound: An interaction paradigm for virtual auditory worlds. In *ICAD*, 2004.

[40] Niklas Röber and Maic Masuch. Leaving the screen: New perspectives in audio-only gaming. In *11th Int. Conf. on Auditory Display (ICAD)*. Citeseer, 2005.

[41] Joseph D Rogers and Marc E Rogers. Three-dimensional virtual environment, October 23 2013. US Patent App. 14/061,711.

[42] Francis Rumsey and Tim McCormick. Sound and recording. 2014.

[43] Jaime Sánchez, Nelson Baloian, Tiago Hassler, and Ulrich Hoppe. Audiobattleship: Blind learners collaboration through sound. In *CHI'03 Extended Abstracts on Human Factors in Computing Systems*, pages 798–799. ACM, 2003.

[44] Jaime Sánchez, Marcia de Borba Campos, Matías Espinoza, and Lotfi B Merabet. Audio haptic videogaming for developing wayfinding skills in learners who are blind. In *Proceedings of the 19th international conference on Intelligent User Interfaces*, pages 199–208. ACM, 2014.

[45] Jaime Sánchez and Mauricio Lumbreras. Virtual environment interaction through 3d audio by blind children. *CyberPsychology & Behavior*, 2(2):101–111, 1999.

[46] Lauri Savioja, Jyri Huopaniemi, Tapio Lokki, and Ritta Väänänen. Creating interactive virtual acoustic environments. *Journal of the Audio Engineering Society*, 47(9):675–705, 1999.

[47] M. Schroeder, Thomas D. Rossing, F. Dunn, W. M. Hartmann, D. M. Campbell, and N. H. Fletcher. *Springer Handbook of Acoustics*. Springer Publishing Company, Incorporated, 1st edition, 2007.

[48] EAG Shaw. External ear response and sound localization. *Localization of sound: Theory and applications*, pages 30–41, 1982.

[49] Samuel Siltanen, Tapio Lokki, and Lauri Savioja. Rays or waves? understanding the strengths and weaknesses of computational room acoustics modeling techniques. In *Proc. Int. Symposium on Room Acoustics*, 2010.

[50] Mel Slater, Martin Usoh, and Anthony Steed. Taking steps: The influence of a walking technique on presence in virtual reality. *ACM Trans. Comput.-Hum. Interact.*, 2(3):201–219, September 1995.

[51] Sue Targett and Mikael Fernström. Audio games: Fun for all? all for fun. In *ICAD*, 2003.

[52] Micah T Taylor, Anish Chandak, Lakulish Antani, and Dinesh Manocha. Resound: interactive sound rendering for dynamic virtual environments. In *Proceedings of the 17th ACM international conference on Multimedia*, pages 271–280. ACM, 2009.

[53] Unity Technologies. Audio overview. `http://docs.unity3d.com/Manual/AudioOverview.html`, 2015. [Online; accessed 25-August-2015].

[54] Unity Technologies. Unity. `https://unity3d.com/`, 2015. [Online; accessed 25-August-2015].

[55] Eric Velleman, Richard van Tol, Sander Huiberts, and Hugo Verwey. *3d shooting games, multimodal games, sound games and more working examples of the future of games for the blind*. Springer, 2004.

[56] Michael Vorländer. *Auralization: fundamentals of acoustics, modelling, simulation, algorithms and acoustic virtual reality*. Springer Science & Business Media, 2007.

[57] Hans Wallach, Edwin B Newman, and Mark R Rosenzweig. A precedence effect in sound localization. *The Journal of the Acoustical Society of America*, 21(4):468–468, 1949.

[58] Thomas Westin. Game accessibility case study: Terraformers–a real-time 3d graphic game. In *Proceedings of the 5th International Conference on Disability, Virtual Reality and Associated Technologies, ICDVRAT*, pages 95–100, 2004.

[59] G. White and G.J. Louie. *The Audio Dictionary: Third Edition, Revised and Expanded*. University of Washington Press, 2005.

[60] Wikipedia. Audio game — Wikipedia, the free Encyclopedia. `http://en.wikipedia.org/wiki/Audio_game`, 2015. [Online; accessed 25-August-2015].

[61] Fredrik Winberg and Sten Olof Hellström. Investigating auditory direct manipulation: sonifying the towers of hanoi. In *CHI'00 extended abstracts on Human factors in computing systems*, pages 281–282. ACM, 2000.

[62] John Wood, Mark Magennis, Elena Francisca Cano Arias, Teresa Gutierrez, Helen Graupp, and Massimo Bergamasco. The design and evaluation of a computer game for the blind in the grab haptic audio virtual environment. *Proceedings of Eurohpatics*, 2003.

APPENDIX A

Questionnaires

A.1 Pre-Test-Questionnaire

Pre-test:

Your age:

Your gender:

male female other

Please indicate your answer to the following questions on the scale from 1 to 7, where 1 means „not at all“ and 7 means “very much”:

How much experience of computer usage do you have?

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

How much experience of virtual reality do you have?

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

How much experience of playing computer games do you have?

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

How much experience of playing audio games do you have?

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

Do you play a musical instrument, sing or make music?

yes no

If yes, how professional are you at it?

1 7

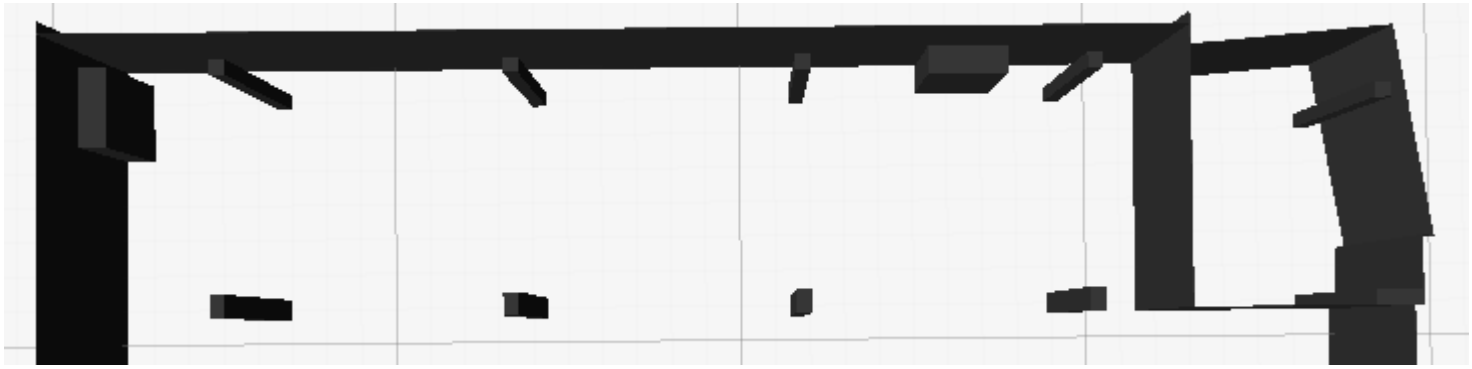
|----------|----------|----------|----------|----------|----------|

not very professional very professional

A.2 Each-Test-Questionnaire Simple Room

Each-test:

Please draw the virtual room you were in, including the phone and bombs.



How difficult was it to draw the room? Please indicate your answer on the following scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very difficult

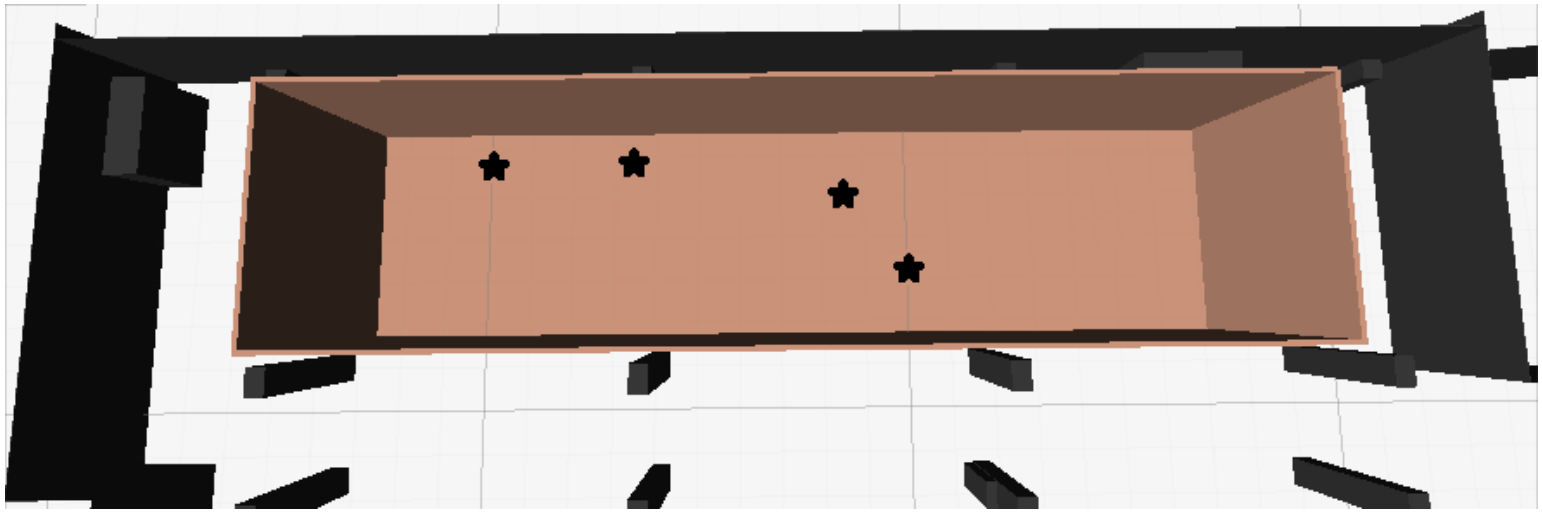
How correct do you think your drawing is? Please indicate your answer on the following scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all absolutely correct

In the following image, you can see the room you were in with the positions of the phone and the bombs. Assign an order to the stars, starting with 1 for the phone and finishing with 4 for the last bomb. Put in the walls.



How difficult was it to assign the order? Please indicate your answer on the following scale:

1 7

|---------|---------|---------|---------|---------|---------|

not at all very difficult

How difficult was it to draw the walls? Please indicate your answer on the following scale:

1 7

|---------|---------|---------|---------|---------|---------|

not at all very difficult

How correct do you think your solution is? Please indicate your answer on the following scale:

1 7

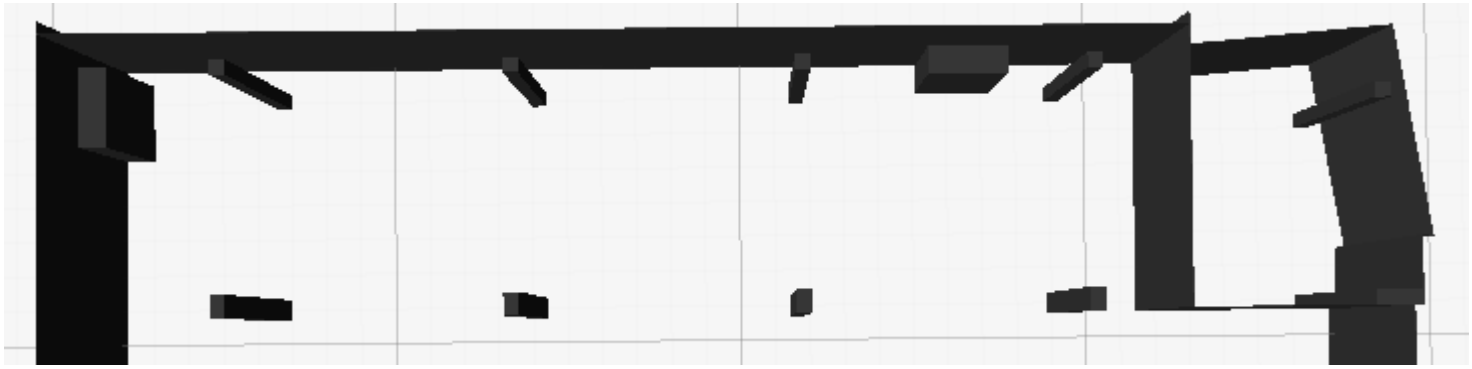
|----------|----------|----------|----------|----------|----------|

not at all absolutely correct

A.3 Each-Test-Questionnaire Complex Room

Each-test:

Please draw the virtual room you were in, including the phone and bombs.



How difficult was it to draw the room? Please indicate your answer on the following scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very difficult

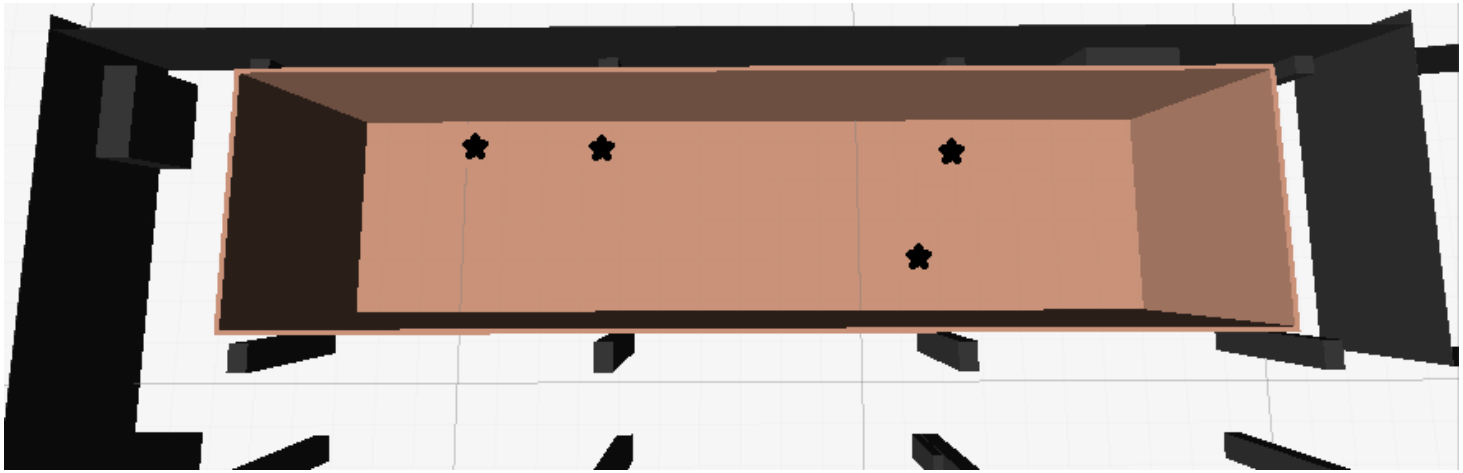
How correct do you think your drawing is? Please indicate your answer on the following scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all absolutely correct

In the following image, you can see the room you were in with the positions of the phone and the bombs. Assign an order to the stars, starting with 1 for the phone and finishing with 4 for the last bomb. Put in the walls.



How difficult was it to assign the order? Please indicate your answer on the following scale:

1 7

|---------|---------|---------|---------|---------|---------|

not at all very difficult

How difficult was it to draw the walls? Please indicate your answer on the following scale:

1 7

|---------|---------|---------|---------|---------|---------|

not at all very difficult

How correct do you think your solution is? Please indicate your answer on the following scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all absolutely correct

A.4 Post-Test-Questionnaire

Post-test:

Simulator-sickness questionnaire:

Circle how much each sypmtom is affecting you right now:

General discomfort	None	Slight	Moderate	Severe
Fatigue	None	Slight	Moderate	Severe
Headache	None	Slight	Moderate	Severe
Eye strain	None	Slight	Moderate	Severe
Difficulty focusing	None	Slight	Moderate	Severe
Salivation increasing	None	Slight	Moderate	Severe
Sweating	None	Slight	Moderate	Severe
Nausea	None	Slight	Moderate	Severe
Difficulty concentrating	None	Slight	Moderate	Severe
« Fullness of the Head »	None	Slight	Moderate	Severe
Blurred vision	None	Slight	Moderate	Severe
Dizziness with eyes open	None	Slight	Moderate	Severe
Dizziness with eyes closed	None	Slight	Moderate	Severe
Vertigo	None	Slight	Moderate	Severe
Stomach awareness	None	Slight	Moderate	Severe
Burping	None	Slight	Moderate	Severe

To what extent was the introductory session helpful for the task fulfilment? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very helpful

The following questions concern the first test session:

During the testing session, how aware were you of the events occurring in the real world around you? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not aware at all very aware

How realistic was your sense of moving around inside the virtual environment? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not realistic at all very realistic

To what degree did you feel confused or disoriented during the session? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

To what degree did you feel confused or disoriented at the end of the session? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

How involved were you in the virtual reality experience? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

How quickly did you adjust to the VE experience? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very slowly very quickly

How difficult was it for you to localize the ringing phone in the room? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very easy very difficult

How difficult was it for you to localize the first bomb? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very easy very difficult

How difficult was it for you to localize the second bomb? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very easy very difficult

How difficult was it for you to localize the third bomb? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very easy very difficult

How much did you like the sound in general? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

How realistic did the sound in general seem to you? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not realistic at all very realistic

Did you hear any sound reflections?

yes no not sure

If yes, how realistic did they seem to you? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not realistic at all very realistic

How helpful or confusing did they seem to you? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very confusing very helpful

The following questions concern the second test session:

During the second testing session, how aware were you of the events occurring in the real world around you? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not aware at all very aware

How realistic was your sense of moving around inside the virtual environment? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not realistic at all very realistic

To what degree did you feel confused or disoriented during the session? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

To what degree did you feel confused or disoriented at the end of the session? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

How involved were you in the virtual reality experience? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

How quickly did you adjust to the VE experience? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very slowly very quickly

How difficult was it for you to localize the ringing phone in the room? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very easy very difficult

How difficult was it for you to localize the first bomb? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very easy very difficult

How difficult was it for you to localize the second bomb? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very easy very difficult

How difficult was it for you to localize the third bomb? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very easy very difficult

How much did you like the sound in general? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not at all very much

How realistic did the sound in general seem to you? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not realistic at all very realistic

Did you hear any sound reflections?

yes no not sure

If yes, how realistic did they seem to you? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

not realistic at all very realistic

How helpful or confusing did they seem to you? Please answer on the scale:

1 7

|----------|----------|----------|----------|----------|----------|

very confusing very helpful

Please give us any type of feedback, your comments are very welcome: