Underwater Acoustic Simulator for Communications

Steven Kah Hien Wong
steven@zensaki.com

This report is submitted as partial fulfilment of the requirements for the Honours Programme of the School of Computer Science and Software Engineering, The University of Western Australia, 2005

Abstract

In terms of acoustics, the underwater environment is very chaotic. For example, in shallow region of water, sound waves will experience a high level of reverberation due to waves reflecting off the surface and seabed. The time of day can also affect the temperature of the water surface, which will then cause propagating sound waves to refract differently. These are just two of the several factors that can influence underwater sound wave propagation. Because of these large varieties in conditions, signal processing methods need to be specialised for each environment [56]. Testing and developing these signal processing algorithms solely through field experimentation is both expensive and time-consuming. Using a simulator is cost-efficient and quicker to setup.

The NASRay simulator is one such simulator for underwater sound. It cannot be a highly accurate model, since the physics for underwater acoustics is too complex. But the simulator implements enough models to be useful for preliminary testing.

However, the processing time of the simulator was slow. To overcome this shortfall, a block-based processing algorithm was implemented. This allowed operations to be performed on blocks of sample points at a time, as opposed to one sample point at a time. By operating in blocks, less function calls were made, and hence the overall overhead from calling functions several times were reduced.

Orientable grouped movement, for simulating the movements and rotations of a floating hydrophone array, were also added. The orientations are represented using quaternions. This method of representing orientations allowed "smooth" rotations between any two orientations. It also avoided the problem of the Gimbel lock that expressions in Euler angles suffer from.

The simulator also did not support scheduled source transmissions. The old implementation would always immediately start all transmission at the beginning of the simulation. Due to this, it was not possible to create sources that only transmitted when they received a "ping". So support for events were added.

In the end, it became apparent that the scope of the project was too large. Many of the features were implemented, but at the cost of quality. The physics of underwater acoustics was more complicated than expected, and much time was spent with research. Even now, there are numerous additions to the model that can still be added to the simulator to improve realism.

Keywords: communication, Doppler effect, noise, ocean, underwater acoustic model
CR Categories: A.2, I.7.2

Acknowledgements

Janet Graham
Nautronix Ltd.

Nick Spadaccini
University of Western Australia

Anthony Zaknich
University of Western Australia

Matthew Comi
Nautronix Ltd.

Jason McSweeney
Nautronix Ltd.

Contents

Abstract **ii**

Acknowledgements **iii**

1 Introduction **1**

2 Background **3**

2.1 History of Underwater Acoustics 3

2.2 The Nature of Underwater Sound 5

3 Underwater Acoustic Physics in the Ocean **7**

3.1 Propagation Loss 7

3.1.1 Absorption 7

3.1.2 Spreading Loss 11

3.2 Boundary Interactions 13

3.2.1 Reflection and Refraction Coefficient Assumptions 14

3.2.2 Reflection 15

3.2.3 Refraction 20

4 Existing Work **21**

4.1 Funkhouser 21

4.2 AcoustiKit 22

4.3 SEASIM 22

4.4 Geng and Zielinski 22

5 Simulator **23**

5.1 Building and Development Environment 23

5.2 Assumptions and Simplifications 24

5.3 Ray-path-Based Modelling 24

5.4 Pipe Architecture 25

5.5 Time Delays 26
5.6 Reflection 26
5.7 Block-Based Processing 27
5.8 Caching Frequently-Calculated Results 27
5.9 Automated Buffer Size Calculation 28
5.10 Grouped Movement 30
5.11 Scheduled Transmissions 30

6 Conclusion **32**
6.1 Further Work 32

A Original Research Proposal **34**
A.1 Background 34
A.2 Aim 35
A.3 Method 36
A.3.1 Overview 36
A.3.2 People 36
A.3.3 Optimisation 36
A.3.4 Grouped Movement 37
A.3.5 Scheduled Transmission 37
A.3.6 Reflection Model 38
A.3.7 Reverberation Model 38
A.3.8 Refraction Model 38
A.3.9 Occlusion Model 38
A.3.10 Graphical User Interface 38
A.4 Software and Hardware Requirements 39

List of Tables

3.1 Typical relaxation frequencies for seawater at a temperature of 14°C . . . 10

List of Figures

2.1 Energy absorption of sound and light in water 4
2.2 Sound wave travelling down a tunnel 6
2.3 Location of waves represented by a single ray-path 6

3.1 Sound wave with a period equal to medium's relaxation time 9
3.2 Sound wave spreading spherically from source 12
3.3 Sound wave spreading cylindrically 13
3.4 Reflected and refracted rays at a medium interface 14
3.5 Deep-water situations without one or both major reflective bodies 16
3.6 Reflection off a pressure release surface 17
3.7 First and second-order specular reflection ray paths 18
3.8 Source image method for finding n-order reflections 19

5.1 Paths for a configuration of sources and receivers 25
5.2 Old NASRay implementation 28
5.3 New block-based NASRay implementation 29

CHAPTER 1

Introduction

Today, many ranged, wireless communications are achieved through radio waves, a subset of the electromagnetic spectrum. Recently, even the visible light spectrum has been exploited in the form of fibre optics. For several decades, electromagnetic waves, especially radio waves, have seen extensive use in long range communications.

When the propgating medium is water, however, electromagnetic waves find limited use. Such waves can only travel a few metres before experiencing significant attenuation. This can be demonstrated by shining a torchlight underwater and observing how far the beam travels. In certain areas in the world, it is impossible to see past your own outstretched hands under the water, even with intense illumination [8].

It became very clear in the beginning of the 20th century that sound waves could propagate in water much more efficiently than electromagnetic waves [38]. But underwater sound waves were still far from an ideal communication method. Acoustic signals propagating through the ocean can be distorted in many ways. Such distortions include the wave becoming:

- delayed,
- reverberated (multiple echoes of the same signal),
- stretched or compressed, and
- diminished in strength.

In fact, it is most likely a combination of all the above. It is the task of advanced signal processing algorithms to analyze and restore these signals back to their original form.

Individual underwater environments can also vary greatly. For instance, shallow water environments will experience high reverberation; the shorter distance between the surface and seabed will produce more reflections. In deep water, signals are generally less distorted, but it is still significant, especially at longer ranges where attenuation effects become prevalent.

These are a few real-world examples of sound distortion. Firstly, the layer of ice on the surface of Arctic regions are found to reflect sound waves with less scatter. This is because the surface of ice is comparatively flatter than the undulating surface of liquid water. In the tropics, temperatures at different times of day, or different seasons, affect the temperature of the water surface. This leads to changes in the way sound waves will be refracted, and the overall propagational behaviour.

Because of the vast variety of environmental conditions, signal processing methods are commonly specialised for individual environments [56]. Testing and developing these signal processing algorithms solely through field experimentation is both expensive and time-consuming. Using a simulator for preliminary tests are more cost-efficient and faster to setup.

The NASRay simulator is one such underwater acoustic model.

It was performing slower than desired. As the problem was due to excessive function call overhead, I solved this problem by using a block-based processing algorithm.

The simulation of a hydrophone array was also required. To provide this, grouped movement was added with the addition of Quaternion-based orientation and rotation. Quaternions are a mathematical construct used to represent rotations. It provides a way to smoothly interpolate the points between two orientations.

Event-based scheduling was another enhancement made. Before that, sources in the simulator would all transmit sound at the very beginning. With scheduling, sources can now be made to emit a sound only on certain events. These events can actually be propagated through the water, so individual receivers would not detect the event until it had reached them. An example of this use was for ping signals.

Automated memory optimisation additions were done, also. It is quite a hassle to force users to enter in memory requirements before the simulation. An algorithm for automatically determining the minimal amount of buffer memory needed was implemented.

The area of underwater acoustics is extremely complex. The first two chapters will detail the theory behind underwater acoustic wave propagation. There will then be a brief chapter on the simulator and the enhancements I made.

CHAPTER 2

Background

Sound waves can travel through water with comparatively less attenuation than electromagnetic waves. Even though there is still a significant degree of attenuation, sound waves are still more effective than electromagnetic waves. Using information gathered from Clay [8] and Quazi [38], an approximate plot of attenuation levels for sound and light waves in water can be adapted. The result is illustrated in figure 2.1. It is useful for presenting the overall orders of magnitude differences between the wave types. As a general rule, attenuation becomes significant at 10 dB [4, 29]. The dotted lines are individual absorption components for sea water, and can be dismissed for now. The graph's derivation will be discussed further in chapter 3.

Since water is a denser medium than air, sound waves can also propagate faster than they do in air. In air, the speed of sound is accepted to be around 340 m/s. Underwater sounds have speeds of around 1,500 m/s. This is significantly lower than the speed of electromagnetic waves (approximately 300,000 m/s), but still fast enough to provide adequate responsiveness for most applications.

2.1 History of Underwater Acoustics

As stated by McCormick [8], underwater acoustics is an area that is both old and new. The first documented use of underwater acoustics was in 1490, by the famed Leonardo De Vinci [54] - this is how the field is old. He observed that: "If you cause your ship to stop and place the outer extremity to your ear you will hear ships at a great distance from you." But in the time between then and the early 20th century, little progress in underwater acoustics had been made. It was only after that period did the field of underwater acoustics, triggered by a few, key, historical events, begin to grow. This relatively sudden surge of interest in the study is how it is new.

After De Vinci, it was not until nearly two centuries later, in 1687, that a physicist, Daniel Colladon, and mathematician, Charles Sturm, actually measured the speed of underwater sound using a method similar to De Vinci's [27]. They used a long tube and an underwater bell to measure the time taken for the bell's ring, attached to one boat, to reach the listener, on another boat. The speed they came up with was 1,435 m/s – it was only approximately 3 m/s off the speed accepted today!

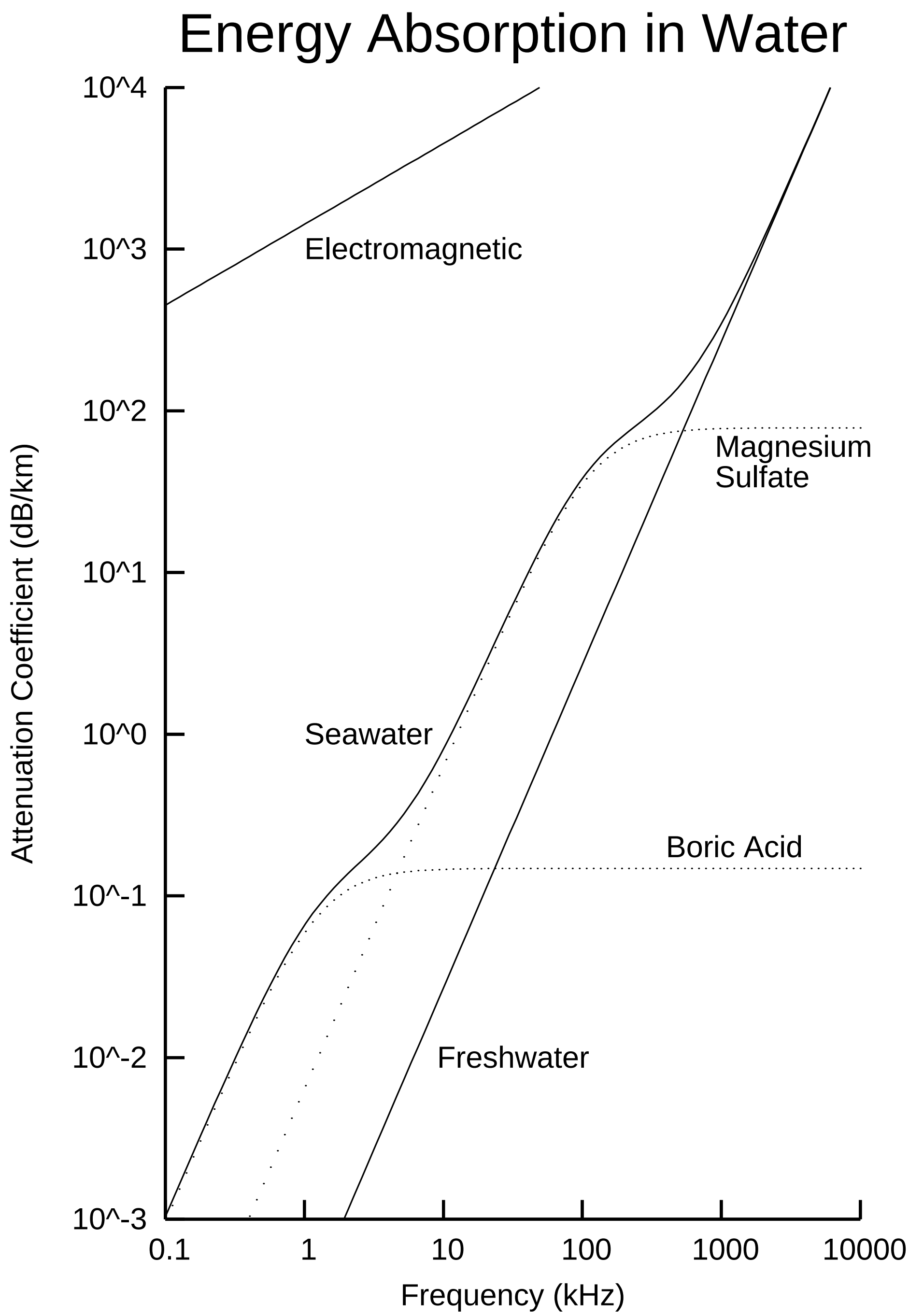

Figure 2.1: Energy absorption of sound and light in water

The tragic sinking of the Titanic in 1912 kindled further research into underwater acoustics [8]. More specifically, the study of echo ranging. This was the technique of sending out short pulses of sound and listening for its return echo. Using this technique, it was possible to detect distant objects in the water. It was in 1914, when R. A. Fessenden successfully detected an iceberg 3.2 km away.

Both World War I and II saw one of the first, most-practical applications of underwater acoustics. This was in the form of *sonar detection*. The first World War saw the initial development of this technology, but it was not ready until the end of that war. The first devices passed current through a quartz crystal, sandwiched between two steel plates, to create rapid oscillations [8]. The sounds generated by these oscillations were the source of "ping" signals. The extensive use of sonar detection for revealing hidden submarines began in World War II. Its success initiated renewed interest into underwater acoustics.

The invention of the "underwater telephone" in 1945, by the Navy, was the first application of underwater acoustics to communications [38]. From here, it is seeing further use in technology for several areas. These included networks of underwater sensors [37], military surveillance probes, and remotely-operated vehicles (R.O.V.). As you can see, the applications of underwater acoustic communications are wide. Today, it is still the only effective means of transmitting information wireless through water.

2.2 The Nature of Underwater Sound

Sound heard by the human ear is the result of vibrations within the air. In this case, the medium that the sound is travelling through is the air. However, in an underwater environment (such as the ocean), the medium is water. Both mediums propagate sound in roughly the same way, but differ in how sensitive they are to individual factors. For example, sound travels faster in water than in the air; water can be seen as more sensitive to pressure variations, and hence react to such variations faster.

An important property of sound is that it is wave-like. In the cases above, the vibrations described are actually variations of pressure. These pressure variations are caused by particle movements in the medium (air or water molecules). When particles move together, they exhibit high pressure; and when they move apart, there is low pressure. These two states are known as *compression* and *rarefaction*, respectively. Figure 2.2 is a representation of a sound wave travelling down a pipe (adapted from Halliday [23]).

Since these are actual particle movements, pressure cannot change instantly from low to high. Instead there must be gradual transitions between the states. In mathematical terms, pressure levels are differentiable at all times. Trigonometric functions such as the *sine* and *cosine* functions are commonly used to model sound waves mathematically. These gradual transitions are what give sound its wave-like form. Due to this, sounds are often referred to as sound waves.

Because sound waves vibrate in the direction they are moving, they are classified as

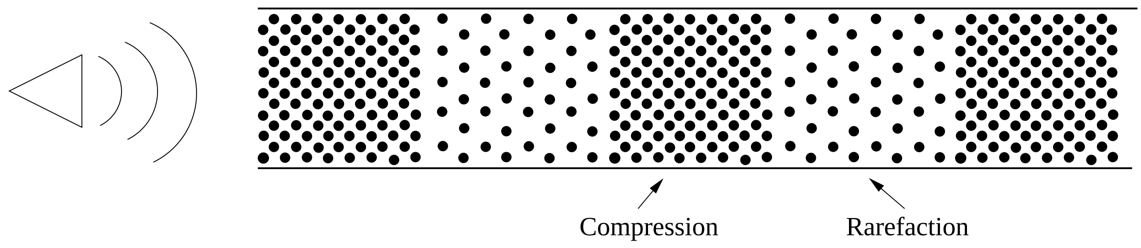

Figure 2.2: Sound wave travelling down a tunnel

longitudinal waves. The other type of waves are *transverse waves*, where the vibrations are perpendicular to the movement of the wave. A common example of a transverse wave is the wave created by shaking one end of a suspended string.

The wave-like nature of sound means it exhibits some interesting behaviours while propagating through water. These behaviours can wreak havoc on communication signals, which is why sophisticated signal processing techniques are required in underwater acoustic communications. Chapter 3 will discuss these behaviours in more detail, along with the underlying physics of underwater acoustic wave propagation.

On a final note, it is sometimes useful to represent sound waves as rays (or *ray-paths*) instead. Many texts about sound propagation will adopt this view of sound waves to ease explanations. These ray-paths are simply the normals of a propagating wave. Figure 2.3 shows where the ray is in a wave reflection.

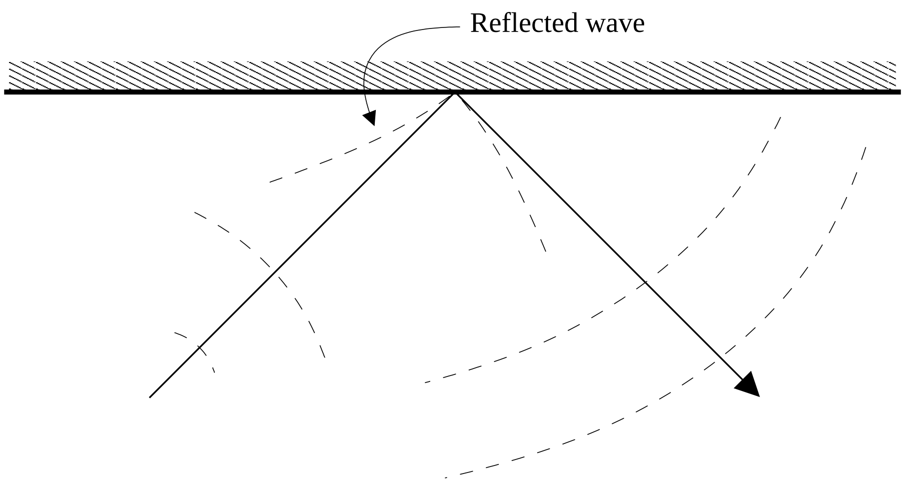

Figure 2.3: Location of waves represented by a single ray-path

CHAPTER 3

Underwater Acoustic Physics in the Ocean

The ocean is a dynamic and complex environment. Water movements are never-ceasing, and conditions are always changing with the time of day and weather. It is generally very difficult to predict.

Acoustic signals travelling through it are distorted by a variety of factors. The major contributors are absorption, refraction and reflection (reverberation). Through these three factors, the signals picked up by receivers are duplicated forms of the original, of varying levels of strength and distorted by a certain degree of spreading or compression. It is the job of modern signal processing techniques to decipher this amalgamation of signals back to its original form.

3.1 Propagation Loss

Propagation loss (also referred to as *transmission loss*) causes a sound wave's amplitude to gradually diminish as it travels. There are two main contributers to this loss. The first is absorption from the actual water medium. The level of absorption experienced is a function of several factors, and will be covered later in this chapter. The second cause of transmission loss is due to spherical spreading. This is where the intensity of the wave decreases as the wave's total power is distributed over its growing spherical surface.

3.1.1 Absorption

One of the ways sound waves can be attenuated is through absorption by the carrier medium. It was initially thought that thermal conductivity was a major factor in this, but this has turned out to be negligible [8]. The two forms of loss are attributed to *shear viscosity* and *bulk viscosity*, with bulk viscosity being the more prominent.

Shear viscosity is caused by frictional forces made by relative motions between layers of liquid. Due mainly to the variations of temperature and pressure at different depths, the ocean is composed of layers of liquid. Each of these layers can move differently to its neighbouring layers and produce frictional forces. The layered nature of the ocean is discussed further in section 3.2.3, where refraction is covered.

Bulk viscosity is the result of volume changes in water as the compressional and rarefaction parts of waves propagate through the water. A process known as *relaxation* is one form of bulk viscosity. Relaxation is the general term given to the processes that occur when the medium encounters pressure changes (compressions and rarefactions). Two forms are relevant for underwater acoustic waves:

Structural relaxation – Molecules absorbing or emitting energy as they move closer or further apart.

Chemical relaxation – Component ions undergoing dissociation or recombination.

The theory of structural relaxation was published by Hall [22]. He theorised that molecules in water could exist in two states: a *normal energy state*, and a *high energy state*, where the molecules are more closely packed. When a group of molecules are compressed (refer to figure 2.2 for an illustration), they will enter the higher state; and when they return back to their neutral positions, they will also return to the normal energy state. To enter the high energy state, energy is absorbed from the sound wave. This energy is released again when the molecule returns to the normal energy state.

Transitions between these two states require a finite amount of time. This time is known as the *structural relaxation time*. If the oscillations of the wave (a function of frequency) are much faster than the relaxation time, then the molecules do not have time to transit states, so no energy is absorbed or returned. If the oscillations are much slower, then the transitions can take place in phase with the oscillations (since it is going by so slowly). The energy is returned also in phase with the oscillations; so again, no loss occurs.

But when the period of the wave is similar to the relaxation time, then significant attenuation can occur. This is demonstrated in figure 3.1. In the second half of the relaxation time, energy is being returned during a rarefaction phase. The returned energy will cancel out the refraction, resulting in a weaker area of rarefaction. Neighbouring particles will then experience less pressure difference, and the wave will weaken as a result.

Similarly, chemical relaxation is another process where the time and energy taken for particle rearrangements can be detrimental on the propagating wave's strength. In this case, the process is related to ionisation. A key ingredient of sea water, magnesium sulfate ($MgSO_4$), is an example of a substance that ionises (or dissociates) in water. Normally, the process of dissociation (breaking up into ionic components) and recombination is at equilibrium. But compressional waves can upset this balance and cause excess recombinations. The subsequent dissociation's afterwards, and time taken for it to complete lead to further relaxation dissipation on the passing wave.

A mathematical model for attenuation due to absorption will now be developed. An expression for absorption is given by Clay [8] as simply:

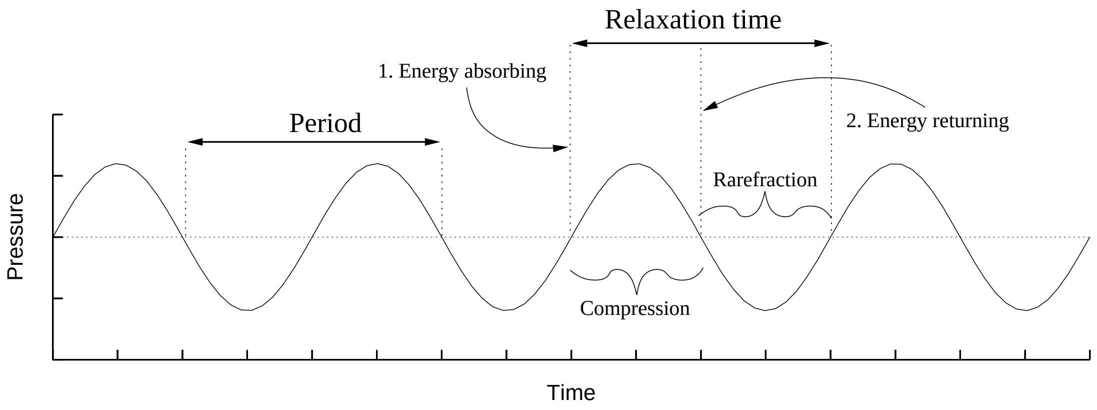

Figure 3.1: Sound wave with a period equal to medium's relaxation time

$$\text{dB loss} = \alpha x \tag{3.1}$$

Where α is known as the *attenuation coefficient* and x is the distance travelled by the sound wave in metres. The attenuation coefficient α is derived from a variety of factors. A freshwater case will be considered first, and the more complicated saltwater case will be built on from there. Taking into account both shear and bulk viscosity, Clay defines the freshwater case (α_F) as:

$$\alpha_F = \frac{1.71 \times 10^2 \; (4\mu_F/3 \; + \; \mu_{\acute{F}})f^2}{\rho_F c_F^3} \tag{3.2}$$

Where: α_F Freshwater attenuation coefficient in dB/m.

μ_F Dynamic (or absolute) coefficient of shear viscosity for freshwater ($\approx 1.2 \times 10^{-3}$ N·s/m^2 for T = 287 K)

$\mu_{\acute{F}}$ Dynamic (or absolute) coefficient of bulk viscosity for freshwater ($\approx 3.3 \times 10^{-3}$ N·s/m^2 for T = 287 K)

f Frequency of the sound wave in hertz (Hz).

ρ_F Density of medium ($\approx$ 1000 kg/m^3)

c_F Speed of sound ($\approx$ 1461 m/s for T = 287 K)

Substituting in the given typical values for freshwater at a temperature of 287 K, which is the equivalent of 14°C, the attenuation coefficient for freshwater can be simplified. This version can be used to produce an approximate, generalised attenuation model for fresh water [53]:

$$\alpha_F = 2.692 \times 10^{-13} \times f^2 \tag{3.3}$$

During World War II, it was discovered that the attenuation experienced by sonar pings were approximately 25 times more in saltwater than freshwater. It was originally thought it was due to the salt content in sea water (sodium chloride, NaCl). However, R. W. Leonard, through experimentation, isolated the cause to be magnesium sulfate ($MgSO_4$), another molecular compound found in seawater [8]. The sodium chloride caused no absorption at all!

At lower frequencies (less than 10 kHz), additional attenuation caused by relaxation occurs. This time it is due to the boric acid ($B(OH)_3$) in seawater. Taking these two additional factors into consideration, Clay presents the attenuation coefficient of seawater as:

$$\alpha_S = \underbrace{\frac{1.71\times10^2\,(4\mu_F/3+\mu_{\acute{F}})f^2}{\rho_F c_F^3}}_{\text{Freshwater}} + \underbrace{\left(\frac{SA' f_{rm} f^2}{f^2+f_{rm}^2}\right)(1-1.23\times10^{-3}P_a)}_{MgSO_4\text{ relaxation}} + \underbrace{\left(\frac{A'' f_{rb} f^2}{f^2+f_{rb}^2}\right)}_{B(OH)_3\text{ relaxation}}$$

Where:

α_S	Saltwater attenuation coefficient in dB/m.
S	Salinity in ppt, parts per thousand ($\approx$ 35 ppt)
A'	2.03×10^{-8} dB/[(Hz)(ppt)(m)]
A''	1.2×10^{-7} dB/[(Hz)(m)]
f_{rm}	Relaxation frequency (Hz) for $MgSO_4$
f_{rb}	Relaxation frequency (Hz) for $B(OH)_3$
P_a	Gauge pressure due to water column in atm ($\approx 1atm$)

Through a series of laboratory and field experiments, Schulkin and Marsh [42] were able to derive an empirical formula for the relaxation frequencies for $MgSO_4$ and $B(OH)_3$, f_{rm} and f_{rb}, respectively. They are given as:

$$f_{rm} = 21.9\times10^{9-\frac{1520}{T}}\ \text{Hz} \quad (3.4)$$

$$f_{rb} = 900\times1.5^{\frac{T-273}{18}}\ \text{Hz} \quad (3.5)$$

Where: T Temperature of water in Kelvins.

Again, assuming a seawater temperature of 287 K (14°C), a typical value for these relaxation frequencies can be derived. These are shown in table 3.1.

Molecule type	Typical relaxation frequency (Hz)
$MgSO_4$	1.107×10^5
$B(OH)_3$	1.234×10^3

Table 3.1: Typical relaxation frequencies for seawater at a temperature of 14°C

Using the above relaxation frequency values and the simplified attenuation coefficient for freshwater (equation 3.3), a final, simplified expression for the attenuation coefficient for saltwater is shown below. A plot of it onto a graph was given in the first chapter (figure 2.1).

$$\begin{aligned} \alpha_S &= 2.692 \times 10^{-13} \times f^2 + \frac{7.858 \times 10^{-2} \times f^2}{f^2 + 1.226 \times 10^10} + \frac{1.481 \times 10^{-4} \times f^2}{f^2 + 1.522 \times 10^6} \\ &= f^2 \left[2.687 \times 10^{-13} + \frac{7.858 \times 10^{-2}}{f^2 + 1.226 \times 10^10} + \frac{1.481 \times 10^{-4}}{f^2 + 1.522 \times 10^6}\right] \text{dB/m} \end{aligned} \quad (3.6)$$

Attenuation due to absorption covers one of the two main causes of propagation loss. It is highly reliant on a large number of variables, including frequency, temperature, and pressure. This makes it difficult to implement in a project of this scope. This is why further simplifications and approximations were explored. Even though absorption is only one part of the physics of underwater acoustic waves, it is an important one. Through attenuation models such as the one above (along with the spherical spreading model discussed next, in section 3.1.2), the communication ranges of acoustic signals can be gauged.

3.1.2 Spreading Loss

When a sound is produced, it will typically spread out in a spherical fashion. Figure 3.2 illustrates this process. The sound source is at the centre, and the wave emitted is the sphere. This sphere will continue to grow outwards as the sound travels further. Spreading loss is attributed to this spherical expansion and is known as *spherical spreading loss*.

In order to figure out the loss associated with spherical spreading, the expression for power is considered first:

$$P = A \times I \quad (3.7)$$

Where:	P	Power expressed in watts
	A	Area expressed in m^2
	I	Sound intensity measured in watts/m^2

The area in this case is the area of a sphere, so A becomes:

$$A = 4\pi r^2 \quad (3.8)$$

Where: r Radius of the sphere in metres

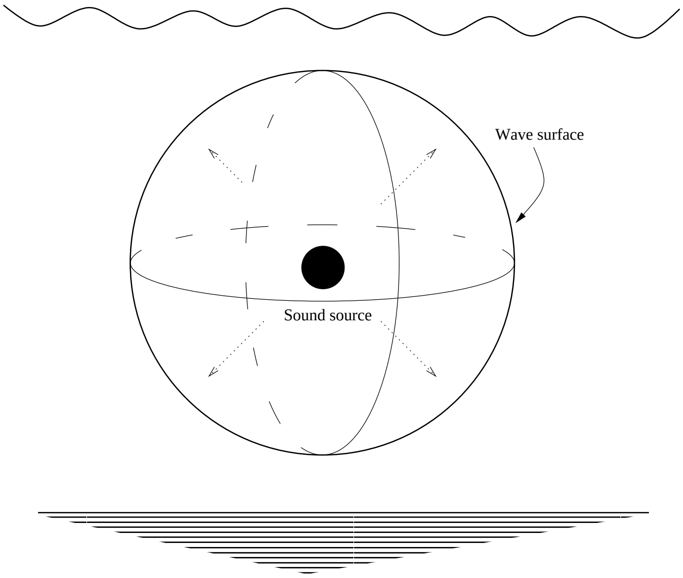

Figure 3.2: Sound wave spreading spherically from source

Substituting (3.8) into (3.7) yields the power over a sphere to be:

$$P = 4\pi r^2 \times I \tag{3.9}$$

Once the source has emitted its sound, the acoustic wave will be carrying a fixed amount of power. Ignoring energy absorbed from the medium, this power will remain mostly constant. The distance travelled by the wave is effectively the radius of the sphere. So as the wave travels further, the radius and surface area of the sphere will grow. Along with this, the intensity of the sound wave (measured as power per unit area) will decrease, as the total power is redistributed over the growing surface of the entire sphere. Representing this relationship mathematically reveals:

$$\begin{aligned} P = 4\pi r_1^2 \times I_1 &= 4\pi r_2^2 \times I_2 \\ r_1^2 \times I_1 &= r_2^2 \times I_2 \end{aligned} \tag{3.10}$$

Where:
- r_1 Initial distance from source in metres
- I_1 Initial intensity in watts/m^2
- r_2 New distance from source in metres
- I_2 New intensity in watts/m^2

It turns out the resultant intensity is a ratio of another intensity, at another distance. The intensity of the sound at a unit distance from the source is always known – it is simply the original power of the sound wave divided by 4π! So substituting $r_1 = 1$, we have an equation for spherical spreading loss (this is known as an *inverse-square* relationship):

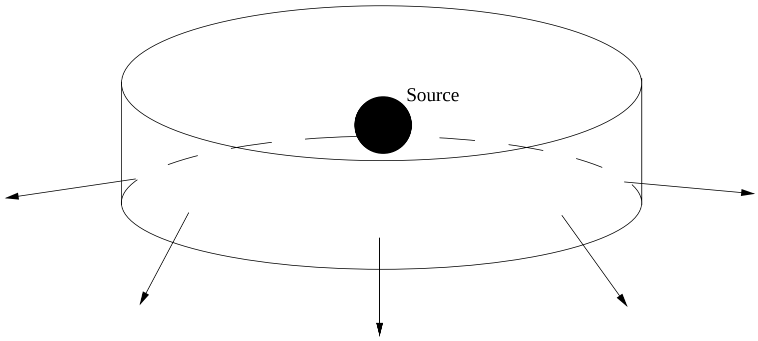

Figure 3.3: Sound wave spreading cylindrically

$$I = \frac{I_o}{r^2} \tag{3.11}$$

Where: I Intensity of sound r metres from the source, in watts/m^2
I_o Original intensity of sound in watts/m^2
r Distance from source in metres

It should be noted that for long-range, shallow-water propagation, the spreading will become cylindrical, as the waves become bounded by the sea surface and sea floor. Using a similar method to before, but taking $A = h \times 2\pi r$ instead (where h is the height or depth of the ocean), spreading loss becomes cylindrical (see figure 3.3 for a visualisation) and no longer inverse-square:

$$I = \frac{I_o}{r} \tag{3.12}$$

This section covered the second cause of transmission loss. The results from this and absorption loss discussed in section 3.1.1 can be applied together to create an adequate model for loss. Note, however, that the cylindrical loss model discussed here assumes the sea floor is flat.

3.2 Boundary Interactions

When a wave passes through a *medium boundary*, it can be both *reflected* and *transmitted*. A medium boundary is defined as the edge between two different, adjacent mediums. By reflected, we mean the wave "bounces" off the boundary. As for transmitted, we mean the wave continues to propagate through the new medium. Note that transmitted waves will undergo *refraction* – the concept that is covered in subsection 3.2.3.

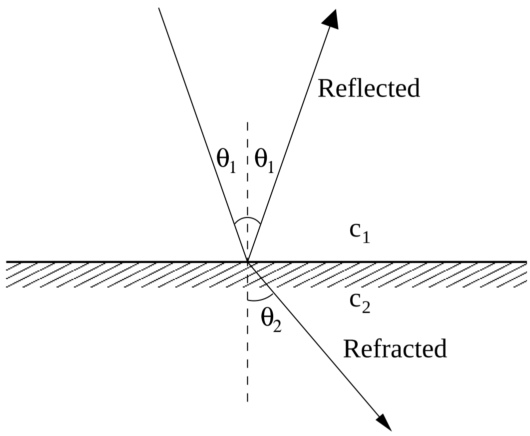

Figure 3.4: Reflected and refracted rays at a medium interface

It is simpler and more intuitive to use the concept of ray paths to describe the behaviours covered in this section. How ray paths relate to sound waves were outlined in section 2.2, and will be further detailed in section 5.3. An example of its intuitiveness is in figure 3.4, which is a representation of a sound wave's boundary interactions using ray-paths. Using waves would have required drawing a series of curves to represent each wave's wavefront. This would not look as simple as this ray-path variant.

3.2.1 Reflection and Refraction Coefficient Assumptions

The redistribution of the wave's energy among the reflected and refracted ray are determined by the boundary's *reflection coefficient* and *transmission coefficient*. These values can be used to determine how much of the incident ray's energy will be reflected and transmitted (then refracted). Unfortunately, the available mathematical expressions for it were too difficult to implement within the available time frame.

To demonstrate this difficulty, the mathematical procedure for finding the reflection coefficient will be outlined. Clay [8] and Tucker [53] present full working for evaluating the *planar* reflection coefficients (for a planar wave – not a spherical one):

$$\mathcal{R}_{12} = \frac{\rho_2 c_2 \cos\theta_1 - \rho_1 c_1 \cos\theta_2}{\rho_2 c_2 \cos\theta_1 + \rho_1 c_1 \cos\theta_2} \tag{3.13}$$

Where:
- $\mathcal{R}_{12}$ Reflection coefficient for a ray travelling from medium 1 to 2
- ρ_n Density of medium n in kg/m^3
- c_n Velocity of propagation (speed of sound) for medium n
- θ_n Angle to normal for the ray in medium n

Since these coefficients are only appropriate for planar waves, a result given by Brekhovskikh [**?**] for calculating a *spherical coefficient* approximation was needed:

$$\mathcal{R} = \mathcal{R}_{12} - \frac{iN}{kR} \tag{3.14}$$

$$N = \frac{\mathcal{R}''_{12} + \mathcal{R}'_{12} \cot \theta_1}{2} \tag{3.15}$$

Where:	R	Distance from the image source to the receiver
	$\mathcal{R}$	Spherical coefficient
	$\mathcal{R}'_{12}$, $\mathcal{R}''_{12}$	Derivatives of $\mathcal{R}_{12}$ with respect to θ_1

Clay comments that even this approximation would fail completely for angles of incidents that are close to the *critical angle*. See the section on refraction in section 3.2.3 for more information about the critical angle.

Modelling the reflection and transmission coefficients was not feasible. There were special critical angle cases to watch out for that would make developing a robust implementation too time-consuming. Instead, it was decided to assume that all medium boundaries were either completely reflective or transmissive. The reflective surfaces such as the ocean surface and seabed would have reflection coefficients of value 1, and transmission coefficients of value 0. When refraction within the water body are modelled, it will be assumed that the boundaries between layers will have transmission coefficients of value 1, and allow passing waves to transfer over completely (and then refract).

3.2.2 Reflection

Like all waves, underwater acoustic waves can be reflected off medium boundaries. Reflections are one of the main causes of reverberation in acoustic signals. Waves travelling along reflected paths will have travelled a longer distances than the direct path. Due to this difference, the reflected path's wave will be more attenuated and delayed longer. Superimposing all the waves, the resultant signal that reaches the receiver will exhibit the properties of being reverberated.

From experimentation, it is known that the *angle of incidence* is equal to the *angle of reflection* in any reflection – this is the law of reflection [23]. The angle of incidence is the angle of the arriving sound wave relative to the normal of the boundary (also known as a *medium interface*). Similarly, the angle of reflection is defined as the angle of the reflected ray, also relative to the normal. Referring to figure 3.4, these angles are marked as θ_1. This type of reflection is commonly called *specular reflection*.

If scattering was taken into account, then it would be possible there will be some reflections where their angle of reflection would not equal their angle of incidence. These types of reflections are known as *non-specular reflections*, and were not modelled in this project. However, non-specular reflections, due to their scattered nature, are usually significantly weaker than specular reflections, so their absence does not severely impede our model.

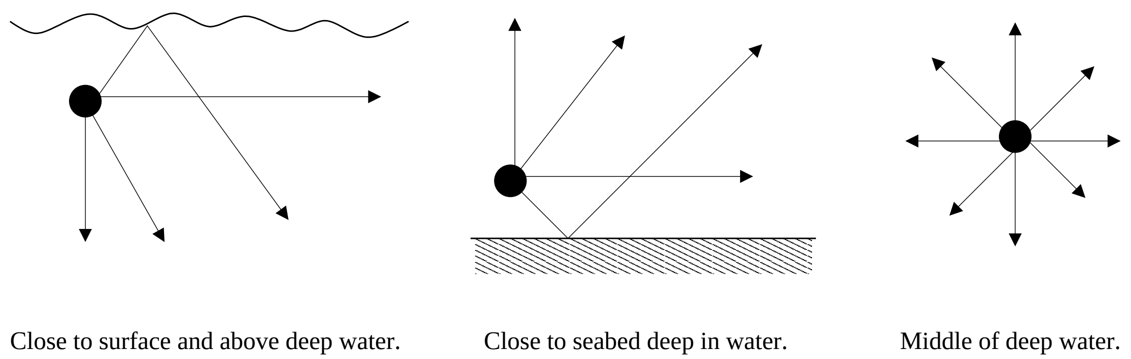

Figure 3.5: Deep-water situations without one or both major reflective bodies

When it comes to determining the main causes of reflections in underwater acoustics, there are only two main bodies to consider: the ocean surface and seabed. In deep-water, one or both of these can even be ignored (see figure 3.5). Note that both the sea surface and seabed are assumed to be perfectly flat, since we have not considered any form of scattering in our reflection model.

The ocean surface is classified as a *pressure release surface*. This means that when pressure variations, such as those of an acoustic wave's, reach it, the surface will absorb this variation. In doing so, the energy from the wave will be converted into potential energy, as surface tension increases. Eventually, the potential energy is reintroduced into the system as surface tension causes the surface particle to return to its neutral position.

Figure 3.6 outlines this whole process, frame by frame. The particle on the very left has a force applied to it that creates a pressure wave, travelling along the right. At $t = 4$, the energy from the wave is transferred into potential energy and the surface of the water stretches. At the next time sequence following that, surface tension returns the energy back into system.

Now we will examine the areas of compression and rarefaction in figure 3.6. These are denoted as 'C' for compression and 'R' for rarefaction. Just before the wave reflects off the surface ($t = 4$), the head of the wave is a compression, and once the wave is reflected the head becomes a rarefaction. In the case where a reflection is off a rigid surface (as opposed to this pressure release surface), the reflected wave would remain a compression. In effect, the pressure release surface of the ocean surface causes the reflected signal to shift phase by half a cycle.

Assuming acoustic waves are represented as sinusoids, phase shifts are jumps to another point of the sinusoidal cycle. For example, a 180° phase shift of a normal sin function will result in the wave being "flipped". The points of strongest compression become points of strongest rarefaction, and vice versa. Zero points stay the same. In the discrete case, every sample point in the wave was multiplied by -1.

Reflection off the seabed is less complicated. Since it is assumed to be a rigid body,

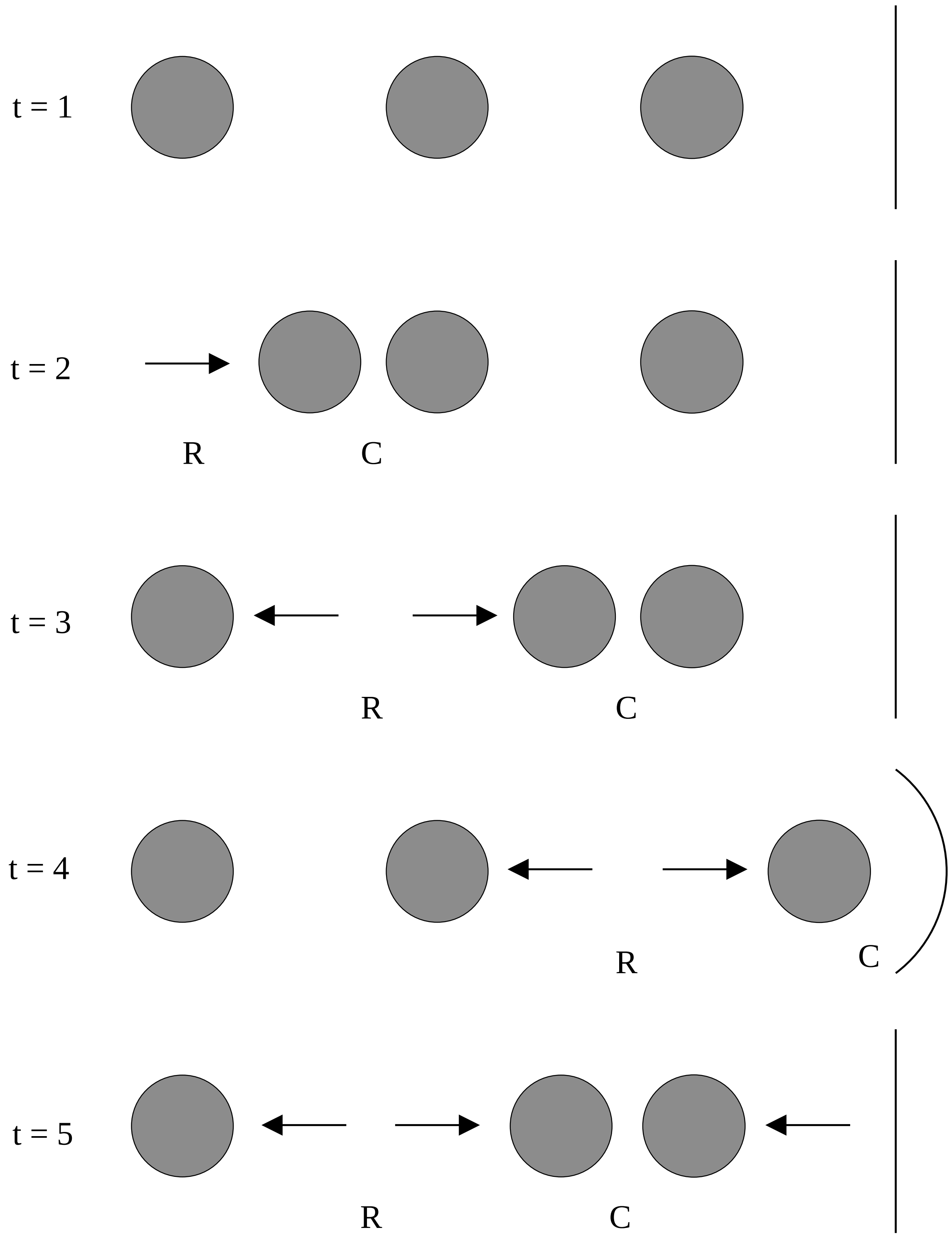

Figure 3.6: Reflection off a pressure release surface

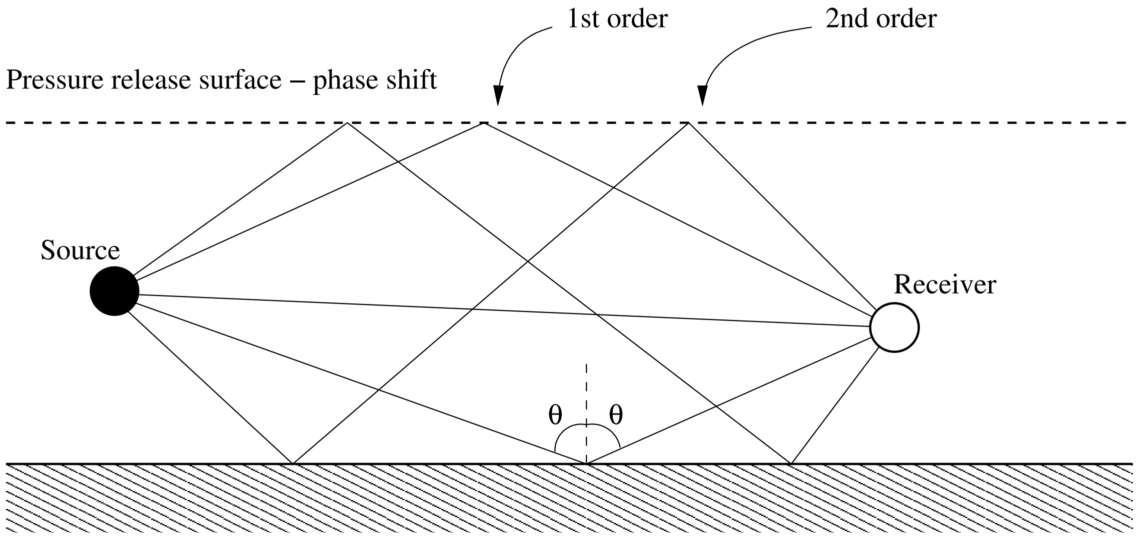

Figure 3.7: First and second-order specular reflection ray paths

reflections off it will introduce no phase shift.

The last point to consider about reflections are the points where a ray must reflect off in order to reach the receiver. Reflective paths are categorised by the count of reflections they make before reaching the receiver. This count is the *order* of the reflection. For example, the first order, the most simplest path, consists of a reflection at one point before reaching the receiver. Figure 3.7 illustrates possible first and second order reflections for a configuration of one source and receiver and different depths.

In cases where only one boundary is present, then only first-order reflections are possible. This is because once the wave has been reflected off on boundary, there is no opposite boundary on the other side to reflect off. When neither boundaries have been defined, then no reflections are possible. So for most simulation runs that wish to examine reflection, defining the top and bottom bounds are recommended.

All reflective paths, no matter what order, are made up of points of reflections along the sea surface or seabed boundaries. Once these points have been found, then it is possible to draw the ray path for such a reflection.

Solving the position of these points along a surface can be accomplished with the *image source* method. The essence of this method involved reflecting the source along the boundary to be reflected. This reflected source is known as the image source, or *virtual source*. The order of the reflective path being generated determines the number of image sources that need to be created. For first order reflections, creating one virtual source is adequate. For second order reflections, the next image source should be created as a reflection of the last-created image source, reflected along the opposite boundary. More orders can be pursued by creating more image sources; each new one reflecting the last source image created, about the opposite boundary.

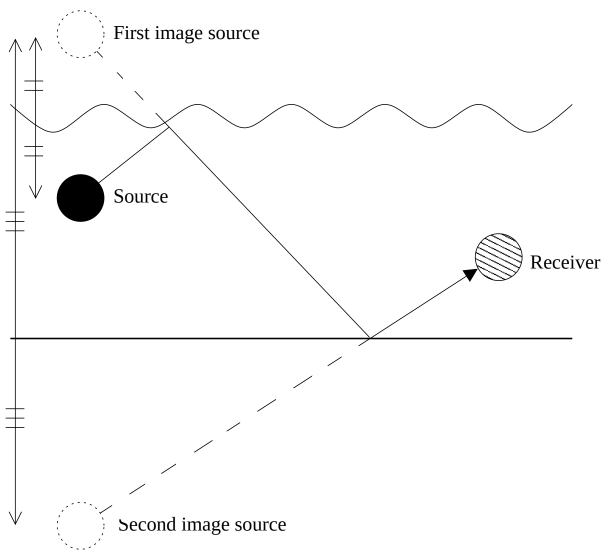

Figure 3.8: Source image method for finding n-order reflections

After sufficient virtual sources have been created, then the points of reflection along the boundaries can be determined. This starts with the last image source created. A straight line is drawn between that image source and the target receiver. The last point of reflection is where this line intersects the boundary. The next reflection point is found similarly. Another line is drawn a between the next-latest image source to the last reflection point found, and the point where that line intercepts the boundary becomes the next reflection point. This process repeats until all image sources have been traversed and the real source has reached last. At this point, the last point of the whole reflective path is the real source's position. The full reflective path is created by successively linking up ray paths from the real source, along each successive reflection point on the boundaries, and finally the receiver. Figure 3.8 is an illustration of the source image method being used to find a second order reflection.

It is also possible to find reflection paths using conventional geometry and trigonometry, and solving for points with equal incidence and reflection angles. But with that method, finding the total distance travelled will require calculating the lengths of each line that make up the path to the receiver. With image sources, the distance can be solved by finding the distance between the receiver and the last source image.

This covers our model for reflection. For high numbers of order reflections, and receivers and sources, this can processing intensive. So using the image source method to generate reverberation is not feasible. But it is more than adequate for modelling reflections of a few orders, which is adequate for most basic testing needs.

3.2.3 Refraction

The ocean is far from a homogeneous medium. The temperature, density (a function of pressure) and salinity will all affect the propagation of an underwater acoustic wave. All these factors will basically affect the medium's *velocity of propagation*. This is the measurement of how fast a sound wave within that medium would travel. Recall figure 3.4, which shows how a ray path is refracted. According to *Snell's law*, the angle of refraction is a function of the speeds of sound in the two particular mediums that share the boundary being crossed by an acoustic wave:

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{c_1}{c_2} \tag{3.16}$$

Where: c_n The speed of sound in medium n
θ_n The angle of the ray to the normal in medium n

When $c_2 > c_1$, and the angle of incidence is greater than the *critical angle*, the ray will not pass through the boundary. Instead, it will just be reflected. This is known as *total internal reflection*. The critical angle can be found by substituting $\sin\theta_2 = 1$ into (3.16) and rearranging:

$$\theta_c = \arcsin\frac{c_1}{c_2} \tag{3.17}$$

Where: θ_c The critical angle

Despite its strong influence on underwater acoustic communications, a refraction model was not feasible in the time-span given for this project, so it was dropped. To implement such a model would require finding refraction paths that would lead from source to receiver. This would involve solving multiple instances of Snell's law simultaneously. Without any immediate algorithms for solving such a complex problem, it was decided it would not possible for a project of this size.

CHAPTER 4

Existing Work

Underwater sound wave propagation is affected by countless factors. Implementing a general simulator, with sufficient fidelity to be useful to researchers, is an extremely difficult task. Instead, current implementations focus on modeling only a specific subset of the physics, catering for a certain field or use.

Despite the lack of an all-encompassing simulator, the use of simulators by researchers are increasing [29]. They are commonly used to perform verification tests. It is hoped that these will, in turn, increase the success rates of comparatively more costly field experiments.

This chapter will cover a few existing acoustic simulators.

4.1 Funkhouser

This is a real-time acoustic simulator designed to work with a virtual reality application [19]. In order to produce real-time sound with limited processing power, the simulator has a pre-processing phase. It takes advantage of the fact that all sound sources in the simulator do not move. This allows it to calculate all existing sound paths possible for each given point in virtual environment.

Instead of using a conventional ray-path-based approach to draw sound paths, Funkhouser uses a beam-based system. Each beam basically represents a block of sound paths within it, with the edges of the beam being the limits of paths possible in that beam. The disadvantage of this approach is the complex culling and geometry mathematics needed to create the correct dimensions of these beams.

Funkhouser uses a hybrid of the image source method to determine reflective paths. Instead of reflecting off all possible surfaces, this simulator only needs to find the limiting reflective paths. Once these have been found, an appropriate beam can be created, and all reflective paths within that beam will be automatically handled.

To improve performance, there is a preprocessing phase. This phase calculates all major sound paths from each source to every possible receiver location. The sources cannot move, so this reduces the amount of calculations. Hence Funkhouser has managed to create apparent real-time calculations through a heavy pre-processing phase.

This simulator is interesting in how it uses beams rather than ray paths to model sound paths. Its real-time generation is limited by the fact that sources are all stationary. For underwater communications, this is not useful, as sources can sometimes be mobile beacons or submersible, remotely-operated vehicles.

4.2 AcoustiKit

AcoustiKit was developed by Pak-King Wan to model the behaviour of sound waves within rooms. It is written in the MATLAB language, and deals with reflection and absorption. It is mostly focused on reflecting sounds off surfaces, and also the sounds generated by vibrating surfaces. It is quite clear that AcoustiKit was designed for room acoustics, and not underwater acoustics. As such, it does not concern itself much with refraction. An underwater acoustics simulator is more concerned with the time delays, signal attenuation, and refraction due to larger size of the ocean relative to a room.

4.3 SEASIM

SEASIM (Surface Escort Anti-Submarine Warfare Simulation) is an aging simulator initially made in 1980, and written in BASIC, followed by a FORTRAN implementation [2]. The BASIC version has visualisation model so that the results of a scenario can be viewed in real-time for investigating tactics. The FORTRAN variant is used for gathering statistics about the performance of anti-submarine warfare technologies. It does this by running the same scenario hundreds of times and storing records of how each individual scenario progressed. This is possible because the FORTRAN implementation is orders of magnitudes faster than the BASIC version.

The simulator includes models for cruisers, sonar systems, submarines and the torpedo-bearing craft intercept used to intercept them called LAMP (Light Airborne Multipurpose System) helicopters. In order to run the automated tests, it also has automated strategies for controlling vehicles. These artificial navigators are rudimentary at best, and many simply just head towards any enemy contacts (cruisers), while others try to avoid them (submarines).

4.4 Geng and Zielinski

Geng and Zeilinski [20] developed model based on eigenpaths, (also known as ray-paths). The interesting facet about this model is that each path is randomly distorted using a complex set of statistical probability functions. The maths behind it is not easily accessible, but it claims to be able to simulate extremely complicatedextremely complicated tasks such as reverberation.

CHAPTER 5

Simulator

The NASRay simulator models several aspects of underwater acoustic wave propagation. It is by no means a high-fidelity and accurate representation, but it is more than appropriate for quickly setting up preliminary testing environments. It currently has models for:

- transmission loss,
- time delays,
- specular reflection, and
- additive noise models.

This chapter will cover the implementation details of NASRay. It will first cover ray-based modelling, and how it is used to represent sound wave propagation. How its various models (such as transmission loss) are applied will then be outlined.

5.1 Building and Development Environment

The simulator is written with standard ISO C++, and consists of around 303 classes. Being standard C++, it should be compilable on most architectures, with minimal modifications. NASRay has been built and tested on both Linux and Windows operating systems.

There are a few external library dependencies required by the simulator. The Standard Template Library (STL) is used extensively, and is available with most modern C++ compilers, especially due to its recent addition as part of the standard C++ library. The library collection from the Boost Project [5] is another dependency. This library is used mostly for its random number generation libraries, used especially by the noise models. Finally CppUnit [11] is also utilised for the creation and execution of each of the classes' unit tests.

5.2 Assumptions and Simplifications

In order to simplify the implementation of the simulator, several assumptions are made about the environment being modelled. These are outlined below:

- The ocean is an isogradient medium. There are no layers of differing temperatures, salinity, pressures, or sound speeds. No refraction is handled in the simulator at this time.
- Objects in the simulator cannot travel faster than sound. Otherwise special supersonic physics might need to be considered.

5.3 Ray-path-Based Modelling

NASRay is a ray-based model. This means that it models sound wave propagation by drawing paths (also known as ray-paths) between each source and receiver. In a simple model with no refraction (bending of waves), reflection, or occlusion (interference from other solid bodies in the medium), this path will simply be a direct, straight line between source and receiver.

How a propagation model based on straight lines can be derived from this can be explained with Huygens's principles. Christian Huygens was a physicist-astronomer (1629–1695) who stated that each wavefront (the spherical surface) is made of up numerous smaller wavefronts. Ray path theory models the normals of the spheres (or wavelets) that follow a path to the receiver.

The ray-based approach to underwater acoustic modelling is only valid if the wavelengths of the sound waves are significantly less than the size of the enclosure it is propagating through. In the ocean, our enclosure size is virtually infinite. It is practically unbounded horizontally, and bounded by the sea floor and surface vertically. Also, communication signals need to be transmitted at relatively high frequencies in order to provide sufficient bandwidth for encoding data. Most underwater acoustic communication signals are around 10 kHz. A higher frequency leads to lower wavelengths. For 10 kHz, the wavelength is calculated to be approximately 0.15 m. These two properties ensure that the ray-based model is sufficient for modelling underwater acoustics.

The simulator models sound propagation by building ray paths between each source and receiver. See figure 5.1 for a visualisation of these paths. These paths are used to keep track of the delay times between a source emitting a sound and a receiver hearing it. The delay between a source and receiver is directly a function of distance. If either the source or receiver move, the path is updated again to reflect the new distance. These delay times are used to determine the amount of delay heard by the receiver for a given source.

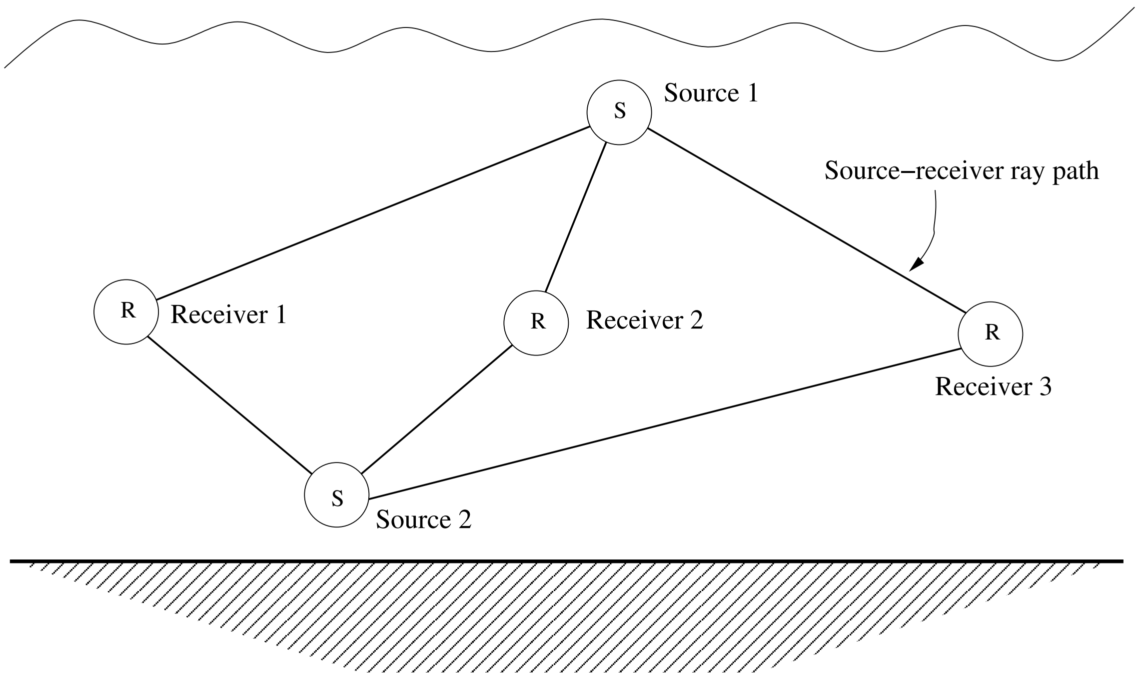

Figure 5.1: Paths for a configuration of sources and receivers

The diagram shown in figure 5.1 is an arrangement of two sources and three receivers. Receivers are entities in the simulator that listen for sounds, and stores them – typically in a WAV or RAW audio format file. Sources are the entities that emit or transmit sound through the water. The lines between the sources and receivers are paths that the simulator will create.

The behavior of two interfering sound waves is the sum of their pressure variations. This means that the individual sounds coming from each path to a certain receiver can be added together. This is how the final waveform for the resultant sound heard by the receiver is generated.

5.4 Pipe Architecture

The noise and absorption models are modelled in a pipe architecture. The undistorted sound is passed into a set of filter classes that will modify it and then pass it to the next filter. This type of architecture is advantageous in that each filter is independent of one another. It makes the software code modular and maintainable.

The noise model can use different random distributions to generate code. These random distributions include typical ones such as Gaussian and uniform random distributions. Ultimately though, all of them are additive noise models. This means they simply add a random value to each sample point.

The maximum level of noise can be specified, too. This is useful for simulating sound that must exhibit a certain *signal to noise ratio* (SNR). The SNR is the average level of the real communications signal to the average level of background noise.

Transmission loss is another filter that can be added to this pipe. This filter takes in a distance for which to apply the loss. The original sound sample is then divided by the distance squared. This is in line with the spherical spreading loss model described in section 3.1.2.

Finally, the last filter is the absorption loss model. This will multiply all sample points passed to it by the *absorption coefficient*. The value of this coefficient is specified in a configuration file. A typical coefficient value was calculated separately using the formulae in section 3.1.1, and then saved into the configuration file.

5.5 Time Delays

Time delays are handled sample by sample. For each discrete point in time, the distance of paths between receiver and source at that time are extracted. These distances are then divided by the speed of sound (specified in the configuration file) to give the time taken for a sample point to reach the receiver at that point in time.

Using these time of flight values, the source is then queried for what it transmitted that long ago. The sample point from that time is then copied into the receiver's buffer, for the current time. This way of retrieving old time values is only effective when only one of the objects are moving.

If both source and receiver are moving at the same time, then this method is not correct.

5.6 Reflection

The reflection model of NASRay consists of first-order specular reflections off the sea surface and seabed. The z coordinates for the sea top and bottom are defined in a configuration file. After checking that the top coordinate is above the bottom, then the simulator will create three paths for each source-receiver. The first is a normal DirectPath class, which handles the direct propagation path between source and receiver. The other two are instances of the ReflectionPath class, one for reflecting off each boundary. Of course, there is only one ReflectionPath per surface, so if none were defined, then there is no additional ReflectionPath's created.

The ReflectionPath class uses the *source image method* described in section 3.2.2 to calculate specular reflected path. As mentioned, phase shifts can occur for reflection points on the surface. The ReflectionPath class can automatically detect which surface it is reflected off, and if it is the top, then it will multiply all sample points by -1 before passing it to the receiver.

5.7 Block-Based Processing

Unlike many conventional signal processing algorithms, NASRay does not represent sound waves as a sinusoidal wave function. Instead, the simulator digitises all sound into sample points. The typical sampling rate of a simulation is 96,000 Hz. This sampling rate can be changed if desired. The initial implementation of the NASRay simulator distorted sound propagations one sample point at a time. For example, to process one second worth of sound, the simulator would loop through and modify one sample point at a time. This meant a there was atleast one loop being executed 96,000 times per second of simulator time.

The program was run through a performance profiler tool, oprofile [35]. Results from this revealed that close to 60% of the processing time was spent on function call overhead.

More specifically, most time was spent on locking calls. The current NASRay implementation did not require these locking calls, as it did not run any tasks concurrently. So even though these calls were being made, they did not actually perform any processing. The locking calls were passed to instances of a NullLocking class, which just accepted locking request calls, but did not carry them out. Since there was interest in multi-threading NASRay in the future, it was not desirable to remove these calls from the code.

My solution was to restructure the NASRay simulator's main processing structure to generate sample points in blocks, instead of single sample points at a time. By performing these operations in blocks, significant function call overhead was removed. Figure 5.2 is an overview of the old processing algorithm for NASRay. It is much simpler, but also highly iterative. Figure 5.3 is a reimplementation of the old implementation, except samples are now being performed in blocks as opposed to single sample points at a time.

5.8 Caching Frequently-Calculated Results

The older implementation of NASRay involved regenerating paths at each point of simulation time. This meant calculating the distance tens of thousands of times, per source-receiver path, every loop. For objects that were static, and always the same distance apart, this was a waste of processing resources.

The Path classes were created to encapsulate a ray path between a source and receiver. These were preserved during each iteration, so that values that were calculated before, such as distance, were kept. If either the source or receiver of a path move, then the path would be marked "dirty" and be recalculated.

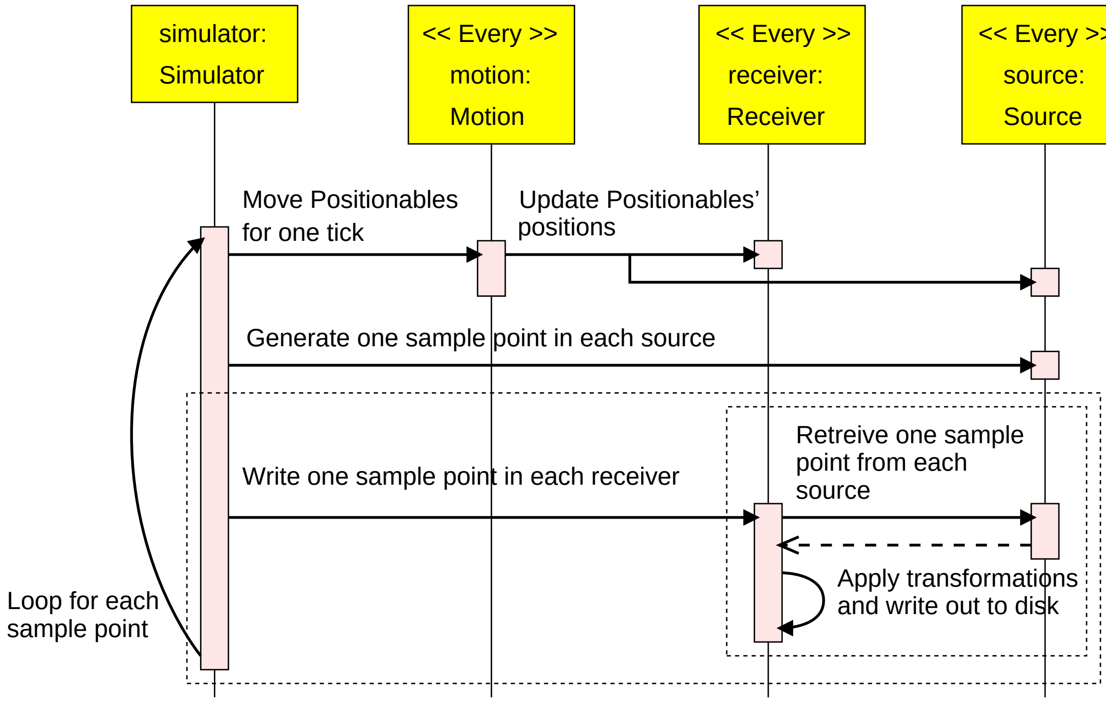

Figure 5.2: Old NASRay implementation

Most simulations in NASRay involve moving objects anyway, so this enhancement did not yield much real-world performance time improvements. However, if there is a need to perform extensive static-position objects in the future, then perhaps this enhancement would prove useful then.

5.9 Automated Buffer Size Calculation

The sounds generated by a source are stored into buffers, so that old values can be retrieved by the time delay model. There was no automated way of determining the sizes of these buffers before, and users basically had to supply their own buffer sizes and hope it was large enough.

So another enhancement made to the simulator was to calculate the largest delay possible, per block of sample points processed. This was done by finding the largest distance between source and receiver within that period of time. A buffer large enough to accommodate this distance could then created, without any interaction from the user.

Once the largest distance within a block of time was found, then an additional distance of $2 \times \text{samplingrate}speedofsound$ was added. This ensured that the buffers were still large enough if the source and receiver moved even more apart between the consecutive time blocks. This method worked because the maximum speed of an object is assumed to be

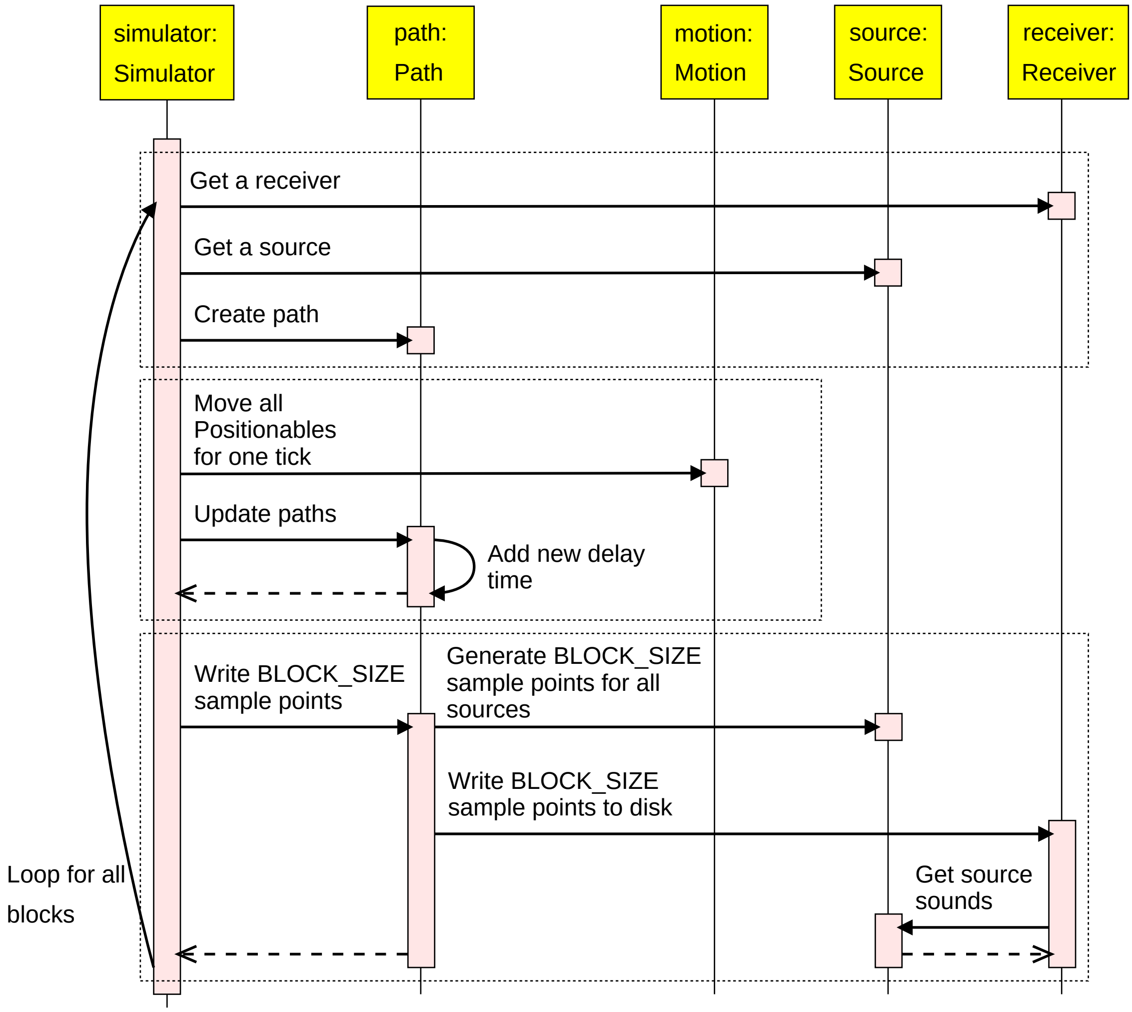

Figure 5.3: New block-based NASRay implementation

less than the speed of sound.

5.10 Grouped Movement

For simulating hydrophones on boats. Movements were re-factored out their own class called Motion. Sources and receivers (both children of Positionable) could then be attached to motions. Each time movement was made, the Motion class would move all positionables attached to it. This was how grouped movement was implemented.

Orientation and rotation support for the group of objects was also added. A mathematical construct called *quaternions* were used to smooth out rotations [45]. This is a method of representing orientations in terms of a four-component value; one real and three complex. The advantage of this quaternion form is that a special rotation function can be applied that generates the rotations necessary to move from one orientation to another. This function is called the SLERP (Spherical Linear Interpolation) function.

Basically, orientations are represented in the more-intuitive euler form, which is a three-component value: yaw, pitch, and roll. This is how orientations are typically visualised by the human mind. To get the rotations necessary to reach another orientation, the orientations are temporarily converted to quaternions, operated on by the SLERP function, and then converted back to euler-form orientations.

This algorithm provided a surprisingly simple way of calculating rotations on a group of objects, as long as the underlying mathematics was not studied too closely. Such as how quaternions represent orientations in their 4-component values, or how its rotation orientations are calculated were. By just using the given equations and functions, implementation was a simple task.

5.11 Scheduled Transmissions

Before, every source began transmission immediately at the beginning of the simulation. Sometimes this is not desirable, especially for modelling stations that transmitted only on replies to received telemetry signals. A form of scheduling transmissions was required.

A special type of source was created, called a SchedulableSource. This source contained another real source in it. When the time it was scheduled to transmit had not arrived yet, then the ScheduleableSource would emit sample points values of 0 (no sound). Only once the scheduled time had arrived did it invoke the real source it stored to retrieve transmission values. As far as the stored source was concerned, it was sending out its samples at time zero, the SchedulableSource hid away the scheduling details away from it.

Event-handling was done by adding an event checking phase into the processing loop. Each loop it would check for occurrences of any events. If an event had arrive, then it would perform the appropriate action.

It was also possible to propagate actual events in the simulator. For instance, the simulator allowed sources to emit a ping event. These ping events moved through the simulated environment like a normal sound wave (at the speed of sound, towards all receivers). When the event had propagated far enough to reach a receiver, then the receiver would be notified of the event. The abstraction of pings into an event made it simpler to implement ping replies, rather than to write a ping detector on the actual receiver, and letting it figuring it out based on the actual sounds it was hearing. That solution would have been too processing-intensive.

CHAPTER 6

Conclusion

The NASRay simulator is useful for setting up quick basic tests in a simulated environment. It does not distort the sound in a sufficient degree to truly allow successful simulation runs to mean the product is ready for real-sea environments, but it does provide a general means of just validating the initial rudimentary functionality of a signal-processing algorithm.

Time became a limiting factor in this project. It is suspected that the initial scope of the project was too large to begin with. It was not expected that the physics of underwater acoustics would be so complex, so a large amount of time had to be dedicated to research.

6.1 Further Work

Some possible directions for future work on NASRay include:

Refraction model – One of the most interesting and unique phenomena in underwater acoustic physics. Sound propagation in air is not affected by this as much as underwater is. A model of refraction would significantly add to the realism of the simulator. Temperature variations due to whether could also be implemented.

Scattering – Again, this is another major source of distortion of underwater sound waves. Backscattering, where sound waves reflect back to the source as it travels forward, is one of the problems that signal processors must take account of for sources that are also receivers. The backscatter can add a significant amount of reverberation to the final received signal. No scattering model will be complete without a model of scattering from rough surfaces such as the ocean bed and sea surface. The roughness of the sea surface would also be a function of the weather.

Statistic reverberation and fluctuations – There are some parts of underwater acoustic physics that are computationally too intensive to be performed by today's current technology. This includes a thorough reverberation model (since there can be a infinitesimal amount of reflections) and random fluctuations caused by particles in the water medium. These can all be implemented statistically using probability like Geng and Zielinski [20].

Multi-threading – There are several components in the processing parts of NASRay that would benefit from being run concurrently, especially across a parallel computing network. Since sound can be easily superimposed, and their individual components are unique to what other waves are emitting, it is easy to break down calculation into parts to be divided among different processors.

Occlusion - This will be the proper implementation of the reflection and transmission coefficients. The reflection model will need to be rewritten, as there will no longer just be a top and bottom reflective boundary.

Visualisation – As the complexity of the simulator increases, there is more need to visualise its results. Stettner and Greenberg [49] provide a good overview on methods of visualise acoustic space. A graphical user interface for the configuration of the simulator would be useful as well.

APPENDIX A

Original Research Proposal

A.1 Background

Underwater acoustic communication has only seen significant use in the past few decades. For sending signals through water, especially over long distances, acoustics are the favoured method. Areas it is used include sonar systems, underwater positioning, and surveying seabeds. It also serves as a replacement or backup for large, unwieldy cable run through the ocean, between communication nodes.

Conventional communication systems run through air using radio waves. These have little use underwater though, as radio waves are quickly absorbed by the water; severely limiting its range. Underwater sounds waves still attenuate, but no where near as much as radio waves. However, there are still other factors which make its application more challenging.

Sound travels underwater at a speed of around 1,500 m/s. This speed can vary depending on pressure, water composition, and temperature. This is slow relative to electromagnetic waves, which move around the speed of light (around 300,000,000 m/s). Acoustic signals will not reach their targets as quickly as electromagnetic waves; so there will be relatively high latency. Also, if the distance between the sound source and receiver are changing (i.e, one or both of them are moving), then the signals will undergo significant Doppler; they will stretch or compress.

The level of background noise in the sea can also interfere with the signal heard by a receiver. This noise can come from the actual source or receiver; like the motor on a boat. Even seemingly far-away objects can cause significant noise, due to the higher conductivity of sound in water. Noise is also generated by the weather (such as from storms and winds).

In addition, the paths travelled from source to receiver, by an underwater acoustic wave, is not a simple, direct, straight one. Waves can arrive from being reflected or scattered by the surface and seabed. The type of reflection or scattering produced is also dependent on a variety of factors, such as the composition of the seabed layers, and roughness of the ocean surface. Also, the differing conditions at varying depths (such as temperature and pressure), will cause waves to be refracted (path bending). Refraction can limit the range of sound waves, as waves from a source could be bent completely away from the receiver.

All these problems are what underwater communication technologies need to tackle. Since running field tests costs a significant amount of resources, it is desirable to be able to first thoroughly test underwater acoustic communication systems in a simulated environment. This will increase the probability of success at the actual field tests, and hence less need to repeat them again.

Nautronix is a commercial company specialising in water communication and positioning technology. One of their projects is an "underwater GPS" called NASNet. It works by deploying a handful of sensors in an area of water to be tracked, and then produces positioning information of various objects in the area, with the help of receiver stations on the water surface.

The system works on large and complicated software. As with all complex systems, it is crucial that it is well-tested. To aid in this task, Nautronix have another project, called NASRay, tasked with the creation of an underwater acoustic simulator. This simulator takes sound samples emitted by sources, adds the appropriate distortions, and writes out the generated output. This output can then be played-back to a receiver for testing.

A.2 Aim

As the name implies, NASRay uses a ray-based approach in its modeling. It needs improvements in many areas, which are all outlined below:

1. Optimise the algorithms used in sound generation.

   To generate and process samples, at the moment, takes a long time.

2. The sound physics model:

   - Support for a sound speed profile.
     This profile contains the speed of sound at varying depths of the water, and is useful for figuring out the amount of refraction in the path.
     This is basically a refraction model.
   - Implement a reverberation model.
   - Implement a basic occlusion model.
     Sound waves can then be reflected or absorbed by objects.

3. The configuration system:

   - Support grouped movement of points (sources and receivers).
   - Support scheduled transmissions.

4. Develop a graphical user interface.

A.3 Method

A.3.1 Overview

The general order of completion for each of the aims (described above) are as follows:

Task	*Duration (months)*
Optimisation	2
Grouped Movement	1
Scheduled Transmission	1
Reflection Model	1
Reverberation Model	1
Refraction Model	1
Occlusion Model	1
Graphical User Interface	1

A.3.2 People

The project work will be carried out by three people, Matthew Comi, Jason McSweeney, and Steven Wong.

Matthew Comi is a recent graduate of Computer Science from Edith Cowan University, and is working on this project for his Honours.

Jason McSweeney is a Computer Science graduate from the Curtin University of Technology. As the technical lead for the original project, he will be providing invaluable experience and advice.

Steven Wong is an undergraduate of Software Engineering from the University of Western Australia. This project will form the basis for his final year project.

A.3.3 Optimisation

The optimisation of the NASRay simulator's speed will be tackled first. This will provide an excellent opportunity to become familiar with the code. With a good understanding of the program, conflicts will be minimised when adding the other new features. Without a good background on the existing program, modifications will not be as clean, and likely to create even more bugs.

NASRay will be run through a profiler tool (such as gprof or oprofile). This will generate a report on the most times taken in different areas of code. The most time-consuming sections can then be examined for optimisation. Existing calculations used in the models were not originally written to be efficient, so there is much room for improvement.

The code will be broken down into threads. One possibility is to have each receiver's perceived sound generated from within its own thread. Performing calculations in a parallel and distributed manner will also be investigated.

Redundant operations will be removed. An example is to scan for ranges in time where no sound will be heard by a receiver, and forgoing unnecessary calculations during those times. This is a likely scenario as most NASNet stations only transmit range pulses about 1/30th of their time. It is then possible to approximate the window where there will be a sound. The approximation is not trivial to determine though as it must be wide enough to account for the Doppler Effect.

The main generation loop, where calculations for each sample are made, will need to be rewritten. At the moment, it generates one sample value at a time, this leads to typically 96,000 loops per second, for generating sound at a 96 kHz sampling rate. This excessive looping is causing a high amount of function call overhead. The solution is to change the current algorithm from generating one sample per loop, to generating blocks of samples per loop.

A.3.4 Grouped Movement

Grouped movements support means allowing a transformation matrix to be applied to a group of points (sources or receivers). The matrix will depend on the requested motion: rotation or translation. An additional set of intuitive configuration options will also need to be designed to allow the user to setup these grouped movements.

The point of this is to allow reproducing the motions of a rocking boat sailing along the surface of the water. That is, the points must be able to rock back and forth, oscillating up and down, and move forward, all at the same time.

This will be especially useful for the NASNet project, when it needs to simulate a beamformer. This is a structure of around eight hydrophones mounted to a pole, forming what is known as a *hydrophone array*. The receiver software takes the input from these hydrophones and beam-forms it - enhancing the amplitude of a particular source with respect to the background noise. Without the ability to group the eight receiver points on the beam-former together, simulating a beam-former is not possible.

A.3.5 Scheduled Transmission

A configuration option will be added to allow scheduling a specific sound sample to play at a given time. This is ultimately for allowing telemetry commands to be broad-casted at specific times, and having actual NASNet stations (plugged into the simulator) to respond.

A.3.6 Reflection Model

The reflection model will be extended to go further than just first-order reflections. First-order reflections are defined as sounds waves which reach their destination by bouncing off exactly one point. Second-order reflections bounce off exactly two points before reaching their destination, and so on. Only specular reflections are calculated - where the incident and reflected angles are equal. Supporting non-specular reflections are beyond the time frame of this project.

Reflections are initially assumed to occur only on the seabed and water surface. Once the occlusion model is implemented, reflection will also come from other solid bodies in the water.

A.3.7 Reverberation Model

Reverberation is a computationally impossible phenomena to replicate, instead a rough estimation must be made. Due to attenuation, reverberation is very soft. This noise-like appearance and weak strength of reverberation makes it easier to emulate.

A.3.8 Refraction Model

Support for a sound velocity profile of the water will be added. This profile basically describes the speed of sound at various depths. From these values, it will be possible to compute each sound wave's angle of refraction as it goes deeper or shallower.

A.3.9 Occlusion Model

A new model will be added to support additional surfaces in the water other than just the seabed and water surface. Initially, these surfaces will only be capable of reflecting sound waves, as this is only a basic model. Absorption will be a possible feature in future releases.

A.3.10 Graphical User Interface

Currently, the configuration system basically involves manually writing a text file containing key and value pairs. A graphical user interface will be developed to allow users to easily set configuration options and generate a configuration file for them.

A.4 Software and Hardware Requirements

All necessary software and hardware will be provided by Nautronix, at their Fremantle office. The NASRay simulator is written in C++ under the Linux operating system, but it can also capable of building under the Microsoft Windows platform. It requires a substantial amount of computing power to generate the sound samples in a reasonable amount of time. Distributed computing optimisations will hopefully lower this requirement.

Bibliography

[1] Afek, Y., Brown, G., and Merritt, M. A lazy cache algorithm. In *Proceedings of the first annual ACM symposium on Parallel algorithms and architectures* (Santa Fe, New Mexico, United States, 1989), ACM Press, pp. 209–222.

[2] Arrigan, J. M., and Shao, D. M. Antisubmarine warfare simulation on a minicomputer. In *13th Conference on Winter Simulation* (Atlanta, Georgia, 1981), IEEE Press, pp. 53–58.

[3] Audet, Jr, J. J., and Vega, G. G. A flexible sound speed retrieval technique for underwater acoustic models. In *OCEANS '75* (Sept. 1975), IEEE, pp. 1–7.

[4] Baggeroer, A. B. Acoustic telemetry - an overview. *IEEE Journal of Oceanic Engineering OE-9*, 4 (Oct. 1984), 229–235.

[5] Boost C++ Libraries Project. [WWW] Available at: http://www.boost.org/, 2004.

[6] Borsboom, P. P., and Levy, M. F. Scattering with parabolic equation methods: Application to rcs computation. *IEEE Colloquium on Common Modelling Techniques for Electromagnetic Wave and Acoustic Wave Propagation* (Mar. 1996), 5/1–5/4.

[7] Chang, C.-Y., and Chen, M.-S. A new cache replacement algorithm for the integration of web caching and prefectching. In *Proceedings of the eleventh international conference on Information and knowledge management* (McLean, Virginia, USA, 2002), ACM Press, pp. 632–634.

[8] Clay, C. S., and Medwin, H. *Acoustical Oceanography: Principles and Applications.* Wiley-Interscience, United States of America, 1977.

[9] Coates, R. Underwater acoustic communications. In *OCEANS 1993, 'Engineering in Harmony with Ocean'* (Oct. 1993), IEEE.

[10] Coates, R., and Miller, J. Cutting the rov umbilical: Inferential positioning, acoustic communication and machine intelligence. *IEEE Colloquium on Control and Guidance of Underwater Vehicles* (Dec. 1993), 8/1–8/3.

[11] CppUnit Project. [WWW] Available at: http://cppunit.sourceforge.net/, 2004.

[12] de Sousa, J. B., and Göllü, A. A simulation environment for the coordinated operation of multiple autonomous underwater vehicles. In *1997 Winter Simulation Conference* (Atlanta, Georgia, 1997), IEEE Press, pp. 1169–1175.

[13] Deville, Y. A low-cost usage-based replacement algorithm for cache memories. *SIGARCH Computer Architure News 18*, 4 (1990), 52–58.

[14] Edelmann, G. F., Hodgkiss, W. S., Kim, S., Kuperman, W. A., Song, H. C., and Akal, T. Underwater acoustic communication using time reversal. In *OCEANS 2001* (Nov. 2001), vol. 4, IEEE, pp. 2231–2235.

[15] Essebbar, A., Loubet, G., and Vial, F. Underwater acoustic channel simulations for communication. In *OCEANS '94, 'Oceans Engineering for Today's Technology and Tomorrow's Preservation'* (Sept. 1994), pp. III/495–III/500.

[16] Essebbar, A., and Vercelloni, V. Simulation of communication system for underwater acoustics. In *OCEANS '95, 'Challenges of Our Changing Global Environment'* (Oct. 1995), IEEE, pp. 1204–1207.

[17] Freitag, L., Stojanovic, M., Singh, S., and Johnson, M. Analysis of channel effects on direct-sequence and frequency-hopped spread-spectrum acoustic communication. *IEEE Journal of Oceanic Engineering 26*, 4 (Oct. 2001), 586–593.

[18] Fröhlich, T. The virtual oceanarium. *Communications of the ACM 43*, 7 (2000), 94–101.

[19] Funkhouser, T. A., Carlbom, I., Elko, G., Pingali, G., Sondhi, M., and West, J. A beam tracing approach to acoustic modeling for interactive virtual environments. In *Computer Graphics Proceedings, SIGGRAPH '98* (July 1998), ACM.

[20] Geng, X., and Zielinski, A. An eigenpath underwater acoustic communication channel model. In *OCEANS '95 'Challenges of Our Changing Global Environment'* (Oct. 1995), IEEE, pp. 1189–1196.

[21] Gomes, J., and Barroso, V. Ray-based analysis of a time-reversal mirror for underwater acoustic communication. 2981–2984.

[22] Hall, L. The origin of ultrasonic absorption in water. *Physics Review*, 75 (1949), 1415.

[23] Halliday, D., Resnick, R., and Walker, J. *Fundamentals of Physics Extended.* John Wiley & Sons, United States of America, 1997.

[24] Hazelwood, R. Controlling noise to improve underwater positioning. *IEEE Colloquium on Control and Guidance of Underwater Vehicles* (Dec. 1993), 7/1–7/3.

[25] Hoffman, J. P., Penrose, J. D., and McMahon, D. R. Acoustic propagation prediction in shallow water. In *Australian Acoustical Society Conference* (Nov. 2000), Curtin University of Technology, Defence Science and Technology Organisation, Australia.

[26] Johnson, M., Herold, D., and Catipovic, J. The design and performance of a compact underwater acoustic network node. In *Proceedings of OCEANS '94 Oceans Engineering for Today's Technology and Tomorrow's Preservation* (Sept. 1994), IEEE, pp. 467–471.

[27] Kahark, V., Bradley, D., Brink, K., Clark, C., Fox, C., Mikhalevsky, P., Munk, W., Nystuen, J., and Weir, G. *Beyond Discovery™: The Path from Research to Human Benefit.* National Academy of Sciences, Washington, D.C., Mar. 1999, ch. Sounding Out the Ocean's Secrets.

[28] Kessler, R. E., and Hill, M. D. Page placement algorithms for large real-indexed caches. *ACM Transactions on Computer Systems 10*, 4 (1992), 338–359.

[29] Kilfoyle, D. B., and Baggeroer, A. B. The state of the art in underwater acoustic telemetry. *IEEE Journal of Oceanic Engineering 25*, 1 (Jan. 2000), 4–27.

[30] Korupolu, M. R., Plaxton, C. G., and Rajaraman, R. Placement algorithms for hierarchical cooperative caching. In *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms* (Baltimore, Maryland, United States, 1999), Society for Industrial and Applied Mathematics, pp. 586–595.

[31] Lam, M. D., Rothberg, E. E., and Wolf, M. E. The cache performance and optimizations of blocked algorithms. In *Proceedings of the fourth international conference on Architectural support for programming languages and operating systems* (Santa Clara, California, United States, 1991), ACM Press, pp. 63–74.

[32] Lee, D., Schultz, M. H., and Saied, F. Supercomputers in computational ocean acoustics. In *Proceedings of the 1989 ACM/IEEE conference on Supercomputing* (1989), ACM Press, pp. 133–140.

[33] Lombardo, J. S. A demonstration of undersea warfare technologies in a high fidelity simulation on the defense simulation internet. In *Proceedings of the 25th conference on Winter simulation* (Los Angeles, California, United States, 1993), ACM Press, pp. 1030–1035.

[34] Moghaddam, P. H., and Amindavar, H. A new algorithm for multipath time delay estimation in low snr using mle method. 35–38.

[35] OProfile Project. [WWW] Available at: http://oprofile.sourceforge.net/, 2004.

[36] Paster, D. L. Experience with application of modern software management controls. In *Proceedings of the 5th International Conference on Software Engineering* (San Diego, California, United States, 1981), IEEE Press, pp. 18–26.

[37] Proakis, J. G., Sozer, E. M., Rice, J. A., and Stojanovic, M. Shallow water acoustic networks. *IEEE Communications* (Nov. 2001), 114–119.

[38] Quazi, A. H., and Konrad, W. L. Underwater acoustic communications. *IEEE Communications Magazine 20*, 2 (1982), 24–30.

[39] Robinson, T. Odin: An underwater warfare simulation environment. In *Proceedings of the 33nd Conference on Winter Simulation* (Arlington, Virginia, 2001), IEEE Computer Society, pp. 672–679.

[40] Santhanakrishnan, G., Amer, A., Chrysanthis, P. K., and Li, D. Gd-ghost: A goal-oriented self-tuning caching algorithm. In *Proceedings of the 2004 ACM symposium on Applied computing* (Nicosia, Cyprus, 2004), ACM Press, pp. 1141–1145.

[41] Savioja, L. Modeling techniques for virtual acoustics. Master's thesis, Telecommunications Software and Multimedia Laboratory, Helsinki University of Technology,, 1999. Report TML-A3.

[42] Schulkin, M., and Marsh, H. W. Absorption in seawater. *Journal of Acoustical Society of America*, 35 (1962), 864–865.

[43] Sen, S., and Chatterjee, S. Towards a theory of cache-efficient algorithms. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms* (San Francisco, California, United States, 2000), Society for Industrial and Applied Mathematics, pp. 829–838.

[44] Sharif, B. S., Neasham, J., Hinton, O. R., and Adams, A. E. Doppler compensation for underwater acoustic communications. 216–221.

[45] Shoemake, K. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer Graphics and Interactive Techniques* (1985), ACM Press, pp. 245–254.

[46] Sochacki, J. S., Mitchum, D., O'Leary, P., Ewing, R. E., and Sharpley, R. C. Distributed computation of wave propagation models using pvm. In *Proceedings of the 1993 ACM/IEEE conference on Supercomputing* (1993), ACM Press, pp. 22–31.

[47] Sozer, E. M., Stojanovic, M., and Proakis, J. G. Underwater acoustic networks. *IEEE Journal of Oceanic Engineering 25*, 1 (Jan. 2000), 72–83.

[48] Spivack, M. Integral equation methods for wave propagation over rough surfaces. *IEEE Colloquium on Common Modelling Techniques for Electromagnetic Wave and Acoustic Wave Propagation* (Mar. 1996), 9/1–9/3.

[49] Stettner, A., and Greenberg, D. P. Computer graphics visualization for acoustic simulation. In *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques* (1989), ACM Press, pp. 195–206.

[50] Stojanovic, M. Recent advances in high-speed underwater acoustic communications. *IEEE Journal of Oceanic Engineering* (Apr. 1996), 125–136.

[51] Tatro, P., and Spofford, C. Underwater acoustic models. In *OCEANS* (Sept. 1973), pp. 206–216.

[52] Tsingos, N., Funkhouser, T., Ngan, A., and Carlbom, I. Modeling acoustics in virtual environments using the uniform theory of diffraction. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (2001), ACM Press, pp. 545–552.

[53] Tucker, D. G., and Gazey, B. K. *Applied Underwater Acoustics.* Pergamon Press, London, England, 1966.

[54] Urick, R. J. *Principles of Underwater Sound.* McGraw-Hill, United States of America, 1983.

[55] Wasserblat, M., and Tabrikian, J. Underwater acoustics communications in a time-varying environment. In *21st Convention of the Electrical and Electronic Engineers in Israel* (Apr. 2000), IEEE, pp. 452–455.

[56] Zaknich, A. An integrated sensory-intelligent system for underwater acoustic signal-processing applications. *IEEE Journal of Oceanic Engineering 28*, 4 (Oct. 2003), 750–759.

[57] Ziomek, L. J., and Richard J. Blount, J. Underwater acoustic model-based signal processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing]* (Dec. 1987), 1670–1683.