NTNU – Trondheim
Norwegian University of
Science and Technology

Hydroacoustic Channel Emulator - HACE

Leo Heggem Hauge
Frank Iversen Hetland

Master of Science in Electronics
Submission date: June 2015
Supervisor: Ulf R Kristiansen, IET
Co-supervisor: Tor Arne Reinen, Sintef
Arne Lie, Sintef
Thor Husøy, Kongsberg

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Preface

The aim of the thesis is to design and build a system simulating acoustic behavior for underwater sound propagation. This master thesis was performed during the spring of 2015 and is a further development of the project thesis written in the autumn of 2014. Both theses have been conducted in cooperation with Kongsberg Maritime to satisfy their needs. The background for the project was that Kongsberg wanted a new and simple way to test their hydroacoustic solutions as a total system, with the purpose of reducing the number of sea trials required. A reduction in the number of sea trials will reduce costs and save time. During the development of our system, Kongsberg Maritime have been having weekly meetings with the purpose of influencing and shaping our solution to their needs. They have also provided the necessary utilities for testing the system.

Trondheim, 2015-06-06

Leo Hauge Frank Hetland

Acknowledgment

The authors would like to thank Kongsberg Maritime's Principal R&D Engineer Thor Husøy for suggesting this thesis topic, and for his guidance throughout the work. His motivation, fast responses and positivity have been a great help during the master thesis. The authors would like to thank SINTEF's senior researchers Tor Arne Reinen and Arne Lie for their role as supervisors and for their support and assistance in the fields of acoustics and communication. The weekly meetings have been a great help in keeping up our motivation, and inspiring us to do "that little extra." A special thanks goes to Stian Michael Kristoffersen and Magne Pettersen from Kongsberg Maritime for assistance during the sea trail, and for the testing and feedback regarding the emulator. Thanks to Tim Cato Netland from NTNU for his advice and supplying the necessary appliances. Also, thanks to Kongsberg Maritime for lending out equipment for our thesis and for covering the costs for the trip to Kongsberg Maritime in Horten to participate in the sea trail. We would also like to extend our thanks to an unknown number of persons at SINTEF and Kongsberg Maritime for their contributions towards making this master thesis realizable. Last but not the least, we thank our friends, family and classmates at NTNU for constructive questions and feedback.

Leo Hauge & Frank Hetland

Problem Formulation

The work will be to design and build a device mimicking the acoustic behaviour of the ocean for underwater navigation and communication equipment. Actual hydroacoustic equipment will be communicating with each other through the emulator. The equipment will be electrically connected to the emulator in a lab environment replacing the transducers and the water in a normal setup. This solution will greatly contribute to efficient testing of hydroacoustic systems reducing the number of sea trials.

The project work can be divided in to several subtasks and distributed between more than one student depending on interest. One subtask might be oriented towards system integration with finding hardware and doing software for the framework. Other subtasks might be oriented towards signal processing with implementation of propagation delay, transducer directional characteristics, reflection from seabed and sea surface, time- and Doppler spreading, time variability etc.

Kongsberg Maritime are to provide actual hydroacoustic equipment in form of Kongsberg Maritime Universal Transponder Boards (UTB), and supporting electronics for conversion of signal levels between the UTB cards and the audio interface.

- Kongsberg Maritime

Acronyms

APOS Acoustic Position Operator Station system software

ASIO Audio Stream Input/Output

DSP Digital Signal Processing

FIR Finite Impulse Response

GUI Graphical User Interface

HACE Hydroacoustic Channel Emulator

NSL Noise Spectral Level

SNR Signal-to-Noise Ratio

USB Universal Serial Bus

UTB Universal Transponder Board

.dsp Digital Signal Processing (toolbox in MATLAB)

Definitions

Here is a list defining words and phrases used in the text

ASIO - is a sound card protocol for digital audio.

Buffer - temporary data storage.

Crosstalk - signal from a circuit or a channel interfering with another circuit or channel.

Emulator - a system duplicating the functions of another system. For this project it means replacing the water channel and transducers with a system where the outer observations show the same characteristics as it would have for the real-world scenario.

Frame - segment of data acquisition in MATLAB in real-time.

Hydroacoustics - acoustics of underwater sound.

Jack - analog audio plug.

Jitter - random deviation of signal delay.

Latency - undesired delay from external hardware and software. It is a measure of how long it takes for an input to have an effect on the output.

Propagation delay - introduced and wanted delay. The delay comes from the time it takes for an underwater acoustic wave to propagate over a distance.

Pseudocode - a description of how a program or an algorithm works

Queue - stacking up data for the next frame while the frame is being processed.

Simulator - used to recreate or imitate properties of a real-world system. For this project it can be seen as the mathematical models applied in the emulator to recreate the underwater acoustic phenomena.

Sonar - an acoustic technique for measuring distances, communicate or locate objects underwater.

Telemetry - automated communication process for transmitting and receiving data for monitoring inaccessible/remote equipments.

Transceiver - a device able to both transmit and receive signals

Transducer - is the underwater antenna for sound waves. It converts electrical signals into pressure waves, and vice versa.

Transponder -automatically answers a known signal.

Abstract

Kongsberg Maritime wanted a channel emulator [1] for testing their hydroacoustic equipment before deploying it at sea. The benefits of such a system are that it detects problems in an earlier phase of development, thus conveniently reducing the number of expensive sea trials necessary. This master thesis describes how a channel emulator with hydroacoustic properties can be made. The emulator will replace the transducers and water with a computer simulating the acoustics, an audio interface and voltage attenuation. Figure 1 illustrates the equipment that has been replaced within the dashed squares, thus making the test arena possible to perform in an office or at a work station.

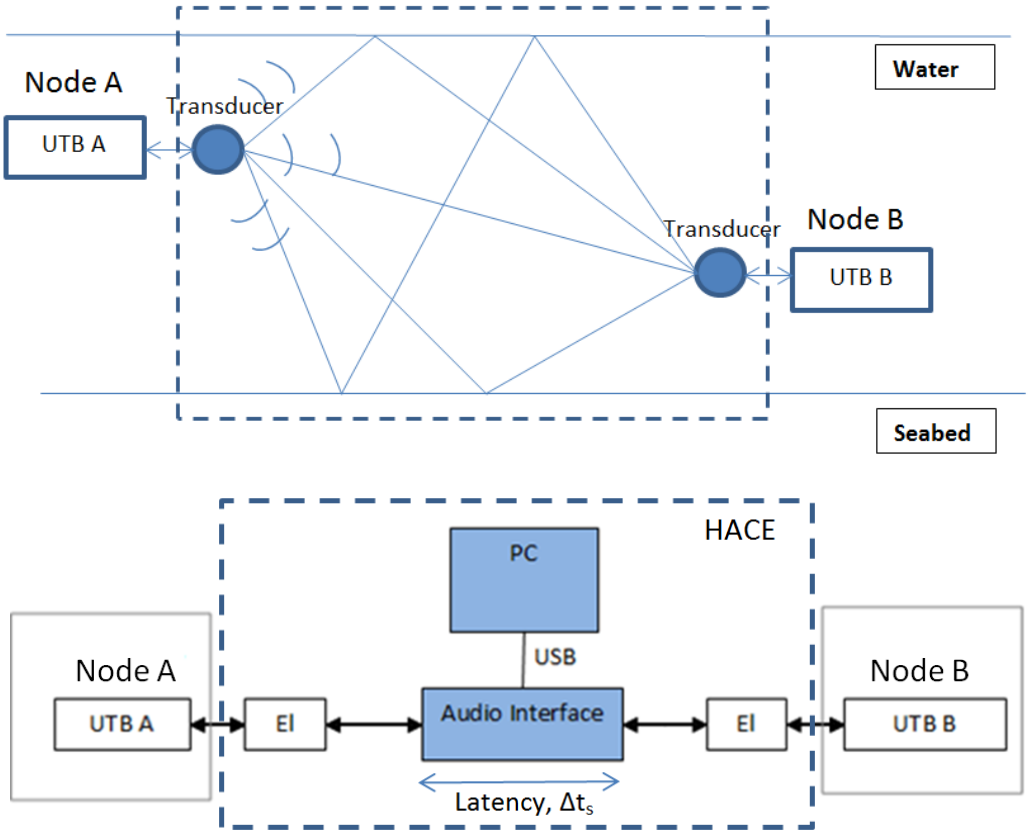


Figure 1: The dashed squares show the replacement between the real world and the hydroacoustic channel emulator, HACE. HACE consists of the computer, the audio interface and the attenuation boards (EI).

Our approach has been to develop a stable and user-friendly channel emulator with a basis in

[1] Channel emulator - a system duplicating the functions of another system, i.e. here it means to replace the water channel and transducers with a system where outer observations show the same characteristics.

acoustic wave theory. The hydroacoustic channel emulator, *HACE*, includes acoustic simulation models where the user is allowed to change acoustic parameters and place the positions of transducers for both point-to-point and network communication. This thesis has focused on advanced acoustic models such as Doppler spread, surface scatter, varying seabed and surface in 3D, sound speed profile with ray tracing, and network communication, together with the fundamental models such as, propagation loss and delay, and reflections.

In order to meet the requirements of this masters thesis, with respect to latency and jitter, a good programming platform is important. MATLAB was chosen due to the huge library of built-in-functions, especially with respect to digital signal processing. To control the system a user interface was created with the focus on simplicity, where the interface allows the user to control the system and adjust parameters.

The real-time requirement for the system was a latency with a maximum of 100 ms. Since the latency is dependent on both software and hardware, and varies from setup to setup, a calibration function was developed to ensure the best performance for each individual system. HACE has full control over the system latency and exploits it when adding the propagation delay. The minimum latency was measured as 34.2 ms, which resulted in a minimum distance between two nodes using a sound speed of 1500 m/s, being 51.3 meters. For the system to model other distances correctly, this latency must be taken into account when adding propagation delay. Ideally, zero latency would have been preferred so that all distances could be simulated.

Two tests were performed to determine the performance of the total system, one that compared the real world impulse response with the simulated impulse response, and secondly to verify the propagation delay in HACE against the measured distance from APOS. The results showed that the simulated ranges corresponded well with the ranges measured in APOS, with an offset of around 20 cm throughout all the results.

Impulse response measurements were performed at a sea trial in Horten (Breiangen) measuring at horizontal ranges from 0 to 3000 meters between two nodes. Results from the sea trial compared with those of HACE showed very good similarities between the two, with time deviations between the first and second arrival being from 0 - 3 ms (0 to 15 %), where the largest deviations were found at the shortest ranges.

HACE has already been put to use by Kongsberg Maritimes engineers, and has aided in the discovery of an error in the detection of long range propagation in Kongsberg Maritime's equipment. This problem was corrected by Kongsberg Maritime shortly afterwards, thereby demonstrating how HACE can be useful as a new testing platform.

Sammendrag

Kongsberg Maritime ønsket seg en kanalemulator[1] for å teste sitt hydroakustiske utstyr på en enkel måte, og for å se om utstyret fungerer før sjøtesting. Fordeler med et slikt system er å finne feil i en tidligere fase av utviklingen, og samtidig redusere antall kostbare sjøtester. Denne masteroppgaven presenterer hvordan en kanalemulator med hydroakustiske egenskaper kan bli laget. Emulatoren vil erstatte transdusere og havet, med en datamaskin som simulere de akustiske egenskapene til havet, lydkort for å konverte signaler fra analog til digital og en spenningsdemper. Figur 2 illustrerer med de stiplete firkantene hva som byttes ut. Som et resultat vil dette være en testbenk som er mulig å benytte inne på et kontor eller en arbeidsplass.

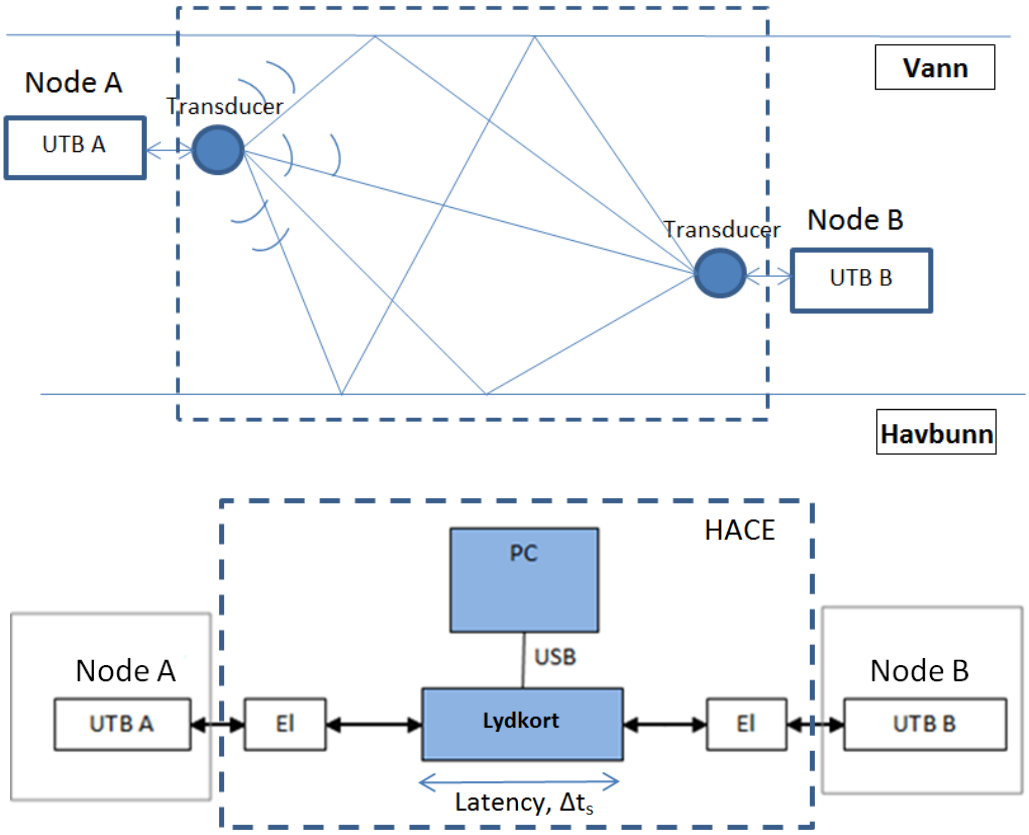


Figure 2: De stiplete linjene viser bytte av den virkelige verden med HACE. Transduserene og havet i det øverste bildet blir byttet ut med HACE. HACE består av en datamaskin, et lydkort og spenningsdempere (El)

[1]Kanalemulator - et system som dupliserer funksjonene til et annet system, det vil si at vi her bytter ut havet og transducere med et system hvor ytre observasjoner resulterer i samme karakteristikk.

Målet vårt har vært å utvikle en stabil og brukervennlig kanalemulator med tyngde i akustisk bølgeteori. Den hydroakustiske kanalemulatoren, *HACE*, inkluderer akustiske modeller hvor brukeren har mulighet til å gjøre forandringer på de akustiske parameterene og posisjonene til transduserene for både punkt-til-punkt- og nettverkskommunikasjon. I denne masteroppgaven har vi hatt fokus på å gjenskape mer avanserte akustiske modeller, som Doppler spredning, overflate spredning, varierende havbunn og havoverflate i 3D, lydfartprofil med strålegang og nettverkskommunikasjon. I tillegg til disse er også de fundamentale modellene som propageringstap, propageringsforsinkelse og refleksjoner med i HACE.

En god programmerings platform var viktig for å kunne tilfredstille de kravene som var gitt i oppgaven med tanke på latenstid og jitter. MATLAB ble valgt på grunnlag av dens store bibliotek av innebygde funksjoner, da særlig med tanke på digital signal behandling. Lydkortet "OCTA-CAPTURE" fra Roland ble valgt basert på god ytelse med hensyn på latens og dens antall kanaler. Brukergrensesnittet ble laget med stort fokus på enkelhet. Grensesnittet gjør det enkelt for brukeren å kontrollere systemet og justere parametere.

Sanntidskravet for systemet var en maksimum latenstid på 100 ms. Denne latenstiden er avhengig av både programmvare og maskinvare og vil variere fra oppsett til oppsett, det var derfor nødvendig å innføre en kalibreringsfunksjon for å finne den optimale ytelsen for hvert individuelt system. HACE har kontroll over, og utnytter, latensen når propageringsforsinkelse legges til. Den minste latens tiden ble målt til 34.2 ms, og med en lydhastighet på 1500 m/s resulterer dette i en avstand på 51.3 meter. For at HACE skal klare å simulere avstandene riktig må disse latens tidene bli tatt hensyn til, ideelt hadde null latens vært å foretrukke slik at alle distansene hadde blitt simulert.

To tester ble gjennomført for å verifisere ytelsen til HACE, en sammenligning med impulsrespons målinger gjennomført på en sjøtest i Horten (Breiangen) fra 0 til 3000 meter, og en annen for å sammenligne propageringsforsinkelsen i HACE med APOS [1]. Resultatene viste at de simulerte avstandene i HACE samsvarte veldig bra med målingene i APOS, og impulsrespons målingene i Breiangen var veldig lik de simulerte i HACE med et tidsavvik mellom første og andre mottak fra 0 - 3 ms (0 til 15 %).

HACE har allerede blitt tatt i bruk av Kongsberg Maritime sine ingeniører, og har i løpet av dens levetid funnet en feil i langdistanse propagering i Kongsberg Maritime sitt utstyr. Feilen ble raskt rettet av Kongsberg Maritime like etter, og dermed vist at HACE kan være et brukendes program som en ny test platform.

[1] APOS - Akustisk posisjonerings program utviklet av Kongsberg.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Over the years, more and more equipment dealing with underwater acoustic positioning, navigation and communication has been developed. Many simulation tools and theoretical models have been developed to account for underwatersound propagation, e.g. BELLHOP[1] and Comsol[2]. Programs like these provide a good insight into how the hydroacoustic equipment will behave theoretically but, in order to know what actually happens when the equipment is submerged, actual testing is required. Sea trials and testing are expensive and time consuming. To reduce cost and save time Kongsberg Maritime started looking for a new line of testing with the intention of reducing the number of sea trials carried out.

Kongsberg Maritime is developing and manufacturing a variety of acoustic solutions for the industry as well as for the navy. As a "first line of defence" Kongsberg Maritime wanted a device that could simulate the oceanic acoustic behavior of underwater communication in real-time, hence the thesis's title *Hydroacoustic Channel Emulator, HACE*. The word "hydroacoustics" originates from the Greek language, where "hydro" means "water" and "acoustics" means "to listen".

Underwater wireless communication has many similarities with terrestrial radio communication, but the underwater channel is more complex and has a variety of limitations due to the physics of sound propagation in water. While electromagnetic waves are highly attenuated by the ocean, acoustical waves can propagate over long distances underwater, and this is the reason why acoustical waves are being used. Mathematical models can be used to describe the different phenomena of underwater acoustics, including reflection loss, multipath, absorption and spread loss.

[1] BELLHOP is ray tracing program, see `http://oalib.hlsresearch.com/Rays/`

[2] Comsol is a multiphysics modelling software, see `http://www.comsol.no/`

Nesting these phenomena together and creating a real-time communication channel, with a basis in the physics of underwater sound propagation, is the aim of this thesis.

Because of the sound speed and frequency ranges used in underwater communication, it is possible to perform real-time simulations without any specialized equipment. The sound speed results in blunt requirements with respect to latency, which can be compared to radio communication, where the speed of light dwarfs the sound speed, and latencies in the order of milliseconds are way too high. Also, the ranges used for radio communications are often larger than those used underwater. Both the latency and ranges for radio waves require computations and processing powers beyond the limits of an average contemporary computer.

This master thesis is a continuation of the previous project performed during the autumn of 2014. The project gave the authors insight and expertise concerning the subject, resulting in the creation of a solid foundation on which to build and develop the system. Since the end of the project, the complexity of the simulations in the emulator have increased from having the most basic acoustical models for point-to-point, to a more robust and adaptive system dealing with network communication with varying sound speed profiles among many other models. Some of the most drastic changes are: adaptation and increased performance in real-time, network communication, Doppler spread, surface scatter, 3D models for seabed and surface and ray tracing.

Literature Survey

Jens M. Hovem, in his book *Marine Acoustics* [7], has described most of the physical phenomena an acoustic wave experiences in a marine environment. It is, in his own words, "a book with an unusually complete text covering the wide gamut of sonar engineering." The book covers theory about underwater sound propagation, reflections, transducers as well as sonar systems, and has been the building block from the acoustic perspective of the thesis. It gives a thorough and detailed description, with mathematical expressions, of underwater acoustics.

There has been a lot of work done studying underwater sound, and many different simulation tools have been developed. Paul C. Etter, in his book, Underwater Acoustic Modeling and Simulation [5], explains how the physics of sound can be implemented and calculated with the use of a computer. The aspiring factor in developing the simulations has been to improve sonar systems by achieving a wider understanding of sound propagation unfolding underwater.

Real-time digital signal processing has been described in a variety of literature. Proakis and Manolakis [16] give a detailed description of many different signal processing techniques. Their work focuses on digital signal processing, and that as long as signals can be sampled, the same

structures and mathematics are valid for any signals. For real-time signal processing, MathWorks [13] has created an easy way of accessing audio in real-time. *Real-Time Digital Signal Processing* by Kuo and Lee [10] is a thorough introduction into real-time signal processing, where examples in both MATLAB and C++ are given. They describe how the analogue-to-digital conversions works with sampling and quantization, and how to process signals in real-time.

1.2 Approach

Having a stable fundament is a key attribute in building any system. In this thesis, the fundament revolves around having a robust and reliable real-time structure. From a robust structure, it becomes easier for the system to evolve and support the required and wanted features. Key points in creating a stable fundament for the system are the choice of sound card and programming language.

Programming Language

The authors' previous knowledge and experience were used as a guideline for choosing the programming language, and this is the reason for the simulations in the hydroacoustic channel emulator being purely written using MATLAB. MATLAB is a high-level language with a wide range of built-in functions [14]. Compared to programming languages like C++, MATLAB is easy to use, with a low threshold for implementing algorithms. It makes the user focus on the algorithm itself rather than on the syntax and how to implement it. Apart from the ease of implementation, the digital signal processing toolbox (.dsp toolbox) in MATLAB makes reading audio and accessing sampled audio signals an easy task.

Sound Card

An audio interface is required for connecting the analog signal to a computer. By choosing an external sound card more options are available, such as adjusting the sampling frequency and buffer size, and it allows an easy connection to analog systems. The sound card used in this thesis was OCTA-CAPTURE from Roland. This sound card was used during the project in the autumn of 2014 and proved to be a good sound card, having several inputs and outputs, and a small latency for AD/DA conversion. Since this sound card was used during development, the system has been tailored for that interface. It is, however, possible to choose any sound card, but the recommendation is to use OCTA-CAPTURE.

Accomplished tasks

The task of creating a flawless system mimicking all aspects of underwater acoustics is a vast venture, expanding above both the expertise of the authors and the time span of the masters thesis. Thus, a prioritization list was drawn up, with the intention being to implement as much as possible during the time period (see Appendix A.3 for the list). The completed tasks in this masters thesis are evolutions of the earlier work with the project thesis, where important improvements concerning implementation, stability, code structure and much more have been made. The goal of the project thesis was to get HACE "up and going," focusing on a simple ray tracing method with constant sound speed, plane mirror reflections, a flat sea bed and surface with no scatter. In rebuilding the entire system, this master thesis explores a whole new level, with many more advanced and improved acoustic features. There has been a heavy focus on the implementation of the network communication and the variations in 3D, with advanced ray tracing methods from BELLHOP which have resulted in a much more sophisticated channel emulator. HACE is developed in MATLAB and converted to an .EXE file.

The completed tasks in this masters thesis are:

- Sea trial in Horten (Breiangen) measuring channel impulse responses from 0 to 3000 meter.
- Comparing the measured real-world results with HACE
- Discovered error for long range propagation in the Kongsberg Maritime's equipment.
- Developed a real-time hydroacoustic channel emulator, HACE, with :
  - Improved graphical user interface, with adjustable parameters
  - Possibility of adding transducer properties
  - Network communication in 3D
  - Improved point-to-point communication
  - Recording option
  - Signal analysis option
  - Latency calibration/system adaptation
  - Varying seabed in 3D
  - Varying sea surface based on sea state in 3D
  - Varying sound speed
  - The possibility of deciding which acoustic features contributes in the simulation
  - Gain calibration
  - Acoustic properties to the system:
    - Reflection loss

    * Geometric spread
    * Propagation delay
    * Ray tracing
    * Transducer properties
    * Frequency dependent ambient noise
    * Frequency dependent absorption
    * Doppler spread
    * Scatter

Where the system requirements are:

- Latency no larger than 100 ms with small jitter, less than 3 ms
- Perform range test using APOS [1] and set telemetry [2].
- Possible to use on an average contemporary computer.

1.3 Limitations

By focusing on creating a working emulator within the time span of this master's thesis, some simplifications and limitations have been made for the system. With respect to the acoustic part, the simplifications are mainly to limit the number of mathematical models implemented. The models used are described in the theory chapter. Other acoustical simplifications and limitations are:

- Minimum range between nodes
- 2D axisymmetric geometric rays
- Plane wave reflection from seabed
- Non-movable objects
- Identical transducer data for all transducers in each setup.

There are also a few limitations due to the sound card, computer and software. As will be mentioned in Section 2.9, latency is always present, and is dependent on an unknown number of factors which results in a minimum distance between the nodes. The latency is dependent on the processing power of the user's computer. It is recommended to use the 2014 MATLAB version or later, and to have a reasonable amount of memory ($\sim$ 8 GB) and a processing capability of $\sim$ 2 GHz. The number of sound card channels limits the number of nodes, e.g. OCTA-CAPTURE has 8 channels, i.e. possible to simulate a network of 8 nodes.

[1] APOS - Acoustic positioning software developed by Kongsberg.
[2] Telemetry - Protocol for wireless communication to fetch information and to read status

1.4 Report Structure

This report is divided in four parts: theory, implementation, results, and discussion and conclusion. Chapter 2 gives the fundamental theory behind understanding how HACE works. It supplies some of the main theory for describing underwater sound propagation mathematically, transducer properties, and how real time processing works. Chapter 3 covers the implementation of the theory described. A block scheme of the system can be seen, how to connect the equipment, and explain the algorithm behind the program. The 4th chapter presents the results, while chapter 5 contains discussion of the findings, and makes a conclusion with recommendations for further work.

Chapter 2

Theoretical Background

Today, many underwater applications depend on acoustics, e.g. echo sounding [7, p. 6]. Echo sounding is a form of sonar[1], often used to measure water depth and to locate fish. It works by transmitting an acoustic wave, and then analyzing the reflected sound. Simply by measuring the time for the reflected wave to return, the sea depth can be found. How the waves behave plays an important role in the accuracy of the echo sounding. Taking this acoustic technique a little further, underwater wireless communication can be achieved.

Terrestrial wireless communication is performed using electromagnetic waves, such as radio waves. Underwater communication, on the other hand, is achieved with sound waves. The main reason for using sound waves is that electromagnetic waves are greatly attenuated underwater. According to Butler [3] electromagnetic waves at 10 kHz have an attenuation of approximately 5.4 dB per meter in sea water, compared to an attenuation of 0.0012 dB per meter with acoustic waves with the same frequency showed by Hovem [7]. For long range propagation it is fair to say that use of electromagnetic is not applicable as a result of the huge absorption loss. As will be discussed more, the signal frequency is a very fragile and important aspect in the underwater environment, not only for underwater electromagnetic waves.

This chapter will focus on the theory concerning underwater acoustic wave propagation, and how to implement propagating waves in real-time simulations for underwater communication. First, the basic physics of underwater wave propagation will be explained, followed by acoustical models in the underwater channel. And, lastly describe the concepts of frame based real-time signal processing for both network and point-to-point communication.

[1] Sonar - an acoustic technique for measuring distances, communicate or locate objects underwater

2.1 Underwater acoustics

Compared to electromagnetic waves that can travel through vacuum, acoustic waves require some sort of medium through which it can propagate. The acoustic wave is initiated by a mechanical vibration, where the vibration causes particles to move. This causes pressure variations to occur inside the medium which results in a sound wave. The direction of propagation is the same as the direction of the vibrations, i.e. acoustic pressure waves are longitudinal waves.

Studying how the vibrations and the density changes are related to the flow of mass in both time and spatial position, the *wave equation* can be derived. The wave equation is a way to mathematically describe the motion and behavior of the acoustic waves. There are several ways of representing the wave equation, some use the velocity potential, others the acoustic pressure or displacement. In order to derive the equation there must be, according to Hovem [7, p. 14], *conservation of mass, momentum balance* and an *equation of state*. The homogeneous wave equation for velocity potential ϕ can be expressed as

$$\nabla^2\phi - \frac{1}{c_0^2}\frac{\partial^2\phi}{\partial t^2} = 0 \tag{2.1}$$

where c_0 is the nominal sound speed, t the time and ∇ is the differential operator. Or by introducing a source term in the wave equation, resulting in what is known as *the inhomogeneous wave equation* expressed as:

$$\nabla^2\phi - \frac{1}{c_0^2}\frac{\partial^2\phi}{\partial t^2} = -q(t)\delta(\mathbf{r}) \tag{2.2}$$

where $\delta(\mathbf{r})$ is dirac pulse[1] creating a point source with strength $q(t)$ and $\mathbf{r}$ is the source position. In spherical coordinates expressed as

$$\delta(\mathbf{r}) = \frac{1}{4\pi r^2}\delta(r) \tag{2.3}$$

Taking the Fourier transform of Equation 2.2 results in the inhomogeneous *Helmholtz equation*:

$$\left[\nabla^2 + \kappa^2(\mathbf{r})\right]\Phi(\mathbf{r},\omega) = -Q(\omega)\delta(\mathbf{r}-\mathbf{r_0}) \tag{2.4}$$

where

$$\kappa(\mathbf{r}) = \frac{\omega}{c_0(\mathbf{r})} \tag{2.5}$$

is the wave number. Now, if the sound speed is assumed to be complex, absorption can be added

[1] Dirac pulse - will be introduced in section 2.8

into the formula. Making the substitution in Equation 2.5 yields

$$\hat{\kappa} = \frac{\omega}{c_0(1 - i(c_i/c_0))} \approx \frac{\omega}{c_0} + i\omega\frac{c_i}{c_0^2} \tag{2.6}$$

The approximation in Equation 2.6 can be done when $c_i << c_0$, where c_i is the imaginary part of the complex sound speed. Thus, the second part in the approximation is defined as the absorption $\alpha = \omega\frac{c_i}{c_0^2}$, and expressed in a plane wave as

$$\Phi(x,\omega) = \Phi_0 \exp(i\hat{\kappa}x - \alpha x) \tag{2.7}$$

Acoustic absorption will be discussed further in Section 2.3, where we will see that the absorption is frequency dependent.

Solving the Helmholtz equation for sound pressure in spherical coordinates results in

$$p(r,t) = i\omega\rho_0\frac{Q(\omega)}{4\pi r}\exp\left[i(\kappa r - \omega t)\right] \tag{2.8}$$

where it can be seen that the pressure is inversely proportional to the range r, and by this introduces a new term called *spread loss*.

2.2 Spread loss

Spread loss, or geometric spread, is a measure of how much the signal intensity is diminished as a function of distance, and is the most significant form of transmission loss for long range underwater wave propagation. The sound intensity, I, is a function of sound pressure from Equation 2.8 with sound speed, c, and density, ρ, defined as

$$I(r) = \frac{p^2(r)}{\rho c} \tag{2.9}$$

where it is seen that $I \propto p^2$. Now, consider a point source transmitting an acoustic signal. At distance $r_0 = 1m$ the signal has an intensity of I_0. Further away, at distance r_d the signal intensity is I_d. In order to keep the energy conservation, the total power P at distance r_0 must be equal to the total power at distance r_d, that is

$$P = A(r_0)I_0 = A(r_d)I_d \tag{2.10}$$

where $A(r)$ is the total area of a sphere at distance r. And, if geometric spread is the only form of loss, we can define transmission loss TL in dB as

$$TL = 10\log\left(\frac{I_0}{I(r)}\right) = 20\log\left(\frac{r}{r_0}\right) \tag{2.11}$$

where $r_0 = 1m$ and $I \propto r^2$, with a spread factor of 2. Figure 2.1 show spherical spread as a function of distance in water.

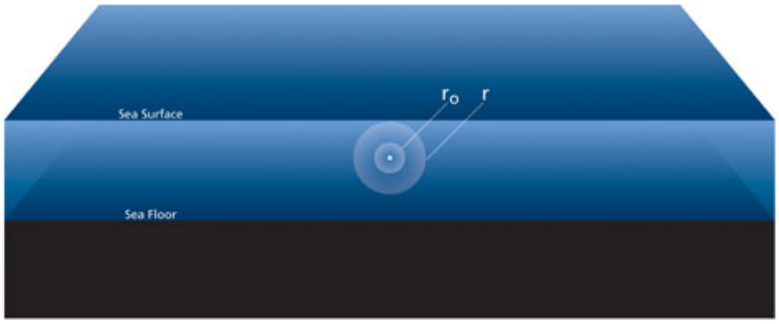


Figure 2.1: Spherical spread underwater, courtesy of DOSITS [4]

When the distance r is much larger than the ocean depth the spread loss can be considered cylindrical, see Figure 2.2, where from solving the Helmholtz equation for cylindrical coordinates results in $I \propto r$, or a spread factor of 1. The transmission loss in dB is then

$$TL = 10\log\left(\frac{r}{r_0}\right) \tag{2.12}$$

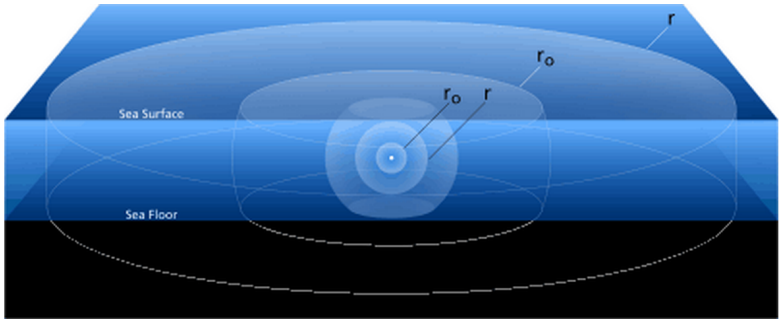


Figure 2.2: Cylindrical spread underwater, courtesy of DOSITS [4]

It can be said that both models for geometric spread are only approximations for the actual spread loss in the real world environment. The underwater world is not ideal as these two models describe, cylindrical and spherical spread. There are acoustic phenomena such as bending rays

(described Section 2.6) that can guide more sound intensity in one direction more than in other directions. Therefore, it can be advantageous to use a spread loss factor in between the cylindrical and the spherical model. Acoustics-Lab [1] refer to an approximated spread factor of 1.5, while Porter [15] uses a spread factor of 1.7 in BELLHOP.

2.3 Acoustic absorption

Acoustic absorption is a sound wave's energy loss when propagating through a medium. Most of the lost energy is transformed into heat. Different type of media have different absorption; for example, fresh water has a lower absorption than sea water. The reason for the difference between sea- and fresh water is that, sea water contains boric acid and magnesium sulfate that contributes to the absorption.

When a wave propagates in a medium there will be a change in pressure and volume, according to Section 2.1. These changes cannot happen without a delay in time. This time delay is called the *relaxation time* τ, and its occurrence is described by the *relaxation frequency* f_i.

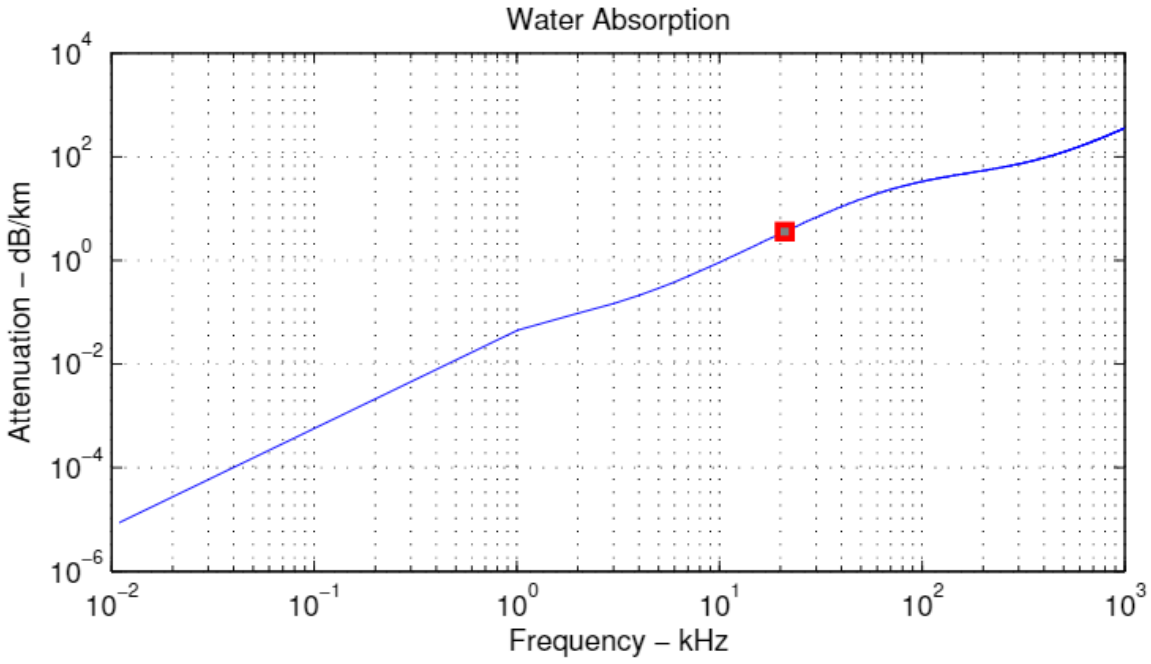


Figure 2.3: Acoustic absorption in sea. Loss in dB per km. Based on Eq: 2.17 with pH = 7.8, Depth= 40 m, Temperature=10 °C, sound speed, c=1492 m/s and Salinity = 35 ppt.

Figure 2.3 is plotted from equations developed by Francois and Garrison [6]. To develop a single acoustic absorption equation for seawater, three contributions are combined: boric acid, magnesium and fresh-water. These three contributions are based on both theoretical and experimental calculations, and is well proven by engineers and researchers in the field of underwater acoustic[7]. The equations for the three contributions boric acid, magnesium, and fresh-water are shown in Equation 2.13, 2.14 and 2.15.

Boric acid contribution

$$\begin{aligned} A_1 &= \frac{8.86}{c} \times 10^{(0.78pH-5)} \left[\frac{dB}{km\ kHz}\right] \\ P_1 &= 1 \\ f_1 &= 2.8(S/35)^{0.5} \times 10^{(4-\frac{1245}{\theta})} [kHz] \end{aligned} \tag{2.13}$$

Where f_1 is the relaxation frequency, $\theta = T + 273K$ is the temperature expressed in Kelvin and c is the sound speed in the water based on the formula:

$$c = 1412 + 3.21T + 1.19S + 0.167D \tag{2.14}$$

T is the temperature in Celsius, S is the salinity in the ocean ppt (parts per thousand), and D the depth in meter.

Magnesium sulfate contribution

$$\begin{aligned} A_2 &= 21.44\frac{S}{c}(1 + 0.025T) \left[\frac{dB}{km\ kHz}\right] \\ P_2 &= 1 - 1.37D \times 10^{-4} + 6.2D^2 \times 10^{-9} \\ f_2 &= \frac{8.17 \times 10^{8-\frac{1990}{\theta}}}{1 + 0.0018(S-35)} [kHz] \end{aligned} \tag{2.15}$$

Fresh-water viscous contribution

$$\begin{aligned} &For\ T \leq 20°C \\ &\quad A_3 = 4.937x10^{-4} - 2.59 \times 10^{-5}T + 9.11 \times 10^{-7}T^2 - 1.5 \times 10^{-8}T^3 \left[\frac{dB}{km\ kHz}\right] \\ &For\ T > 20°C \\ &\quad A_3 = 3.964x10^{-4} - 1.146 \times 10^{-5}T + 1.45 \times 10^{-7}T^2 - 6.5x10^{-10}T^3 \left[\frac{dB}{km\ kHz}\right] \end{aligned} \tag{2.16}$$

Combining these equations, Equation 2.13, 2.15 and 2.16, results in the absorption loss α expresses in dB per km as:

$$\alpha = \frac{A_1 P_1 f_1 f^2}{f^2 f_1^2} + \frac{A_2 P_2 f_2 f^2}{f^2 + f_2^2} + A_3 P_3 f^2 \tag{2.17}$$

The equation is divided into three parts and is the three contributions, as mentioned earlier, boric acid, magnesium sulfate and fresh-water, where f_1 is the relaxation frequency of the boric acid and f_2 is the relaxation frequency of the magnesium sulfate, and the frequency f is the signal frequency.

The absorption loss is dependent on both distance and frequency. From Figure 2.3 it is easy to see that the absorption is increasing with the frequency.

2.4 Reflection

Reflection loss is loss of acoustic energy when a wave is reflected at the boundary between two media. Dependent on the media parameters, some of the incident wave is transmitted through the boundary, and is called transmission. The *grazing angle*[1] θ_1, in Figure 2.4 plays an important role in the amount of transmitted and reflected energy. For small grazing angles, all of the incident wave is reflected, while for larger angles the wave is also transmitted through the boundary. From this we define a *critical angle* θ_{crit}, where the critical angle is the angle where the transition from total reflection to transmission occurs, described in Hovem [7, p. 65-72].

At a boundary between two media there must be, according to Hovem [7, p. 67], identical vertical particle velocity and pressure on both sides, and the horizontal wave number must be the same in both media. These conditions are expressed in Equation 2.18, where $\kappa_i = \omega / c_i$ is the wave number in the different media. From trigonometry, we see, in Equation 2.18, that k is the horizontal component of the wave number, called the horizontal wave number. Figure 2.5 shows the decomposition of the wave number into the horizontal and vertical contributions.

$$\kappa_1 \cos\theta_1 = \kappa_2 \cos\theta_2 = k \tag{2.18}$$

Since we have the initial condition that the horizontal wave number is constant for the two media, we identify Snell's law as

[1] Grazing angle: Angle with reference to the horizontal direction, rather than the surface normal

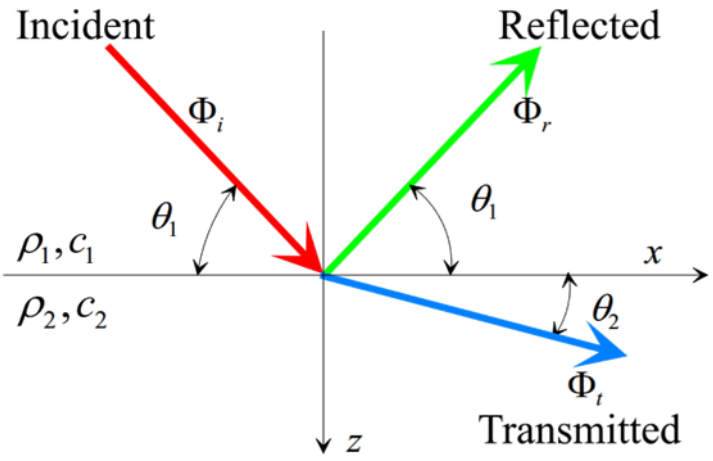


Figure 2.4: Reflected and transmitted waves at an interface between two fluid media: an incident plane wave Φ_i with the angle θ_1 produces a reflected wave Φ_r with the same angle and a transmitted wave Φ_t with the angle θ_2. Obtained from Hovem [7, p.66]

$$\xi = \frac{cos\theta_1}{c_1} = \frac{\cos\theta_2}{c_2} \tag{2.19}$$

Where ξ is the ray parameter, c_1 and θ_1, respectively, is the sound speed and grazing angle in medium 1 and, c_2 and θ_2 in medium 2. From this, it can be said that there must exist an angle θ_1 that gives $\theta_2 = 0$ when $c_1 < c_2$. The value of θ_1 is then called the critical angle θ_{crit} and is given by Equation 2.20 as

$$cos\theta_{crit} = \frac{c_1}{c_2} \tag{2.20}$$

The vertical wave number γ, on the other hand, is changing from medium to medium, dependent on the sound speed in the media. From the wave number κ and the horizontal wave number k, we can use Pythagoras's theorem to calculate the vertical wave number γ in two media as

$$\begin{aligned} \gamma_1 &= \frac{\omega}{c_1}\sin\theta_1 = \sqrt{\frac{\omega^2}{c_1^2} - k^2} \\ \gamma_2 &= \frac{\omega}{c_2}\sin\theta_2 = \sqrt{\frac{\omega^2}{c_2^2} - k^2} \end{aligned} \tag{2.21}$$

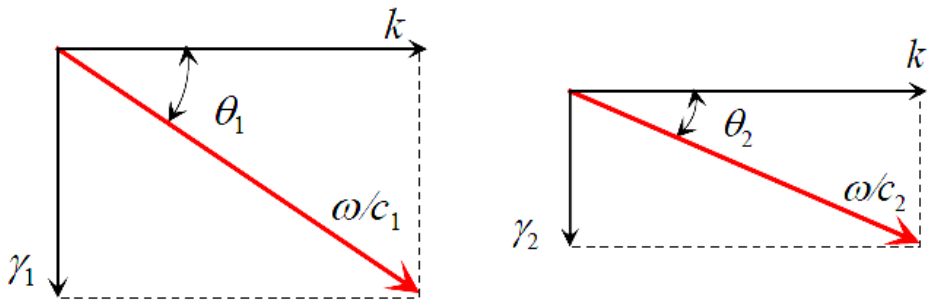


Figure 2.5: The wave number components for the first medium, one the left, and for the second medium, one the right. The horizontal wave number component k is the same in both media, while the vertical wave number components $\gamma 1$ and $\gamma 2$ are different. Obtained from Hovem [7, p.68]

The incident wave in medium 1 can be described complex as

$$\begin{aligned} \phi_i(x,z,t)_i &= \Phi_i exp((i\kappa_1 x\cos\theta_1 + i\kappa_1 z\sin\theta_1)exp(-i\omega t) \\ \Rightarrow \Phi_i(x,z,t)_i &= \Phi_i exp(ikx + i\gamma_1 z)exp(-i\omega t) \end{aligned} \tag{2.22}$$

where it can be seen that the horizontal wave number $k = \kappa_1 \cos\theta_1$ and vertical wave number $\gamma_1 = \kappa_1 sin\theta_1$.

The reflected wave in medium 1 is described as

$$\begin{aligned} \phi_i(x,z,t)_r &= \Phi_r exp(i\kappa_1 x\cos\theta_1 - i\kappa_1 z\sin\theta_1 exp(-i\omega t) \\ \Rightarrow \Phi_i(x,z,t)_r &= \Phi_r exp(ikx - i\gamma_1 z)exp(-i\omega t) \end{aligned} \tag{2.23}$$

and the transmitted wave in medium 2 described as

$$\begin{aligned} \phi_i(x,z,t)_t &= \Phi_t exp(i\kappa_1 x\cos\theta_2 + i\kappa_2 z\sin\theta_2 exp(-i\omega t) \\ \Rightarrow \Phi_i(x,z,t)_t &= \Phi_t exp(ikx + i\gamma_2 z)exp(-i\omega t) \end{aligned} \tag{2.24}$$

It is the vertical wave number that determines the loss of energy at a boundary. If the vertical wave number had been the same at both sides, all the energy would be transmitted. When the incident wave is less than θ_{crit}, there will be as mentioned total reflection, but also an imaginary transmission often called a refracted wave. This phenomena will not be covered in this thesis, see Hovem [7, p. 72-77] for more information.

Boundary conditions require equal pressure in both media at the interaction, $z = 0$. With the sound pressure p is defined as $p = -\rho\partial\phi/\partial t$ and the particle velocity given by $v = \nabla\phi$. Applying these two conditions to Equation 2.23 and 2.24 with time and z equal to zero, results in

$$\begin{aligned} p_1(x,0,0) &= p_2(x,0,0) \\ \Rightarrow i\omega\rho_1(\Phi_i + \Phi_r) &= i\omega\rho_2\Phi_t \end{aligned} \tag{2.25}$$

$$\begin{aligned} v_z,1(x,0,0) &= v_z,2(x,0,0) \\ \Rightarrow \gamma_1(\Phi_i - \Phi_r) &= \gamma_2\Phi_t \end{aligned}$$

Equation 2.25, solved with respect to Φ_r and Φ_t, gives the reflection and transmission coefficients:

Reflection coefficient

$$\begin{aligned} R_\phi &= \frac{\Phi_r}{\Phi_i} = \frac{\rho_2\gamma_1 - \rho_1\gamma_2}{\rho_2\gamma_1 + \rho_1\gamma_2} \\ \Rightarrow R_\phi &= \frac{\rho_2 c_2 \sin(\theta_1) - \rho_1 c_1 \sin(\theta_2)}{\rho_2 c_2 \sin(\theta_1) + \rho_1 c_1 \sin(\theta_2)} \end{aligned} \tag{2.26}$$

Transmission coefficient

$$\begin{aligned} T_\phi &= \frac{\Phi_t}{\Phi_i} = \frac{2\rho_1\gamma_1}{\rho_2\gamma_1 + \rho_1\gamma_2} \\ \Rightarrow T_\phi &= \frac{2\rho_1 c_2 \sin(\theta_1)}{\rho_2 c_2 \sin(\theta_1) + \rho_1 c_1 \sin(\theta_1)} \end{aligned} \tag{2.27}$$

By inserting the sound pressure expressed $p = i\omega\rho\Phi$ in the equation, the coefficients can be expressed as

$$\begin{aligned} R_p &= R_\phi \\ T_p &= \frac{\rho_2}{\rho_1} T_\phi \end{aligned} \tag{2.28}$$

And using the vertical particle velocity is defined as $v_z = \partial\Phi/\partial z$ the coefficients expressed with particle velocity are

$$\begin{aligned} R_v &= -R_\phi \\ T_v &= \frac{\gamma_2}{\gamma_1} T_\phi \end{aligned} \tag{2.29}$$

Alternatively, the reflection and transmission coefficient can be calculated with acoustic impedance as seen in Equation 2.30 and 2.31. Acoustic impedance is the impedance the wave is subjected to based on the density, sound speed and angle of the wave.

$$\begin{aligned} Z_1 &= \frac{\rho_1 c_1}{\sin(\theta_1)} \\ Z_2 &= \frac{\rho_2 c_2}{\sin(\theta_2)} \end{aligned} \tag{2.30}$$

$$\begin{aligned} R_p &= \frac{Z_2 - Z_1}{Z_2 + Z_1} \\ T_p &= \frac{2Z_2}{Z_2 + Z_1} \end{aligned} \tag{2.31}$$

The intensity of a plane wave is expressed as:

$$\frac{I_r}{I_i} = -Re\left\{R_p R_v^*\right\} = \left|\frac{\rho_2\gamma_1 - \rho_1\gamma_2}{\rho_2\gamma_1 + \rho_1\gamma_2}\right|^2 \tag{2.32}$$

$$\frac{I_t}{I_i} = -Re\left\{T_p T_v^*\right\} = \left\{\frac{4\rho_2\gamma_1\rho_1\gamma_2}{\left|\rho_2\gamma_1 + \rho_1\gamma_2\right|^2}\right\} \tag{2.33}$$

$$I_i = I_r + I_t \tag{2.34}$$

The reflected intensity can be expressed as $I_r = |R_\phi|^2$. In Figure 2.6 an example of intensity reflection and transmission is shown. As one can see the reflected intensity is 1 for total reflection up to critical angle, for a grazing angle around 28° degrees. For larger grazing angles the wave transmits into the next layer and the transmission intensity increase.

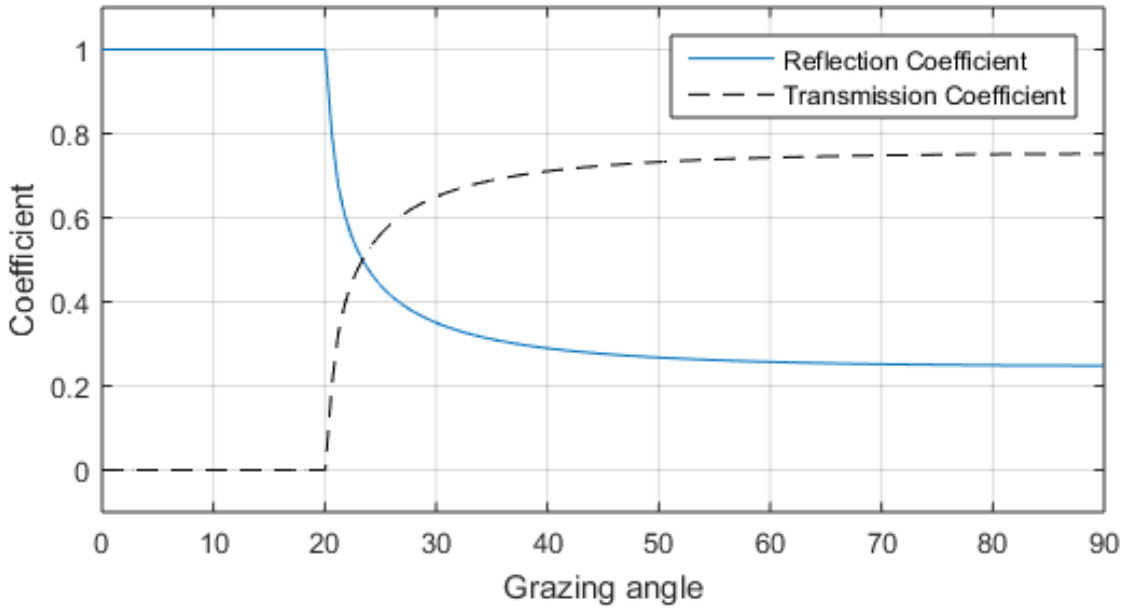


Figure 2.6: Reflection and transmitted coefficients for a case where the sound speed in medium 2 is higher than that in medium 1, where: $\rho_1 = 1000 kg/m$, $c_1 = 1500 m/s$, $\rho_2 = 2800 kg/m$,$c_2 = 1600 m/s$

2.4.1 Scatter

When a sound wave is reflected off an uneven, or corrugated, surface the wave will be spread out in both space and time. This spreading phenomenon is called *scatter*. We can separate into two types of scatter, coherent and diffuse (or incoherent), where coherent scatter is mostly spread in the specular direction, see Figure 2.7. The diffuse scatter in the figure shows that some of the signal is scattered back in the direction of the source, and is called back scattering, while the scatter in the forward direction is called forward scattering.

In this thesis the seabed is considered to be locally flat. Of course in reality totally flat surfaces does not exist, but based on the findings from Applied-Physics-Laboratory [2, p. IV-18] telling of the insignificance with bottom scatter, the bottom scatter has been neglected. A more compelling situation is the sea surface scatter.

The signal intensity of a surface reflection can be separated into coherent and incoherent scatter intensity as

$$I_{tot}(t) = I_{incoh}(t) + I_{coh}(t) - \alpha_{bubbles} \tag{2.35}$$

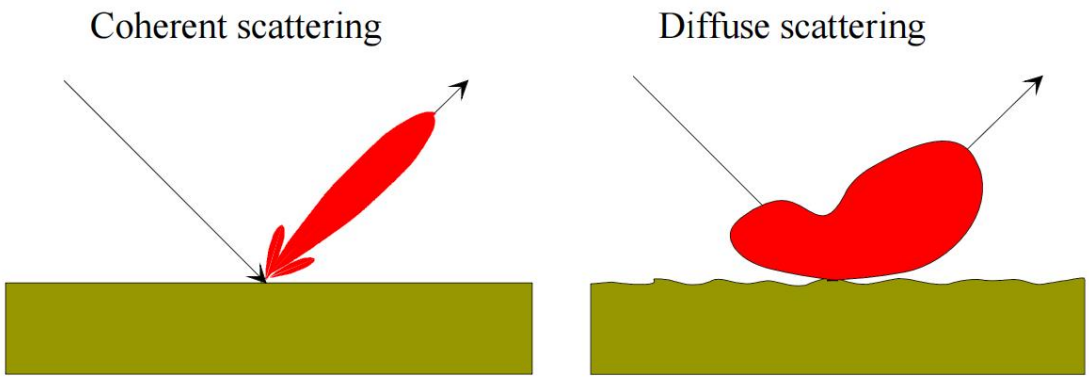


Figure 2.7: Coherent and diffuse scatter from a surface. Courtesy of Hovem [7, p. 78]

where $\alpha_{bubbles}$ is the absorption from bubbles at the surface expressed in dB as

$$\begin{aligned} \alpha_{dB}(U) &= \frac{1.26\times10^{-3}}{\sin\theta}U^{1.57}f^{0.85}, \quad U \geq 6m/s \\ \alpha_{dB}(U) &= \alpha_{dB}(U=6m/s)\exp(1.2(U-6)), \quad U < 6m/s \end{aligned} \tag{2.36}$$

where U is the wind speed, f is the signal frequency in kHz and θ is the specular grazing angle.

For the sea surface, the roughness is considered to be large compared to the signal wavelength, which is valid for most mid to high frequencies according to Applied-Physics-Laboratory [2, p. II-29]. From this, the coherent scatter can be neglected and Equation 2.35 is simplified to

$$I_{tot}(t) \approx I_{incoh}(t) - \alpha_{bubbles} \tag{2.37}$$

When a wave is scattered, it will be spread out in time, called elongation, which is similar to reverberation time in a room Kinsler et al. [8, chap. 12]. From Applied-Physics-Laboratory [2, p. II-29], we have the characteristic elongation time L for $I_{incoh}(t)$ with one reflection as

$$L = \frac{2r_1r_2}{r_1+r_2}\frac{s^2}{c}(1-e^{-\theta/\gamma_0}) \tag{2.38}$$

where

- r_1 and r_2 is the incident and scattered slant ranges
- c is the sound speed

- s^2 is the mean square slope given in Equation 2.39
- $\gamma_0 = \tan^{-1}(s)$

and

$$\begin{aligned} s^2 &= 0.0046 \log_e(2.1U^2), \quad U \geq 1 m/s \\ s^2 &= 0.0034, \quad U < 1 m/s \end{aligned} \tag{2.39}$$

By applying Equation 2.38, the scattered intensity can be approximated as

$$I_{incoh}(t) = \frac{I_0 A_0}{(r_1 + r_2)^2} \begin{cases} 0 & \tau \leq, 0 \\ \Phi(\sqrt{\tau / L}) & ,0 \leq \tau \leq \tau_0 \\ \Phi(\sqrt{\tau / L}) - \Phi(\sqrt{(\tau - \tau_0)/L}) & ,\tau \geq \tau_0 \end{cases} \tag{2.40}$$

where

- $\tau = t - t_s$
- t_s is the time for the specular reflection to propagate a distance with reference at time t = 0
- τ_0 is the transmit pulse length
- I_0 is the intensity of the transmitted pulse
- A_0 is the unit area
- Φ is the Gaussian error function

Figure 2.8 show the results from using Equation 2.40 together with Equation 2.36 for three different wind speeds. The plot in Figure 2.8 has been normalized to total power of the direct path signal. The signal pulse length is 4 ms as the length of the direct path shows. After the surface reflection, the signal is scattered and spread out in time.

2.4.2 Doppler spread

The ocean is under constant changes, where, perhaps, the most influential for underwater sound propagation is the movement of surface waves. These changing conditions at sea give rise to another element when dealing with scatter, namely *Doppler spread* (or time spread). During a time period the channel is assumed to be constant (channels will be explained more in Section 2.8), is called *coherence time L*. The inverse of the coherence time is called *coherence bandwidth or Doppler bandwidth* D_s. The Doppler bandwidth tells how rapid the channel changes, and is given as

$$D_s = \frac{1}{L} \tag{2.41}$$

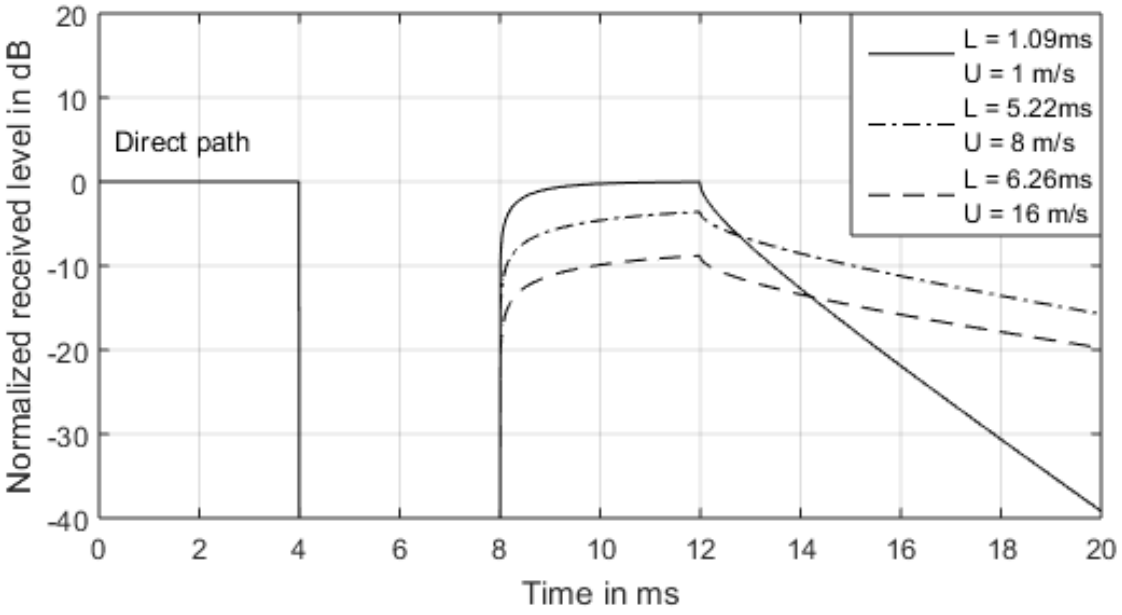


Figure 2.8: Normalized scattered intensity with different wind speeds. Signal frequency is 21 kHz, the grazing angle is 10° and the source and receiver are located at 55 meters depth, 1000 meters apart.

In some literature the coherence bandwidth is increased by a factor of 4 to illustrate a more rapid varying channel, e.g. Tse and Viswanath [19, p. 31]. So, lets say the Doppler bandwidth is 100 Hz, new channel representations must be drawn 100 times per second. And, since the ocean waves are not moving at infinite speed, there is a relation between adjacent coherence times. This relationship is assumed to have a Pierson-Moskowitz wave spectrum [2, p. II-32], thus a 3 dB spectral width in hertz of the forward scatter is given by

$$\Delta f_{3dB}(Hz) = 0.128 U f_c \sin\theta \tag{2.42}$$

where f_c is given in kHz, U is the wind speed in m/s and θ is the grazing angle.

Figure 2.9 shows how a channel is changing in time according to the model. Inserting in Equation 2.42 a frequency $f_c = 21$ kHz, a wind speed $U = 2$ m/s and a grazing angle $\theta = 33°$, the 3 dB spectral width is found to be approximately 3 Hz.

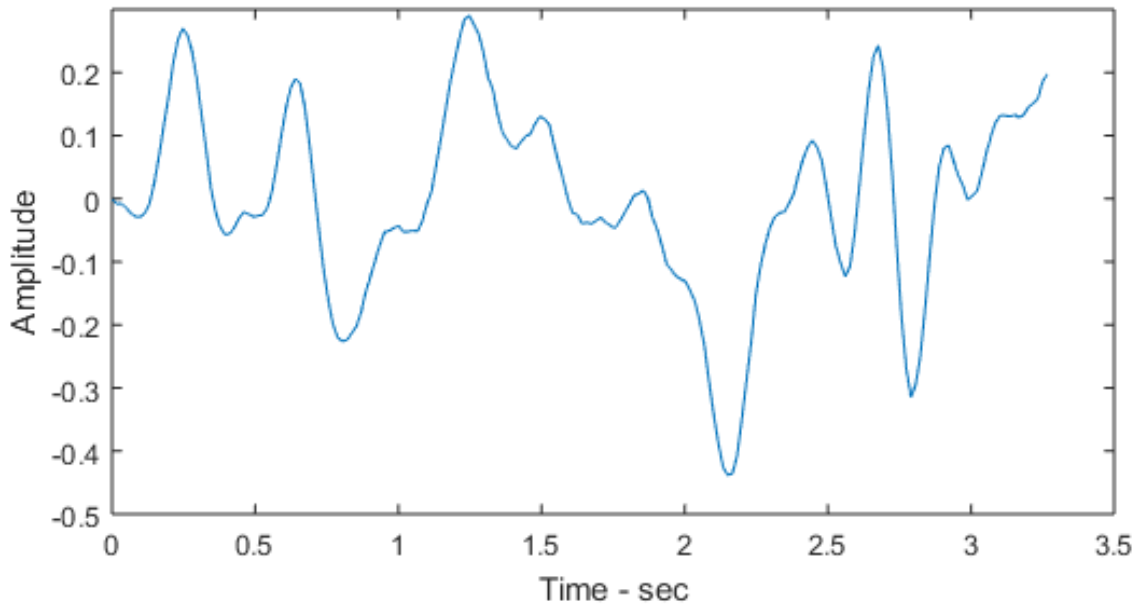


Figure 2.9: Doppler spread with 3 Hz spectral width.

2.5 Directivity and Sensitivity

Directivity and sensitivity are two concepts used for describing the receiving and transmitting property of transducers, described thoroughly in Hovem [7, ch. 9]. Directivity factor D is a measure of radiating intensity $I(\theta = 0)$ in the maximum radiating direction versus intensity of an omnidirectional source with intensity I_{ref}.Often the directivity factor is expressed in a logarithmic scale named the directivity index DI. Mathematically, the directivity factor and index can be described as

$$\begin{aligned} D &= \frac{I(\theta = 0)}{I_{ref}} \\ DI &= 10 \log D \end{aligned} \tag{2.43}$$

Figure 2.10 shows a plot of the directivity index of a sinc as a function of angle, where DI is the logarithmic representation of D. At 0° it has a maximum value, this means that most of the power is radiating straight ahead. Around ±80° the peaks of the two largest side lobes appear. In some cases it is advantageous to have a high directivity in one direction, for instance when the receiver position is known or for long range transmission when most radiating power is in the main

direction, and as a result, large side lobes are unwanted. High directivity increases the SNR[1]. Often in underwater acoustics, the directivity pattern is approximately omnidirectional, by this meaning that the intensity is equally distributed over all angles. The reason being, either the receiver or transceiver does not know the location of the other, or there is a network of receivers the transceiver wants to communicate with.

Sensitivity is divided in two, where one is the receive sensitivity M_v and the other being the transmit sensitivity S_v. Both sensitivity and directivity are physical properties of the transducer. Let's first look at the transmit sensitivity. Many transducers are built up of piezoelectric crystals [7, p. 184]. These crystals react with an electric response when suppressed to pressure, and vice versa. In other words, they behave like converters, converting mechanical energy to electrical energy. Transmit sensitivity is how the sound pressure from the source relates to input voltage, e.g. if S_v = 100 dB/V re 1 μPa with 1 V input voltage, the sound pressure level transmitted is 100 × 1 = 100 dB re 1 μPa. Receiver sensitivity is similar to the transmit sensitivity, but now sound pressure is transformed to an electric signal. Using a 100 dB re 1μPa sound pressure level, a receiver with sensitivity M_v = -100 V/dB re 1 μPa will cause an electric response of 1 V.

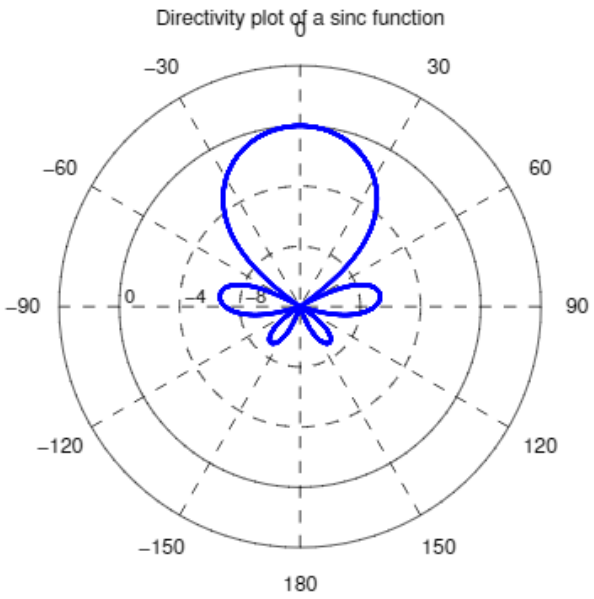


Figure 2.10: Example of a directivity index produced by a sinc function

Both directivity and sensitivity are functions of frequency, where Appendix F.3 shows the frequency dependency of a transducer. Low frequency produces more omnidirectional pattern and the sensitivities are reduced, while higher frequencies will cause more directionality and thus also increase the sensitivity. A plot of the directivity of the transducer used is found in Chapter 4 in Figure 4.5b.

[1]SNR - Signal-to-Noise Ratio is the relationship between noise and signal power

2.6 Ray tracing

As the name indicates, *ray tracing* is a way to track the rays' paths from one location to another. In underwater acoustics, the technique is used to locate the pathways of propagating sound waves. For the ray tracing model to be valid it is assumed that sound waves move along lines (or rays) perpendicular to the sound waves' wavefronts, as shown in Figure 2.11. The surface normal to the wavefront $W(x,y,z)$ is called *eikonal e* originating from the eikonal equation [7, p. 113], where $s(x,y,z)$ is the path generated.

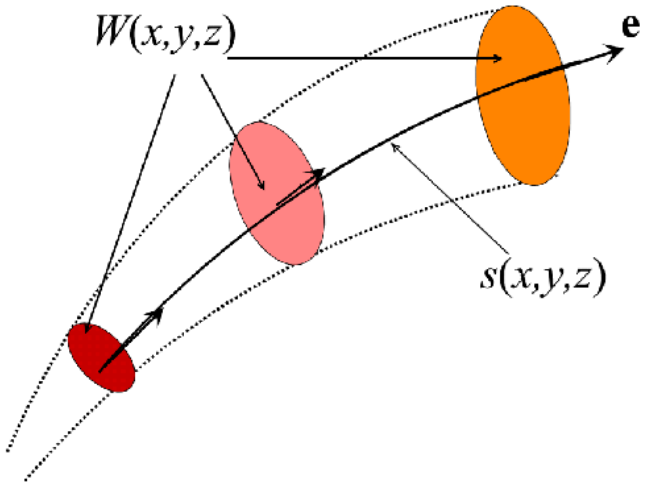


Figure 2.11: Sound wave propagating along a ray $s(x,y,z)$ with wave fronts $W(x,y,z)$. The wave front's surface normal is the eikonal e, from Hovem [7, p. 114]

The principle of ray tracing is fairly simple. In two dimensions consider a starting position (x,y). The next position will then be $(x+\Delta x, y+\Delta y)$, where Δx and Δy is the distance of movement in x- and y-direction, respectively. By always updating your current position a path will be revealed.

From Equation 2.14 the sound speed is varying with depth, temperature and salinity, resulting in a sound speed profile $c(z)$ given as a function of the depth z. A way to use the ray tracing technique is then to discretize the water column into layers with depth Δz, thus, making ray tracing depth, or sound speed dependent. The distance from layer i to $i+1$ is then calculated by

$$r_{i+1} - r_i = \frac{1}{\xi g_i}\left[\sqrt{1-\xi^2 c^2(z_i)} - \sqrt{1-\xi^2 c^2(z_{i+1})}\right] \tag{2.44}$$

where g_i is the sound speed gradient in the layer, and ξ is the ray parameter from Snell's law 2.19.

If $\xi^2 c^2(z_{i+1}) \geq 1$, the direction of the ray turns and the distance between two elements is

$$r_{i+1} - r_i = \frac{2}{\xi g_i}\sqrt{1 - \xi^2 c^2(z_i)} \tag{2.45}$$

Within each layer the ray paths are considered locally straight, as can be seen from Equation 2.44 and 2.45 (see Hovem [7, Ch. 6.2] for derivation). And at the interface between adjacent layers, the direction of the rays from one layer to another are given by the ray parameter ξ. An illustration of how and where the rays bend as function of sound speed is given in Figure 2.12. The two top figures illustrate that increasing sound speed bends the sound waves up, while the two lower figure show the sound waves bending downwards. If the sound speed is constant throughout the whole water column, the rays will follow straight lines.

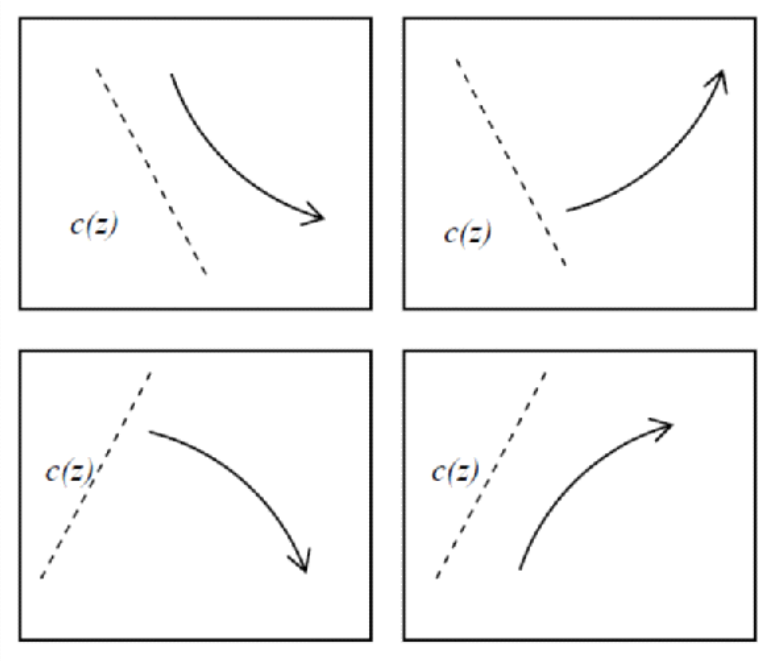


Figure 2.12: How sound waves bend according to increasing or decreasing sound speed c(z). Courtesy of Hovem [7, p. 114]

Figure 2.13 shows how rays from a source at a depth of 1000 meters propagate over a distance. The beamwidth of the source is 40.6° (±20.3° from horizontal orientation), and the ocean depth is 5000 meters. The black curves are reflections, while the red curves are direct path. On the left side of the figure is the plot of the sound speed profile used to calculate the rays' paths. We see that the rays are bending according to Figure 2.12. With the sound speed having its minimum velocity at a depth around 1400 meter, the rays will bend towards the depth of 1400 meters resulting a

waveguide.

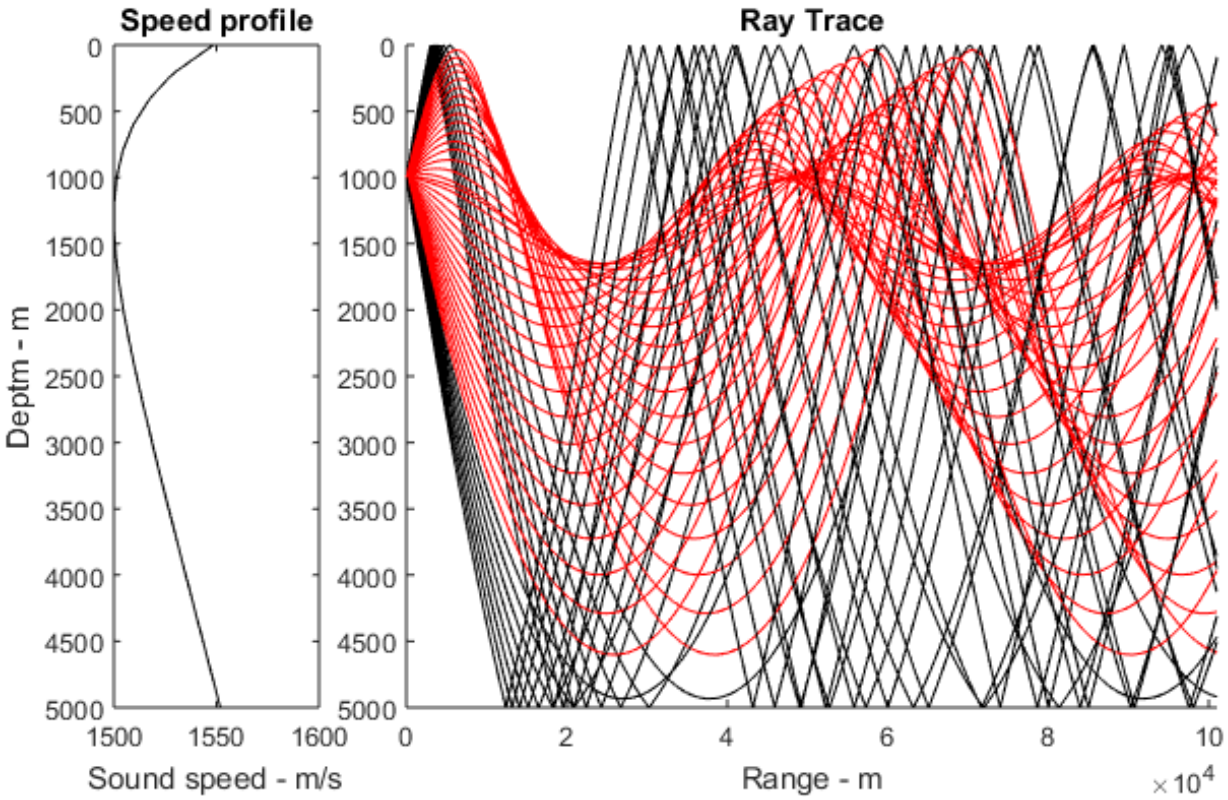


Figure 2.13: Plot of a ray trace produced with BELLHOP [15] for a 50Hz signal. The range is 100 km and the water depth is 5 km. The red curves are direct paths, while the black curves have one or more reflections from either the surface or the seabed.

2.7 Ambient noise

From Hovem [7, p. 330], there are several sources creating underwater noise, in the low frequency the main contributor is often human interaction and ship traffic. Underwater noise created by ocean waves, wind and rain is called *ambient noise*. High in frequency, above 100 kHz, the dominating noise contributor is thermal noise as can be seen in Figure 2.14. Along with these, there are local differences and marine animals, e.g. snapping shrimps, ships and whales creating powerful noise levels [21].

The noise spectral level (NSL) of a simplified ambient noise is shown in Figure 2.14 for five different wind speeds. Often the noise level is expressed in terms of sea states, rather than wind speed, which is only another way representing the noise level in therms of the ocean condition.

Ambient noise can be modelled according to the Knudsen curves described in Knudsen et al. [9]. The larger the waves, the noisier it gets.

In the range 100-1000 Hz the noise power is approximately constant, but from 1 kHz the power decays around 17 dB per octave. At around 100 kHz the thermal noise starts interfering, and is outside the frequency range to consider. A measure of the influence of noise to the signal is signal-

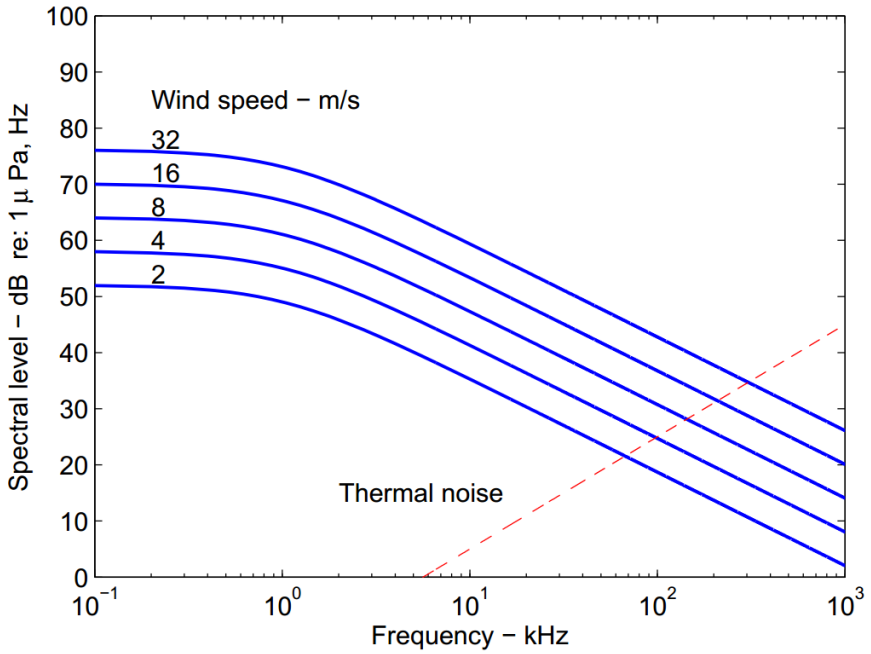


Figure 2.14: Noise spectral level of ambient noise with four different wind speeds according to the simplified Knudsen curves, from Hovem [7, p. 330].

to-noise ratio, or SNR. It is defined as

$$SNR = 10\log\frac{P_{signal}}{P_{noise}} \tag{2.46}$$

where P_{signal} is the signal power and P_{noise} is the noise power.

2.8 Channel

Soni et al. [17] stated that: "Underwater acoustic channels are generally recognized as one of the most difficult communication media in use today." Some of these reasons are described in Sections 2.1 to 2.7. A communication channel is used to convey information from one place to another. One concept often associated with communication channels are *reciprocity*, or that a channel is

reciprocal. The meaning of reciprocity is that a channel from A to B is identical to the channel from B to A.

To simulate a channel digitally, it is required to find a way of representing the channel mathematically. In signal processing, the channel can be represented by its *impulse response* $h(t)$. An impulse response is defined as the output of a system after being subjected to an impulse signal. Often used for the impulse is a Dirac delta function, where the function is expressed as

$$\delta(t) = \begin{cases} +\infty & t = 0 \\ 0 & t \neq 0 \end{cases} \tag{2.47}$$

and

$$\int_{-\infty}^{\infty} \delta(t)\mathrm{d}t = 1 \tag{2.48}$$

The Dirac pulse has properties of great advantage in analyzing systems, where the convolution between the pulse and a time signal is

$$\begin{aligned} x(t) * \delta(t-T) &\equiv \int_{-\infty}^{\infty} x(\tau)\delta(t-T-\tau)\mathrm{d}\tau \\ &= x(t-T) \end{aligned} \tag{2.49}$$

where it is seen that the signal $x(t)$ is shifted by time T given in the Dirac pulse to $x(t-T)$. Either if the system is continuous in time or time-discrete, the properties of such a pulse as the Dirac, given in Equation 2.49, can be exploited. Proakis and Manolakis [16, Ch. 2] thoroughly describe time-discrete system behaviour when introduced with a pulse signal.

2.8.1 Point-to-point communication

The simplest example of a communication channel is a point-to-point channel, meaning we have one source and one receiver. When a signal is transmitted from the source, the surroundings influence the signal, resulting in a different/changed received signal. Mathematically, the input/output relation of a system is given as

$$y(t) = h(t) * x(t) + w(t) \tag{2.50}$$

where $y(t)$ is received signal, $h(t)$ is the channel impulse response, $x(t)$ the transmitted signal and $w(t)$ is noise. Thus, by exploiting the property of the Dirac pulse from Equation 2.49, using it as the input signal $x(t)$ and ignore the noise $w(t)$, the system impulse response can be given as

$$\begin{aligned} y(t) &= h(t) * x(t) \\ &= h(t) * \delta(t) \\ &= h(t) \end{aligned} \tag{2.51}$$

Figure 2.15 shows a simple plot of how a communication channel between the source Tx and receiver Rx can be seen. The blue line is the direct path from the source to the receiver, and the black lines are reflections. The impulse response of such a simplified case can look like the plot in Figure 2.16, represented only with some loss and propagation delay.

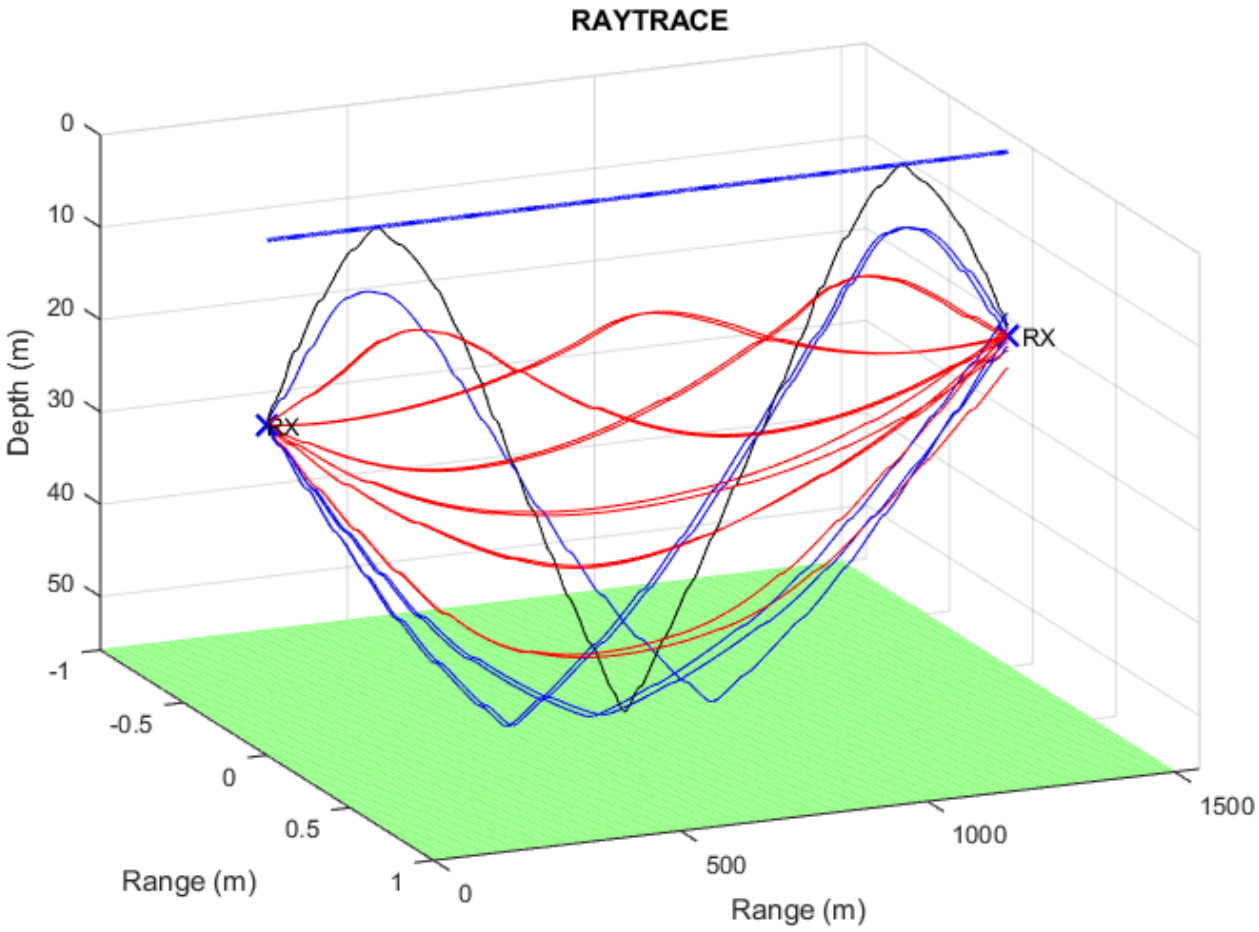


Figure 2.15: Channel representation underwater for point-to-point. The surface are given by the blue line, seabed with the green area, surface reflections are black while seabed reflection is blue, and the direct path is shown by the red lines. The range is 1500 meters, with a depth of 58 meters.

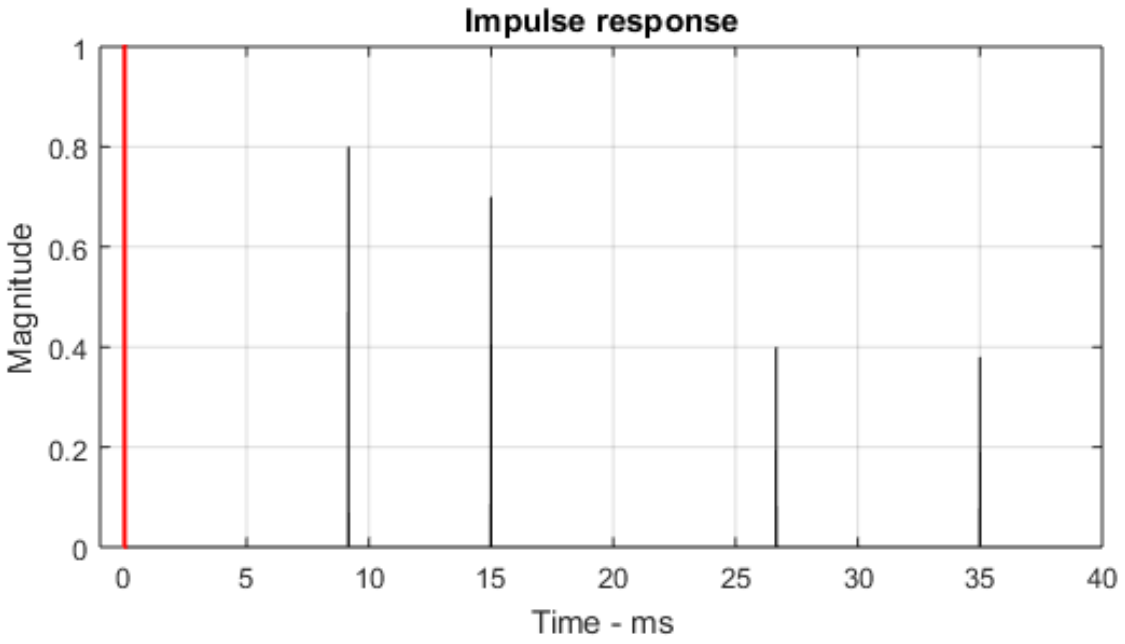


Figure 2.16: Example of how an impulse response can look. The red peak is the direct path, and is the first arrival, while the black taps are reflections.

2.8.2 Network communication

Network communication is defined as a group of transmitting and receiving devices that are connected together, where the devices in the network are referred to as nodes. There are numerous benefits from having a network of nodes, such as the familiar Internet, where the ability of sharing information is utilized to great extension. In underwater acoustics, networks of connected nodes can be used, for example, in acoustic positioning systems to track or navigate by triangulation, where Kongsberg Maritime[11] is one of the leading developers in that field.

Isolated, two nodes works as the point-to-point communication link explained earlier, but for all nodes to be connected with each other, the complexity increases. The complexity $M_{channels}$ increases as a function of number of nodes N as

$$M_{Channels} = \frac{(N-1)N}{2} \tag{2.52}$$

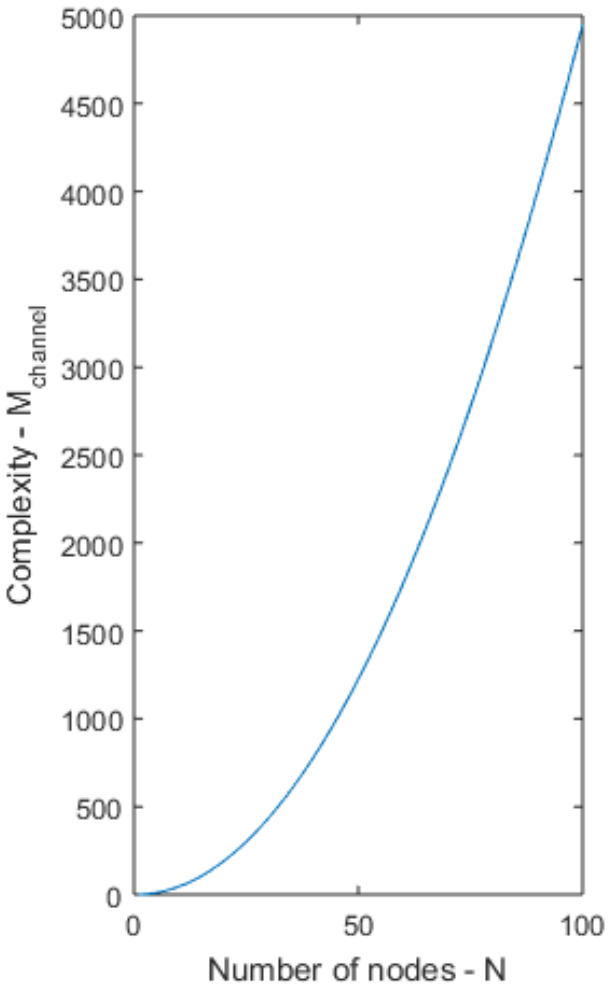


Figure 2.17: A graph showing the increasing complexity $M_{channels}$ as a function of number of nodes N

Figure 2.17 shows how the complexity increases as a function of number of nodes. It can be seen that the complexity from increasing the number of nodes quickly approach complicated systems. For instance, if there are 4 nodes it will require 6 individual channels for the 4 nodes to be connected, while doubling the number of nodes to 8 require 28 channels. Doubling the number of nodes from 4 to 8 increases the complexity by a factor of $\frac{28}{6} \approx 4.67$.

Figure 2.18 shows an overview 4 nodes in a network, with 6 different channels connecting every node. Between nodes, the channels are represented by the rays based on the ray tracing principle described in Section 2.6. The nodes are marked with Rx and it can be seen that every node has

$N-1$ connections.

Figure 2.19a shows a table for 4 nodes, giving an overview over which nodes are connected through the corresponding channels h_i according to Figure 2.19b. The received signals for the 4 nodes are given as

$$\begin{aligned} y_1(t) &= x_2(t) * h_1(t) + x_4(t) * h_2(t) + x_3(t) * h_3(t) + w(t) \\ y_2(t) &= x_1(t) * h_1(t) + x_3(t) * h_5(t) + x_4(t) * h_4(t) + w(t) \\ y_3(t) &= x_1(t) * h_2(t) + x_2(t) * h_4(t) + x_4(t) * h_6(t) + w(t) \\ y_4(t) &= x_1(t) * h_3(t) + x_2(t) * h_5(t) + x_3(t) * h_6(t) + w(t) \end{aligned} \tag{2.53}$$

where x_j, for $j \neq i$, is the output signal from node j. The output signals are convolved with the corresponding channels given in the table in Figure 2.19a, and noise $w(t)$ is added.

A more general representation can be seen in Equation 2.54. The diagonal in the channel matrix $\vec{H}_{N,N}$ is zero, telling that the the nodes are not communicating with themselves. Since it is assumed that the channels are reciprocal, there is symmetry along the diagonal in the matrix. N is the number of nodes, for N≥2.

$$\vec{Y}(t) = \vec{H}(t) * \vec{X}(t) + \vec{W}(t) \tag{2.54}$$

where

$$\vec{H}_{N,N} = \begin{pmatrix} 0 & h_{1,2}(t) & \cdots & \cdots & h_{1,N}(t) \\ h_{2,1}(t) & 0 & \cdots & \cdots & h_{2,N}(t) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & h_{N-1,N}(t) \\ h_{N,1}(t) & h_{N,2}(t) & \cdots & h_{N,N-1}(t) & 0 \end{pmatrix} \text{ and } \vec{X}_N = \begin{pmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ \vdots \\ x_N(t) \end{pmatrix} \quad \vec{W}_N = \begin{pmatrix} w_1(t) \\ w_2(t) \\ w_3(t) \\ \vdots \\ w_N(t) \end{pmatrix} \tag{2.55}$$

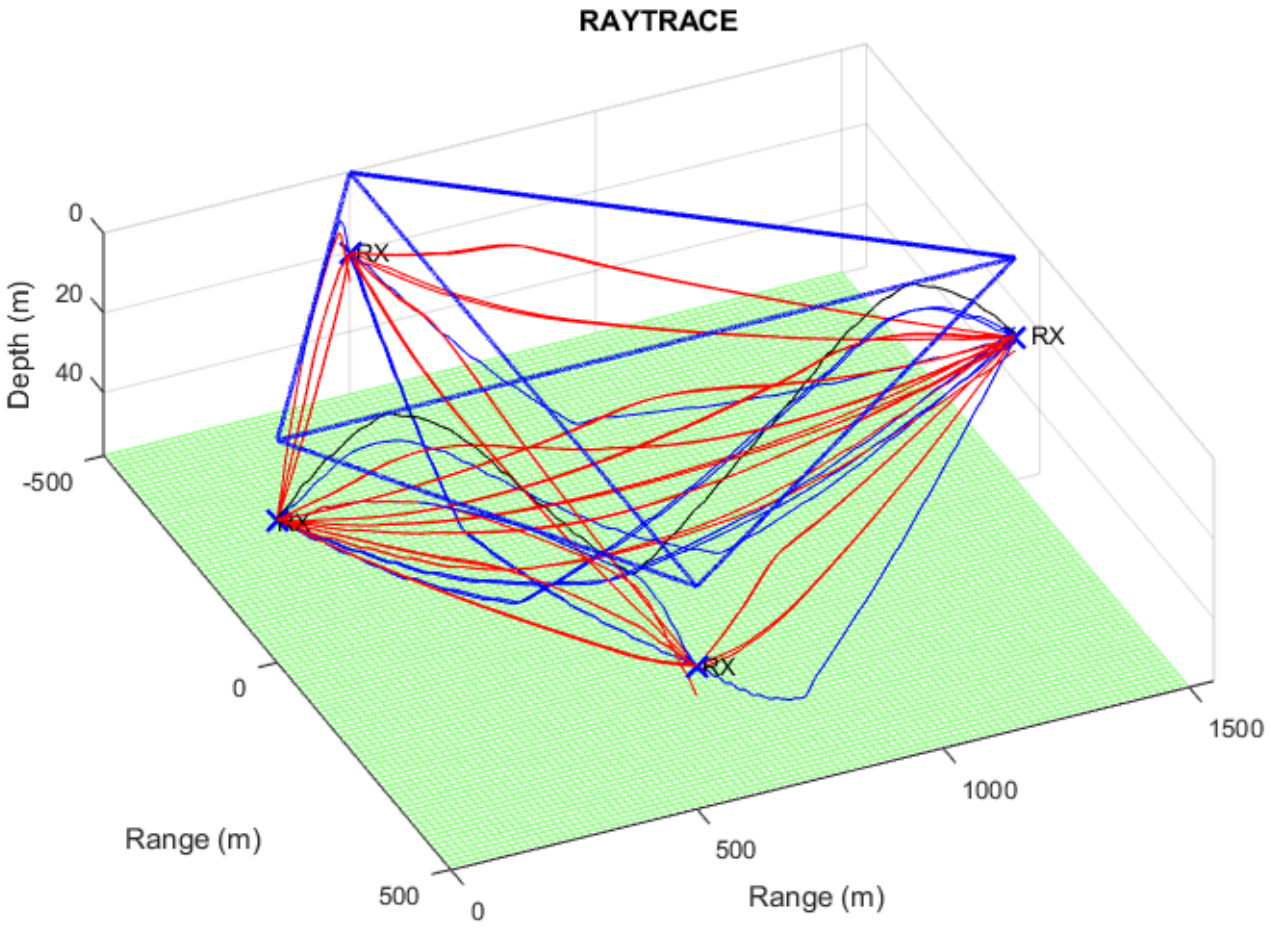
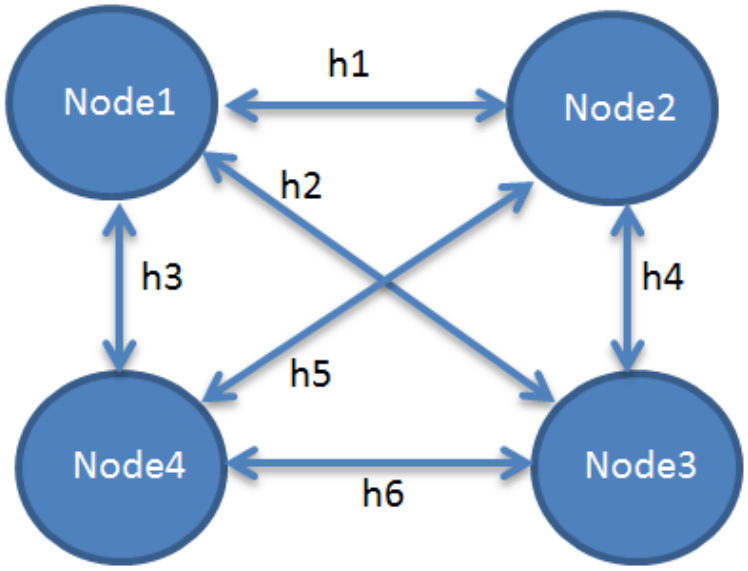


Figure 2.18: A 3D representation of network communication with four nodes, creating six different channels. The nodes are placed in X,Y,Z coordinates: $(0,0,20)$, $(500,500,20)$, $(500,-500,20)$, $(1500,0,20)$. Figure produced using HACE.

	Node 1 *Receiver*	Node 2 *Receiver*	Node 3 *Receiver*	Node 4 *Receiver*
Node 1 *Transmitter*		h1	h2	h3
Node 2 Transmitter	h1		h4	h5
Node 3 *Transmitter*	h2	h4		h6
Node 4 *Transmitter*	h3	h5	h6	

(a) Table representing which channels connect the nodes for 4 nodes. The grey fields shows the active channels



(b) Source tree between the 4 nodes represented with 6 channels. It is assumed the channels are reciprocal

Figure 2.19

2.9 Digital Audio Processing in Real-Time

To understand digital audio processing, sampling is the most basic concept to start with. Sampling in signal processing is a method of converting a continuous time signal into a discrete signal [10]. How often the continuous signal is discretized every second is given by the sampling frequency, or sample rate, f_s defined as

$$f_s = 1/T \tag{2.56}$$

Where T is time interval between two adjacent sampling points. To be able to do an analog to digital conversion (ADC) or digital to analog conversion (DAC) of the signal properly, the two Shannon's sampling theorem conditions must be fulfilled

1. The analog signal, $x(t)$ must be band-limited by a bandwidth f_m of the signal
2. The sampling frequency , f_s must be at least twice the maximum frequency component f_M in the analog signal $x(t)$ that is $f_s > 2f_m$

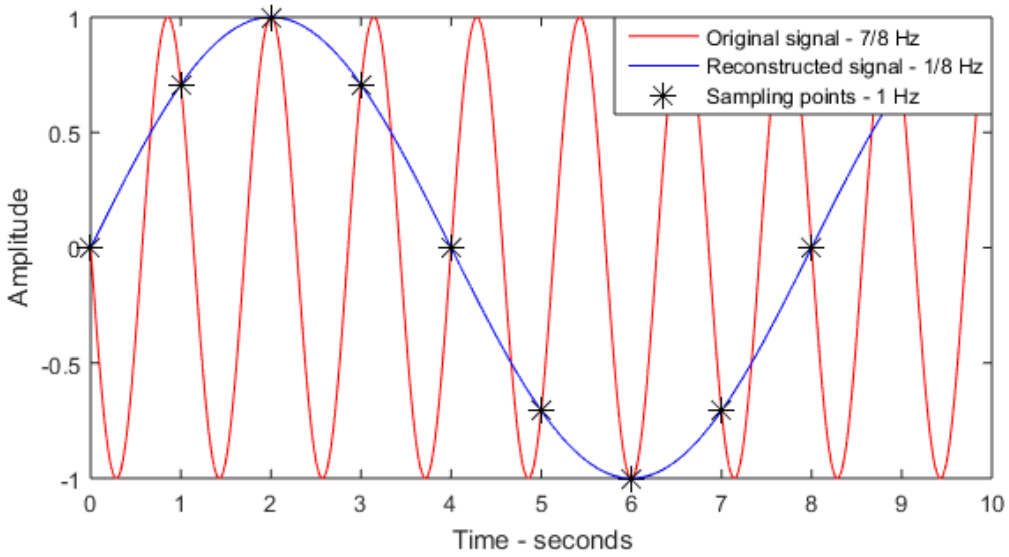


Figure 2.20: Example of aliasing where the input signal is 7/8 Hz and sampled with 1 Hz, results in a signal with 1/8 Hz.

If the bandwidth is too large, or the sample rate is too low, the reconstruction will be imperfect and result in *aliasing* [1]. Figure 2.20 show an example of aliasing where a sine signal of 7/8 Hz is

[1]Aliasing - a sampled signal that contains too few sample to be able to represent the original signal

sampled at 1 Hz results in a sampled sine with frequency 1/8 Hz, which is not desired. There is usually used an anti-aliasing filter[20], low pass filter at $f_s/2$, before the sampling to reduce the high frequency signal components.

The process from analog to digital representation is called quantization, Figure 2.21 show an example of 2 bits quantization of a sine. A quantization error is the error between the quantized representation and the sampled signal. This analog/digital conversion is often handled by a sound card. For more information about this subject and derivations see Proakis and Manolakis [16, ch 1.4]

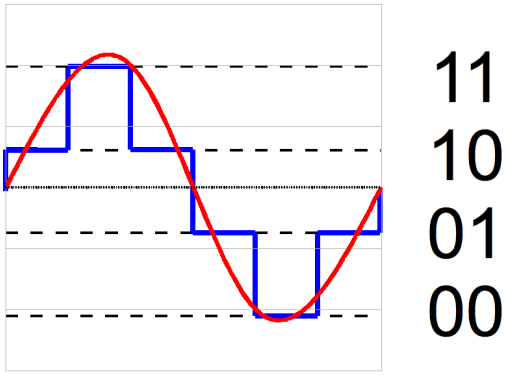


Figure 2.21: Example of how a 2 bits quantization of a sine is

2.9.1 Real-Time

A system operates in real-time if inputs are collected constantly, while returning a response from the current input. This means that the system have to be very efficient, with as small *latency*[1] as possible, to be able to compute the data flow within the "real-time" requirement of a system. The time aspect of the allowed latency for real-time system varies, where the requirement often correlates to the speed of the environment the system is placed in.

A block scheme of a real-time audio system is shown in Figure 2.22. The analog input signal $x(t)$ is converted to a digital signal $x(n)$ by the ADC block, here operated by a sound card, then signal processed by, e.g. MATLAB. From there, the output $y(n)$ is then converted back to an analog signal $y(t)$ with the help of a DAC, where the time between the input $x(t)$ enters and leaves the system Δt_s is defined as the latency in the system.

The processing speed of the system must be so that the queue, or *buffer*[2], of data inputs does not increase in time. If the computations takes more time processing than the signal length, the transmitter will have to “wait” for the signal, thus, be subjected to an unwanted time latency. The lag can also cause information loss if the output does not contain all the necessary elements due to lack of signal at the transceiver at the time instant of transmission.

A way to perform real-time processing is to divide the signal into frames, hence the name *frame-based processing* [12]. Each frame contains a small fragment of the total signal. Figure 2.23 give

[1] Latency - the time it takes for an input to affect the output
[2] Buffer - temporary data storage.

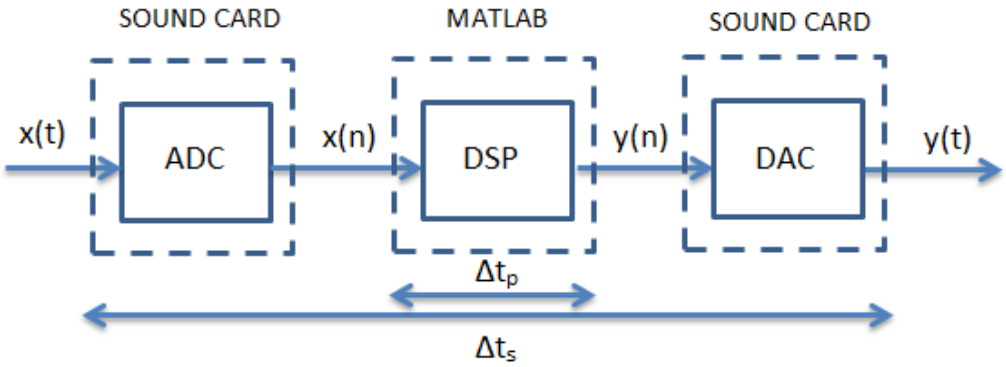


Figure 2.22: Block diagram of real-time audio DSP. Where $x(t)$ analog input, $x(n)$ the digital representation, $y(n)$ the processed signal, $y(t)$ analog output. The real time latency/ system latency, Δt_s, is the time between the signal enters and leaves the sound card. Processing latency Δt_p is the DSP time MATLAB uses to handle the data.

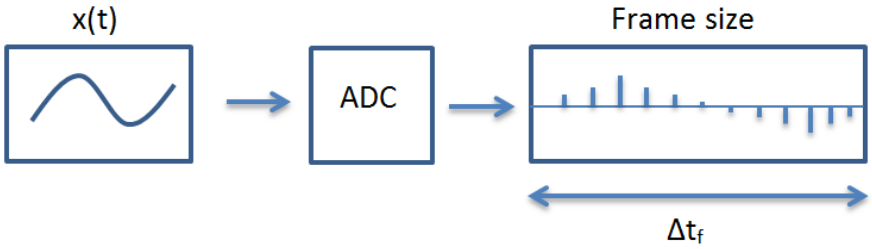


Figure 2.23: Example of frame size with 12 samples

an example of a signal with 12 samples as frame size. The processing time Δt_p cannot be more than the frame size latency Δt_f. For example a frame size at 1536 samples with sample rate of 96 kHz gives a latency, Δt_f, of 1536/96000 = 16 milliseconds. In frame-based processing the system latency is directly proportional to frame size, meaning that if this frame is 16 ms long, an input signal is not registered at the output for at least 16 ms. The total system latency Δt_s will then be $\Delta t_f + \Delta t_p +$ AD/DA conversion. Thus, to minimize the system latency Δt_s, the frames must be chosen as small as possible, but choosing the smallest frame size is not always advantageous. With respect to system stability, a small frame can cause the system to be unstable having events where the queue is either full or lacking information. Hence the frame size must be chosen so that the system achieves stability meanwhile having a short "enough" latency. How large the latency is, also highly depends on the sound card.

Stability means in this context that there is very small or no variation in latency, *jitter*[1], affecting

[1]Jitter is random occurrence of delay in signal [18]

the output signal. If the signal is unstable there will be problem to get credible results. Stability often correlates with the frame size of the system, the larger frame, the more stabile system. This relationship will be demonstrated later and shown in Appendix E.6.

Sometimes a larger frame size is needed. A larger frame size results in having the ability to process more data per frame and increase stability, but will produce a larger latency. For a real-time system to be able to handle large operations meanwhile having a small latency, the frame size must be varying as a function of computational requirements. For the real-time processing performed in this master, the increase in computations are mainly related to number of reflections and range.

Figure 2.24 give an easy example of frame-based processing. From left, the input signal is sampled and put in a frame, marked with the yellow square. The frame is then handled according to the system description. Lastly, the finished processed frame is transmitted as an output signal, and then continues with the next frame and so on. In the figure here, the signal processing does not do anything with the signal.

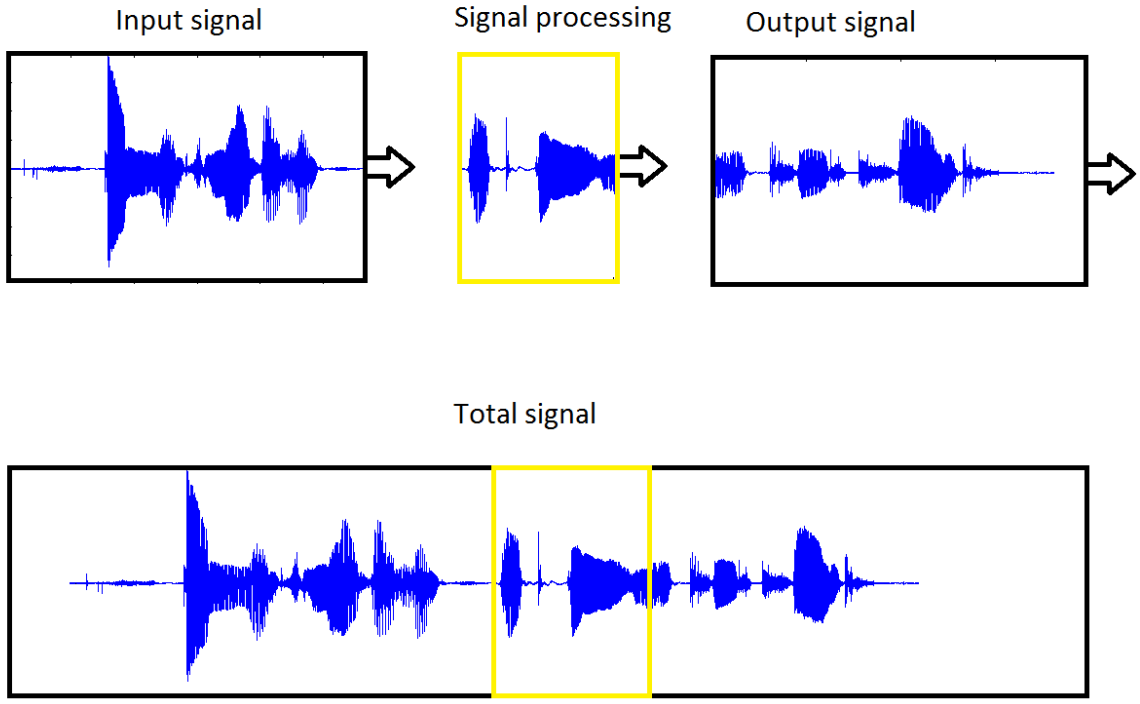


Figure 2.24: Each frame is extracted from the signal, processed and then transmitted (top). Shows the total signal after being processed (bottom).

To sum up the constraints of a DSP systems for real-time applications is that the bandwidth of

the system is limited by the sample rate. The processing speed Δt_p determines the rate at which the analog signal can be sampled. A real-time DSP system demands that the signal processing time, Δt_p, must be less than the frame size latency Δt_f in order to complete the processing task before a new frame introduces.

2.10 Implementation of Theory

A list of what has been implemented in the thesis from the theory section.

- 3D representation of surface and seabed
- 3D model axisymmetric, 2D representation of ray waves
- Plane wave reflection and transmission
- Spread loss
- Acoustic absorption across the entire frequency spectrum.
- Forward sea surface scatter
- Time variating channel from Doppler spread.
- Ambient noise, wind speed/sea state dependent.
- Transducer directivity and sensitivity
- Varying sound speed profile
- Point-to-point and Network communication
- Real-time simulation

In Section 3.2.4 is an overview of where the parts from the theory is added into HACE.

Chapter 3

Implementation and Execution

In Chapter 2 the physics behind waves underwater was presented, with an introduction into frame based real-time signal processing, transducer properties and network communication. This chapter will focus on how the implementation of HACE was performed and how it can be used together with hydroacoustic equipment to perform tests. This section will start to explain how the different equipment was connected, followed by a total system overview of the simulations and how the signal processing in real-time was done. This chapter will also show how the hydro acoustic equipment fit together with HACE, for both point-to-point and network communication. At the end, it is shown how real world underwater measurements were conducted on the field trip to Horten.

3.1 Channel Emulator Setup

Figure 3.1 shows a block diagram of HACE with two nodes. The figure contains two Kongsberg Maritime Universal Transponder Boards (UTB), two analog attenuation circuits (El), a computer (PC) and a sound card (audio interface). The sound card, the attenuation circuit, and the computer with the simulation program sums up to become HACE. Node A is considered to be the main node and it is connected to the computer. The computer contains the software, APOS, and uses it to gather information and to control the communication protocol (see Appendix A.2 for further information about APOS). The UTBs are modems, modulating and demodulating signals for transmission and reception, respectively. Node A consists of UTB-A and the computer with APOS, while node B is UTB-B. Node A transmits a signal to node B, and node B answers.

Each UTB card requires a voltage of 12- 14.4 V to work. From the UTBs, the output voltage is amplified to the order of ~100 V, thus the signal $x(t)$ has to be attenuated to the range

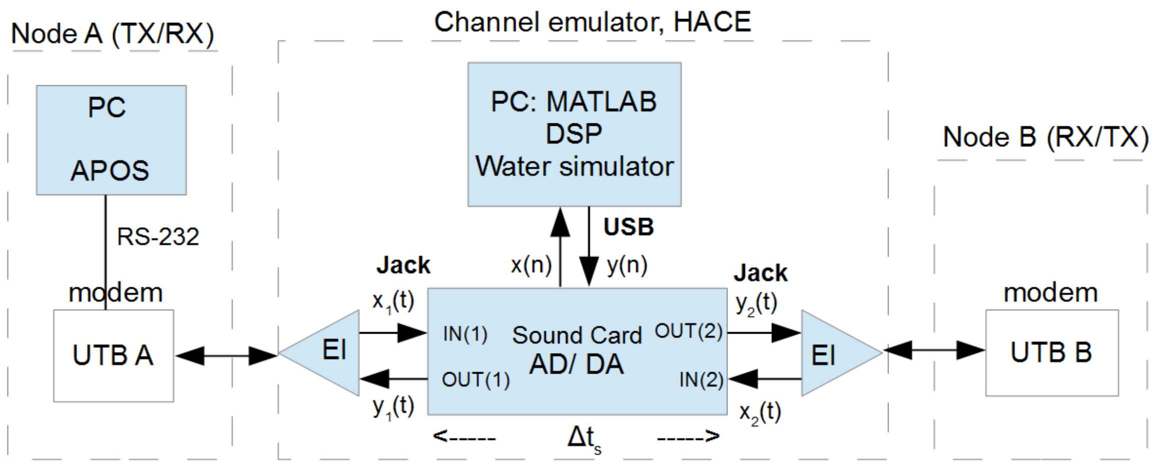


Figure 3.1: Block diagram of HACE during point-to-point communication. The UTBs are modems. UTB A is connected to the computer that which controls the node via APOS. $x_1(t)$ is the analog input signal from node A after been attenuated by El, and $x_2(t)$ is the analog signal from node B. $y_1(t)$ and $y_2(t)$ are the received analog signals after being exposed to the simulations. $x(n)$ and $y(n)$ are the discrete representations of the analog signal input and output signals, respectively.

that matches the sound card (0 - 3 V). This attenuation is performed by the El card having a 51 dB attenuation for transmitting and 58 dB attenuation for the receiving. Both the UTBs and Els are made by Kongsberg Maritime and connected together by their design. In Appendix F.1 a schematic diagram of the El card is shown. The El cards are connected to the sound card with jack cables as can be seen in Figure 3.2. In this figure there are four nodes connected to the sound card, four for receiving and four for transmitting signals.

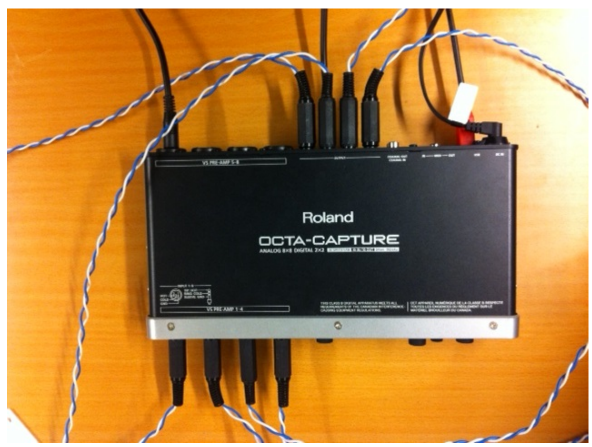


Figure 3.2: OCTA CAPTURE connected to four nodes, four inputs and four outputs.

For the point-to-point setup in Figure 3.1, let node A be the transmitter, and node B be the receiver. The analog signal $x_1(t)$ from node A is first attenuated in the El card before being connected to the sound card's input channel 1. The sound card digitalizes the analog signal and is accessed by the computer through an USB cable from the sound card as $x(n)$. Afterwards the signal has been processed by the simulations on the computer, $y(n)$ is sent back to the sound card, and from the sound card transmitted through the sound card's output channel 2 as $y_2(t)$ where node B receives the signal. When node B replies, ìt can be seen as the transmitter. The output signal $x_2(t)$ is con-

nected to the sound card's input channel 2 and enters the simulation. After the simulations are applied to the signal, it exits as $y_1(t)$ through the sound card's channel output 1.

3.1.1 Network communication

To enable network communication several nodes are required. Described in Section 2.8 the number of multipath channels increase more rapidly than the increase in the number of nodes. An illustration on how four nodes in a network are connected via the multipath channels is shown in Figure 2.19a and 2.19b, The figures show that node 1 communicates with node 2, 3, 4, and node 2 communicates with 1, 3 and 4, and so on. In a more general therm, every node communicate with all other nodes in the network.

Figure 3.3: Transponder, cNODE, from Kongsberg Maritime during test. The transponder is connected to the sound card.

Same as for the point-to-point communication, one node is connected to the computer running APOS in order to control the communication and log data. The nodes are connected to the sound card's corresponding channels, as seen in figure 3.4 (node 1 to input and output channel 1, node 2 to input and output channel 2 and so on). The color coded paths in the figure shows the routing of signals from the input to the output channels. More detail about the routing, and how the processing of the data stream was handled is shown in next section. Figure 3.5 shows the lab setup of the channel emulator during test. The sound card used in this setup is OCTA-CAPTURE and is shown in Figure 3.2. In Figure 3.3 a picture of one of the transponders during test is shown. The node is sealed inside the yellow and back cylinder to avoid crosstalk and it is connected to the sound card. A list of equipment used in the setup of HACE during test is found in Appendix A.

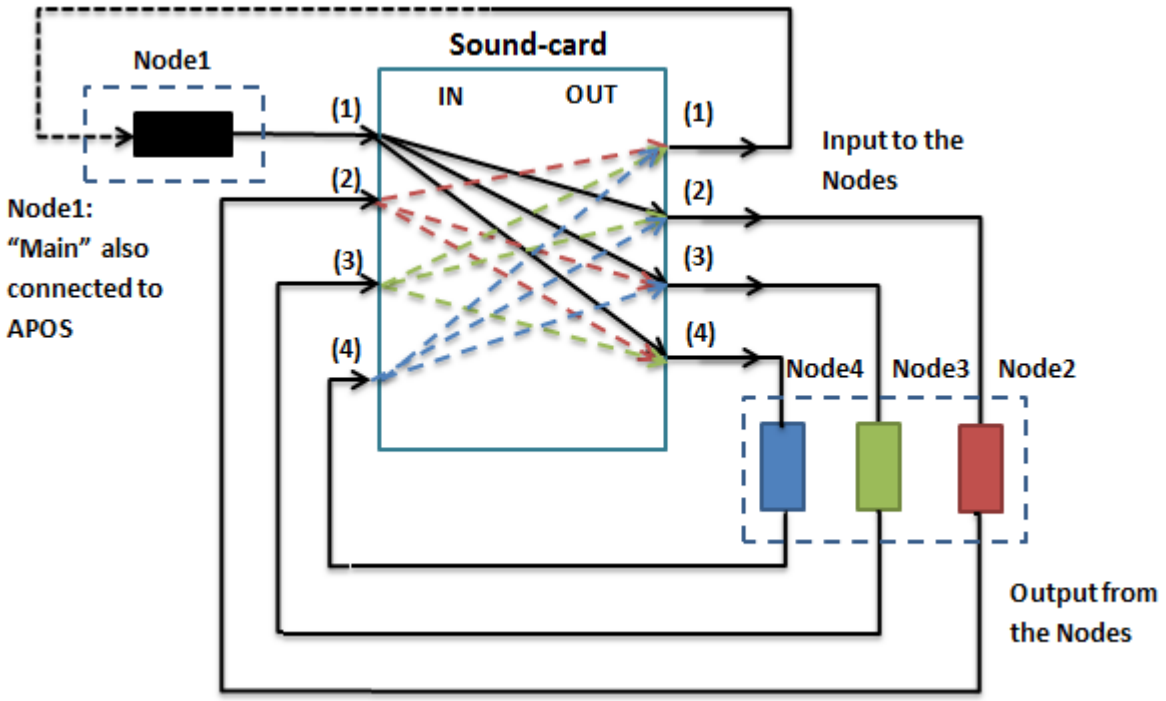


Figure 3.4: Network block scheme with four nodes. The nodes are connected to the sound card, and routed. Color coded paths between the nodes shows where the signal from one node are distributed.

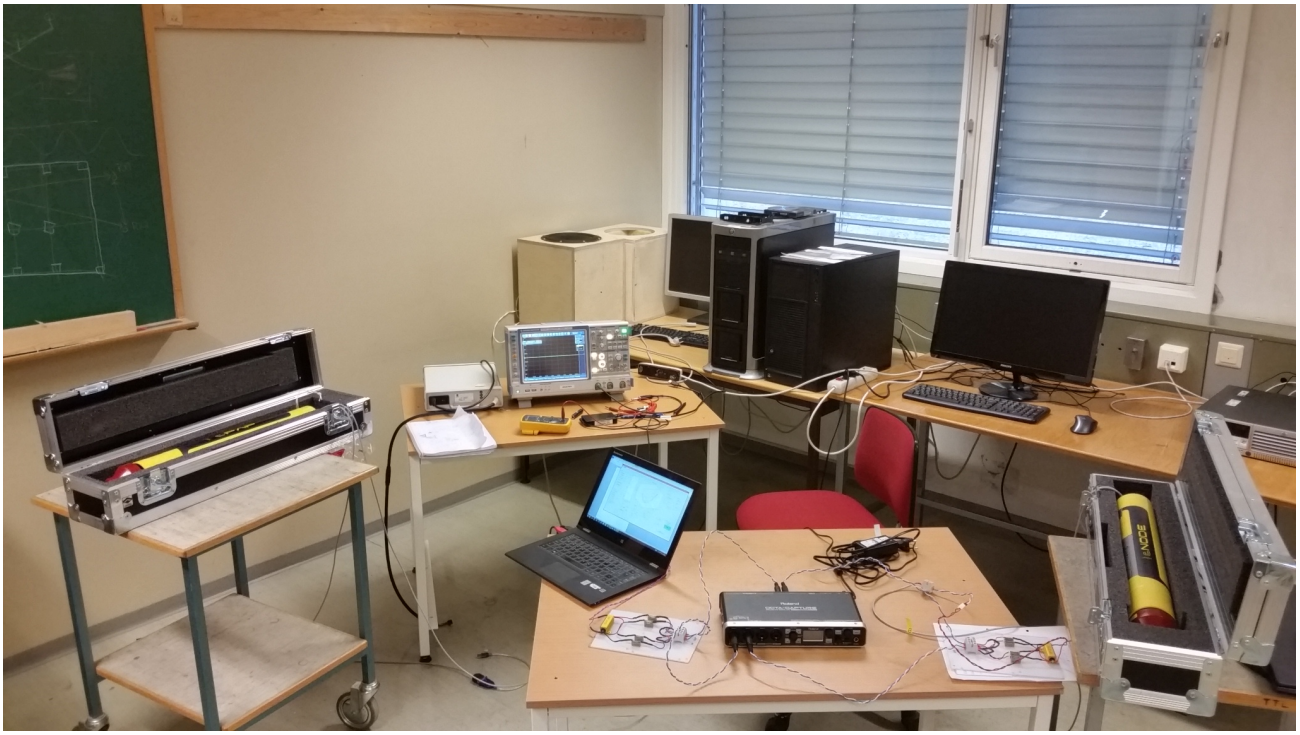


Figure 3.5: Lab setup of the channel emulator: Two nodes connected to the simulator via El attenuation board and the sound card.

3.2 Creating the channel

HACE can been described as a black box containing the algorithms for simulations and ADC/DAC, with an input and output signal, as shown in the Figure 3.6. The input signal is the analog audio signal from the nodes and the output signal is the analog signal after being exposed to HACE, preferably acquiring hydroacoustic behavior. This section covers how the simulation process was performed.

Figure 3.6: The system described by a black box with input and output signal

When the signals from the transmitters have become digital using the sound card, the acoustical models explained in Chapter 2 can be applied to the signals. The more acoustical models are applied, the more processing and calculations are required. Since, the latency is essential for real-time systems, as explained in Section 2.9, it is desirable to have the processing of data in real-time at a minimum. Thus to minimize the amount of processing in real-time, all possible calculations for the simulations are performed before running the simulations in real-time, as shown later in the pseudocode in Algorithm 1.

The general approach has been to compute the acoustical models from Chapter 2 and creating an impulse response from these models. The finished calculated models are inserted into a large matrix H, where it ends up containing all the acoustical models. Then the correct signals are convolved with the corresponding impulse responses. By having an impulse response only a convolution operation in real-time is required.

An impulse response between two nodes is shown in Figure 3.7, and consists of three separate paths between the nodes. The nodes are placed 480 meters apart in the horizontal direction, 20 and 180 meters below the surface and with an ocean depth of 185 meters. Arriving first is the direct path, second is a seabed reflection and last a surface scatter from a sea surface reflection. Both the first arrival and seabed reflection look like peaks, but actually they are filters with coefficients according to the water absorption given Section 2.3.

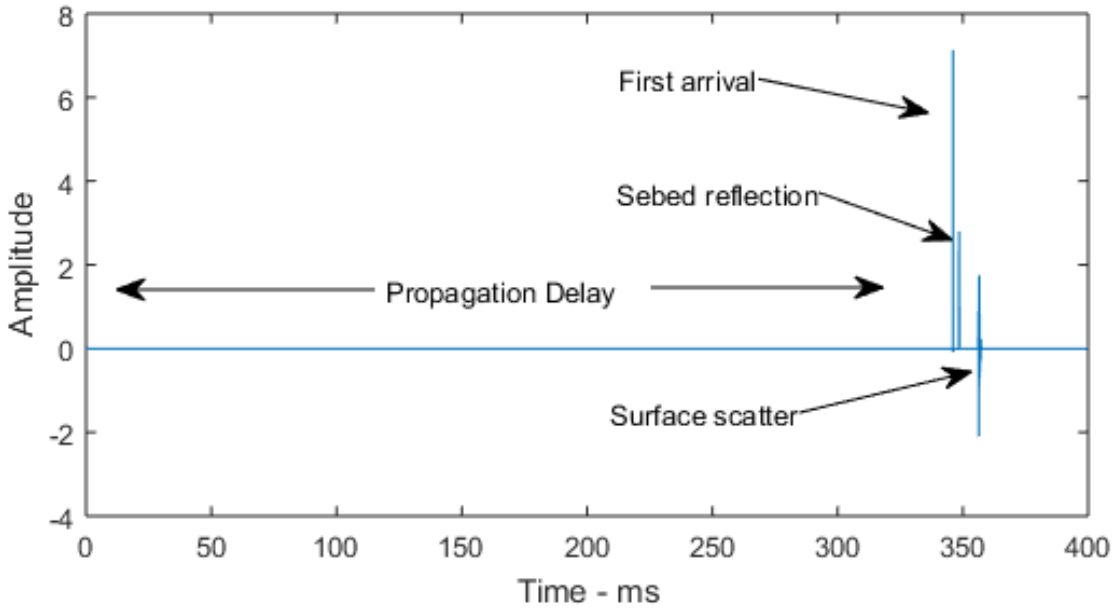


Figure 3.7: Impulse response for a multipath channel. The propagation delay is 340 ms. There is one direct path between the nodes, and two reflections. One of the reflections is a seabed reflection, while the other is a surface scatter reflection. Noise is not present.

3.2.1 Doppler Spread and Scatter

Figure 3.7 showed how an impulse response can look for one time instance, but in order to introduce a changing channel according to the Doppler spread, new channel representations must be drawn with the bases in the Doppler bandwidth. The number of different channel representations are given as a function of sampling frequency divided by frame size, since this is the highest number of possible changes per second. For a frame size of 1536 samples the maximum number of channel representations per second is then $\frac{96000}{1536} = 62.5 \Rightarrow 62$ for a sampling frequency of 96 kHz. To achieve a 3 dB spectral width Δf_{3dB} the scatter is filtered using a low pass filter with cut off frequency f_{cut} given by the spectral width from Equation 2.42 in Section 2.4.

Figure 3.8 shows a plot of how a surface scatter can look. The plot has been normalized in amplitude, and limited when the amplitudes are 15 dB lower than the maximum value. Surface scatter has the function of spreading signals in time as described in Section 2.4.

Shown in Figure 3.9 is how the channel is changing for a surface reflection. The time span of the Doppler spread is 2 seconds. There is a relationship between adjacent scatter given by the 3 dB spectral width of the Doppler spread. During the 2 seconds, 50 different channels are experience. This means that the Doppler bandwidth is 25 Hz, as was explained in Section 2.4.

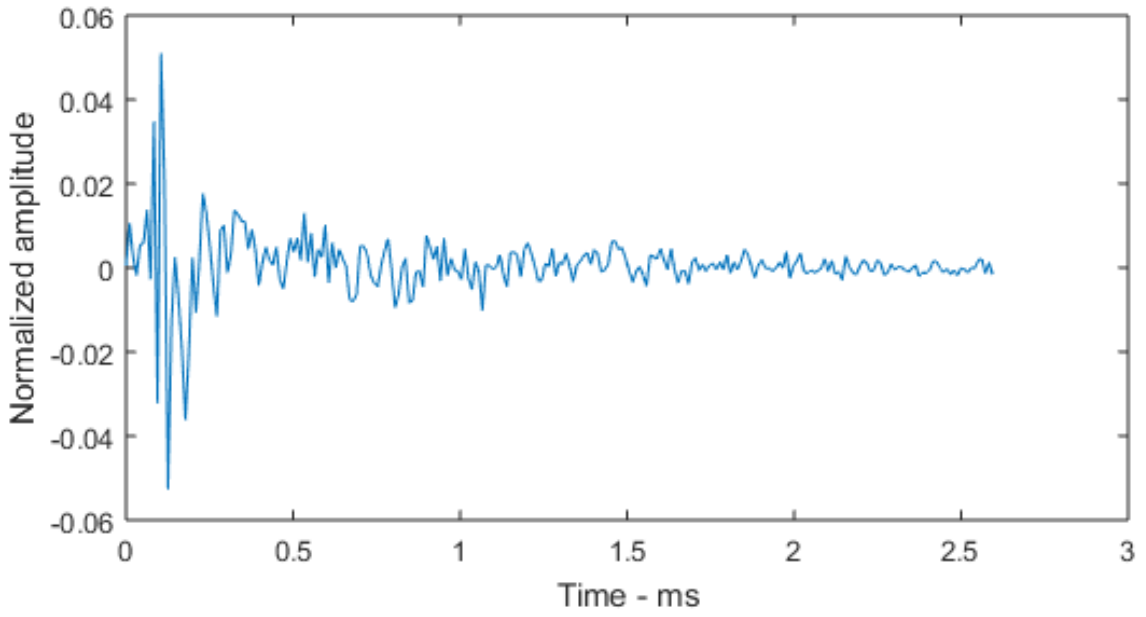


Figure 3.8: Plot of normalized surface scatter. The scatter is limited when the drop in amplitude is more than 15 dB relative to the highest peak.

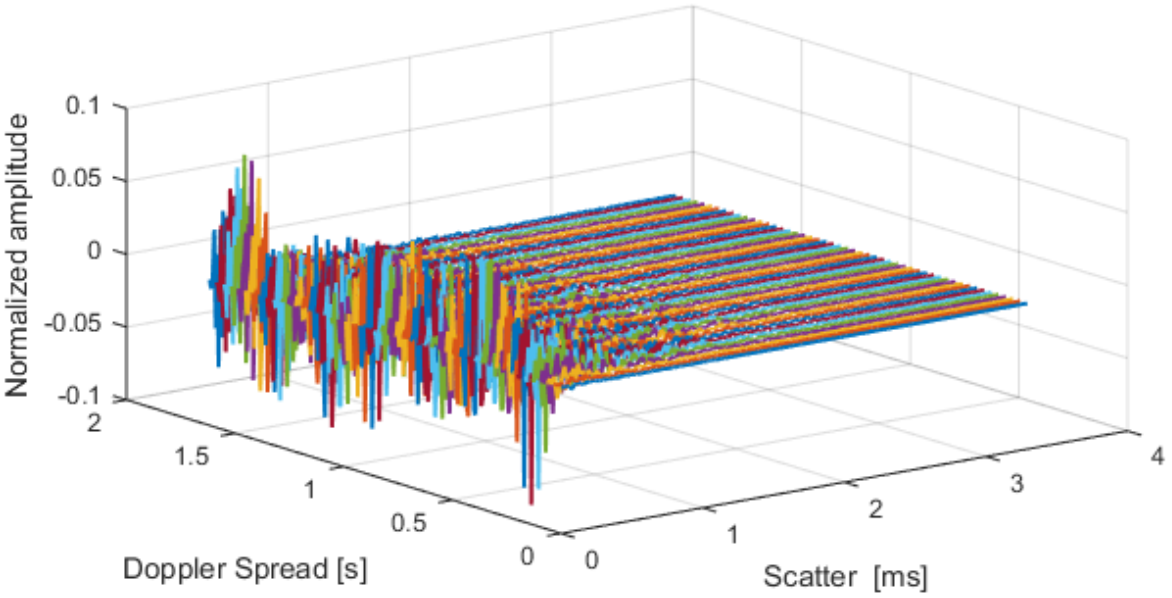


Figure 3.9: Plot of how the scatter is changing in time given by the Doppler spread. The 3 dB spectral width is 3 Hz.

3.2.2 Absorption filter

To add the frequency dependent water absorption from Section 2.3, *FIR*[1] filters were created. The filters are functions of distance, and since the absorption is increasing by distance, new filters must be created for all rays. The number of coefficients are fixed at 31, and were chosen based on two conditions, first by studying the deviation in filter response for an ideal scenario for different ranges, and secondly, we know from Section 2.8 and Appendix C.1 that convolutions create delay, and all filters are put into the channel impulse response. Thus, to have similar delay for all ray paths the number of coefficients was chosen to be fixed. A frequency response of an absorption filter in dB for a distance of 150 meters is given in Figure 3.10. In Appendix E.5 several examples of water absorption filters are displayed.

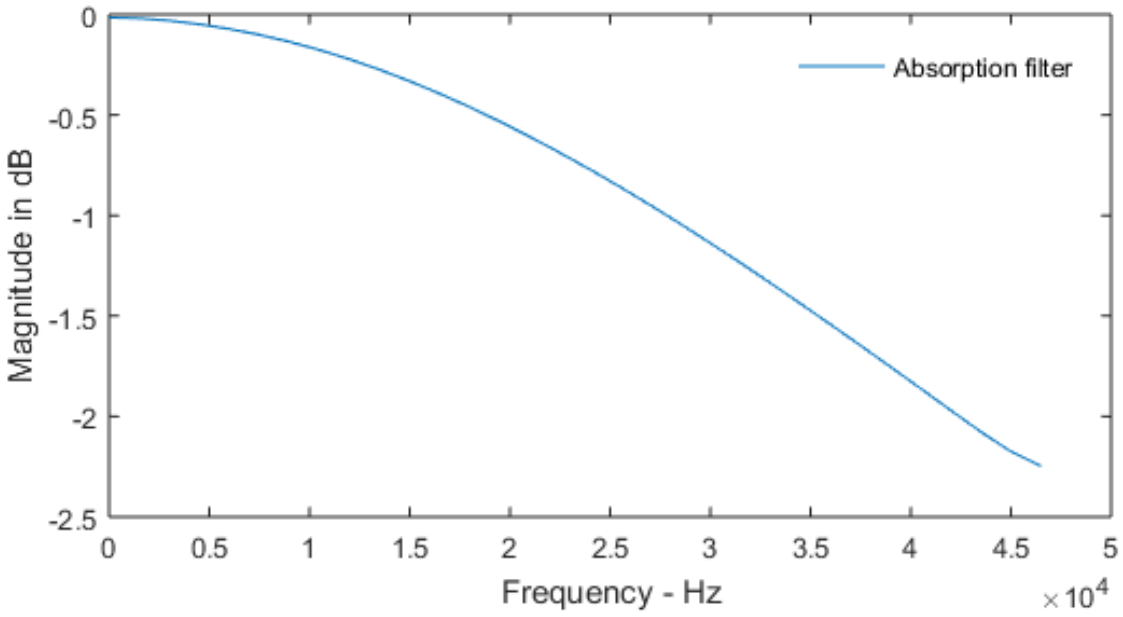


Figure 3.10: Frequency response of an absorption filter for 120 meters. The filter has a low pass characteristics. The sampling frequency used is 96 kHz, resulting in a maximum filter frequency of 48 kHz.

[1] FIR - Finite impulse response

3.2.3 Choosing frame size

From Section 2.9, we have that a larger frame size gives the ability to process more data per frame. Equation 2.54 from Section 2.8, combined with how the propagation delay t_d in samples $N = Fs \times t_d$ can be used from Appendix C.1, it is easy to see that the amount of computations increase as a function of the range, or more precisely the number of samples N. Thus, for the system to be able to handle a wide range of propagation delays, and large computations from acoustical models occurring in real-time, the frame size must be varying as a function of propagation delay.

Point-to-point

Between two nodes, there are several paths (reflections and direct) with different propagation distances or propagation delays. To select the correct frame size for a given setup, the propagation delay of the shortest path t_{ds} must be found, and compared with the system latency Δt_s for the different frame sizes. From Figure 3.7 we see that the propagation delay t_{ds} is around 340 ms. Thus, by exploiting this propagation delay it is possible to have a system latency Δt_s of up to 340 ms and still be able to simulate without loss.

Network

One key factor in creating HACE was to enable network communication with several nodes. We have that point-to-point communication requires one multipath channel, but the number of multipath channels increase by $\frac{(N-1)N}{2}$ for N nodes. In the network, the distances separating the nodes are varying, as shown in Figure 2.18 in Section 2.8. Between two nodes, the distance can for example be 100 meters, while for two other nodes in the same network the separating distance can be 600 meters.

By using the same technique as was performed for point-to-point communication, selecting the correct frame size for network communication can be done, the only difference is that the shortest propagation delay of all multipath channels in the network must be located and used to choose the corresponding frame. Of course, based on this there are limitations to the system. These limitations will be discussed further in Section 5.1 .

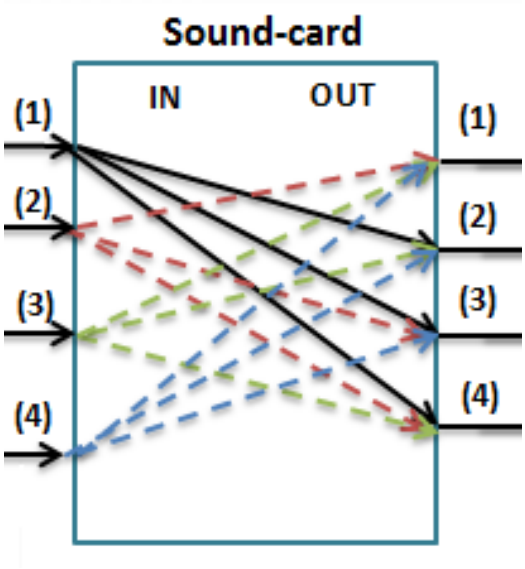


Figure 3.11: Digital routing of signal between nodes in a network

3.2.4 Real-Time Processing

The digital signal processing toolbox (.dsp) in MATLAB allows for accessing audio data in real-time. To enable the real-time computing using MATLAB, a recorder- and a speaker-object must first be created. These objects have options such as frame size, number of input/output channels on the sound card, buffer size and sampling frequency, to modify in making them behave the way necessary or preferable. Making the right adjustments is essential for the system performance, especially with respect to latency and jitter. It is crucial to enable ASIO[1] in MATLAB before running the system.

When the adjustments for the speaker- and recorder-objects have been tuned to meet the system requirements, the real-time fetching of data can begin. The amount of data is given by the frame size and number of sound card channels (nodes). Once a frame of data is collected, it is represented as a matrix in MATLAB with dimensions $frame\ size \times nodes$.

Figure 3.11 shows how the routing is done for four nodes, where the left side of the sound card shows four input signals from the nodes, and the right side shows the four output signals. Here, we see that the signal from Node 1 is received at Node 2, 3 and 4, the signal from Node 2 is received at Node 1, 3 and 4, and so on. Each of the input signals is convolved with its corresponding channel impulse responses. On each output channel there are, in this example, three contributions form the three other nodes which is summed up before transmitting the output signal.

[1]ASIO - an audio stream input/output protocol

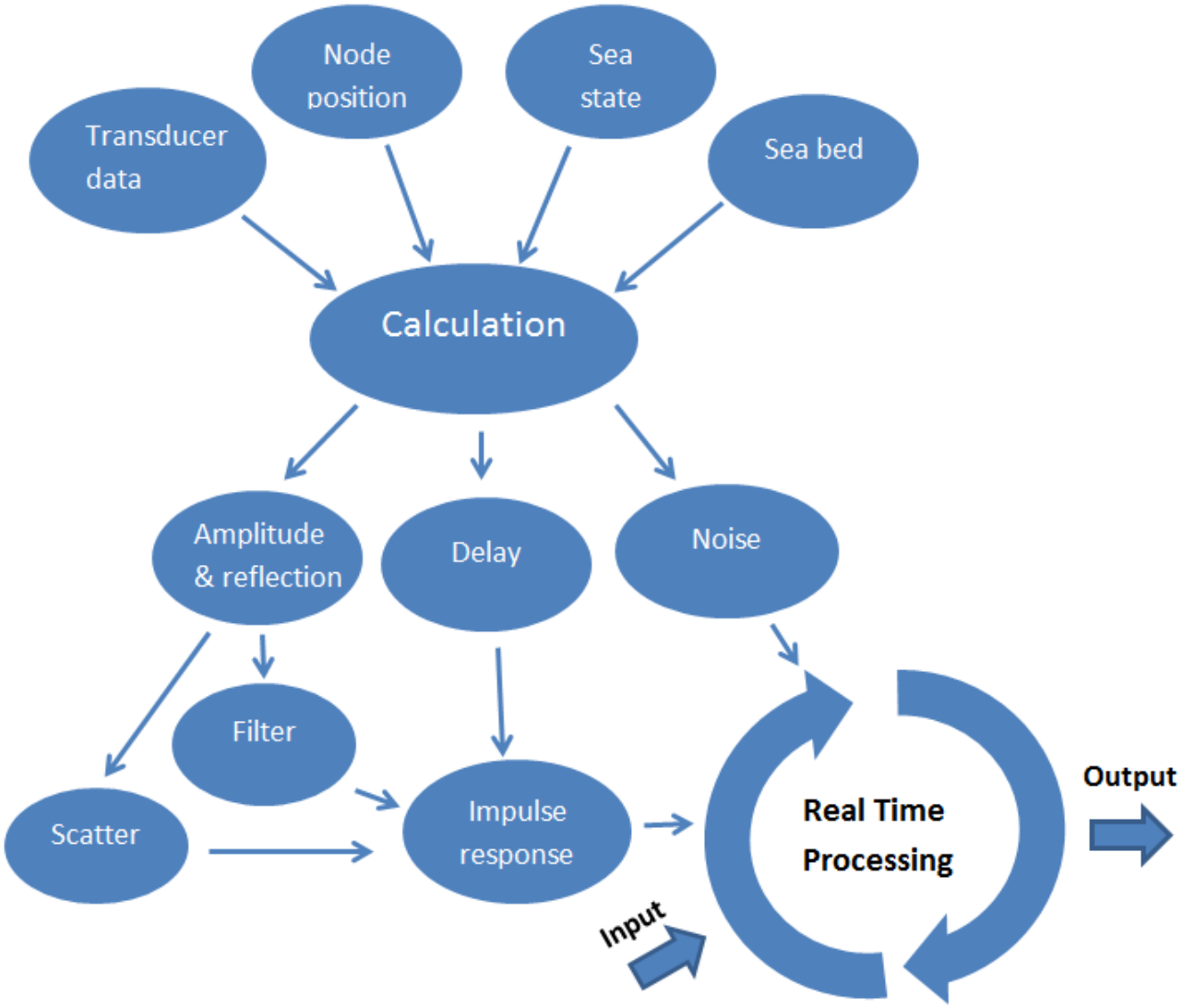


Figure 3.12: Flow chart of HACE. There are only three elements inserted into the real-time process: impulse response, noise and input signal.

Figure 3.12 shows an overview of the system, and how it is built up. It can be seen that the initial conditions are: node positions, sea state, seabed and transducer data. These properties are then used to calculate the arrivals with reflections and amplitude, how long the propagation delay is, and the noise is created with the correct level according to the sea state. Afterwards every calculation is done, the impulse response is created and made ready for real-time processing together with the noise. How the propagation delay and sound pressure calibration is done can be seen in Appendix C.1 and C.2, respectively.

Algorithm 1 shows how the real-time processing is done. As the pseudocode shows, most of the calculations occurs before entering the real-time loop of the system. When the program enters the real-time loop, a frame of the sound signal are read from the input channels from the sound card, named "inputSignal." The input signal is then convolved with the corresponding impulse response ("impulseResponse"), and lastly transmitted through the output of the right channel on the sound card ("outputSignal"). As can be seen from the algorithm, the loops are adaptive with respect to number of devices or nodes in the simulations.

```
Data: In signal from the sound card
create speaker and microphone objects;
find distances/propagation delay and transducer directivity;
create water absorption filters;
calculate noise, doppler spread, surface scatter, 3D network multipath channels, 3D seabed ;
calibrate sound pressure;
add all models in a large impulse response matrix;
while in real-time do
    fetch inputSignal;
    for i = 1 to number of nodes do
        for j = 1 to number of nodes do
            if i ≠ j then
                temp<- convolution [ impulseResponse[j], inputSignal[i] ];
                outputSignal[j]<- outputSignal[j] + temp;
            end
        end
    end
    outputSignal <- noise + outputSignal;
    transmit outputSignal;
end
```

Algorithm 1: Pseudocode for communication in real-time

3.2.5 Latency calibration

To obtain full control over the system latency, but also to be adaptive with changing conditions in form of different hardware or software in use, a method of acquiring the latency was needed. The latency calibration is performed by connecting a Jack-cable from one input channel on the sound card to a different output channel. By running the calibration, test signals are created and transmitted through the output channel. Both the output signals and the input signals are recorded and compared.

The test signal consists of peaks separated in time. These peaks are then compared to the recorded signal, where the difference in time delay between peaks are the latency as can be seen in Figure 3.13. Since the frame size is directionally proportional to the latency, the calibration loops over a set of different frame sizes. The calibration returns whether or not it was successful. When the detections have gone according to the algorithm, the two peaks are compared, and the latency in time are found. From the time and sampling rate, the latency in samples are calculated for all the frame sizes. The number of samples can then be used directly in estimating which frame size to use for a certain setup. Figure 3.13 shows a plot of the test signal and the recorded signal, where the separating distance is the system latency.

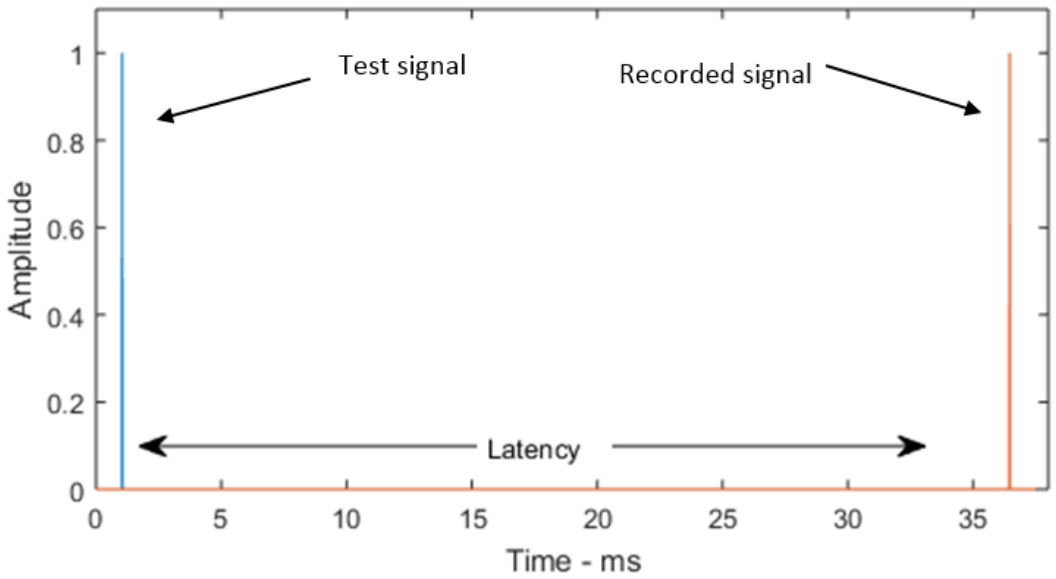


Figure 3.13: Shows two peaks, one for the original signal and one for the recorded signal. Peaks are compared, and latency is measured as the separating distance

3.3 Sea trail

A field trip to Horten was made 15.04.2015 where the goal of the trip was to measure the underwater channel for point-to-point communication in a real world scenario. A transponder was located on the ocean floor and was programmed to transmit a 30 seconds long PN-sequence every 5 min to measure the channel impulse response. See Appendix C.4 for more information about PN-sequence. An ITC hydrophone with a preamplifier was used to measure the received signal from the transponder. The analog to digital converting equipment was the sound card OCTA-CAPTURE with a sampling frequency at 96 kHz, and the signal was recorded using *Audiacity*[1].

Before heading out to sea it was required to test how the equipment behaved, and to tune the settings to get the best measurements. The hydrophone and the transducer were placed 200 meters apart, 2 meters below the surface. The recording showed that the signal amplitude was low, and in order to get better results higher signal amplitudes are preferable. A way to adjust for low amplitudes is to increase the sensitivity on the sound card. But, increasing the sensitivity can lead to information loss if the recorded signal is clipped. The sound card's sensitivity was carefully adjusted so that the signal did not get clipped. At a distance of 200 meters the signal was not clipped for any of the sensitivity increases. So, the sensitivity was set at maximum, at 50 dB gain. Figure 3.14 shows the transponder being submerged at the docks for checking if the equipment worked according to the wanted setup.

Figure 3.14: Thor Husøy deploying the transponder at the harbor.

The channel measurements were performed in Breiangen, a location with relatively flat seabed consisting of clay. Figure 3.17b show the map over the area, where location 1 is the location of the deployed transponder and the distance between Location 1 and Location 2 is where the measurements were conducted. At the two locations, 1 and 2, the sound speed was also measured. From

[1]Audacity - an open source digital audio editor and recorder software

location 1 channel measurements were performed at distances: 0, 500, 1000, 1500, 2000, 2500, 3000 meter, ending up at location 2.

To check if it was possible to get a signal from the transponder, a dunking transducer from Kongsberg Maritime was used. It was lowered into the sea, together with the hydrophone, through the window, as shown in Figure 3.16a, to around 20 meters below the surface. At every measuring spots, two signal periods were recorded. A picture of the boat, Kongsberg Maritime's *Simrad Echo*, used in the sea trial can be seen in Figure 3.17a.

Figure 3.15 shows a drawing of how the setup looked, with the transponder located at the seabed and the hydrophone measuring received signal at the boat. The arrows represents different ray paths.

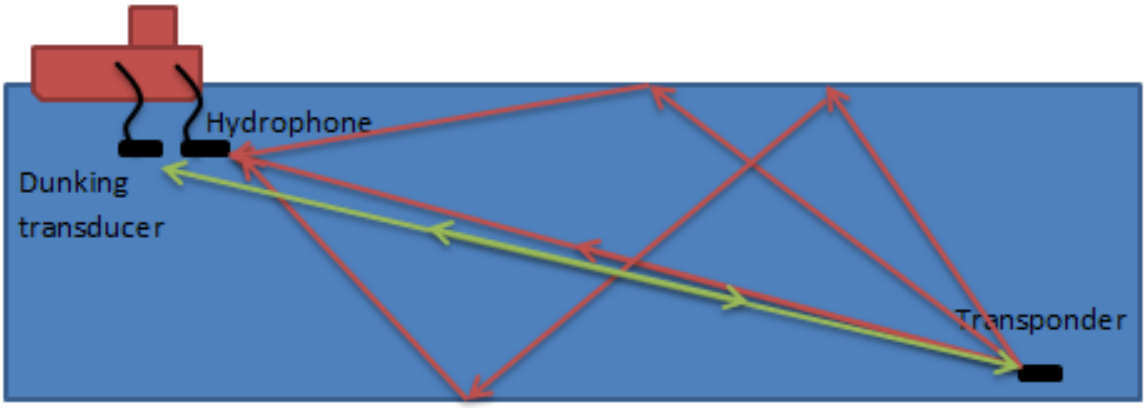


Figure 3.15: Ray paths with arrows between the hydrophone/dunking transducer and transponder

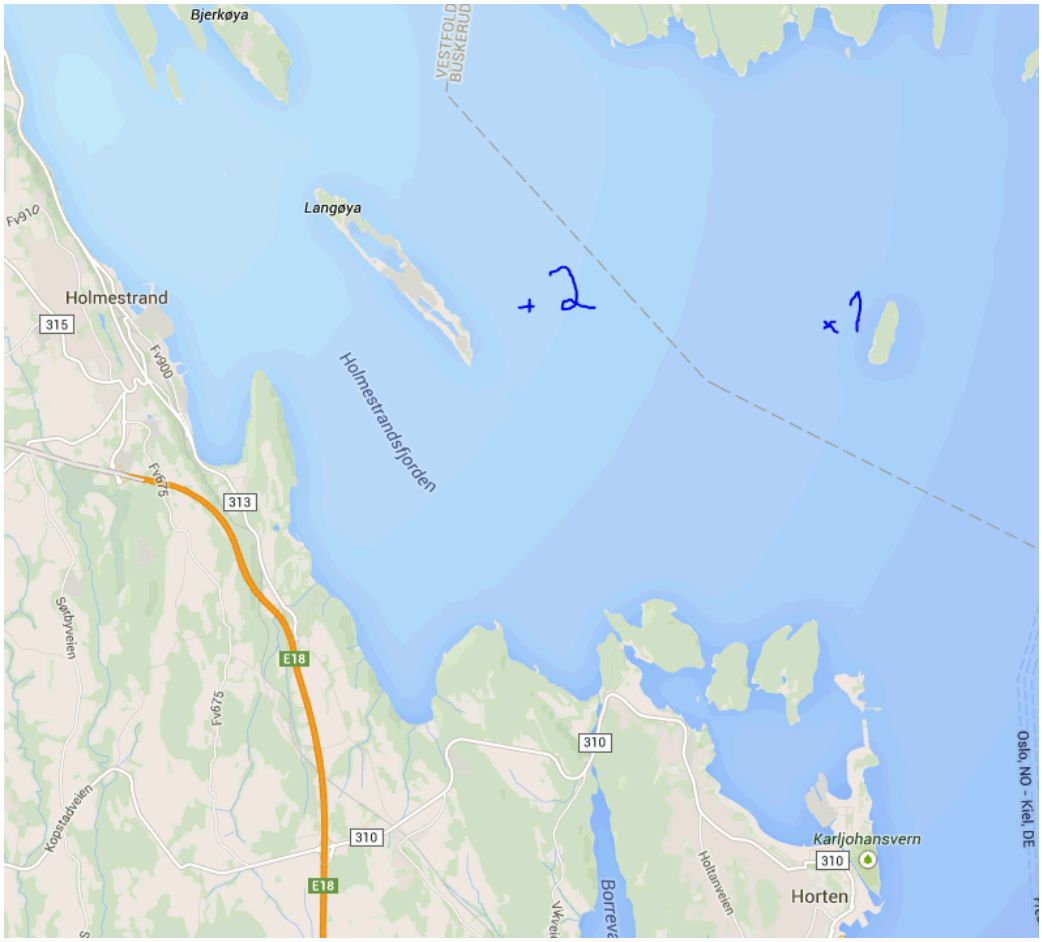
Figure 3.16a shows a picture of the hydrophone placed through the window on the side of the boat and it was submerged to around 20 meters below the surface. The hydrophone cable was fastened to the hooks on the window, and connected to a computer via the sound card to enable recording of signals. In Figure 3.16b a picture of transponders mounted in a basket is shown. The basket was then submerged and placed on the seabed at location 1 in the map in Figure 3.17b. List of equipment used during the sea trail is found in Appendix A.1

(a) A picture of the window from inside the boat. The hydrophone is placed out through a window, and connected to a computer via the sound card.

(b) Types of transponders, in this experiment only one is used

Figure 3.16

(a) Boat: Simrad Echo



(b) Map of the area, point 1 marks the start while point 2 markes the end point

Figure 3.17

3.4 Hydroacoustic Channel Emulator Settings

How the real world channel measurements were performed with hydroacoustic equipment was described in Section 3.3. In this section a list of the settings inserted into HACE for the two nodes are given. The list is used in HACE to compare the simulated channel impulse response to the real world (See Appendix D.1 for the manual for using HACE). Since the seabed in the measurements was relatively flat, a flat seabed was used in the simulations, seabed parameters was set to a density of 2 g/cm^3 and a sound speed of 2000 m/s corresponding to clay, and sea state 1. To get the most correct results, the measured speed profile from the sea trial was inserted into the simulation. In table 3.1 the positions inserted into HACE are given. Node 1 is the transponder located at the seabed, while node 2 is the measuring location near the surface.

- sea state: 1
- sea bed shape: flat
- sea bed: Clay with parameters
  - density: 2.0 g/cm^3
  - sound speed: 2000 m/s

Table 3.1: Simulation locations given in Cartesian coordinates for the two nodes, node 1 is the transponder, while node 2 is the hydrophone.

Order	Node	x-direction	y-direction	z-direction
	1	0 [m]	0 [m]	180 [m]
1	2	10 [m]	0 [m]	20[m]
2	2	500 [m]	0 [m]	20 [m]
3	2	1000 [m]	0 [m]	20 [m]
4	2	1500 [m]	0 [m]	20 [m]
5	2	2000 [m]	0 [m]	20 [m]
6	2	2500 [m]	0 [m]	20 [m]
7	2	3000 [m]	0 [m]	20 [m]

Chapter 4

Measurements and System Performance

In this chapter, system performance and results of HACE is shown with respect to the objectives stated in Chapter 1. This chapter will show the features and the possibilities of the software, verify the results using APOS and compare the results with the real world measurements. A more detailed user manual of the HACE software is found in Appendix D.1.

To show the diversity of HACE, measurements have been done for a large diversity of ranges, where the different ranges varies from 0 to 3000 meters. All the distances are shown in table 3.1, except for the 100 meter case where the two nodes were 20 meters below the surface and separated by a horizontal distance of 100 meters. More detailed results concerning calibration, stability and filter is found in Appendix E.

4.1 Graphical User Interface

The *Graphical User Interface*, in short GUI, for HACE was designed and created with the focus to be user friendly, but also to have the ability to detect errors in different parts of the code during development. A further explanation of GUI is located in Appendix C.3.

Many of the acoustical models can be enabled and disabled via check boxes in the GUI, as seen in Figure 4.6. From the developers' point of view, the ability to run without certain parts of the system have been beneficial, especially with respect to error handling. Running separate parts isolated allows for studying the effects of certain acoustical model, and to see how these models effect the signal.

This ability to run isolated parts of the system can be beneficial for the user as well, for instance, it allows the user to test hydroacoustic equipment under minimal conditions, or with one or two acoustical models present. Or, as a final step, run all together. By adding one or two models, the

user can by himself locate under what circumstances the equipment fails.

From the user friendly point of view, it can be said most of the options are intuitive and plots appear to visualize what happens when settings are changed. The different options have been separated into sections in the GUI. Each section is given a header telling what they control. More on the separate parts later in Section 4.2. What the GUI controls and simulates in real-time, a look at Figure 4.1 can be of help. A view of the GUI is seen in Figure 4.2. The adjustable and attachable parameters in the GUI are:

- Node positions
- Ocean depth
- Multipath or direct path.
- Salinity
- Temperature
- Type of seabed material with density and sound speed
- Ambient noise from sea state/ wind speed
- Carrier frequency
- Transducer data
- Transducer angle re horizontal axis
- Loading speed profiles
- Record and save signals
- Shape of the seabed, with the possibility of choosing from a preset selection or draw your own
- Doppler spread
- Plot and analyze the results
- Sound card
- Sampling frequency
- Latency calibration
- Start/Stop the simulator

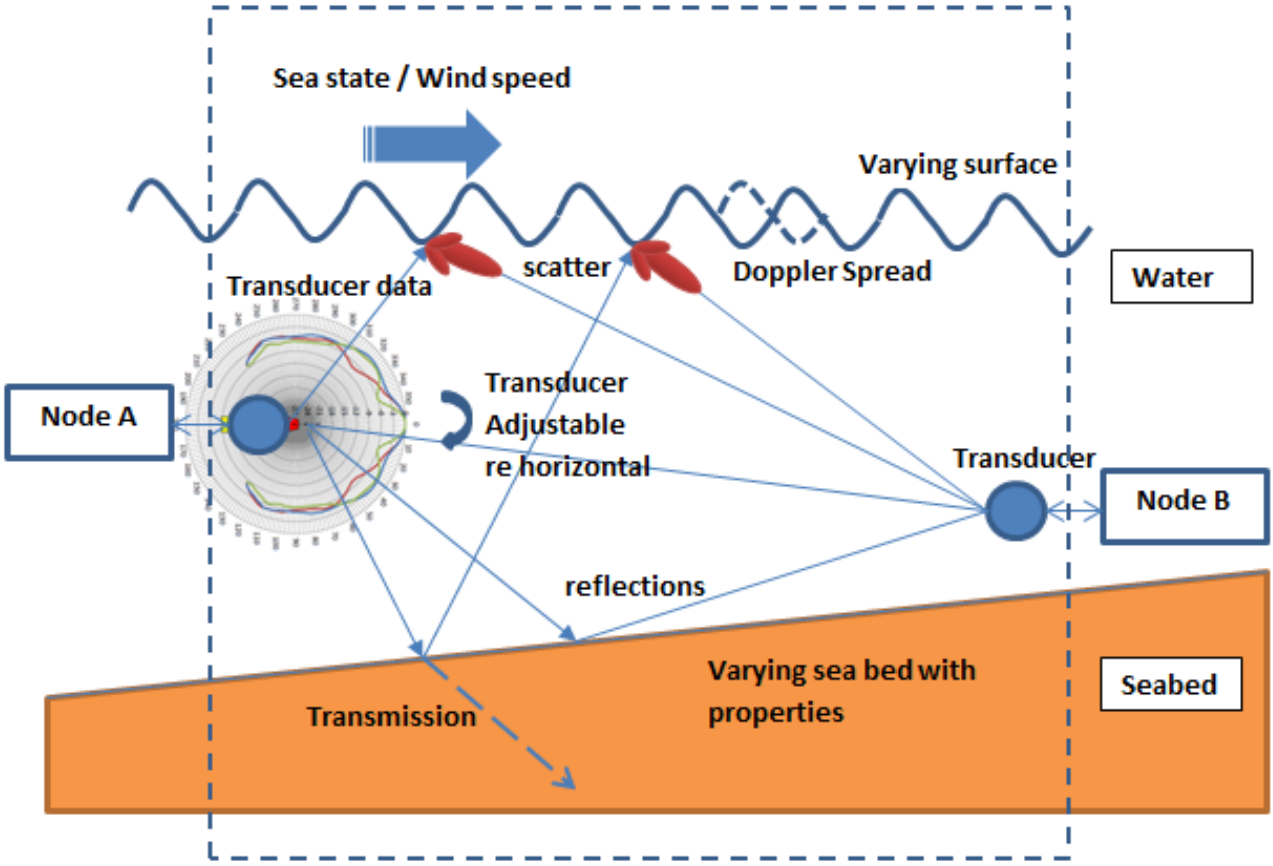


Figure 4.1: Overview of the controllable parameters in HACE.

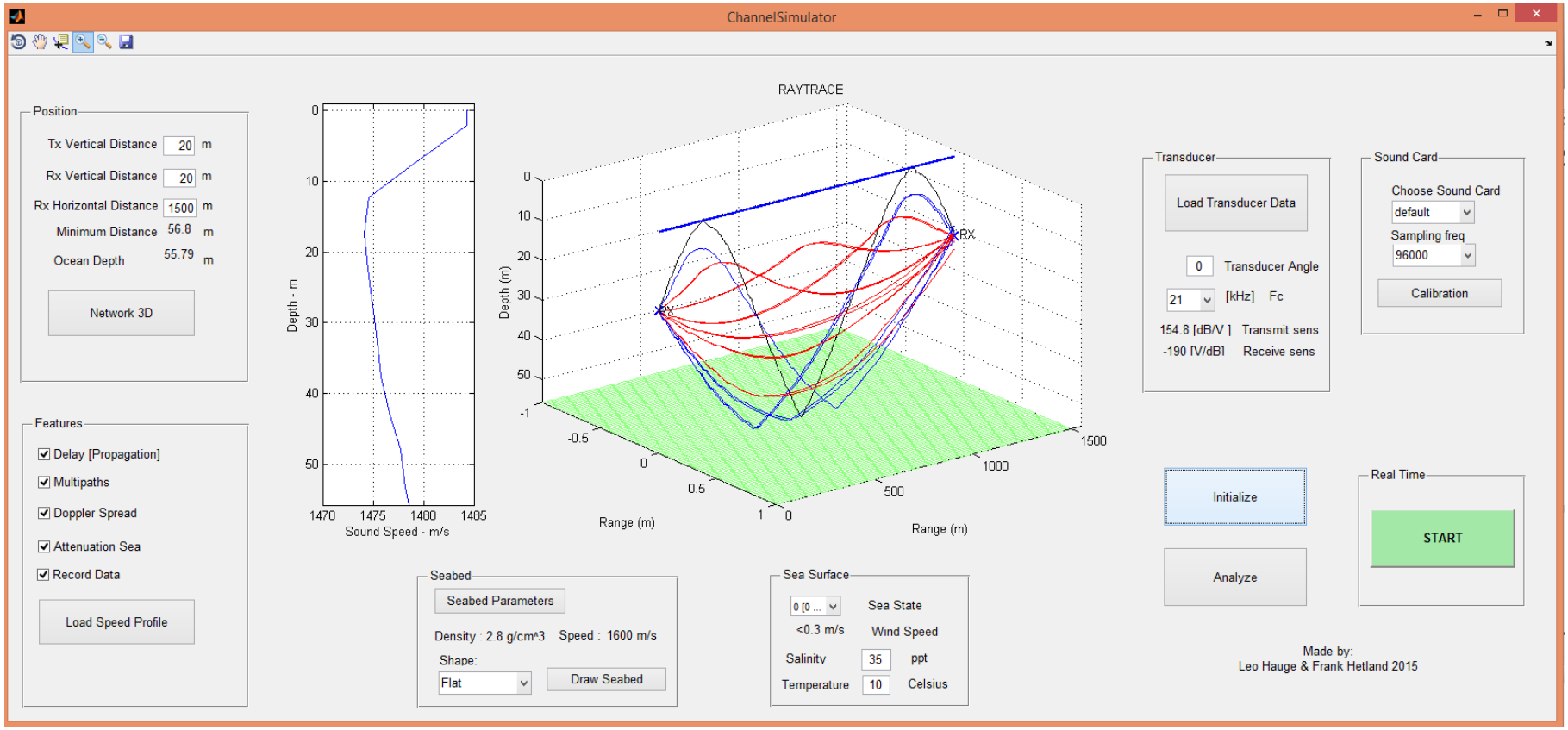


Figure 4.2: HACE's GUI. A ray trace for point-to-point communication is seen in the middle of the window, with the corresponding sound speed profile at the left side. Surrounding the window is several sections for adjusting and adding the acoustic features, and choose the different options.

4.2 Settings and Features

As described earlier, in Section 4.1, HACE support several features and adjustable parameters. In this section an introduction to these features is described. This includes how to set and initialize the sound card, calibrate the system latency, sampling frequency, transducer properties, sea bed and surface, point-to-point and network communication, start/stop the system, how to enable the acoustical features and analyze them.

Sound Card

At the upper right hand side of the GUI in Figure 4.2, “Sound card” section is placed. There are two different pull-down menus, one for choosing the sound card and one for deciding the sampling frequency. A "calibration" button is also located here. The “sound card” section is displayed in Figure 4.3.

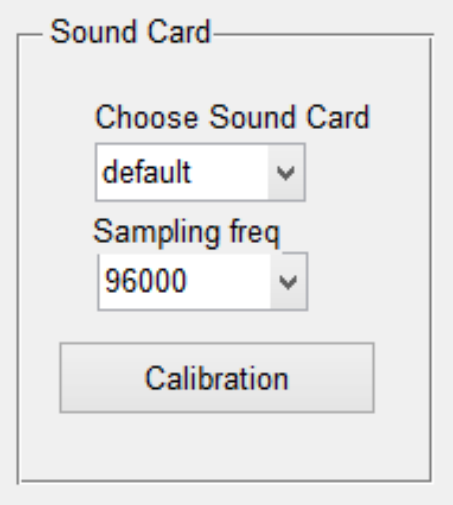


Figure 4.3: Sound card section for setting the sound card options

In the first pull-down it is possible to choose what sound card to use. There are three different options to select from, “default”, “OCTA-CAPTURE” and “UA-25”, where the two last are sound cards from Roland. The two sound cards have both been used during development of HACE and have been implemented into the software. To use other sound cards they must be added in the code, an explanation of how to add other sound cards are found in Appendix D.1. The default option is actually the computer’s chosen sound card, and should work for all other sound cards as well, if the correct settings have been switched on. From measurements, the default option have proven to be a little slower and unstable with respect to latency and jitter compared to enabling the correct sound card.

The second pull-down menu is where the sampling frequency is selected, currently there are 4 different choices. The 4 choices are 44.1, 48, 96 and 192 kHz. The choice of sample rate is given by the frequency of the input signal from the user. We have from Section 2.9 the Shannon’s sampling theorem that says the sampling frequency must be at least two times the highest signal frequency, thus, a sample rate of 96 kHz is sufficient in the frequency range from 21 to 31 kHz. Located below the two pull-down menus a “Calibration” button is located. Pressing this button opens a new

window, seen in Figure 4.4.

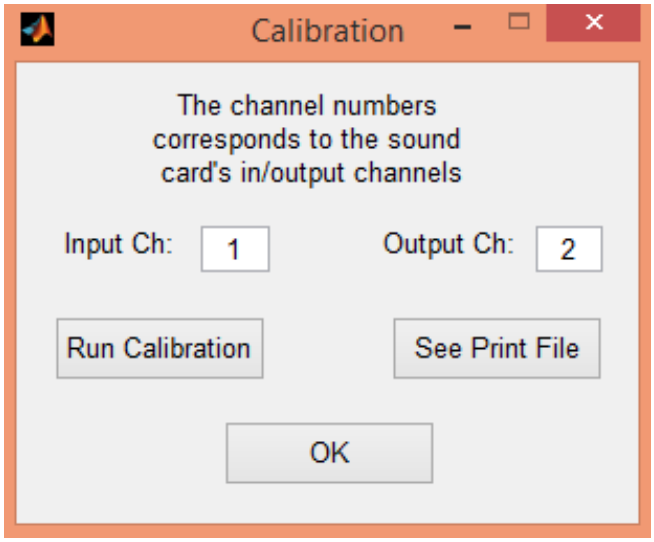


Figure 4.4: GUI window for running the latency calibration

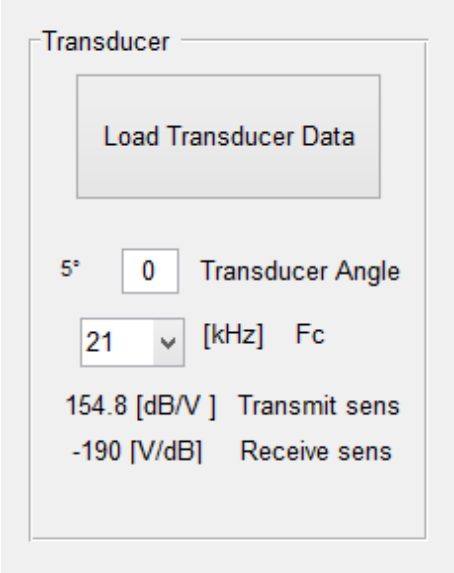
The calibration measures the system latency for a set of different frame sizes. In Appendix E.1 and Section 4.5 statistics of 100 calibrations are shown. To run the calibration, a Jack cable must be connected to the sound card from one input channel to one output channel. There are two input boxes for choosing where the input and output channels the cable are connected. Afterwards the correct numbers are inserted, the calibration begins by pressing the "Run Calibration" button, and it stats to run according to the description in Subsection 3.2.5. During the latency calibration, a wait bar appears to illustrate the progress of the simulation. It takes about 2 minutes to finish the calibration.

On the right hand side of the "Run Calibration" there is a button called "See Print File". The button opens the text document where the results from the latency are given. These calibrated values are then being used in HACE to create exact range simulations. In Appendix E.2 the calibrated results from one calibration is shown. 9 different frame sizes have been used (1024, 1536, 2048, 4096, 6144, 8192, 16384, 32768, 65536 samples), and it is easy to see from the results in appendix that the latency is increasing as a function of frame size.

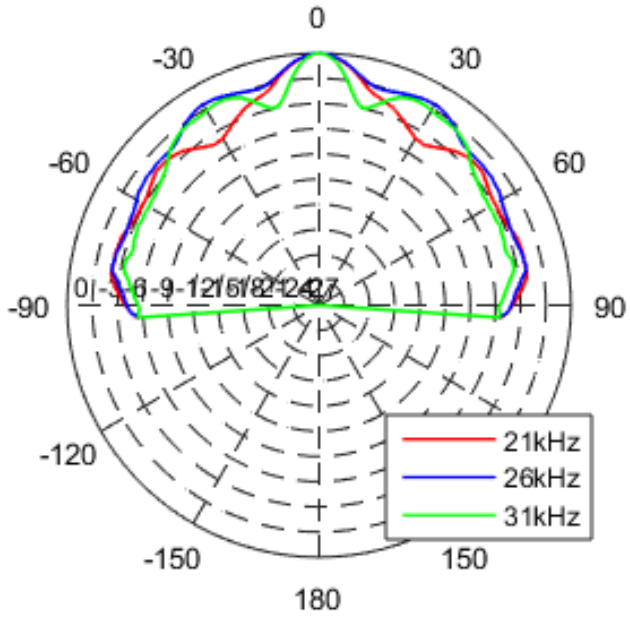
Transducer

Located next to "Sound Card" is the "Transducer" section as seen in Figure 4.5a. This section contains three options for the user to modify "Load Transducer Data", "Transducer Angle" and "Fc". Located at the top is the "Load Transducer Data" button. This button makes loading transducer properties possible, i.e. loading directivity and sensitivity (both receiver and transmit). When the transducer data has been added, a plot of the directivity appears, and the receive and transmit sensitivity are displayed together with an angle. In the figure the angle is 5°, transmit sensitivity is 154.8 [dB/V] while the receive sensitivity is -190 [V/dB].

The angle is limited to point-to-point communication and it shows the direction of the main



(a) Transducer properties for loading beam pattern, carrier frequency, angle at the transducer. The transmit and receive sensitivity shown is the loaded transducer data for 21 kHz.



(b) Directivity plot of the transducer, carrier frequencies of 21, 26 and 31 kHz.

Figure 4.5

node with respect to the horizontal plane and the location of the second node. It is calculated from the *line of sight*[1] between the two nodes. Next to this calculated angle an input box is found. The purpose of the box is to be able to adjust the direction of the main node, where all angles between -90° and 90° are valid inputs. If the input angle in Figure 4.5a is set at 5° the main node will adjust itself to have the main direction facing the second node. As seen in Figure 4.5b all beam patterns have their main direction with maximum gain at 0° and are symmetric on both sides.

Selecting which of the transducer data to use is done by choosing the carrier frequency Fc since the directivity and sensitivity are functions of frequency. In Figure 4.5a the selected frequency is 21 kHz. An overview of the data used is found in Appendix F.3. In the data sheet, there are values for three different frequencies 21, 26 and 31 kHz. Polar plots of the transducer's beam patterns for the three frequencies can be seen in Figure 4.5b.

[1] Line of Sight - a straight line between two points

Features

Located at the lower left side in the main GUI window "Features" is found. In this section the user have the possibility of adding or excluding certain acoustic properties. The different choices are “Delay [Propagation],” “Multipaths,” “Doppler Spread,” “Attenuation Sea,” and “Record Data”, these are showed in Figure 4.6. All the options in the figure are enabled, seen by the check mark. Located below is a “Load Speed Profile” button, by pushing it, speed profiles can be inserted into the system. There are several predefined profiles to choose from, alternatively, the user can create their own profiles. How to make a speed profile are found in Appendix D.1.

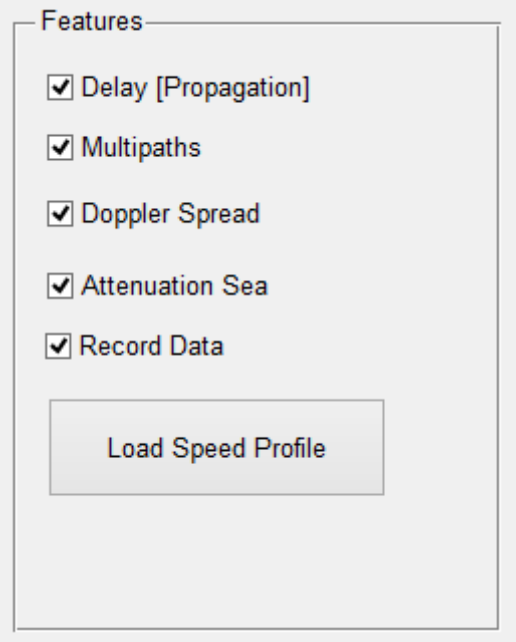


Figure 4.6: Feature section to enable or disable the acoustic features

Disabling the “Delay [Propagation]” means that the simulations are run in another subroutine where the signals does not encounter any of the acoustical models. What happens is that the signals are routed according to Section 3.2.4, no propagation delay is added, and the only delay present is the system latency. If “Multipaths” is enabled, the calculated rays are used, otherwise only one direct transmission is taken into account. “Doppler Spread” enables or disables a time varying channel, and “Attenuation Sea” decides whether or not the water absorption from Section 2.3 is added. It is also possible to record the signals by checking on the “Record Data” box.

Analyze

The recordings can be analyzed with the use of the “Analyze” key in the GUI, shown in Figure 4.2. A new window opens up and is displayed in Figure 4.7. The analyze window is separated into two parts, where the plotting options are located on the left side, and the playback section is located on the right side. This element of being able to analyze recordings have been of great help during development. Time continuous frequency spectrum of the signals can be seen by loading the recorded file and running it. In Appendix **??** the different plots from this window are shown.

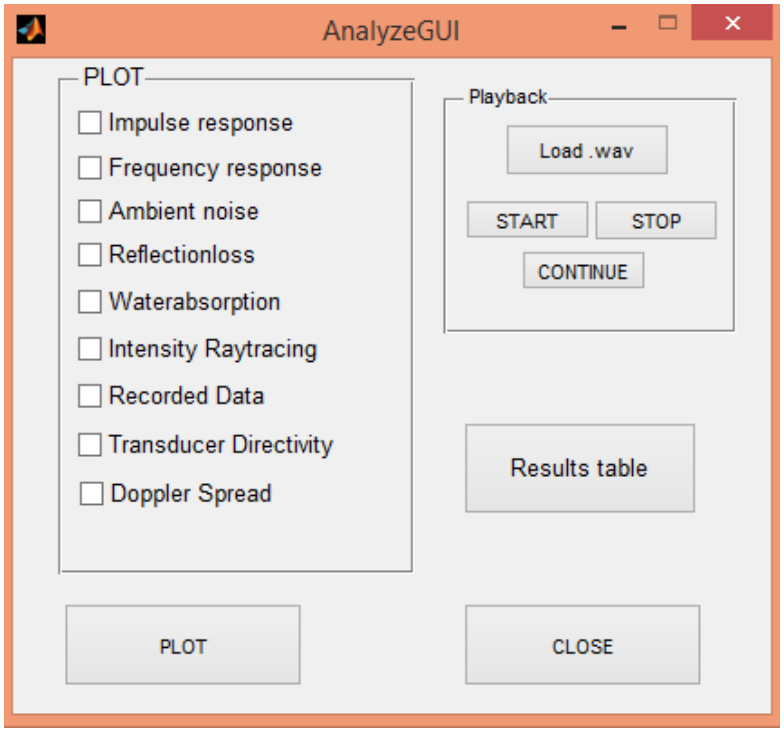


Figure 4.7: Analyze section with options for plotting, playback and show the results in a table

Surface and Seabed

Two modules are located in the lower midsection of the main GUI window named “Seabed” and “Sea Surface”, shown in Figure 4.8. In these two sections the adjustable parameters for the seabed and the surface are accessed. The possible changes for the seabed is shape and substance, where the parameters for density and sound speed are accessed by pressing "Seabed Parameters". When the button is pressed a new window opens, and shown in Figure 4.9. The new window contains a list of typical compositions of seabed sediments, with their corresponding sound speeds and densities, input boxes and a plot of the transmission and reflection coefficients as functions of grazing angle. Shaping the seabed can be done in two ways, where one is to choose from the set of predefined shapes, and the other to draw your own shape. More information about these settings is located in Appendix D.1.

Sea state, salinity and temperature is decided in the “Sea Surface” section. The sea state level are from 0 to 9, where the 9 is the maximum. A sea state of 9 corresponds to surface waves of above 14 meters in height. The wind speed for the selected sea state is given in meters per seconds, and is shown bellow the sea state. Salinity and temperature are used in the acoustic water absorption model described in Section 2.3.

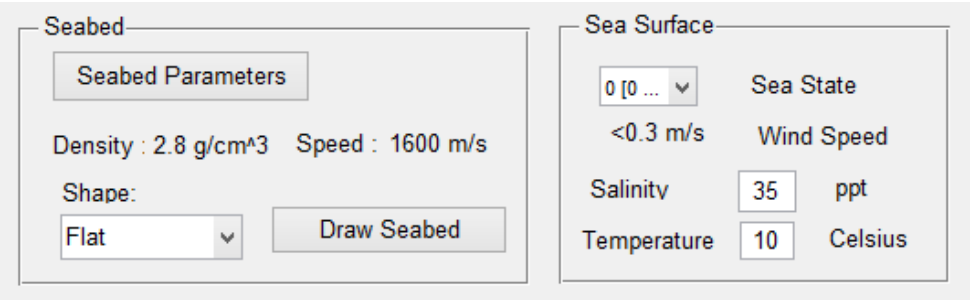


Figure 4.8: To the left the seabed settings is found, while to the right the sea surface settings are. Insertion boxes allows the user to insert the wanted values.

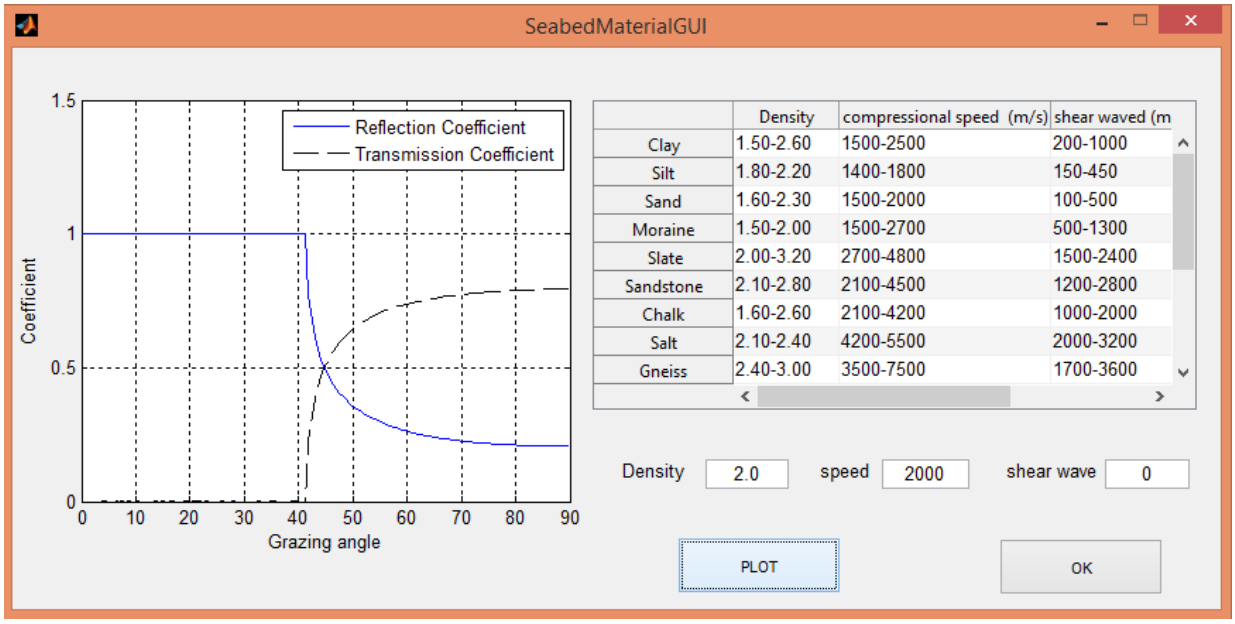


Figure 4.9: Seabed parameters are are shown. The list shows examples of seabed properties for a variety of different compositions. A plot of the reflection and transmission coefficients of the seabed is shown, here the coeffisients for clay is shown.

Positions

At the left hand side in the GUI, shown in Figure 4.10, "Position" is located. As indicated by the name of this section, this is the part where node positions are inserted. The three insertion boxes are used for point-to-point communication, where the possible inputs are vertical positions of the two nodes and horizontal separating distance. Due to the latency described in Section 2.9 there are limits with simulation distances, thus there must be a minimum distance between adjacent nodes. This spacial separation is given and indicated by “Minimum Distance”, in Figure 4.10 the minimum distance is 56.8 meters. Located below is information of the ocean depth given, which is the maximum depth of the speed profile.

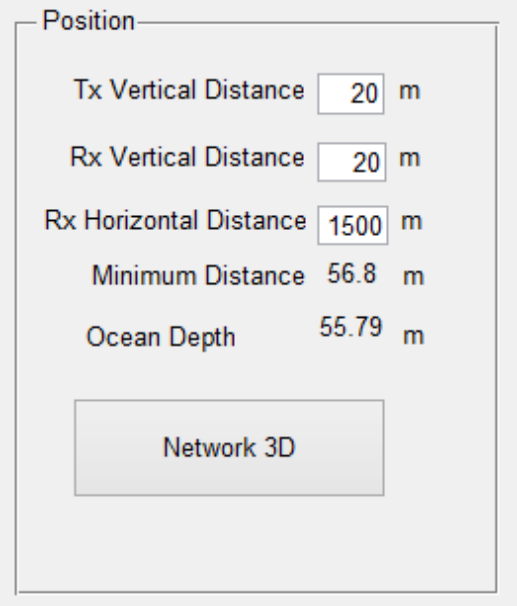


Figure 4.10: GUI section for inserting node positions in Cartesian coordinates

The "Network 3D" opens a new window with the possibility of adding several nodes in a 3D network. Figure 4.11 shows the new window where four nodes have been inserted. Positions of the nodes are added using Cartesian coordinates in the table where the Z-coordinate is the depth. The check boxes to the right are used to add or remove the nodes.

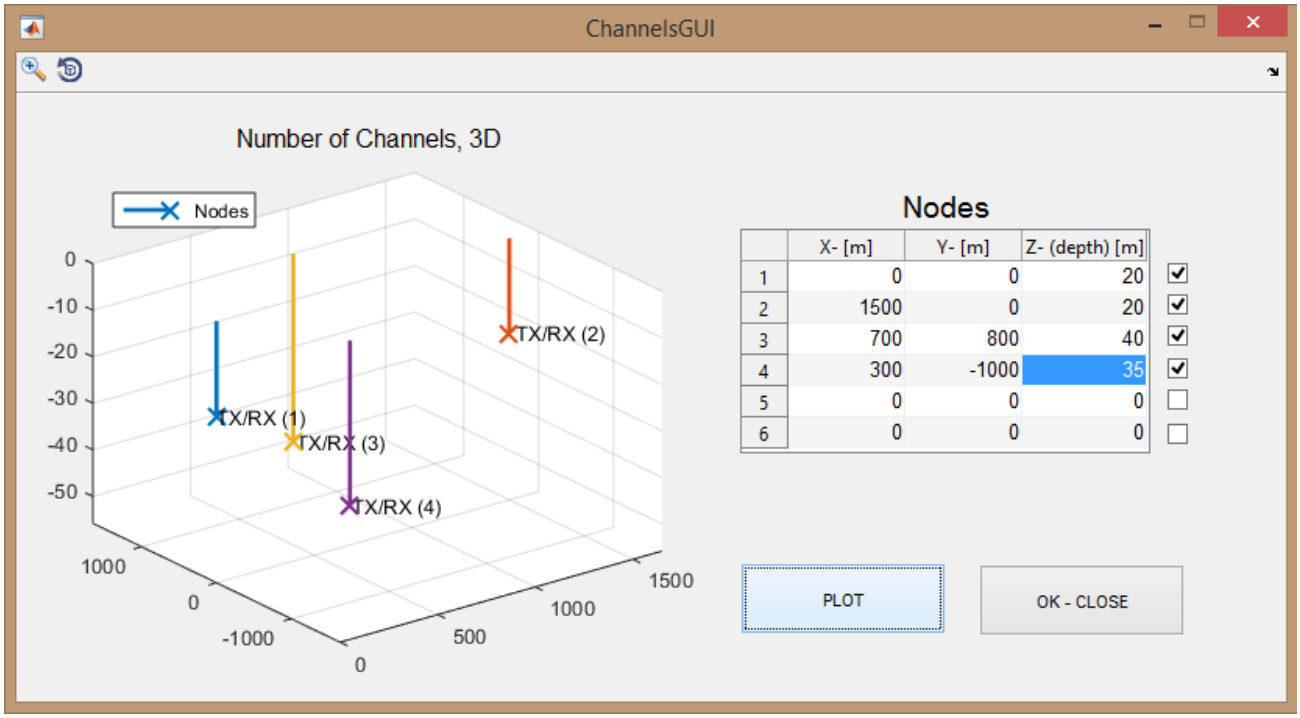


Figure 4.11: Section for inserting the node positions in the network. The positions are given in Cartesian coordinates and the plot button shows placement of the nodes.

Initializing and Running

When the parameters and settings have been decided the acoustic models need to be calculated. By pushing “Initialize” the calculations are performed and the acoustical models are inserted into the channel matrix. A plot of the positions with rays will automatically appear in the main GUI window with the new settings, as shown in Figure 4.2. Plotting gives the user a better view and understanding of the channel. How long the duration of the calculations is, depends primarily on the number of nodes. For any changes to be taken into the simulation, the initialization must be performed.

Once the calculations are finished the real-time simulation can begin. The green “START” button starts the real-time simulation with the initialized settings. When pressed, the “START” button changes appearance to a red “STOP” button. The rea-time simulations stops when the red button is pressed.

For the user to have the ability of running the system without tuning all parameters and settings, HACE has been built with default values. Thus, the user can run the simulations without doing any changes, and as a consequence, save time.

Ray Trace

Figure 4.12 shows an example of a ray trace for point-to-point with a varying seabed, where the seabed has an inverse Gaussian shape. The two nodes are placed 1500 meters apart and 20 meters below the surface. Figure 4.13 shows a similar example of the ray trace, but with four nodes in a network. The four nodes create six different multipath channels, perhaps easiest seen by the blue surface lines. Located in Appendix E.3 are more examples of ray trace plots with several seabed shapes and sea states. These examples have been added to illustrate the diversity of HACE.

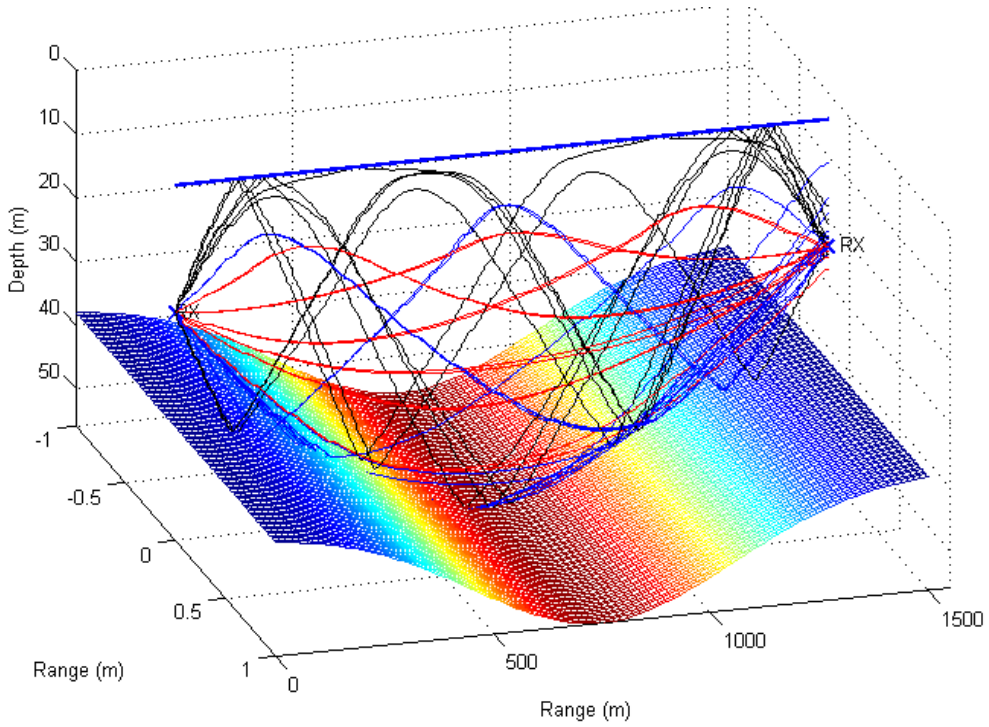


Figure 4.12: Plot of a ray trace for point-to-point communication. The nodes are at a depth of 20 meters and separated by a distance of 1500 meters. The seabed has an inverse Gaussian shape and the blue line at the top is the sea surface.

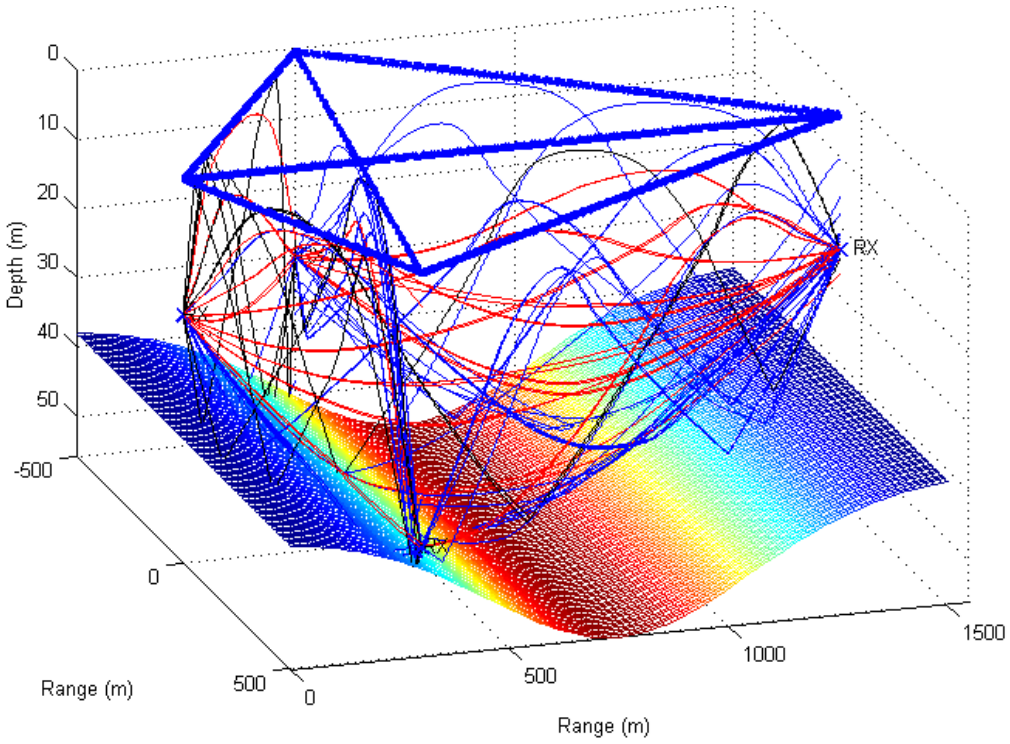


Figure 4.13: Plot of a ray trace for network communication with four nodes. All nodes are placed at a depth of 20 meters and separated with different distances. The seabed has an inverse Gaussian shape and the blue lines at the top are the sea surface.

4.3 Results using APOS

APOS is an acoustic positioning software created by Kongsberg Maritime and is used to operate transducers, measure distances, acquire positions and log data. More information on APOS is found in Appendix A.2. The use of APOS was to verify the performance of HACE, that is to study how well the inserted ranges correspond to the values given by APOS, and whether or not communication through HACE was achieved. Checking the link was done by transmitting a telemetry signal from one node to the next. By using APOS it is possible to measure and log data over time. A way of detecting stability/jitter is to study how the range is changing in time. These jitter results are displayed in Appendix E.6 and in Section 4.5.

Point-to-Point

Table 4.1 shows the results from range measurements with the use of APOS for point-to-point communication. The node positions are given in the table under “Node positions and range”, with the

theoretical calculated range at right side. Under “Measured results APOS” the measured ranges from APOS are shown, along with the deviations between the calculated and the measured ranges, and if telemetry was achieved. The positions of the nodes correspond to the those used during the sea trial, except the 100 meters case. The sound speed was set constant at 1500 m/s, this is to make sure that APOS computes the correct ranges.

Table 4.1: Point-to-point communication and range test results from HACE measured using APOS. The node coordinates show the depth and the horizontal distance.

Node positions and range			Measured results APOS		
Node1 Coordinates [m]	Node2 Coordinates [m]	Range [m]	Measured Dist [m]	Deviation in %	Telemetry
[20,0]	[20,100]	100.00	100.23	0.23	YES
[180,0]	[20,10]	160.31	160.50	0.12	YES
[180,0]	[20,500]	524.98	525.21	0.04	YES
[180,0]	[20,1000]	1012.72	1012.96	0.02	YES
[180,0]	[20,1500]	1508.51	1508.76	0.02	YES
[180,0]	[20,2000]	2006.39	2006.63	0.01	YES

The results in Table 4.1 show that HACE simulate nearly ideal distances, with deviations from 0.01 to 0.23 %. Throughout the results, all measured distances show an offset of ~ 20 cm. HACE has also shown high stability over time, with a jitter of ±1 cm for the different distances measured. These jitter measurements are shown in Appendix E.6. Mentioned earlier, crosstalk can be an issue, Figure E.23 in Appendix E.6 shows this phenomenon where the simulated range of 2000 meters oscillates between 2000 and 1000 meters.

Network

A test of network communication with three nodes was performed. Same as for point-to-point communication, range and telemetry were measured with a constant sound speed of 1500 m/s. The positions and distances are located in Table 4.2 along with the results from APOS. Comparisons with APOS show that HACE has the same small range deviations of 0.01 to 0.02 %, with a constant offset of ~ 20 cm. The plots in Figure 4.15 and 4.16 are plotted from recorded signals in HACE and have been added to illustrate that all nodes are connected to each other, and the signals are distributed throughout the whole network.

Figure 4.14 shows how the communication procedure for a network of three nodes works. Node 1 is the main node connected to APOS. During event 1, Node 1 transmits a signal to Node 2 and 3. In event 2, both Node 2 and 3 responds to the signal. From Node 2 the reply signal is received at

Table 4.2: Results from APOS with network communication using HACE. The node positions are given in Cartesian coordinates where the z-coordinate is the depth. The ranges are numbered according to which of the nodes are connected, that means Node 1 is communicating with Node 2, 1 with 3, and 2 with 3.

Node positions and range (Network)			
Node 1 Coordinates [m]	Node 2 Coordinates [m]	Node 3 Coordinates [m]	Ranges [m] (1-2, 1-3, 2-3)
[0,0,20]	[500,1000,20]	[500,-500,20]	[1118, 707.1,1500]
[0,0,20]	[500,750,20]	[500,-500,20]	[901.4, 707.1,1250]
[0,0,20]	[500,500,20]	[500,-500,20]	[707.1,707.1,1000]

Measured results APOS		
Measured Range [m] (1-2, 1-3, 2-3)	Deviation in %	Telemetry
[1118.3, 707.35, 1500.3]	[0.02, 0.01, 0.02]	[yes,yes,yes]
[901.67, 707.35, 1250.25]	[0.02, 0.01,0.02]	[yes,yes,yes]
[707.37, 707.36, 1000.27]	[0.01,0.01,0.02]	[yes,yes,yes]

both Node 1 and 3, and from Node 3 the reply signal is transmitted to Node 1 and 2. H1, H2 and H3 represent the three multipath channels connecting the nodes.

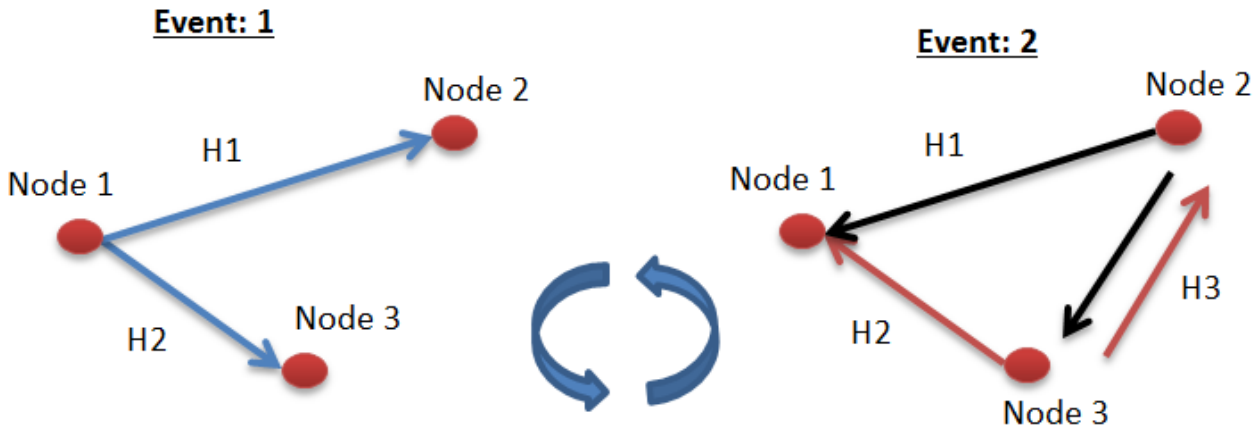


Figure 4.14: Event 1: Node 1 transmits signal to Node 2 and 3. Event 2: Node 2 and 3 responds to the signal from Node 1. Node 2 and 3 responds only on signals from Node 1, therefore the loop.

The nodes are subjected to an extra “Turn-around delay”, which is the time delay the nodes use to receive, decode and respond to incoming signals. This delay has to be subtracted when reading the time difference between arrival times at the nodes, and is by default 125 milliseconds.

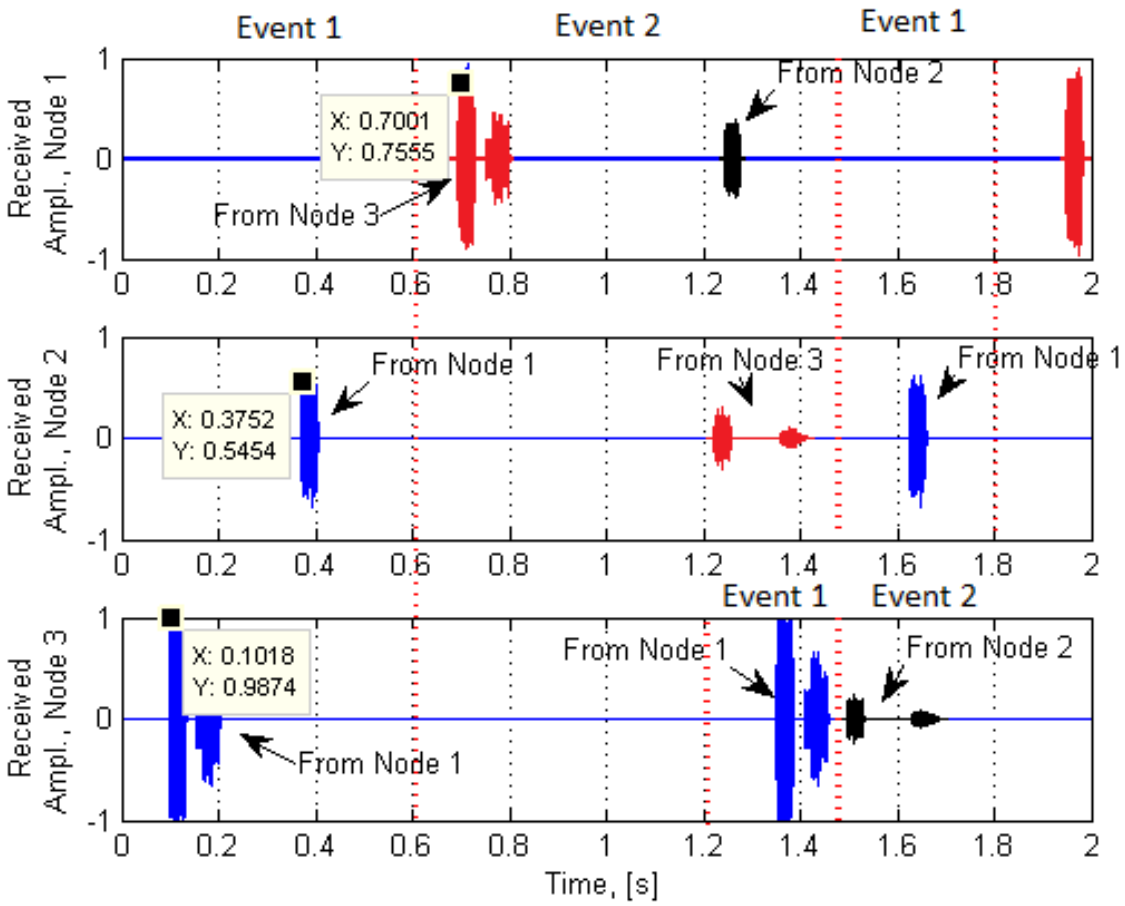


Figure 4.15: Received signals for a three node network (non-overlapping). The signals are color coded where the output signal from Node 1 is blue, the signal from Node 2 is black and the output from Node 3 is red. The node positions and ranges are given in the uppermost line in Table 4.2

Figure 4.15 shows the recorded output signals from HACE during network communication with three nodes. The blue signals are the outputs from Node 1, the black are from Node 2 and the red are the output signals from Node 3. In the figure, the uppermost plot is the received signal at Node 1, in the middle, the received signal at Node 2, and the lower plot is the received signal at Node 3. Between Node 1 and 2 the distance is 1118 meters, between Node 1 and 3, 707.1 meters, and, between Node 2 and 3 the distance is 1500 meters. These distances are shown in Table 4.2 along with the positions of the nodes. The routing of signals was mentioned in Section 3.2.4 and showed in Figure 3.11, a look back might be of help in understanding the procedure.

Table 4.3 shows the time when the signals are received at the nodes during the two events of the three-node-network from the plot in Figure 4.15. First to receive a signal from Node 1, is Node

Table 4.3: The order in when the signals arrive according to Figure 4.15 where the first two arrivals displayed with arbitrary values to correspond to the figure

The received order of 3 nodes in network		
Paths	Delay [ms]	Order
Node 1 to Node 3	101.8	1
Node 1 to Node 2	375.2	2
Node 3 to Node 1	700.1	3
Node 3 to Node 2	1226.8	4
Node 2 to Node 1	1245.8	5
Node 2 to Node 3	1500.8	6

3 seen by the blue signal in the lower plot. Node 2 receives the signal from Node 1 around 274 ms after Node 3 has received it. With a sound speed of 1500 m/s this 274 ms propagation delay results in a distance difference of 274 $ms \times 1500\ m/s = 411\ m$, where the actual difference is supposed to be $1118 - 707.1\ m = 410.9\ m$. When a node is responding to a signal, the 125 ms turn around delay is added. From the table, order number 3 yields 700.1 ms, subtracting the turn around delay, the distance between Node 1 and 3 is $(700.1 - 125 - 101.8)\ ms \times 1500\ m/s = 709\ m$ meaning that Node 3 is around 709 meters away from Node 1. Both Node 2 and 3 reply to the signal. These replies can be seen in the upper plot where the reply from Node 3 is red and from Node 2 is black. The red signal from Node 3 can also be seen arriving at Node 2 in the middle plot, and the black signal from Node 2 can be seen arriving at Node 3 in the lower plot. When Node 1 receives the replies the procedure starts over again.

Figure 4.16 shows another example of the received signals for the three node network. Node 2 and 3 are both placed at a distance of 707.1 meters away from Node 1, and the separating distance between Node 2 and 3 is 1000 meters. The distances are shown in the last row in Table 4.2 along with the positions of the nodes. During the first event, illustrated with the blue color, is the signal from Node 1 seen arriving at Node 2 and 3 at the same time instance after an arbitrary time delay. The two nodes then reply where the replies reach Node 1 at the same time instance (green signal). After around 2 seconds, the signal from Node 2 is received at Node 3 (black signal), and the signal from Node 3 is received at Node 2 (red signal).

Shown in table 4.4 are the different propagation delays for the replies, relative the first arrival from Node 1. The time difference between the received signal from Node 1 and 2 (and 3) is $1807\ ms - 1210\ ms = 597\ ms$, with the turn-around delay of 125 ms and the sound speed of 1500 m/s, the calculated separating distance between Node 1 and 2 (and 3) is $(597 - 125)\ ms \times 1500\ m/s = 708\ m$. Between Node 2 and 3 the calculated distance is $(2000 - 1210 - 125)\ ms \times 1500\ m/s = 997.5\ m$.

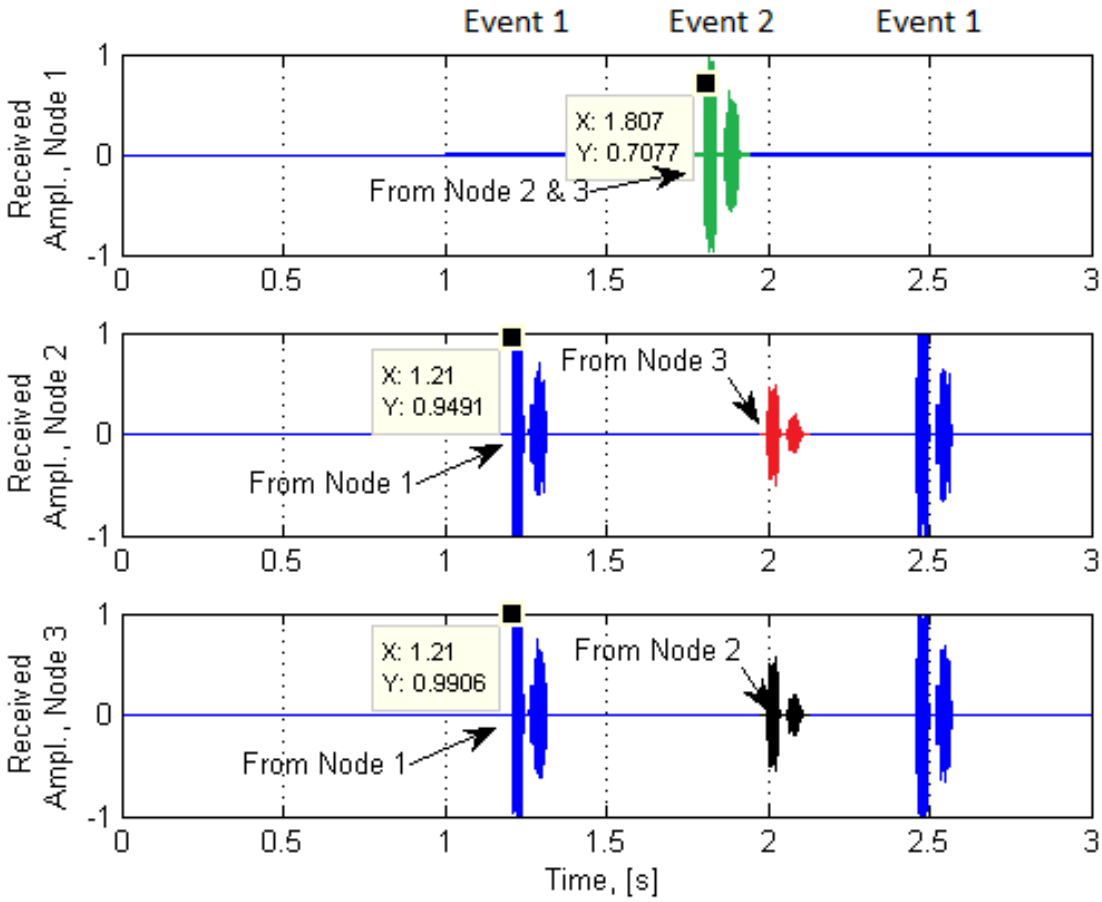


Figure 4.16: Time view of three output channels, recorded by using the HACE. The path distances for Node 2 and 3 is the same, case 3 from table 4.2, as a result, Node 1 receives the signal from Node 2 and 3 simultaneously, marked here as green.

All received signals in figure 4.15 and 4.16 consist of reflections and direct transmissions. Since, in the simulations the depth of the nodes are 20 meters, while the separating distances are above 700 meters with constant sound speed, the surface reflections arrive very close in time to the direct arrival. The first surface reflections are then overlapping with the direct transmissions, ending up looking like only one signal. Between Node 1 and 3, and 2 and 3 in Figure 4.15, there are clear later reflections having longer paths.

Locating the values of the arrival times were done by studying the plot resulting in lowered precision of the calculated distances. In the figures and tables, the first arriving signals are shown and plotted with arbitrary propagation delays. The time delays for the first arrivals are used as the reference values for the later arrivals.

Table 4.4: Arrival time for signals in the three node network. The order in when the signals arrive according to Figure 4.16 where the first two arrivals displayed with arbitrary values to correspond to the figure

The received order of 3 nodes in network		
Paths	Delay [ms]	Order
Node 1 to Node 3	1210	1
Node 1 to Node 2	1210	2
Node 3 to Node 1	1807	3
Node 2 to Node 1	1807	4
Node 3 to Node 2	2000	5
Node 2 to Node 3	2000	6

4.4 Results of Real World versus HACE

In this section the point-to-point measurements from the sea trial in Horten (Breiangen) are presented and compared with results from HACE. The first measuring point was located approximately right above the submerged transducer. From this first location, the next measuring locations were spread out every 500 meter in the horizontal range from the node on the seabed, up to 2500 meters. An impulse response measurement was also taken at a horizontal range of 3000 meters, but the noise was too demanding to get any valuable information. The measurements exploit the PN-sequence to compute the impulse response, mentioned in Section 3.3.

In HACE, Node 1 is placed 180 meters below the surface, on the seabed, and the Node 2 is submerged to 20 meters. The total depth comes from the measured depth at the sea trial and was found to be 185.48 meters. To avoid numerical problems in the simulation the first measuring point was placed 10 meters apart in the horizontal direction, while the next were spread out every 500 meters up to 2500 meters corresponding to the sea trial measurements. Table 4.5 show the positions and order to the nodes during the measurements, the position number will be refereed to during the comparison.

Both the measured impulse responses and the simulated impulse responses are normalized to the highest peak and plotted in dB. The first arrival of the measured impulse response has been synchronized to the first arrival of the created impulse response, where the x-axis shows the propagation delay of a sound wave. To compute the simulated impulse response the sound profile from the sea trial in Breiangen was used. Figure 4.17 shows a ray trace of 2000 meter with the corresponding sound speed profile from Breiangen. Located in Appendix E plot of ray traces for the rest of the positions are shown. The impulse responses from HACE and the sea measurements are

shown in Figures 4.18 to 4.29.

Table 4.5: Simulation locations given in Cartesian coordinates for the two nodes, Node 1 is the transponder, while Node 2 is the hydrophone.

Position number	Node	x-direction	y-direction	z-direction	Range
	1	0 [m]	0 [m]	180 [m]	
1	2	10 [m]	0 [m]	20[m]	160.31 [m]
2	2	500 [m]	0 [m]	20 [m]	524.98 [m]
3	2	1000 [m]	0 [m]	20 [m]	1012.72 [m]
4	2	1500 [m]	0 [m]	20 [m]	1508.51 [m]
5	2	2000 [m]	0 [m]	20 [m]	2006.39 [m]
6	2	2500 [m]	0 [m]	20 [m]	

Since, the two impulse responses are produced differently, the SNR does not give any grounds of comparing the measured and the simulated impulse response. From Section 2.4 we have a time changing channel caused by Doppler spread. The amplitudes of the scatters are changing from one time instance to the next, with the relation given by the Doppler bandwidth. Therefore, comparing the amplitudes of the reflections have been excluded. A way to compare the amplitudes of the reflections is to acquire mean values of both the measured impulse response and the simulated impulse response.

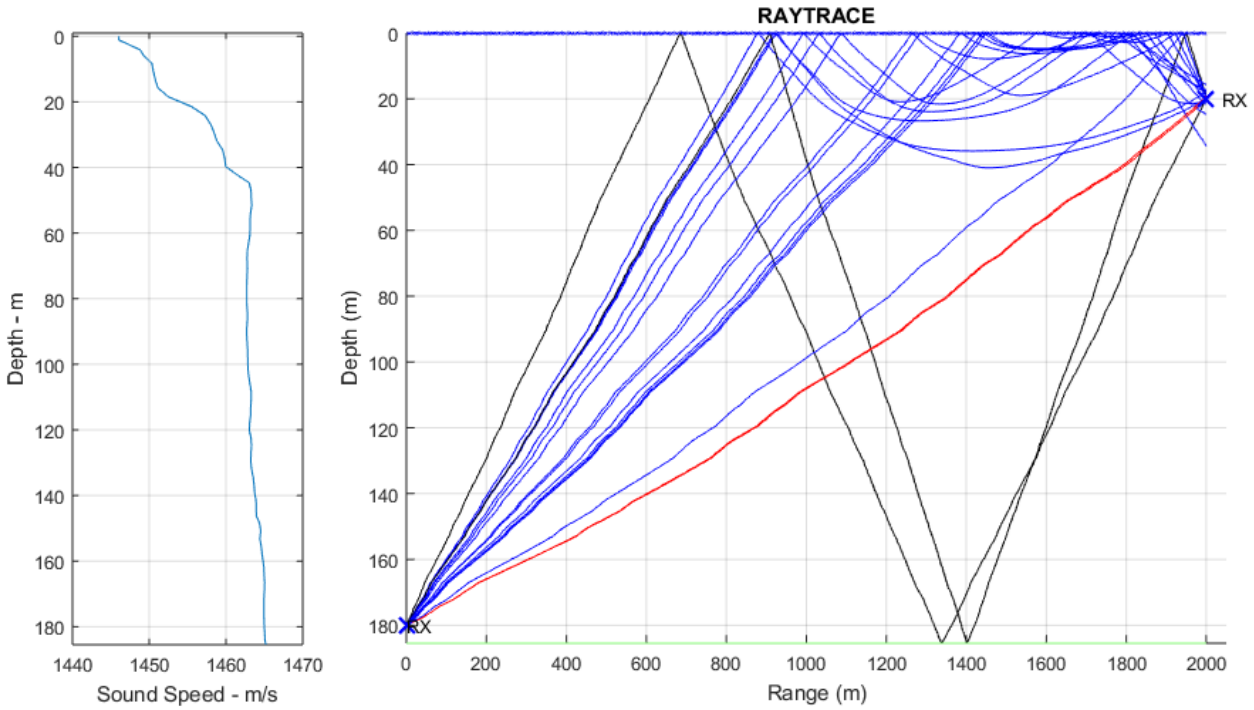


Figure 4.17: Ray trace of where Node 1 is locate at 180 meters depth and Node 2 at 20 meters depth and 2000 meters range. Left side shows the measured sound speed profile in Breiangen. Obtained from HACE.

HACE vs. measurement: Position 1

Figure 4.18 shows the channel impulse response produced using HACE at Position 1 from the values in Table 4.5. A plot of the ray trace is seen in Appendix E.7. The first arrival in the impulse response is the direct path between the two nodes, and arrives after 109.7 ms, while surface reflection is shown 27.9 ms later. The surface reflection produces scatter as can be seen in the figure. Covering the whole plot is the ambient noise described in Section 2.7.

The first direct arrival in Figure 4.19 has been synchronized to the first arrival of the simulated response to make the two impulse easier to compare. The first reflection in the measured impulse response arrived 31.5 ms after the direct. Scatter in the measured impulse response can be seen as the smaller peaks arriving later in time to the first peak of the reflection. The envelope of the two plots look similar, where both seem to only have one direct transmission and one surface reflection. From the two different impulse responses, the reflection in HACE arrives 3.6 ms prior to the measured, or a deviation of 11.42%.

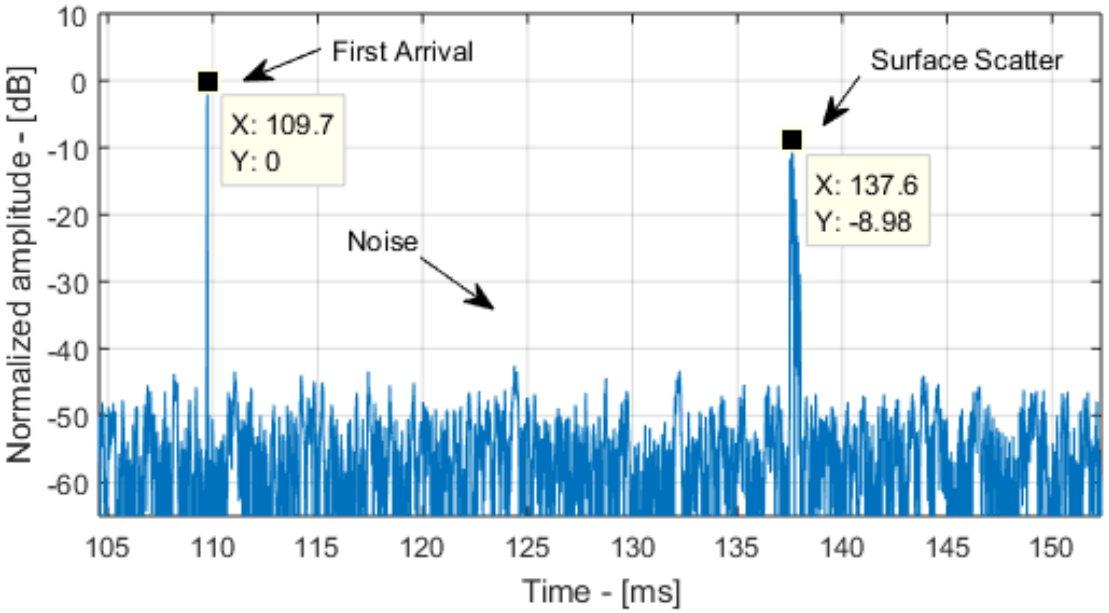


Figure 4.18: Impulse response computed in HACE. The first arrival is arriving after 109.7 ms and the second is a surface reflection arriving after 137.6 ms. Ambient noise is present throughout the window.

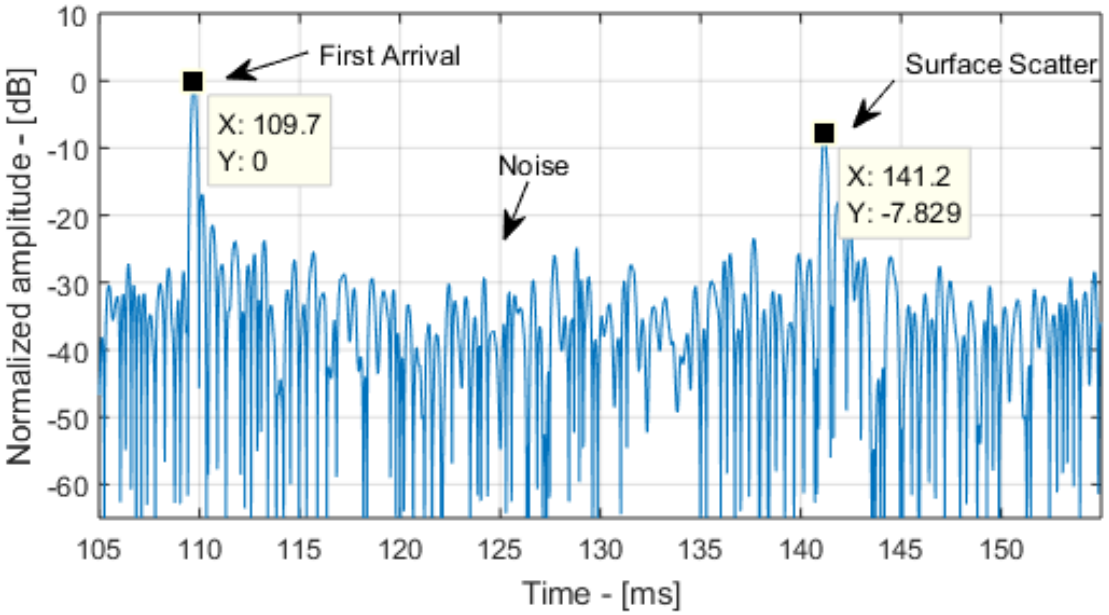


Figure 4.19: Measured impulse response from the sea trial. The first arrival is arriving after 109.7 ms, and the second is a surface reflection arriving after 141.2 ms. Noise is present.

HACE vs. measurement: Position 2

The next two figures, Figure 4.20 and 4.21, show plots of the simulated and measured impulse response for 500 meters range. The direct path arrives after 359 ms. From HACE, in Figure 4.20, the first surface reflection arrives 10.4 ms later than the direct. Since the range between the nodes have increased the angles of the rays are smaller, thus surface reflection is subjected to a larger surface and as a result produce a longer lasting scatter. This means that the reflection is more spread in time. The ray trace for this setup can be seen in Appendix E.8.

The increase in spread can easily be seen in the measured impulse response in Figure 4.21 as well, where the first reflection is detected 12.3 ms after the direct path. After the first reflection the scatter is spread throughout the remainder of the window. The difference in arrival time of the reflection from the simulation and measurement is 1.9 ms. In percent the difference increased to 15.45%.

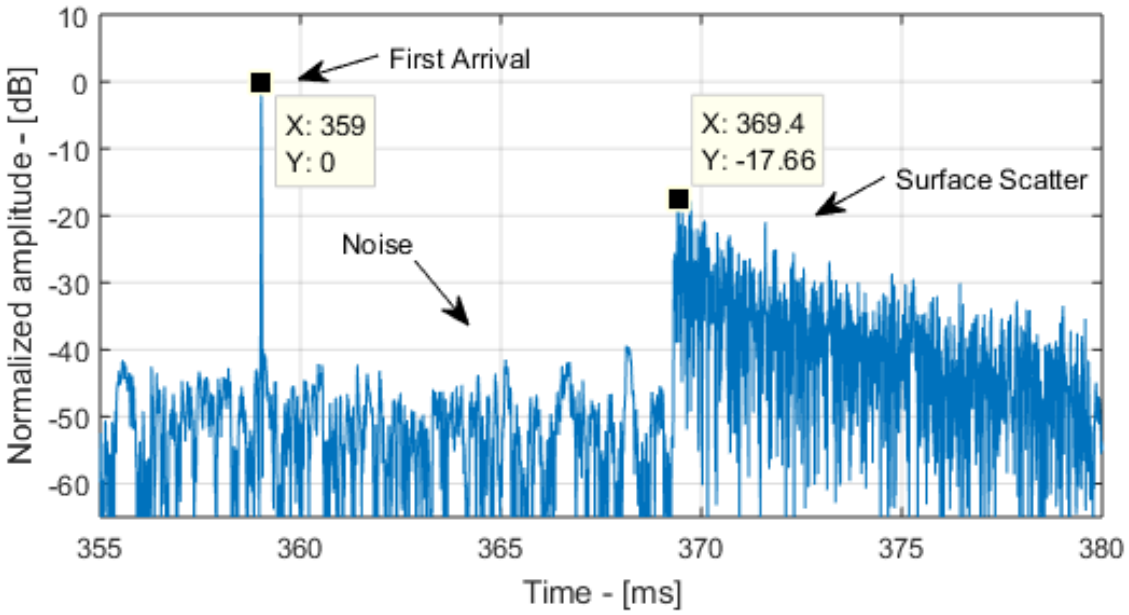


Figure 4.20: Impulse response computed in HACE. The first, direct, arrival is arriving after 359 ms, and the second is the first surface reflection arriving after 369.4 ms. Noise is present.

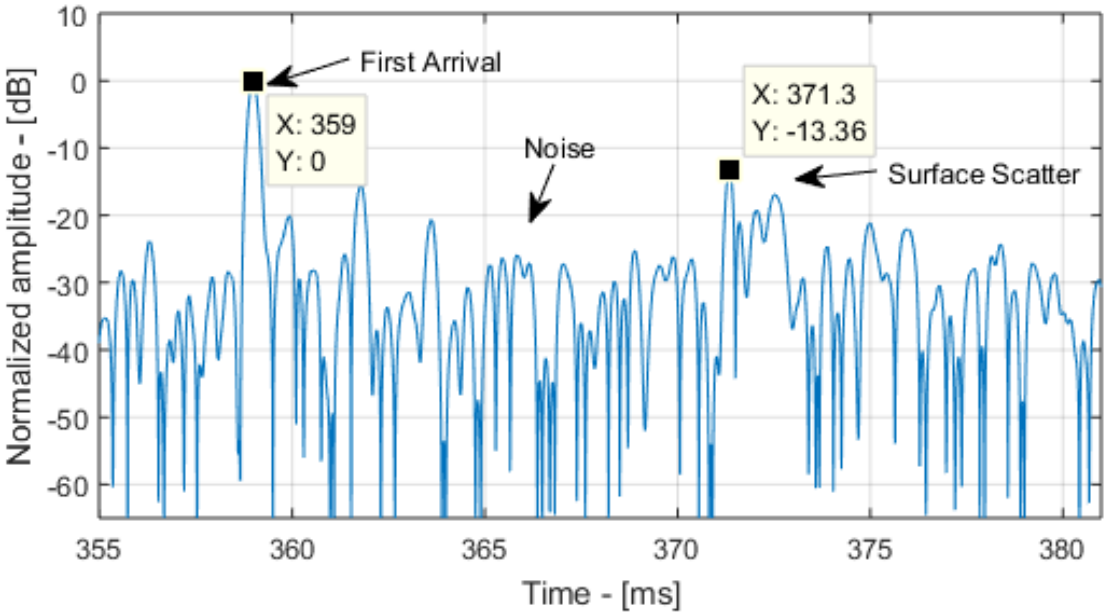


Figure 4.21: Measured impulse response from the sea trial, where the horizontal distance between nodes is 500 meters. The first arrival is arriving after 359 ms, and the second is a surface reflection arriving after 371.3 ms. Noise is present.

HACE vs. measurement: Position 3

Figure 4.22 and 4.23 show the simulated and measured impulse responses for position 3, the 1000 meter separating distance. HACE shows a propagation delay of 692.5 ms for the first arrival and 698.6 ms for the first reflection. The surface scatter consists of several different paths. Combining the increasing number of reflections with an even smaller output angle results in the long spread in time. A view of the ray trace is located in Appendix E.9.

The measured impulse response, in Figure 4.23, has the first reflection 7.2 ms after the direct path. Surface scatter can be seen by the slow decline in amplitudes in time from the first peak. The difference in arrival time between the two is 1.1 ms, or a 15.28% deviation.

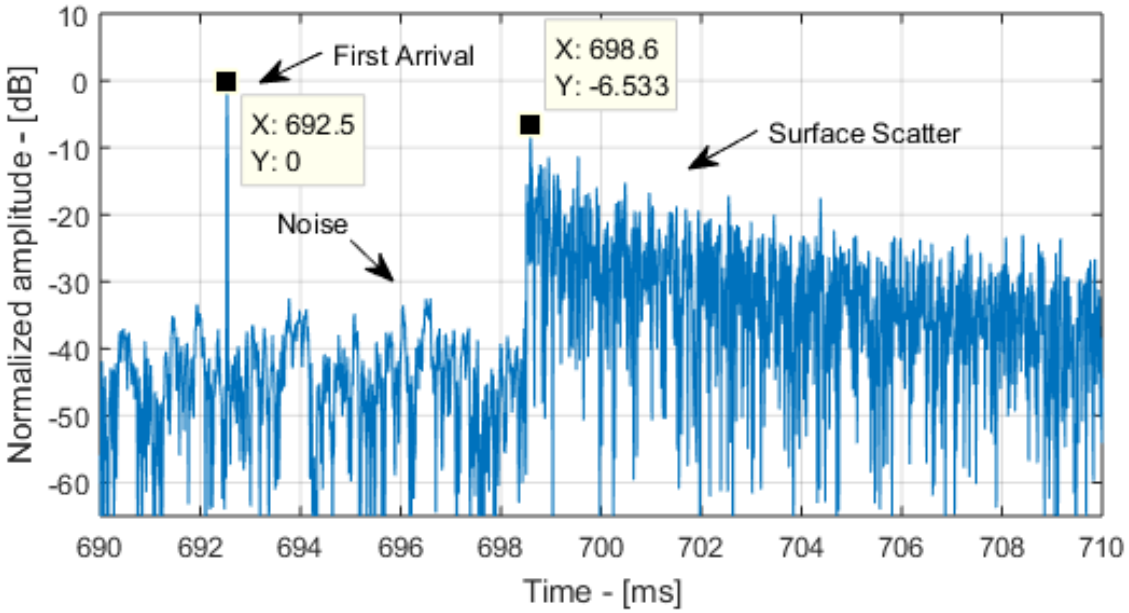


Figure 4.22: Impulse response computed with HACE. The first, direct, path arrives after 692.5 ms, and the second is the first surface reflection arriving after 698.6 ms. Ambient noise is present.

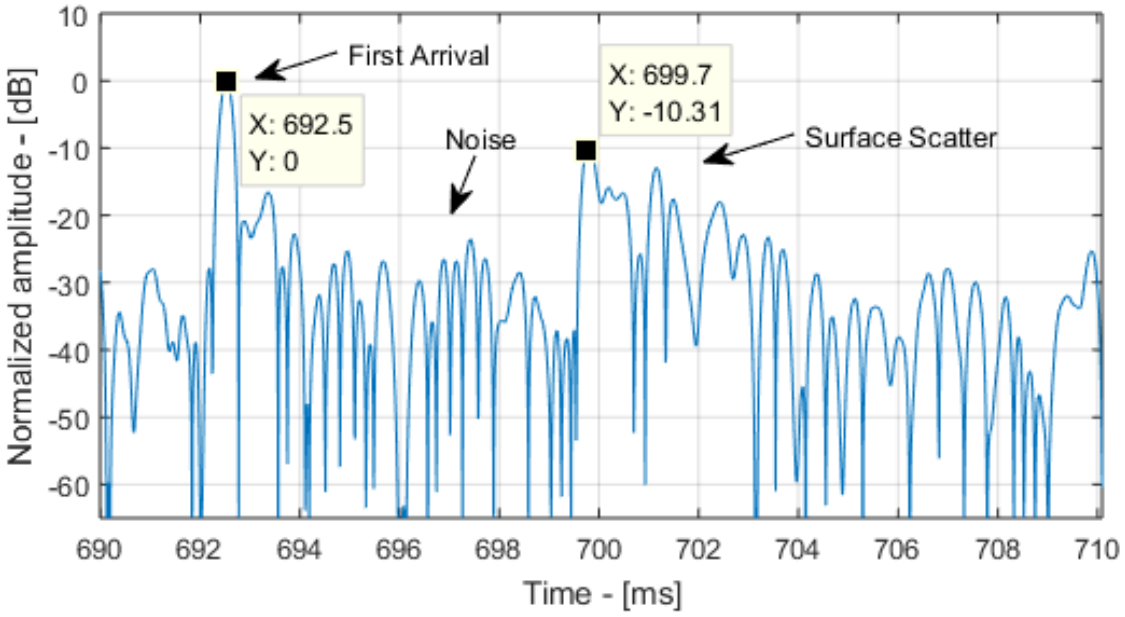


Figure 4.23: Measured impulse response from the sea trial, where the horizontal distance between nodes is 1000 meters. The first arrival is shown at 692.5 ms, and the second is a surface reflection arriving after 699.7 ms. Ambient noise is present.

HACE vs. measurement: Position 4

For this fourth position, results from HACE and the measured impulse response for 1500 meters range is shown in Figure 4.24 and 4.25. From HACE the direct path has a propagation delay of 1031 ms, and the second arrival comes 5 ms afterwards. In the measured impulse response the reflection also arrives 5 ms after the direct path. The surface scatter in both figures shows a decline in peak values from 1036 ms, but from 1043 ms the amplitudes slightly increase, meaning that there are similarities in the envelopes of the two impulse responses. In Appendix E.10 a plot of the ray trace is shown.

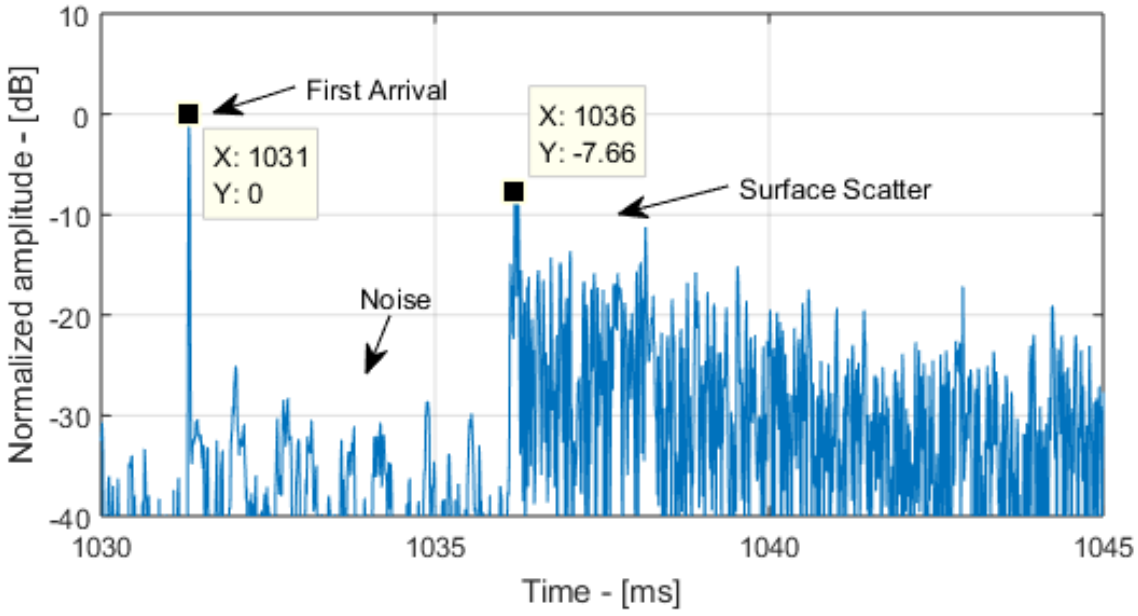


Figure 4.24: Impulse response from HACE. The direct path arrives after 1031 ms and the first reflection is after 1036 ms. There are later reflections seen by the peaks at around 1038, 1043 and 1044 ms.

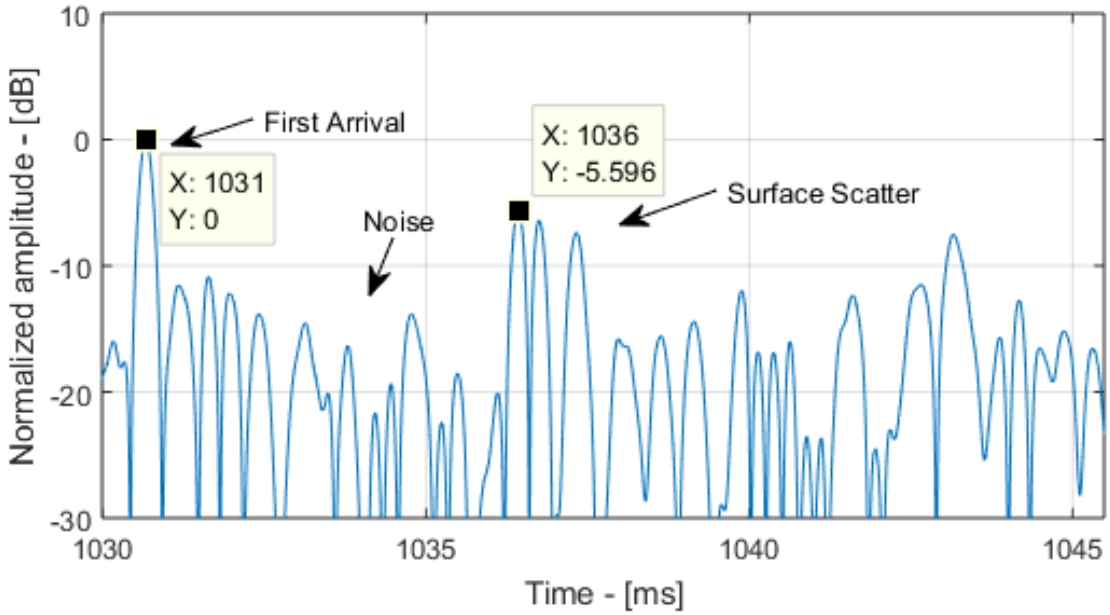


Figure 4.25: Measured impulse response from the sea trial, where the horizontal distance between nodes is 1500 meters. The first arrival is shown at 1031 ms, and the second is a surface reflection arriving after 1036 ms. There are later reflections seen by the peaks at around 1040 and 1043 ms.

HACE vs. measurement: Position 5

The simulated impulse response for a horizontal range of 2000 meters is shown in Figure 4.26 and in Figure 4.27 the measured impulse response is displayed. The ray trace figure for this case is shown in the start of this section Figure 4.17. From HACE the direct path has a propagation delay of 1372 ms, while the first reflection can be seen arriving after 1377 ms, 5 ms separate the two paths in the simulation.

By looking at the measured impulse response, the direct path is found at 1372 ms. After 5 ms the reflection can be seen. The contour of the two impulse responses seem to have similar shapes, where the reflections are scattered in time.

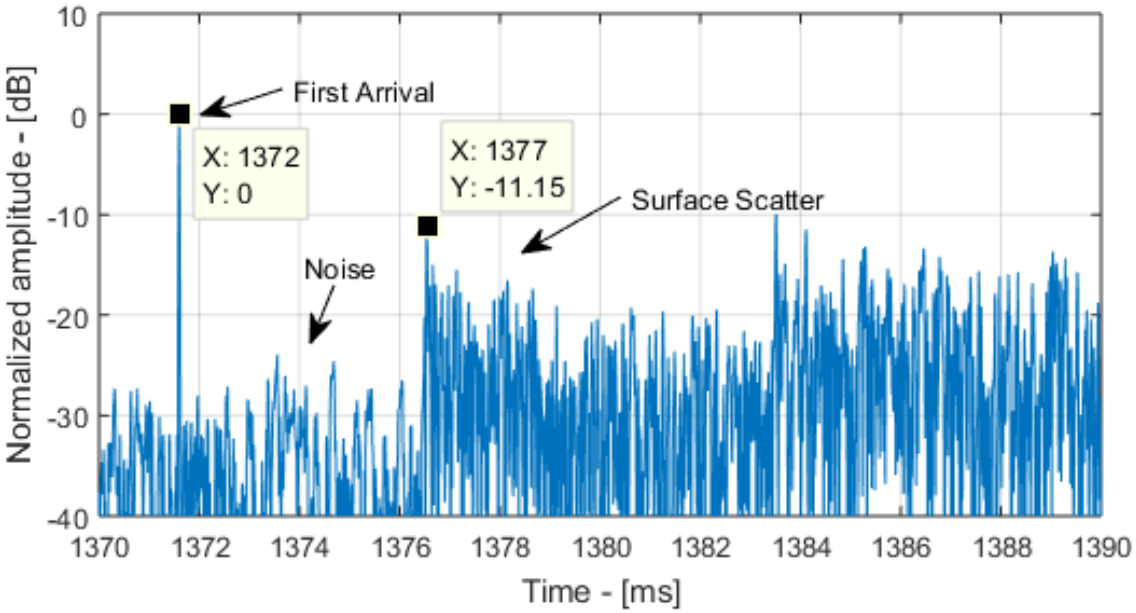


Figure 4.26: Impulse response from HACE. The direct path arrives after 1372 ms and the first reflection is after 1377 ms. There are later reflections seen by the peak at around 1383 ms.

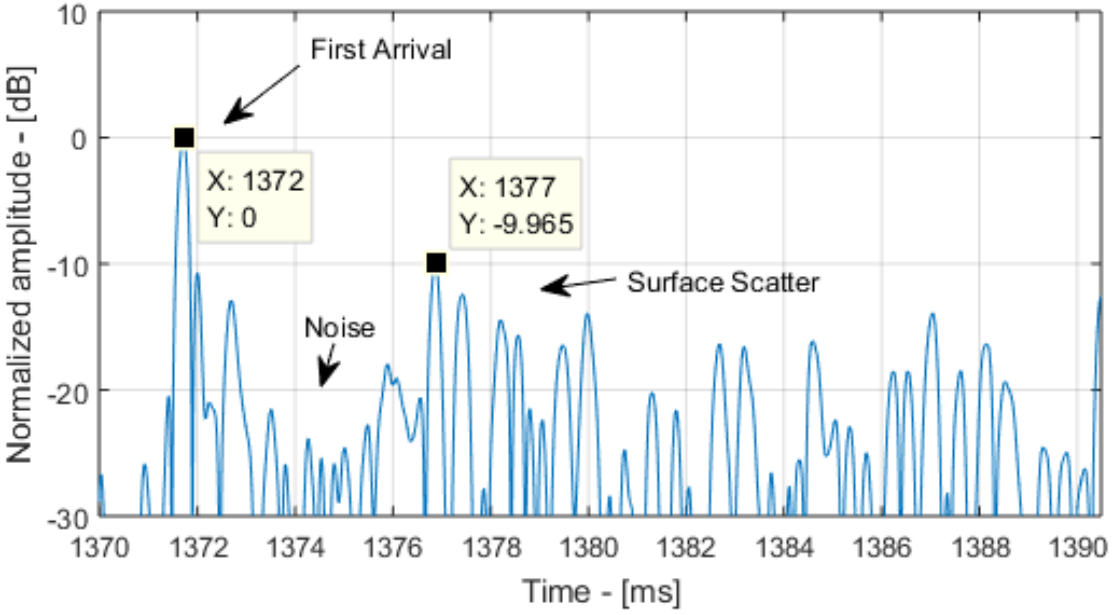


Figure 4.27: Measured impulse response from the sea trial, where the horizontal distance between nodes is 2000 meters. The first arrival is shown at 1372 ms, and the second is a surface reflection arriving after 1377 ms. There is a second reflection at around 1383 ms and one at 1387 ms.

HACE vs. measurement: Position 6

2500 horizontal range was the last measurement with the possibility of producing an impulse response able to get useful information from. The measured impulse response is shown in Figure 4.29 and in Figure 4.28 is the plot of the impulse response from HACE.

The propagation delay of the sound wave over 2500 meters was found to be 1712 ms with a surface reflection arriving 5 ms later in HACE. This 5 ms difference was also found in the measurement. As well as the previous cases, the envelopes of the two impulse responses seem to be quite similar. There are corresponding peaks between the measurement and the simulation at around 1721.5, 1724, 1725.5 and 1727 ms. As well as for the other cases the ray trace is located in Appendix E.12.

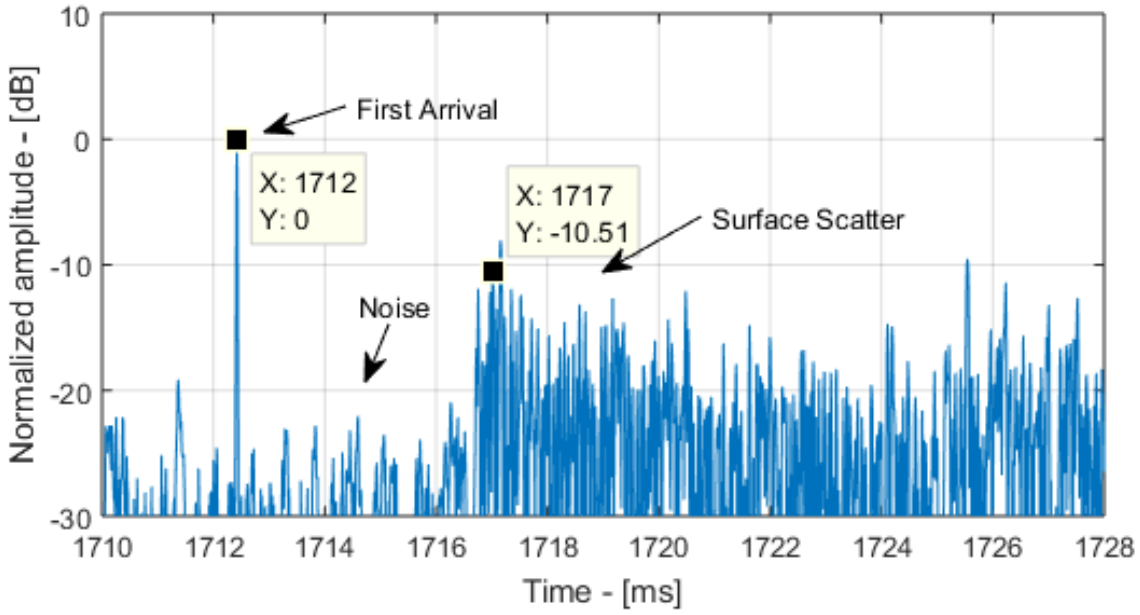


Figure 4.28: Impulse response computed in HACE. The first, direct, arrival is arriving after 1712 ms, and the second is a surface reflection arriving after 1717 ms. At around 1724 and 1725.5 ms there are distinct peaks marking later reflections.

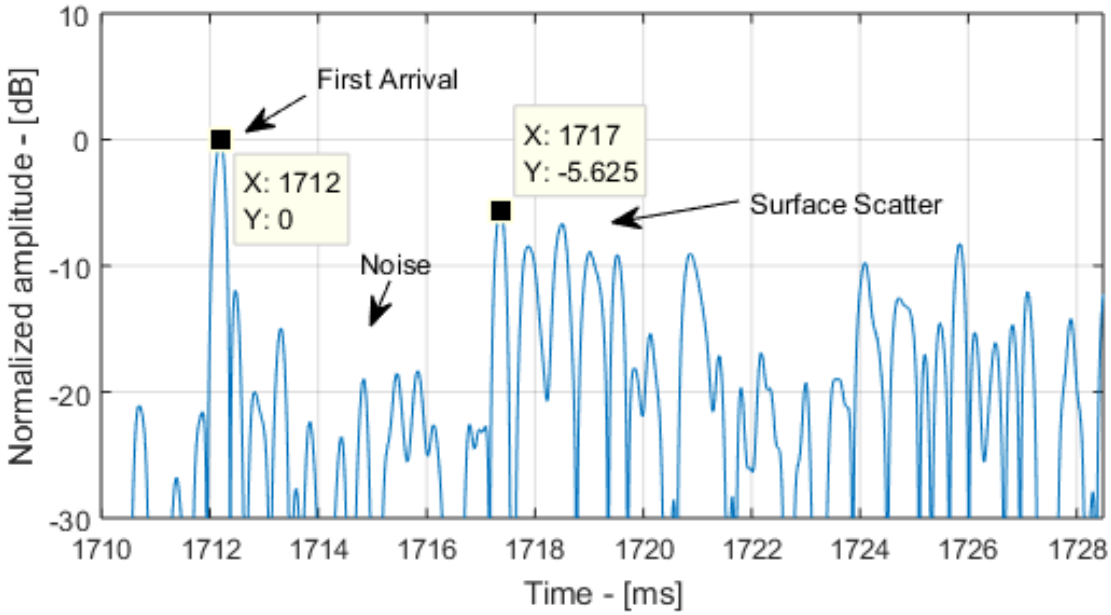


Figure 4.29: Measured impulse response from the sea trial, where the horizontal distance between nodes is 2500 meters. The first arrival is shown at 1712 ms, and the second is a surface reflection arriving after 5 ms after, at 1717 ms. Later reflections can be seen at around 1724 and 1725.5 ms.

Sum up

Throughout all impulse responses, comparing HACE with the measurements reveal recognizable peaks, where the peaks seem to correlate in arrival time with some deviations. For the shortest distance, the direct path and the reflection can easily be seen by the two strong peaks. When the range increases, the number of reflections also increase. These reflections are seen as the increase in scatter length, but also as local peaks in the scatter. Perhaps the most distinct later reflections can be seen in Figure 4.26 and 4.29, where the number of reflections is larger than for the shorter ranges.

For long range propagation in shallow waters, the grazing angle becomes smaller and as a result the path difference between the reflections and the direct path is becoming closer to each other. This mathematical phenomena can easily be seen if we compare Figure 4.18 and 4.26, where the shortest range has a reflection around 30 ms after the direct, while for a range of 2500 meters, the reflection comes 5 ms after.

The results from the impulse responses are shown in Table 4.6. For Position 2 we see the largest deviation. For Position 4, 5 and 6 the deviations are found to be 0%. Actually by studying the plot closely there are small variations for these three positions as well, but since the peak values for

Table 4.6: The results for the measured and simulated impulse response. To the right in the table the time difference and the deviation between the measured and simulated are displayed

Measurements				
Measurement	Delay dir ray	Delay first reflection	Time difference (ms)	Deviation (%)
HACE Pos. 1	109.7 ms	141.2 ms		
Measure Pos. 1	109.7 ms	137.6 ms	3.6 ms	
			100*3.6/(137.6-109.7) =	11.42 %
HACE Pos. 2	359 ms	369.4 ms		
Measure Pos. 2	359 ms	371.3 ms	1.9 ms	
			100*1.9/(371.3-359) =	15.45 %
HACE Pos. 3	692.5 ms	698.6 ms		
Measure Pos. 3	692.5 ms	699.7 ms	1.1 ms	
			100*1.1/(699.7-692.5) =	15.28 %
HACE Pos. 4	1031 ms	1036 ms		
Measure Pos. 4	1031ms	1036 ms	0 ms	
			100*0/(1036-1031) =	0 %
HACE Pos. 5	1372 ms	1377 ms		
Measure Pos. 5	1372 ms	1377 ms	0 ms	
			100*0/(1377-1372) =	0 %
HACE Pos. 6	1712 ms	1717 ms		
Measure Pos. 6	1712 ms	1717 ms	0 ms	
			100*0/(1717-1712) =	0 %

these three cases are rounded up to the closest integer, the corresponding values does not deviate. In the next chapter, sources of errors will be discussed.

4.5 Data System Performance

When creating a new software for a computer, it is necessary to map the required computer power for the program to behave the way it was designed to. There have been used two computers with similar hardware specification, but with different versions of MATLAB. One computer had an Intel i7 2.4 GHz processor, 8GB memory and MATLAB version 2015a, while the other was equipped with Intel i7 1.8 GHz processor, 8GB memory and MATLAB version 2014a.

The computer requirements for the simulations in HACE are linked to the real-time part of the system. Calculations prior to running in real-time does not have the same time criteria. Mentioned previous, keeping the computations in real-time has been a focus during development. Figure 4.30 shows the change in computer power usage during real-time processing for two and four nodes.

The computer used in the example had an Intel i7 1.8 GHz processor, 1.6 % of the processor power was used for point-to-point communication with a 133 MB increase in memory usage. For four nodes 10.6 % of the processor was used with a memory increase of 192.6 MB. Under "Disk" the increase tells that the signals were recorded and saved to a file.

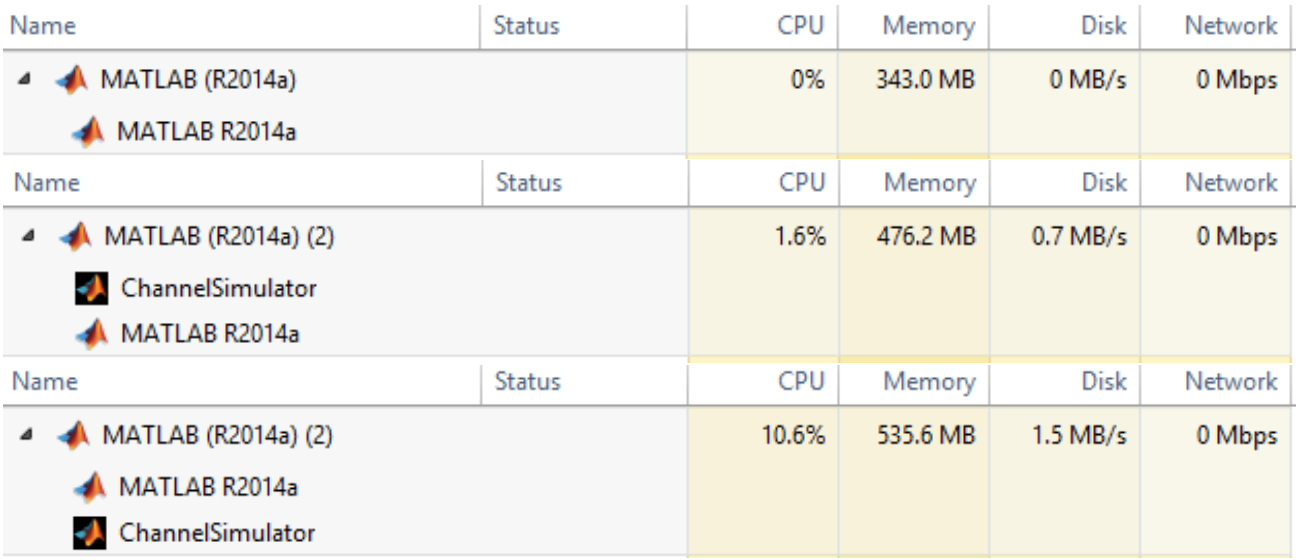


Figure 4.30: Computer performance during real-time simulations. Top figure shows the computer usage only with MATLAB, the figure in the middle shows the computer usage during the real-time simulation with two nodes, while the lower figure shows computer usage during the real-time simulation with four nodes.

To check for the reliability of the latency, 100 calibrations were performed for the computer with 1.8 GHz processor. The statistics from these 100 calibrations are plotted in the box plot in Figure 4.32. In appendix Figure E.1 is a corresponding latency calibration plot of the other computer used.

From Section 2.9 the relation between frame size and delay was explained, the plot in Figure 4.32 shows the latency in milliseconds versus frame size. The "+" signs in the plot represent single deviations, and can be seen for the frame sizes 1024, 1536 4096 and 6144 samples. For the 1024 samples frame size, the latency shows a distribution surrounding 34.2 milliseconds, illustrated by the blue square. This tells that the 1024 samples frame size is not as reliable as the larger frame sizes, but has been used to be able to simulate the shortest ranges. It can be seen from the plot that increasing the frame size, increases system reliability, but also increase the latency.

The importance of choosing the correct frame size also appears when investigating the system stability. In Figure 4.31 the stability of a simulation of 160.31 meter is shown, where a frame of 1536 samples have been used. The stability has shown to be very accurate with a jitter of $\pm$ 1 cm, with a $\sim$ 20 cm offset for all measured ranges.

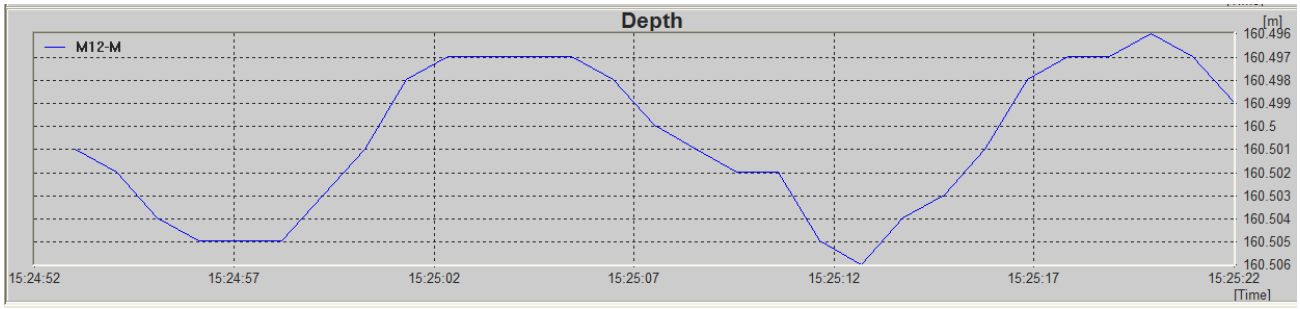


Figure 4.31: Jitter 10 meter(On top), depth 180 meters, and 20m, obtained from APOS

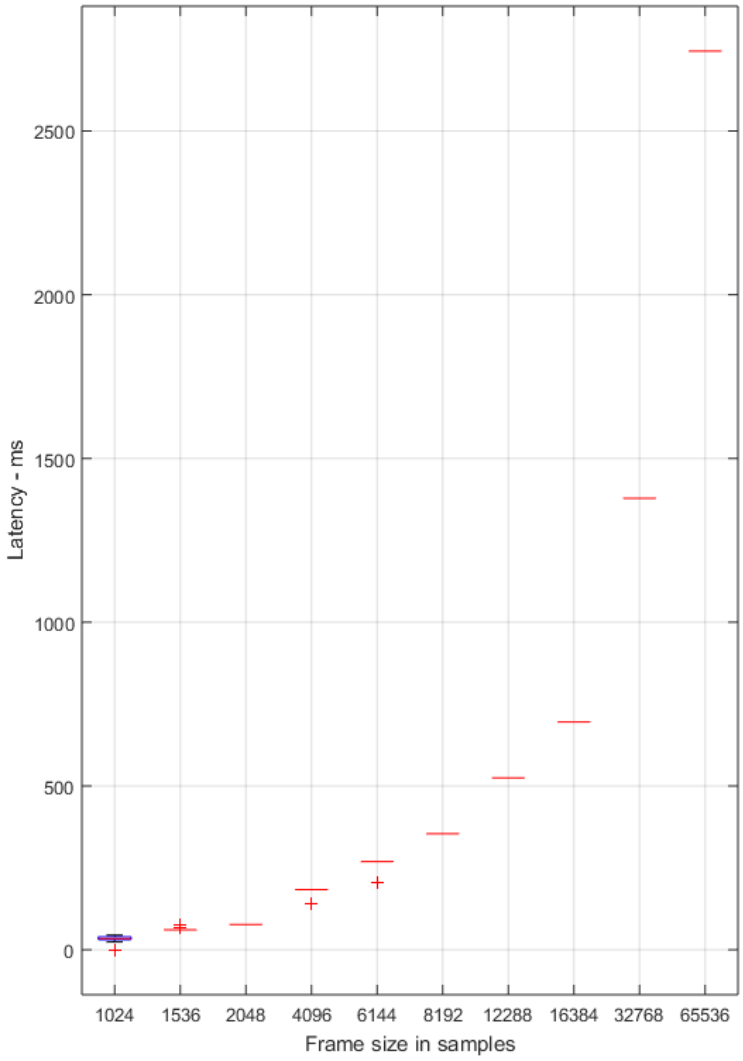


Figure 4.32: Calibration results for frame size versus latency in milliseconds.

Chapter 5

Discussion and Conclusion

In this last chapter the findings and results from the previous chapter, and the sources of errors are discussed and reviewed. A recommendation for further work and a conclusion will be drawn at the end.

5.1 Discussion

Underwater wave propagation in the real world is more complex than that which is implemented in the simulations with more parameters influencing the signal, such as rough surface and seabed, Doppler shift, several seabed layers, different noise sources, fish, and so on. Implementing new mathematical models to simulate oceanic behavior is a time-consuming process, and there will always be potential for improvement. This having been said, the most influential acoustic phenomena for underwater wave propagation have been introduced and implemented into the system.

As explained in the introduction, this thesis is an expanded and improved solution to the previous project thesis. Where the project thesis focused on how to get the channel emulator to work, this master's thesis has focused on the more advanced acoustics phenomena and on real time performance with the possibility of network communication between nodes.

HACE

HACE has been developed from the ground upwards, giving us the possibility of being free in our decisions regarding code structure. Drawing on our previous knowledge, and considering its huge library of build-in-functions, MATLAB was the obvious choice for creating HACE. The digital signal processing toolbox has been crucial in the development allowing real-time access to audio streams.

The choice of programming language has an effect on the efficiency of a program. It is known that MATLAB is not the most efficient programming language, but with the possibility of converting the code to another language later for speed enhancement, this was not considered to be an issue.

The GUI was created with the intention of making it easy to use and intuitively understandable. Decisions concerning the layout of the GUI were discussed with Kongsberg Maritime during development to satisfy their needs and preferences. There have been several layouts, but we ended up with one main window where the user has all the parameters and buttons easily accessible.

Stability in a real-time system is important, as it determines the reliability of a system by reducing jitter. For the simulator, the stability is mainly dependent on the accuracy of the simulated range over a period of time. Jitter will create a varying range, even if the range in the simulation is fixed. Findings show that the emulator was stable, even for long range propagation where the measurements in APOS showed that the jitter, translated into distance, was around ± 1 cm in ranges from 100 to 2000 meters. A 1 cm fluctuation, with a sound speed of around 1500 m/s and a sampling frequency of 96 kHz, does not correspond to one shift in a sample, meaning that this variation in range might not occur from HACE.

The latency of the system is dependent on both the computer's software and its hardware. It was found that there were big latency differences between the MATLAB 2014a version and the MATLAB 2015a version, with more stability and faster computation in the newest version. The latency calibration function gave the channel emulator program serious strength with respect to jitter/stability and precision, for its individual customization option for different computers and sound cards. The minimum system latency having stability was measured as 34.2 milliseconds, which results in a minimum distance of 51.3 meters for a 1500 m/s sound speed.

Reducing the number of calculations in real-time to a minimum was the focus in creating the system. Since the number of calculations are directly proportional to the impulse response, a reduction in computations can be made by decreasing the size of the impulse response. By exploiting the latency, a longer propagation delay can be added without any increase in the number of computations. As a consequence the impulse response is smaller, and fewer computations in real-time are required.

Calculating the ray trace was done with the external software named "BELLHOP". It is a well known and established program in the community of underwater acoustics. Building a ray trace software with similar capabilities as "BELLHOP" would have been a time consuming process. It would therefore have been an enormous job to create a new ray tracing program that could have competed with "BELLHOP" within the time constraints. Instead, we used "BELLHOP" and merged it into our system. By choosing to use "BELLHOP" we had the time to focus on creating a better

channel simulating software all around.

It should be pointed out that HACE creates an ideal situation where the real world has continuously moving objects, noise from different sources and location specific acoustic situations not covered in the simulations. However, with Doppler spread (time variation), surface scatter, varying seabed and parameters, acoustic absorption, sea state, varying sound speed and ambient noise, the simulated channel impulse responses revealed good similarities compared to the measured impulse responses.

The results from range measurements showed an offset of ~20 cm for all ranges. Water absorption filters with 31 coefficients are applied to all paths, see Section 3.2. From Appendix E.5 the filter delay with 31 coefficients is $D_s = 15$ samples. With the sample rate set at 96 kHz during the measurements, a 15 samples delay results in a time delay D_t of

$$D_t = \frac{15}{96000} \approx 0.156 \; [ms]$$

And when the time D_p for a wave to move 20 cm with a sound speed of 1500 m/s is

$$D_p = \frac{0.20}{1500} \approx 0.133 \; [ms]$$

where we can see that the 20 cm offset comes from the filter delay.

Sea trial measurements

To validate how closely the emulator manages to recreate the real world, was the motivation for taking the field trip to Horten. The trip also provided hands-on experience when dealing with hydroacoustic equipment and how underwater measurements were performed. Earlier it was explained that the ambient noise is a function of sea state, or wind speed. The wind speed was not measured, but the sea was calm. An evaluation of the sea state was made in collaboration with the experienced participants, and was estimated to be sea state level 1.

Measurements were done by deploying the hydrophone from the boat in the sea and record the received signal. The exact cable length at each deploying was not measured, but was estimated to be around 20 meters. During measurements the motor was turned off to reduce noise, and the boat was not anchored. An unanchored boat has the tendency of drifting with the current, and during up to 10 minutes of measurements the boat can move a significant distance. This drifting can also affect the hydrophone. If the boat and hydrophone are moving at different paces the hydrophone suspension could lift it closer to the sea surface creating a shorter path for the reflection between

the hydrophone and the surface.

The boat has a large hull piercing several meters into the ocean. Therefore, it is reasonable to assume that the signals were reflected off the hull. These hull reflections could affect the measured signals if the hydrophone is drifting towards the vessel creating a relatively shorter path for these reflections.

At every measuring point two signal intervals were measured to increase the redundancy. Two different types of pulses were used, a PN 255 and a PN 1023, where the time period between the two signals was 5 minutes and 10 seconds. The signal power for the transmitter was set to minimum, because Kongsberg Maritime was concerned that the transponder would overheat during the 30 seconds pulse. By using the minimum power setting the signal was indistinguishable from the noise at ranges above 2500 meters. Increasing the power will increase the SNR enabling detection at longer ranges.

The most significant risks of obtaining false measurements can be summarized as

- Inaccurate measuring position relative to the source
- Drifting vessel and hydrophone
- Seabed scatter
- Cable length of the hydrophone
- Sea state estimation

Kongsberg Maritime's equipment

Kongsberg Maritime supported us with the necessary equipment and its corresponding softwares. A well known problem, from both the previous project thesis and Kongsberg Maritime's own testing, was crosstalk between the communication devices. After the project thesis a way of improving the shielding before the upcoming master's thesis was discussed . The improvement was that the devises were put inside metal cylinders, the same cylinders Kongsberg Maritime use for their underwater equipment. Additionally, the cylinders were placed inside hard case boxes for extra shielding. These two steps contributed greatly in removing most of the crosstalk, but still the nodes needed to be placed meters from each other to totally avoid the issue.

Only two nodes were sealed in metal casings. For network communication one of the three nodes used did not have the same shielding. The lack of insulation for the one node again introduced the problem of crosstalk, even when it was wrapped in aluminum foil, placed 8 meters away from the others and in a different room! In one last attempt the unshielded node was placed inside

an anechoic chamber with a damping factor of above 80 dB for electromagnetic waves. Finally, using the anechoic chamber, the measurements could be taken.

As a comment to the problems observed with crosstalk, Kongsberg Maritime's equipment has been built to function in different circumstances to those used in this thesis. Once the equipment is submerged and separated by a greater distance, crosstalk will not affect the transmission. It is not only the distance of separation that removes the crosstalk, but we also know from the theory in Chapter 2 that electromagnetic waves are highly attenuated by water.

It can be said that our lack of familiarity with Kongsberg Maritime's equipment and software ment that it took time in getting acquainted with the products. APOS was used to communicate with the equipment, and to measure range and stability/jitter. The program has four selectable power levels, from minimum to high. Not all features in the program have been used, but setting up the communication link and telemetry has gone well.

During testing it was noticed that ranges above 2061 meters could not be measured correctly using APOS, e.q. a range of 2500 meters was measured as 439 meters. When the same node positions were used in the simulation, APOS produced equivalent results. Kongsberg Maritime was contacted and a error was found in their software, where the problem was that APOS repeated ranges in periods of 2061 meters. Of course, the issue with range detection was rapidly fixed, but it shows the advantage of having a simple and efficient way of testing in the field of R&D.

5.2 Summary and Conclusions

As stated in Section 2.8, wireless communication through water is a difficult process. With an increasing number of underwater vehicles and sonar operations, underwater communication has become increasingly important. Therefore, a good understanding of how the environment influences signals is essential. A channel emulator is a handy tool for getting acquainted with the effects the water has on underwater acoustic waves and to study how hydroacoustic equipment behaves underwater.

In creating the channel emulator some simplifications were done for the simulations (see Chapter 1.3). The completed and implemented simulations are: reflection loss, geometric spread, propagation delay, ambient noise, acoustic absorption, point-to-point and network communication in 3D, varying sea bed in 3D, varying sea surface, surface scatter, Doppler spread (time variation) and varying sound speed with ray tracing.

With the latency calibration function the simulator opens up a new dimension for HACE, enabling it to adapt to individual systems with respect to computer performance, sound card and

software. The adaptive frame size was developed to maintain stability for different ranges, where every frame size has their corresponding calibrated latency. The minimum system latency was measured to be 34.2 milliseconds for an average contemporary computer with MATALB R2015a, which results in a minimum distance of 51.3 meters for a 1500 m/s.

The results from measuring ranges showed highly accurate values, with an offset of ~20 cm for all ranges measured. All distances measured also showed high stability, where we see from the jitter measurements that the variations for all ranges were ± 1 cm. Most of the offset comes from the filter delay and can be dealt with easily, but tuning the equipment and taking new measurements is time consuming. We concluded that an offset of ~20 cm was adequate. For a range of 2000 meters, a 20 cm offset is an 0.01% deviation from the actual distance. In a later review it would be possible to insert a tunable offset parameter in HACE to enable more accurate simulations.

The field trip to Horten with measurements in Breiangen in April was to measure the underwater channel for communication. A similar setup was used in the simulations to create impulse responses with the corresponding distances and speed profiles. The measured and the simulated impulse responses were then compared revealing good results. Both the measured and simulated responses were normalized in amplitude and the direct arrivals were synchronized. Comparing the time between the direct and first arrival, the largest deviation was 15%, recorded at the 1000 meter range. For longer ranges the deviations plummeted. The impulse responses also showed good similarities where the similar envelopes, or shapes, could be seen in the plots with reflections and scatter.

This master's study has shown that a real-time hydroacoustic channel emulator for point-to-point and network communication can be made. For the real-time simulations using MATLAB, for the completion of a working emulator, simplifications were needed for the simulations. For controlling the simulations, a well-developed GUI was made with an easy and intuitive layout. It has the ability to set position to the nodes, load transducer data, and to adjust/add a number of acoustic properties. Thus, we have attained the objectives in problem formulation, and also met the requirements stipulated by Kongsberg Maritime by having a latency less than 100 ms, with negligible jitter. Range tests with APOS have been performed, and we demonstrated that network communication is possible, with the same accuracy as for point-to-point.

In the short lifespan of HACE it has already been to to use by Kongsberg Maritime, from whom we have received excellent feedback. During the development of HACE, an error with Kongsberg Maritime's software was detected. The error was fixed right afterwards, but it shows how the HACE can be used in an efficient way for testing hydroacoustic equipment. We are proud to have completed a functioning water channel emulator that produces stable and reliable results. There are

opportunities for further development of HACE by adding more sophisticated acoustical models. By using the impulse response, the more advanced models can easily be inserted into the simulations.

5.3 Further work

As stated in the first chapter of the thesis, it was known that mimicking all the ocean acoustics with precision was not possible due to the time constraints. Improvements could be made in order to achieve a more accurate system model, especially with respect to the simulation of the water.

The complexity in creating a system perfectly simulating the ocean is considerable, and perhaps not even possible in real-time simulations using an average contemporary computer. For obtaining an overview of how sound propagation behaves underwater, the simulations are good indicators. The preliminary objectives have successfully been achieved. There are numerous possibilities of improvement and performance enhancement. Suggestions for improvements to be tackled can be seen in the prioritized list.

1. Fix range offset
2. Individual transducer data
3. Uneven/roughness seabed with scatter
4. Backscatter
5. Multiple layered seabed
6. Implement complex noise options: ship, shrimps, rain etc.
7. Doppler shift : communication between moving objects
8. Rewrite for performance enhancement in another programming language
9. Might be implemented on a device, perhaps a FPGA

Bibliography

[1] Acoustics-Lab (2014). *Transmission Loss, Sub-Sea Acoustics Lab*. http://www.sal2000.com/ds/ds3/Acoustics/TL.htm. [Online; accessed 2015-16-04].

[2] Applied-Physics-Laboratory (1994). *APL-UW High-Frequency Ocean Environmental Acoustic Models Handbook*. University of Washington, Seattle, WA.

[3] Butler, L. (1987). *Underwater Radio Communication*. http://users.tpg.com.au/users/ldbutler/Underwater_Communication.pdf. [Online; accessed 2014-10-22].

[4] DOSITS (2002). *Cylindrical vs. Spherical Spreading*. http://www.dosits.org/science/advancedtopics/spreading/. [Online; accessed 2015-03-23].

[5] Etter, P. C. (2013). *Underwater Acoustic Modeling and Simulation*. Taylor & Francis Group, 6000 Broken Sound Parkway NW, Suite 300, 4th edition.

[6] Francois, R. E. and Garrison, G. R. (1982). *Sound absorption based on ocean measurements: Part II: Boric acid contribution and equation for total absorption. J.* Acoust. Soc. Am 72(6), Norwood, MA.

[7] Hovem, J. M. (2012). *Marine Acoustics: The Physics of Sound in Underwater Environments*. Peninsula Publishing, Los Altos Hills, CA, 1st edition.

[8] Kinsler, L. E., Frey, A. R., Coppens, A. B., and Sanders, J. V. (2012). *Fundamentals of Acoustics*. John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, 4th edition.

[9] Knudsen, V. O., Alford, R. S., and Emling, J. W. (1948). *Underwater ambient noise*. J. Mar. Res. 7, 410–429.

[10] Kuo, S. M. and Lee, B. H. (2001). *Real-Time Digital Signal Processing*. John Wiley and Sons, Chichester, New York, Weinheim, Brisbane, Singapore, Toronto.

[11] Maritime, K. (2014). *Kongsberg Maritime*. http://www.km.kongsberg.com/. [Online; accessed 2015-11-05].

[12] MathWorks (2013). *MATLAB, Sample- and Frame-Based Concepts*. http://se.mathworks.com/help/dsp/ug/sample-and-frame-based-concepts.html. [Online; accessed 2015-04-30].

[13] MathWorks (2014a). *MATLAB, Audio Signal Processing in MATLAB*. http://se.mathworks.com/videos/audio-signal-processing-in-matlab-86358.html?form_seq=conf1260&confirmation_page=&wfsid=5820452. [Online; accessed 2015-22-03].

[14] MathWorks (2014b). *MATLAB, The Language of Technical Computing*. http://www.mathworks.se/products/matlab/. [Online; accessed 2014-11-12].

[15] Porter, M. B. (2011). The bellhop manual and user's guide: Preliminary draft. *Heat, Light and Sound Research Inc.*

[16] Proakis, J. G. and Manolakis, D. G. (2007). *DIGITAL SIGNAL PROCESSING, Principles, Algorithms, and Applications*. Pearson Education, Inc., Upper Saddle River, New Jersey 07458, 4th edition.

[17] Soni, S., Prof. Choudhary, S., and Dr. Changlani, S. (2014). *An Extensive Review on Underwater Acoustic Channel*. http://www.ijspr.com/citations/v1n1/IJSPR_0101_006.pdf. [Online; accessed 2014-11-08].

[18] Trischitta, P. R. and Varma, E. L. (1989). *Jitter in digital transmission systems*. Artech House Publishers, Norwood, MA, 4th edition.

[19] Tse, D. and Viswanath, P. (2005a). *Fundamentals of Wireless Communication*. Cambridge University Press, Seattle, WA.

[20] Tse, D. and Viswanath, P. (2005b). *Fundamentals of Wireless Communication*. Cambridge University Press, Cambridge , UK, 1st edition.

[21] Urick, R. J. (1984). *AMBIENT NOISE IN THE SEA*. Undersea Warfare Technology Office, Naval Sea Systems Command, Washington, D.C. 20362.

Appendix A

Additional Information

A.1 List of Equipment

The measurements was done at the Acoustical laboratory at NTNU between Januar and May in 2015 and in Horten (Breiangen) 15.04.2015.

LIST OF EQUIPMENTS:

- Sound card: Roland OCTA-CAPTURE, Model: UA-1010
- Cables: 4X JACK CABLE (mono), Serial-to-USB converter Model: AU0002B, LogyLink
- External equipments developed from Kongsberg Maritime:
  - 2X Transponder (Cnode, HiPap)
  - 3x El board (attenuation board)
  - Battery recharger
  - 1x LNA/PA (power amplification)
  - 1x Sensis transponder board.
- Power: 2 x power supply (14.4V, minimum 1A)
- computer program: MATLAB and APOS (Kongsberg developed)
- Sea trail:
  - Hydrophone ITC with pre amplifier
  - 20 meter hydroacoustic cable
  - CNODE Midi 34-180-SENSIS, serial no 11477
  - Portable transceiver: CPAP 30, serial no 105
  - Dunking transducer: TDD180
  - APOS

A.2 APOS

APOS is a Kongsberg developed software, short for Acoustic Position Operator Station system software. This is the program which the end-user mainly uses out in the field when they are using the transducers. Therefore is it interesting to use that program to make a real-time scenario as explained and showed in section **??**. One of the features in APOS is distance measuring; this is calculated with the response time of the receiver. It can also adjust the output signal level with four different options, they being: minimum, low, medium and high.

A.3 Problems: Prioritized List

1. **Timing, latency and latency variability (jitter):** One key functionality of the emulator is to be able to accurately and precisely control the delay between the terminals. The setup should be tuned for minimum latency and jitter measured. A default latency for the transducer should be accounted for and be controllable in the GUI.

2. **Calibrated levels:** The emulator need to accurately control and interpret the voltage levels at the terminals. The transmit and receive sensitivity of the transducer must be accounted for, controllable from the GUI.

3. **Propagation delay:** The time of flight between the sender and receiver must be modelled in the emulator. A fixed sound velocity should be controllable in the GUI.

4. **Absorption loss:** The frequency dependant absorption loss controllable from the GUI should be modelled in the emulator.

5. **Noise:** Frequency dependant ambient noise should be modelled in the emulator. The GUI should be able to control sea state.

6. **Transducer directional characteristics:** The beam pattern of the transducer can be modelled in the emulator. From the GUI the orientation of the transducer in elevation should be possible.

7. **Reflection from seabed and sea surface:** Mirror sources significantly contributing to the signals should be modelled in the emulator. The GUI should control the main reflection properties of the surfaces.

8. **Ray bending:** Ray bending can be solved using Snell's law, preferably using the Bellhop API. The GUI should handle different sound velocity profiles.

9. **Time and Doppler spreading:** The signal is spread in time due to multipath and diffraction, while in the Doppler domain the signal is spread mainly due to interaction with the moving surface and internal waves. relevant parameters of this stochastic behaviour.

10. **Time variability:** Acoustic channels might be rapidly changing due to internal waves in the ocean, moving sources and receivers, gusts of wind and rain fall, intermittent noise sources etc. The GUI should be able to control relevant parameters of such behaviour.

Appendix B

MATLAB

A presentation of the MATLAB functions used in HACE is shown in the list bellow. Since the amount of code lines used in the creating of the software is so huge only the functions name will be shown to save space. The code it self is found in the .zip file delivered with the masters thesis.

- Functions

  ambientHighFreq - Creates noise with a decrease of 17 dB per decade
  ambientNoise - Creates ambient noise
  amplitudeLimitSelection - Picks out paths with no more than 20 dB drop relative the largest amplitude
  CalibrateLevels - Calibrate amplitude levels with sensitivity and attenuation
  CalibrationFunction - Function for running the latency calibration and write the results to a file
  channelMatrix - Creates the impulse response matrix with Doppler spread
  createSeabed - Creates a seabed from the predefined shapes
  CreateSpeakerAndMic - Creates microphone and speaker object for data fetching
  createSurface - Creates the surface waves
  designAbsorptionFilter - Creates the water absorption filter coefficients
  designFilter - Designs low-pass and high-pass filters
  dopplerSpread - Creates scatter with Doppler spread
  freehanddraw - Function for drawing the users own seabed shapes
  latencyCalibration - Find the system latency
  plotarr2 - Stem plots the arrivals calculated by BELLHOP
  plotati2 - Plots the surface waves

plotbty2 - Plots the seabed shape

plotfunction - Plot function used in analyzing and contains several different plots

plotray - Plots the rays calculated by BELLHOP

rangesDopplerSpread - calculate the ranges used to create surface scatter in the Doppler spread

read_txt - Reads the transducer data from a .txt file

readSpeedAndDepth - Reads sound speed profile and depth from a .txt file

readTXT - Reads sound speed profile and depth from measured sound speeds and write to a .txt file

RealTime - Function running in real-time

transducerAngleGain - Finds the transducer gain and angle, and adds to the signal amplitude

waterAbsorptionCoefficient - Produces water absorption coefficient

writeBellhopInput - Creates files the read speed profile

writeEnvFiles - Write filse used in BELLHOP

- Scripts/GUI

AnalyzeGUI - GUI function with analyzing options

Calibration - GUI function for latency calibration

ChannelsGUI - GUI function for adding nodes

ChannelSimulator - The main GUI function controlling the whole system

headerRealTime - Header file with the most used variables

numberOfMultipaths - Find the number of multipaths for N nodes

SeabedMaterialGUI - GUI function for setting and adjusting seabed paramters

- Additional

Bellhop - Ray tracing program.

caxisrev - Sets the color axis limits and reverses the ticks

crcr - Convert real wave speed and attenuation to a single complex wave speed

newid - Creates a modal dialog box that returns user input for multiple prompts in the cell array ANSWER. PROMPT is a cell array containing the PROMPT strings.

plotshd - Plots a single TL surface in dB

polardb - is a modified version of the regular 'polar' function

read_arrivals_asc - Reads the arrival time/amplitude data computed by BELLHOP

read_env_core - Reads the environmental file

read_shd - Reads the shade file
read_shd_asc - Reads an ascii shade file
read_shd_bin - Reads TL surfaces from a binary Bellhop/Kraken .SHD file
readati - Reads an atifil into the workspace variables
readbty - Reads a btyfil into the workspace variables

Appendix C

Additional Theory

C.1 A way to add the propagation delay

Propagation delay is expressed in number of samples in the algorithm. The delay is added to the signal with use of convolving the input signal with an array A of length N, where $A(N) = 1$ and zero elsewhere. There are other ways of adding this delay, e.g. zero insertion in the input, but with varying delays convolution proved to be the easiest and most efficient way.

N is calculated from a time delay t multiplied with sampling frequency Fs. By taking the system latency into consideration the difference between delay and latency is Δt and N can be calculated from

$$N = \Delta t \times FS \tag{C.1}$$

From this it can be seen that it limits the system by having a minimum delay no smaller than the latency, meaning that there is a minimum distance between transmitter and receiver.

C.2 Sound pressure calibration

To get the right values on the in- and output of the emulator, there is need of calibration. From measurements we have that the input/output relation G is

$$11 = G \Rightarrow G_{dB} = 20\log(G) \tag{C.2}$$

The signal from the UTB is also damped by a factor of 51 dB from the transmitter. The computation of correct sound pressure level SPL is done as

$$SPL = S_v + 51 + G_{dB} + 20\log(Input) \tag{C.3}$$

where S_v is the transmit sensitivity described in section 2.5.

To acquire the correct value on the output the receive sensitivity M_v with a damping of 58 dB must be added. The damping comes from the transition between the UTB and the audio interface.

$$Output = M_v + 58 + SPL \tag{C.4}$$

C.3 GUI

GUI stands for Graphical User Interface, and is a way to show the program for the end user graphical instead of in codes. With push-buttons and input sections the program is more user-friendly and easier to understand. With the use of GUI one also reduce the possibility to mess up the code, and reduces the time spend to understand the program. This was some of the motivation for developing GUI in this project.

C.4 PN sequence

A Pseudo Noise signal (PN signal), is a signal created in a way such that the cross-correlation revels peaks when the signal is detected[16]. It exploits the characteristics similar to the *Barker sequence*, where the cross-correlation of a Barker sequence produces a peak, Proakis and Manolakis [16, p.145].

C.5 Filter Delay

Filter delay D from Proakis and Manolakis [16, p. 660-63]

$$\begin{aligned} D &= \tfrac{N}{2} \quad , N \quad even \\ D &= \tfrac{N-1}{2} \quad , N \quad odd \end{aligned} \tag{C.5}$$

Appendix D

Hydroacoustic Channel Emulator, HACE Manual

D.1 User manual

This is the user manual of the Hydro Acoustic Channel Emulator software created by Leo Hauge and Frank Hetland in 2015. The user manual gives a brief introduction of how to initialize and get started with the emulator. When following the steps listed below the emulator should work as wanted, and give credible results.

Sound card settings:

1. Install driver to the wanted sound card, here «OCTA-CAPTURE», go to Roland website.

2. At the "OCTA-CAPTURE" device (not software), press the button under "Direct Mixer" so it is at the "phono+main" option. Make sure that "MAIN OUT" is valued "max". If not there will be no outsignal at channel 1-2.

3. In "OCTA-CAPTURE"-software seen in Figure D.1: Go to -> Drive -> Drive settings, then select:

    - Audio Buffer Size (select 3)
    - ASIO direct monitor
    - Low latency mode
    - Match with the ASIO sample rate

To make sure that the routing is correct press: Device -> Open the Patch Bay: Make sure that the channels is patched from OUTPUT 1-2 to WAVE OUT 1-2 and like wise for the different channels. See Figure D.2.

- Turn off “Reverb type” which is found in the “MASTER” section in the mixer figure.
- Finally set the “Input Mixer” to -Inf.

4. In MATLAB: find “‘HOME” -> “preferences”, and select ASIO. This will reduce the latency and do the system more stable.

OPTIONAL: If you want to change the sound card to another brand, you have to know the name of the equipment as Matlab would find it. In MATLAB:

1. write in commando window: speaker = dsp.AudioRecoder; speaker.DeviceName=’ (the press TAB) Then you will see the name of your device, remember to select ASIO in "preferences" in Matlab!

2. Change the input sound card option at "INITIALIZATION"

MATLAB ChannelSimulator

1. Always connect the sound card BEFORE starting MATLAB. If not there can be problems with detecting the sound card.

2. Open the folder "GUI", and run the program "ChannelSimulator.m".

3. Press "initialize", a pre computed default file with the parameters from the GUI will be shown at the two figures. Left figure: Sound speed profile Right figure: eigenrays between the nodes (transmitter and receiver)

4. Change the "Sound Card" from "default" to "OCTA-CAPTURE", note that "default" will work, but it is NOT optimized and would be unstable.

5. The default freq. is set on 96000Hz, please check the sound card that the samplings freq is also set on 96000Hz.

6. Press "Calibration". Connect a jack cable from "input" to "output" as explained in the menu. Which input and output channel you want to use you can easy chose. This function finds the inherent latency

- Note: If the calibration is not runned defaults values will be used, some range deviation will then be expected but not crucial.

7. Change the vertical position of the Transmitter and the placement to the receiver at the left.

8. Press "Load Speed Profile" to load different sound speed profiles. It is located in a folder called "CTD". This will update the sound speed and depth. The sound speed profile is based on real measurements in Trondheim fjord and Horten (Breiangen)

9. Features: Features that can be checked in GUI:

   - No Delay : The signal will be sent directly through the sound card without any processing, this is just a feature to test the system
   - No multipaths: Only the direct signal path is sent
   - Doppler Spread: Introduces both scatter and time varying signal
   - Attenuation Sea: Introduces attenuation to the signal.
   - Record Data: Store the signal to a .wav file.

10. Seabed: Possible to select the shape by pre-designed ground shapes, and it is also possible to draw the bottom shape. Parameters to the seabed is found by pressing "Manual Seabottom"

    - Flat
    - Slowly varying sine
    - Rapidly varying sine
    - Gaussian
    - Linear increase
    - Linear decrease

11. Sea Surface: customize choose the level of sea state, salinity and temperature:

    - sea state: 0-9 level
    - Salinity
    - Temperature

12. When parameters are changed, and "Initialized" is pressed a textbox appears, write in your new file name for the project.

13. "Load Transducerdata" -> "directivityTransducer" (optional) "Load Transducerdata" -> "sensitivityTransducer" (optional) Transducer angle (works for point-to-point communication)

14. Before you start the simulator, make sure that you have connected the desired communication devices into the sound card. The routing is simple, for device 1: Input ch 1 and output ch 2, for device 2: Input ch 2 and output 2, for device 3: input ch 3 and output ch 3 and so on.

15. When the settings is ready press “Initialize” and the software will calculate the ray paths and impulse response channel between the nodes. This can take some time, based on the complexity of the input (range, sea state, sea bed, number of nodes etc).

16. After the initialization is done, press “START” and the real time channel emulator starts. (“START” button changes to “STOP” when pressed)

17. Press “STOP” when you want to stop the simulation process.

18. "Analyze" Plots the results, and option for playback

    - Impulse response
    - Frequency response
    - Ambient noise
    - Reflectionloss
    - Waterabsorption
    - Intensity Raytracing
    - Recorded Data
    - Transducer Directivity
    - Doppler Spread

Optional: Create speed profiles. Make a .txt file with distances in column one and sound speed in column two. Sound speed measurements achieved as .cvn files are also supported.

Figure D.1: OCTA-CAPTURE mixer settings,

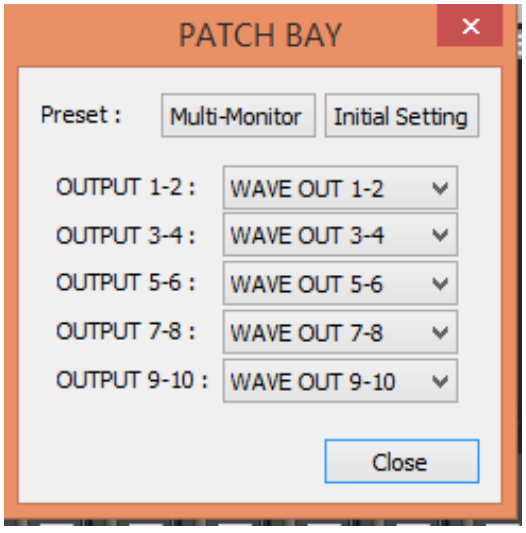


Figure D.2: OCTA-CAPTURE: Patchbay

Appendix E

Results

E.1 Calibration Statistics

The calibrated latency results for 100 runs for two different computer are shown in figure E.1 and E.2. Each plot is shows the relationship between frame size and latency.

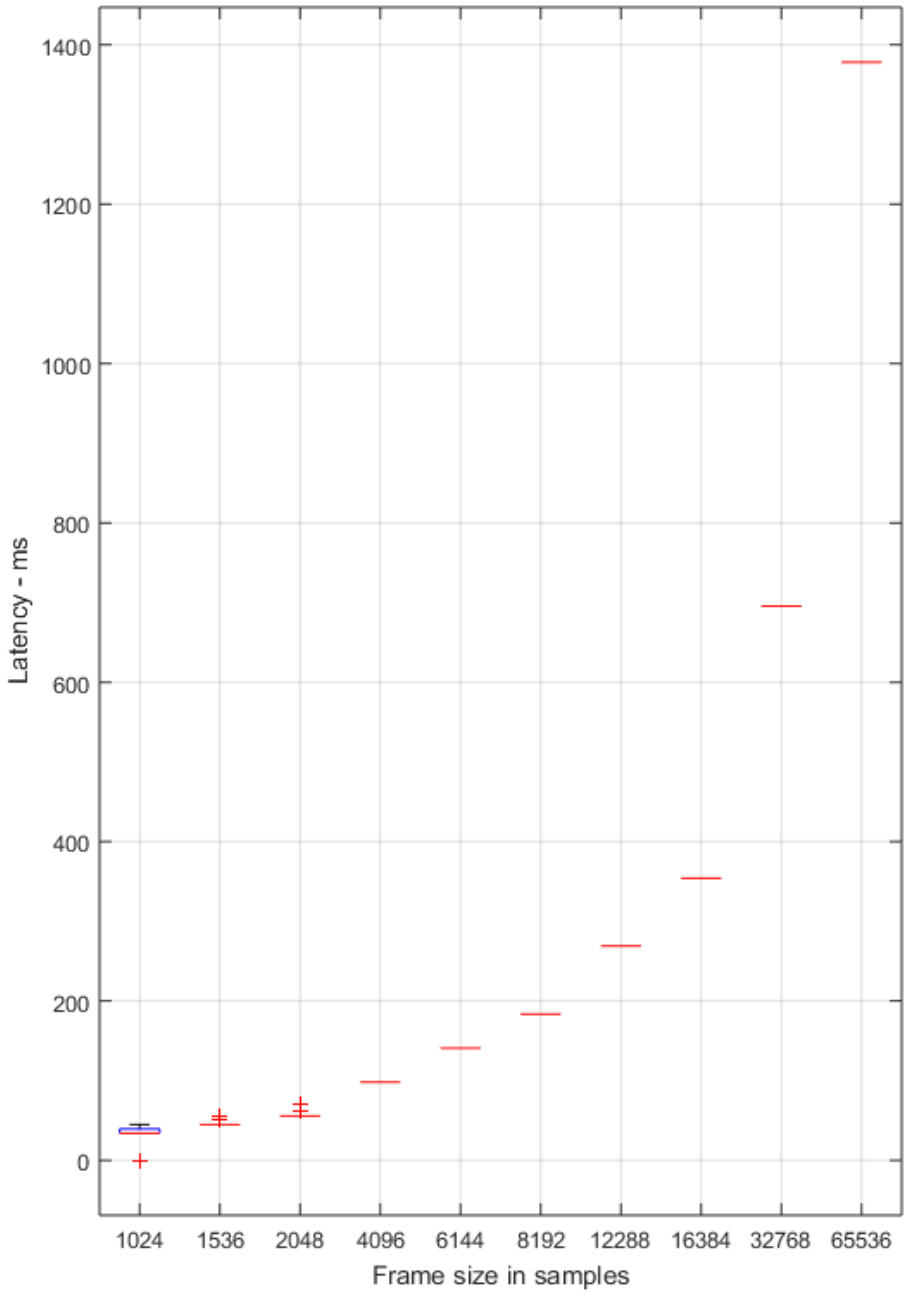


Figure E.1: Calibration results 100 runs. Plotted with frame size versus latency in milliseconds

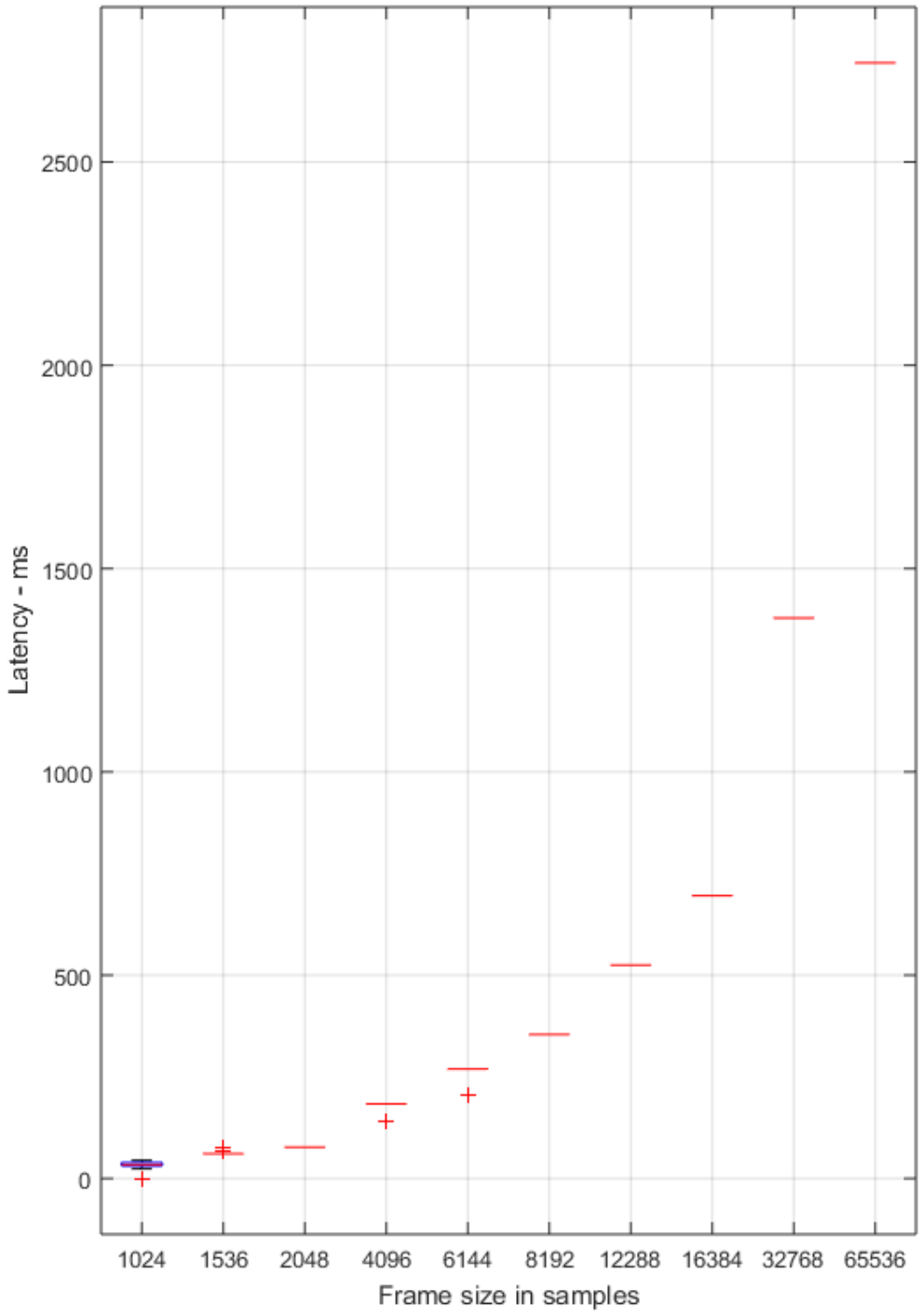


Figure E.2: Calibration results 100 runs. Plotted with frame size versus latency in milliseconds

E.2 Print file from one calibration

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 1024

Latency in ms is : 34.22

Latency in samples is : 3284

Minimum distance is : 52.70

Validation of results : 1

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 1536

Latency in ms is : 44.89

Latency in samples is : 4308

Minimum distance is : 69.12

Validation of results : 1

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 2048

Latency in ms is : 55.55

Latency in samples is : 5332

Minimum distance is : 85.55

Validation of results : 1

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 4096

_____

Latency in ms is : 98.22

Latency in samples is : 9428

Minimum distance is : 151.26

Validation of results : 1

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 6144

_____

Latency in ms is : 140.89

Latency in samples is : 13524

Minimum distance is : 216.96

Validation of results : 1

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 8192

_____

Latency in ms is : 183.55

Latency in samples is : 17620

Minimum distance is : 282.67

Validation of results : 1

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 12288

_____

Latency in ms is : 268.89

Latency in samples is : 25812

Minimum distance is : 414.08

Validation of results : 1

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 16384

_____

Latency in ms is : 354.22

Latency in samples is : 34004

Minimum distance is : 545.50

Validation of results : 1

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 32768

Latency in ms is : 695.55

Latency in samples is : 66772

Minimum distance is : 1071.15

Validation of results : 1

****

****

Sampling frequency: 96000

SoundCard Number of channels: 5

Samples per frame : 65536

Latency in ms is : 1378.22

Latency in samples is : 132308

Minimum distance is : 2122.46

Validation of results : 1

E.3 Plot of ray trace with different seabed

To give a short demonstration of the water simulation program features show some different plots with point-to-point and network with different seabed and sea state will be shown. The first figure 4.12 is a point-to-point communication with 1500 meter range between the nodes, and a depth of 20 meter on both nodes.

As one can see from the plot, the direct transmission is illustrated by the red line, reflections are the black lines, first ground reflection are the blue lines. The sea bed is color coded. Ocean depth can be seen in the z-axis and the horizontal distance can be seen in the x-axis of the plot.

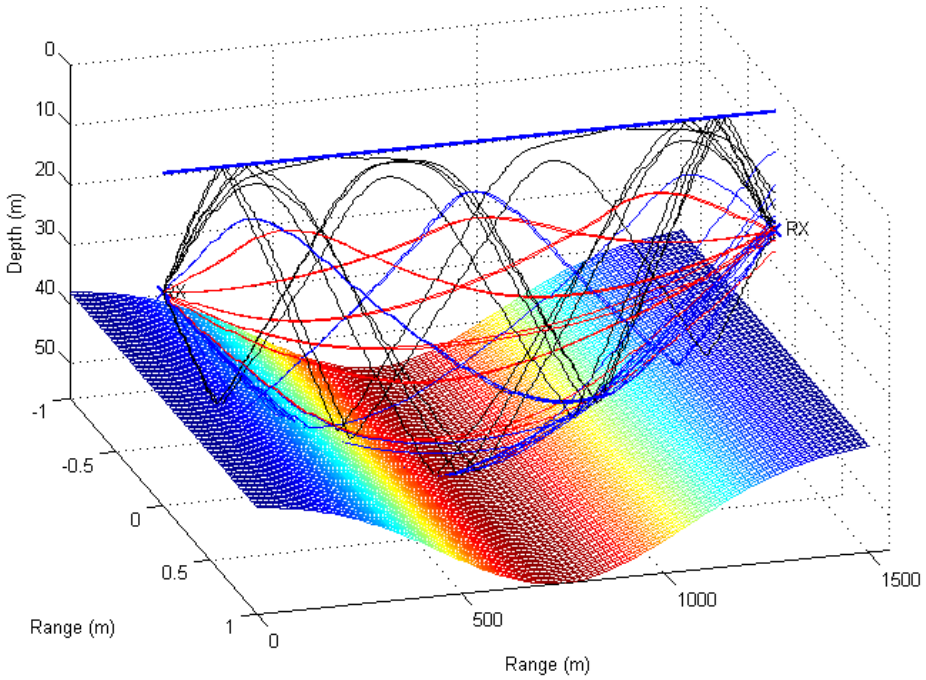


Figure E.3: Point-to-point ray trace. 1500 meters range.

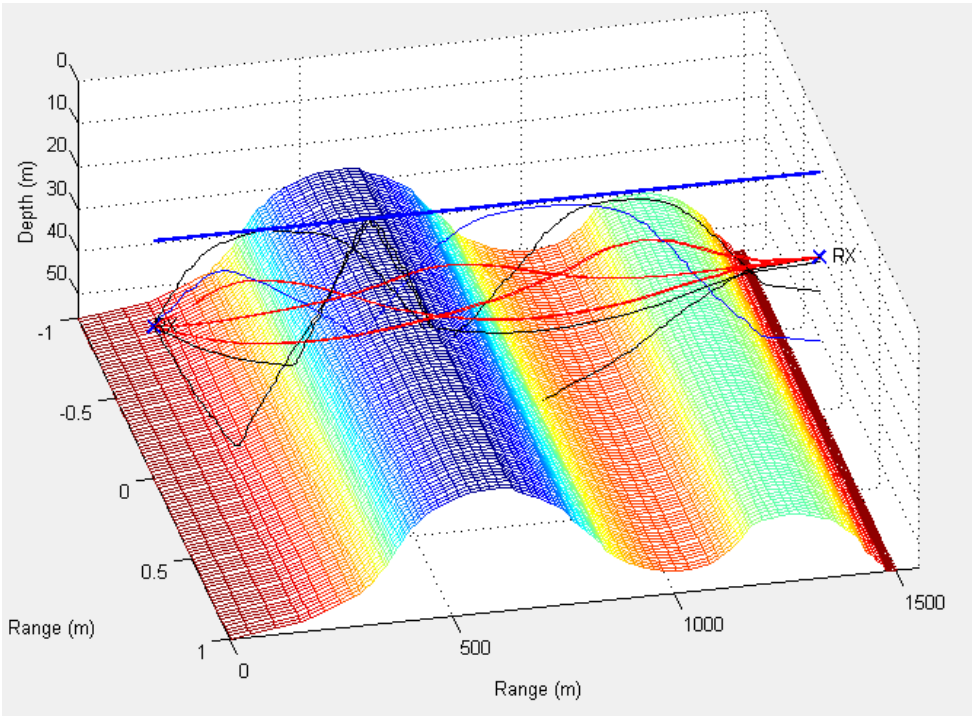


Figure E.4: Point-to-point: Free hand drawing of seabed

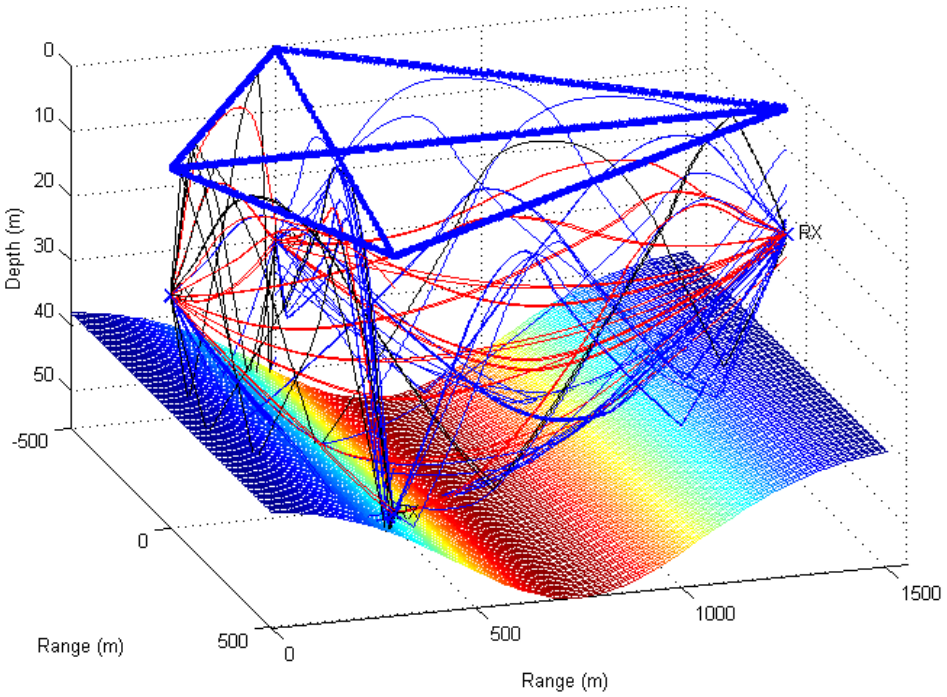


Figure E.5: Network communication. Seabed: invert gaussian, with 20 meter variation

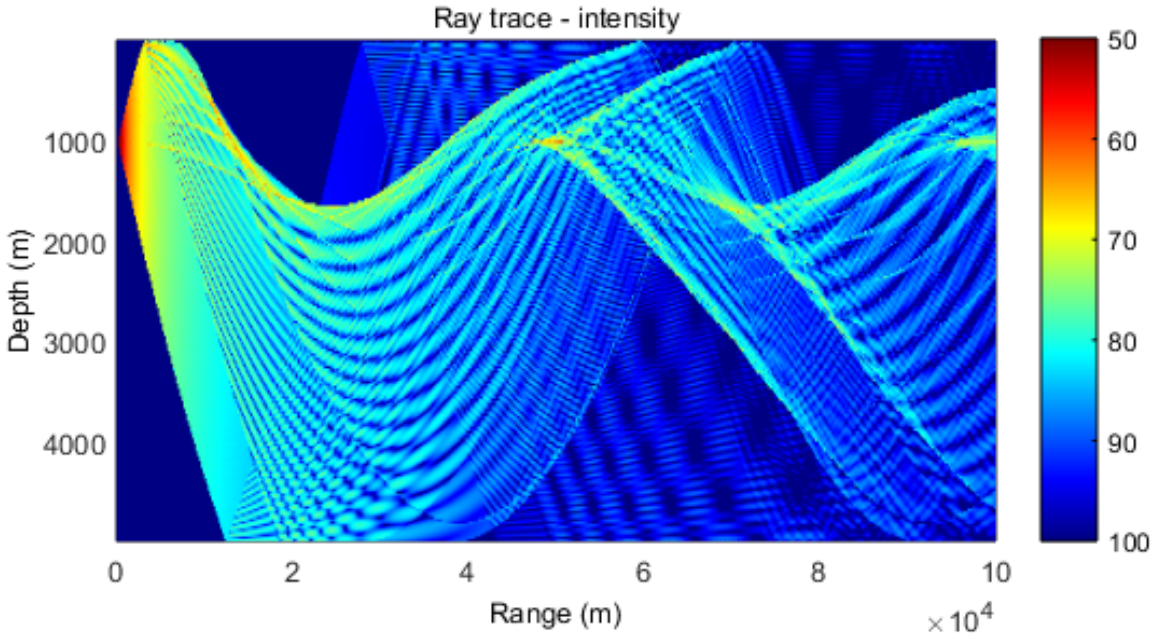


Figure E.6: Intensity ray trace for 10 000 meter range.

E.4 Ray trace according to positions during the sea trial

Ray traces from sea trial in Breiangen simulated in HACE corresponding to position given in Table 4.6.

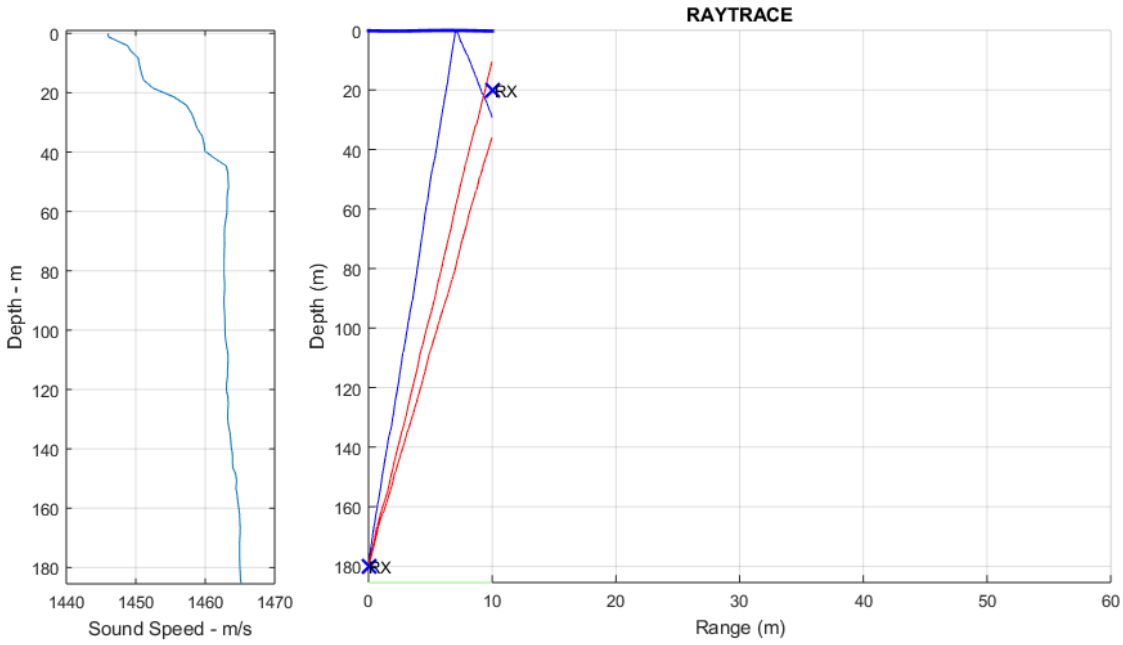


Figure E.7: Position 1: Ray trace at 10 meter horizontal distance between nodes

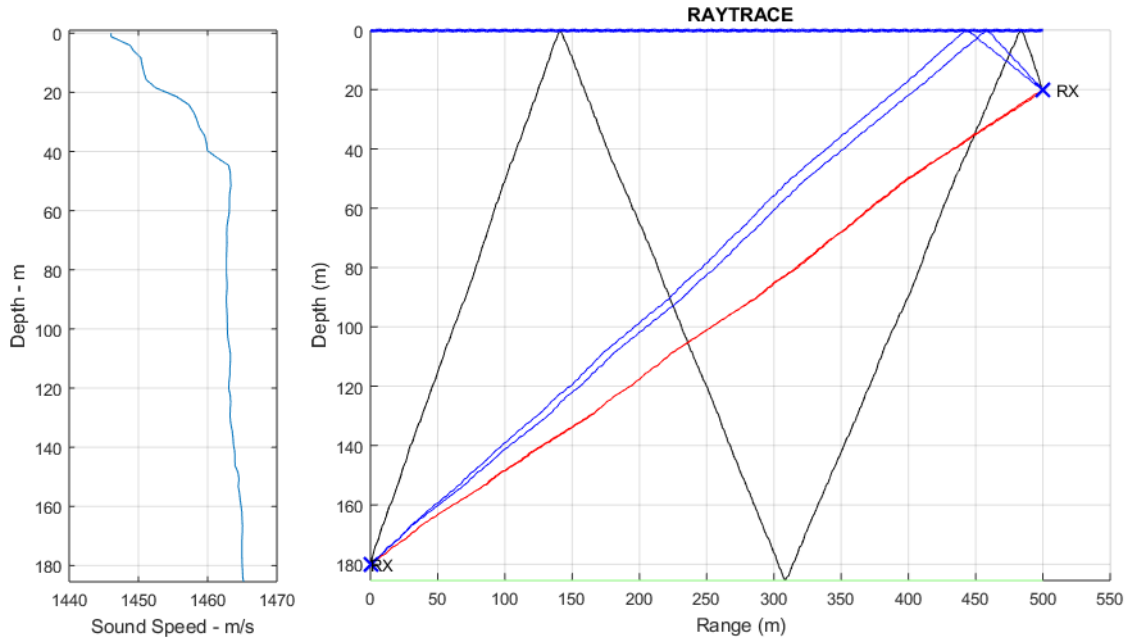


Figure E.8: Position 2: Ray trace at 500 meter horizontal distance between nodes

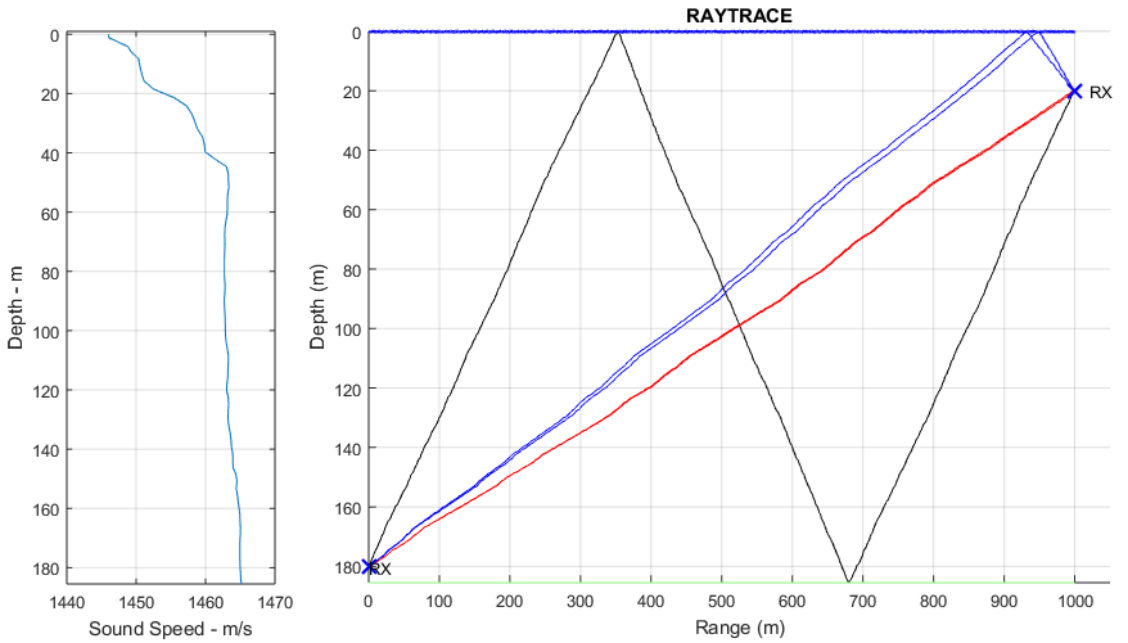


Figure E.9: Position 3: Ray trace at 1000 meter horizontal distance between nodes

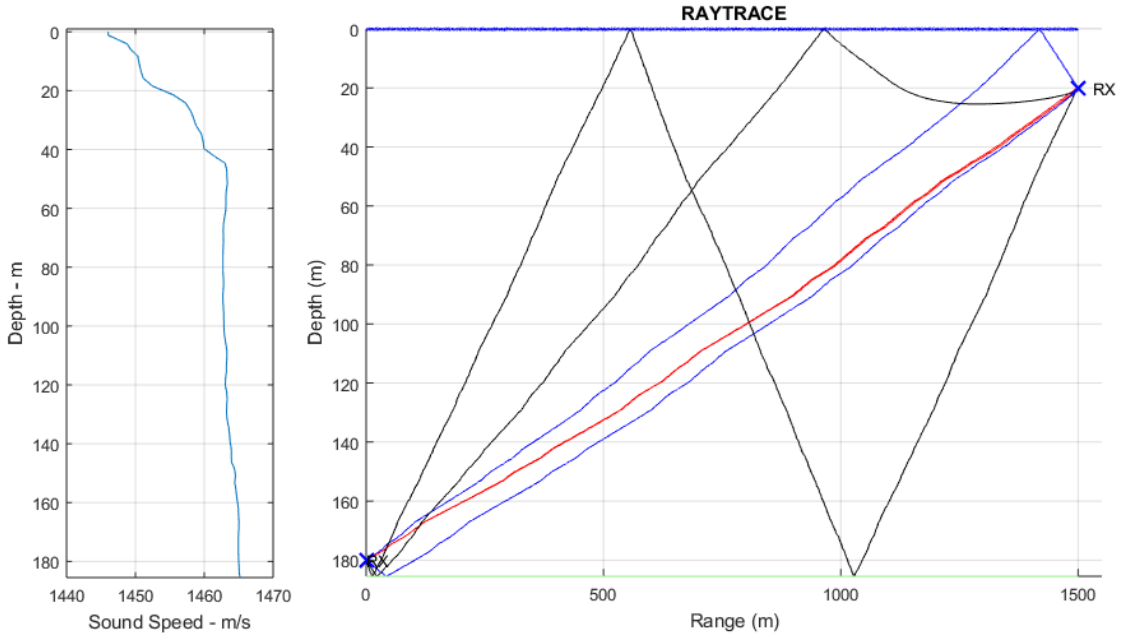


Figure E.10: Position 4: Ray trace at 1500 meter horizontal distance between nodes

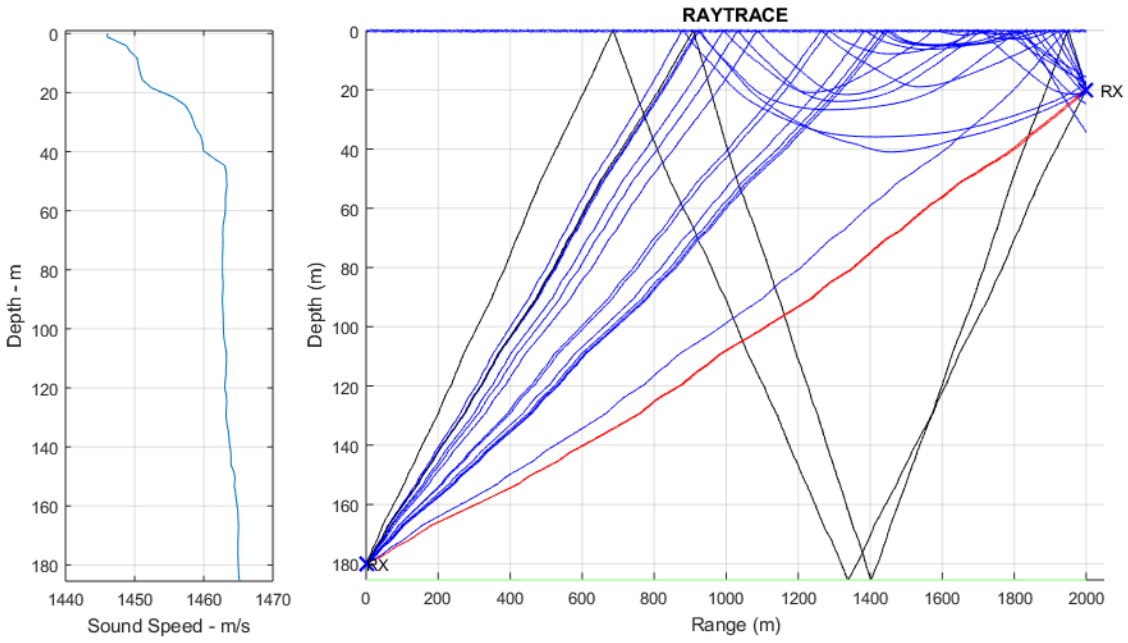


Figure E.11: Position 5: Ray trace at 2000 meter horizontal distance between nodes

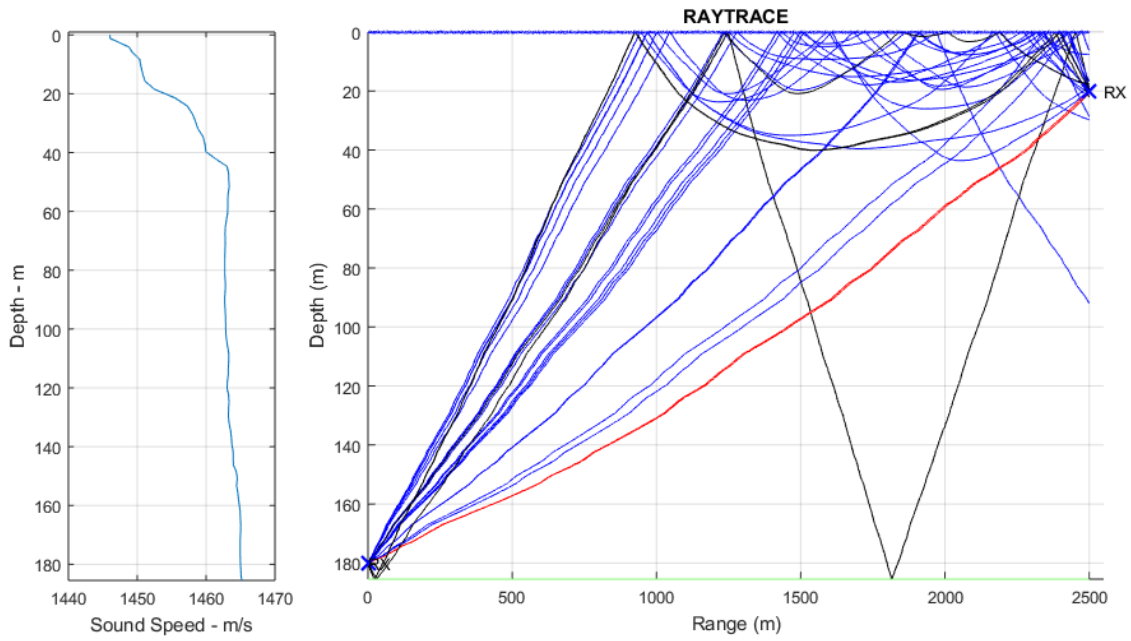


Figure E.12: Position 6: Ray trace at 2500 meter horizontal distance between nodes

E.5 Water absorption filter

The figures show frequency filter responses of the water absorption filter. The designed filter is plotted versus the ideal filter.

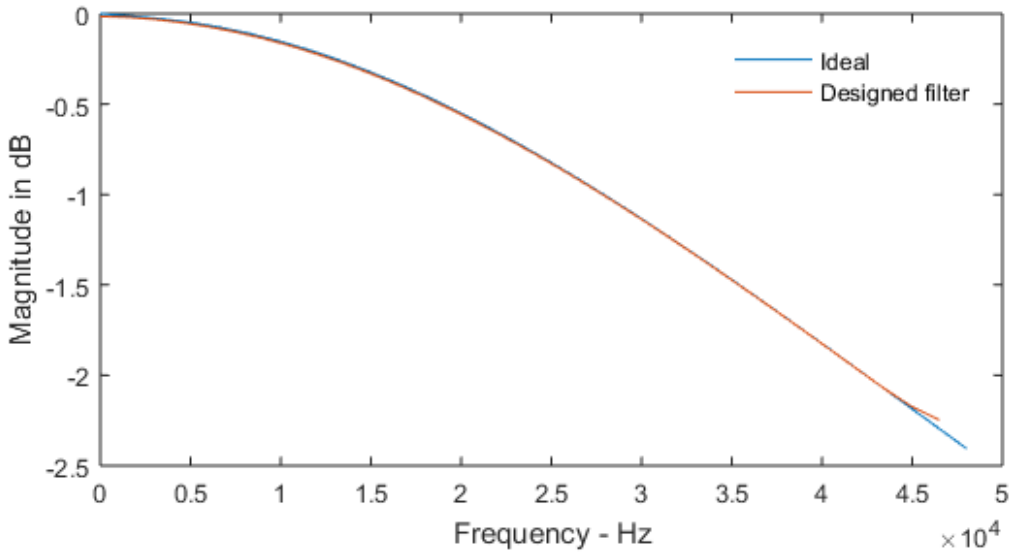


Figure E.13: Frequency response of an absorption filter for 150 meters

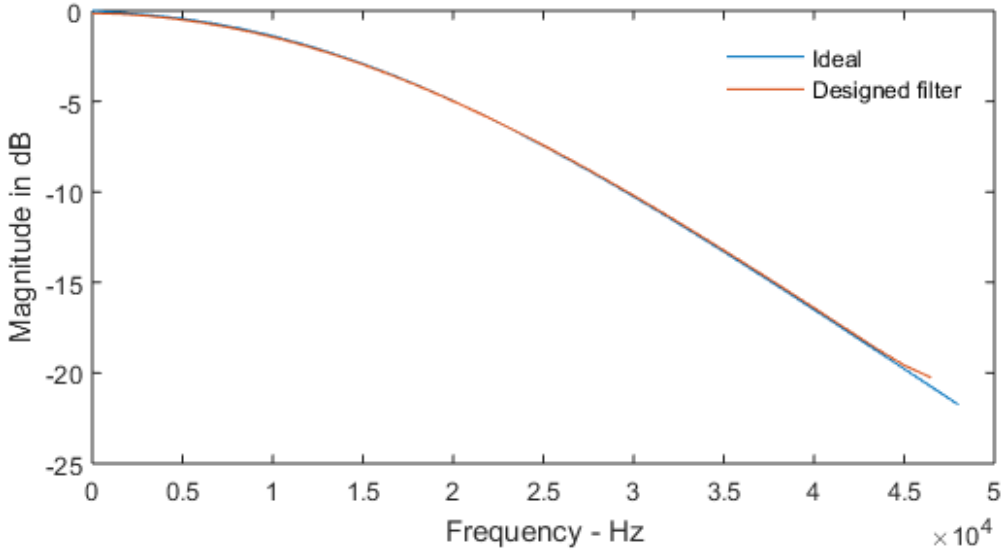


Figure E.14: Frequency response of an absorption filter for 1500 meters

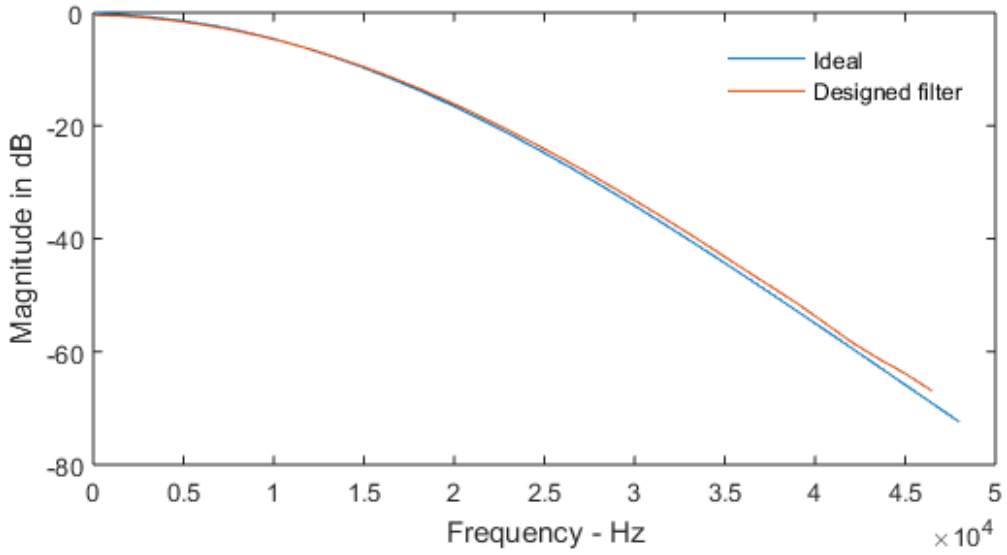


Figure E.15: Frequency response of an absorption filter for 5000 meters. A fixed number of filter coefficients lead to deviations in the high frequencies

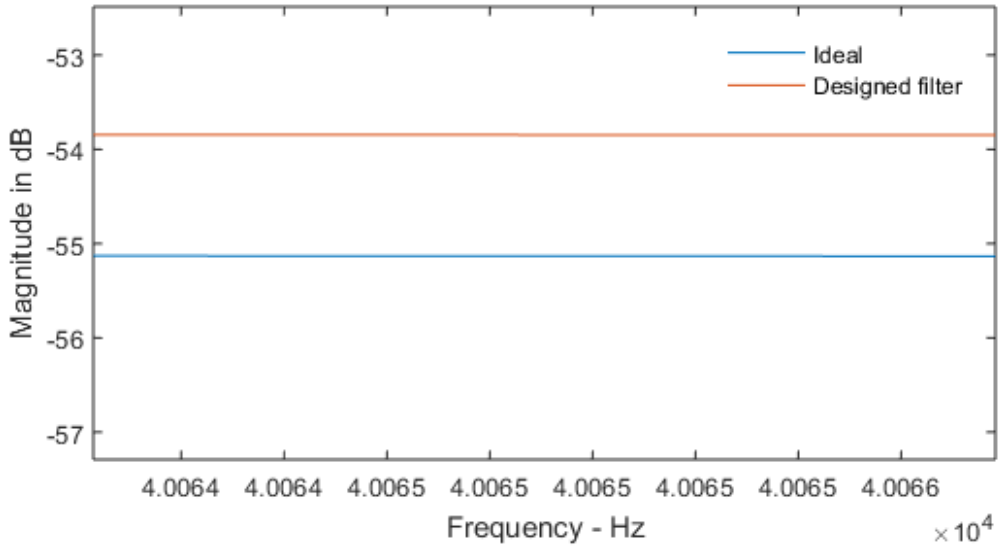


Figure E.16: Frequency response of an absorption filter for 5000 meters. Around 1 dB deviation in the high frequencies

E.6 Jitter

Figure E.17, E.18, E.19, E.20, E.21 and E.22 show how the jitter influence the detected ranges over a time period. In figure E.23 crosstalk is illustrated by the oscillating detection of range, changing from 2000 meters to 1000 meters.

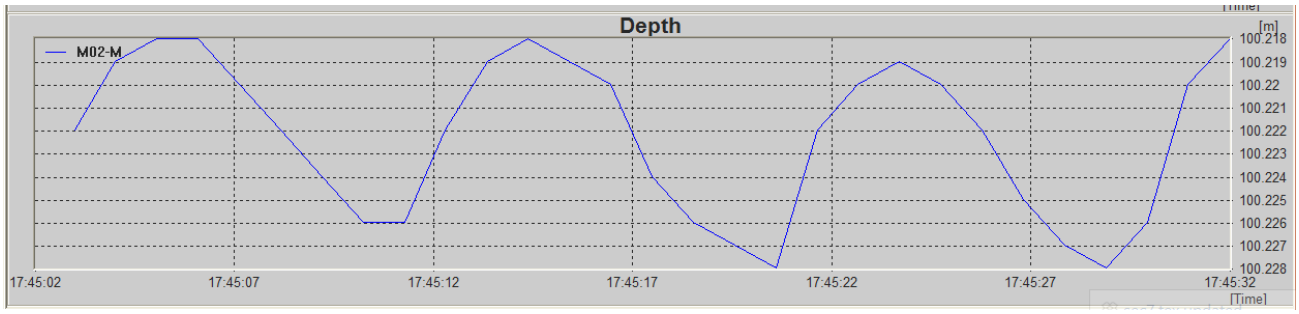


Figure E.17: Jitter 100 meter

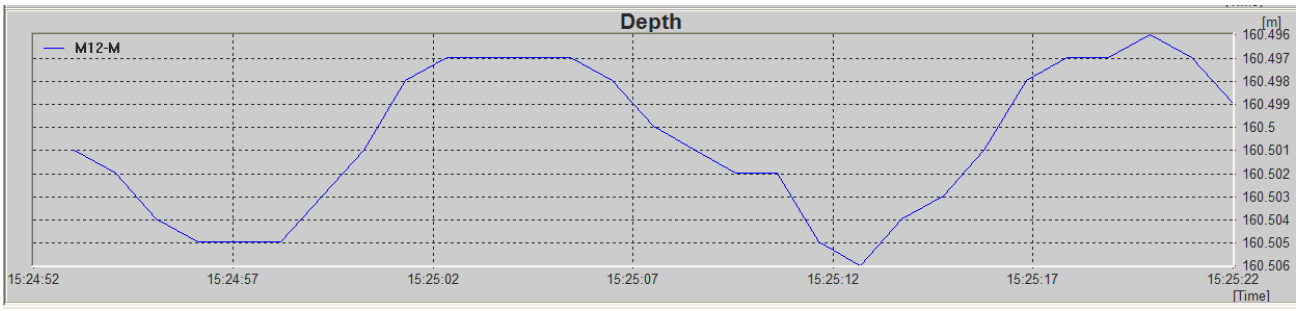


Figure E.18: Jitter 10 meter(On top), depth 180 meters, and 20m

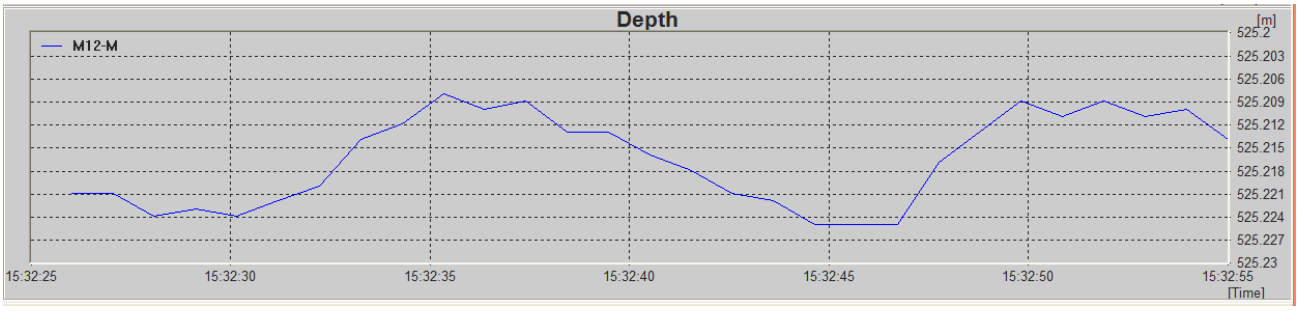


Figure E.19: Jitter 500 meter

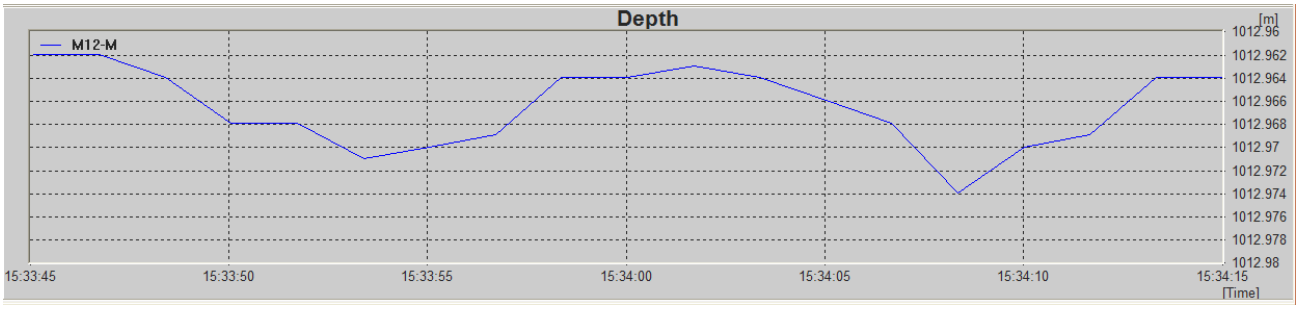


Figure E.20: Jitter 1000 meter

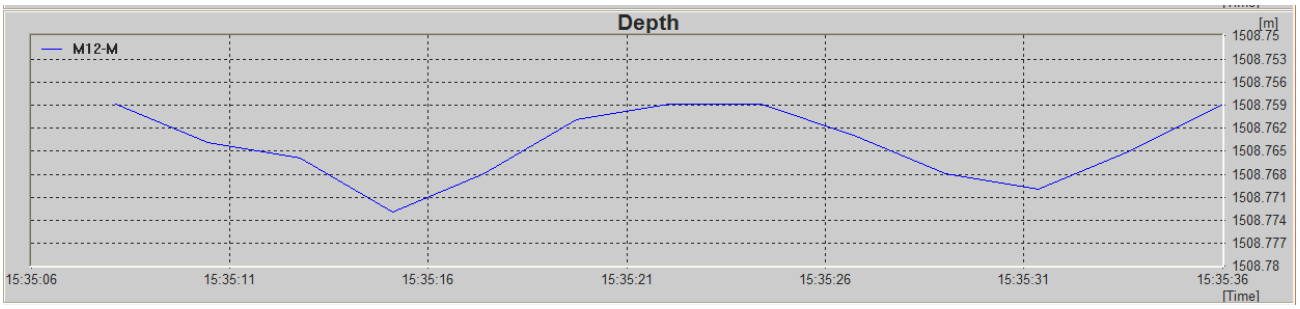


Figure E.21: Jitter 1500 meter

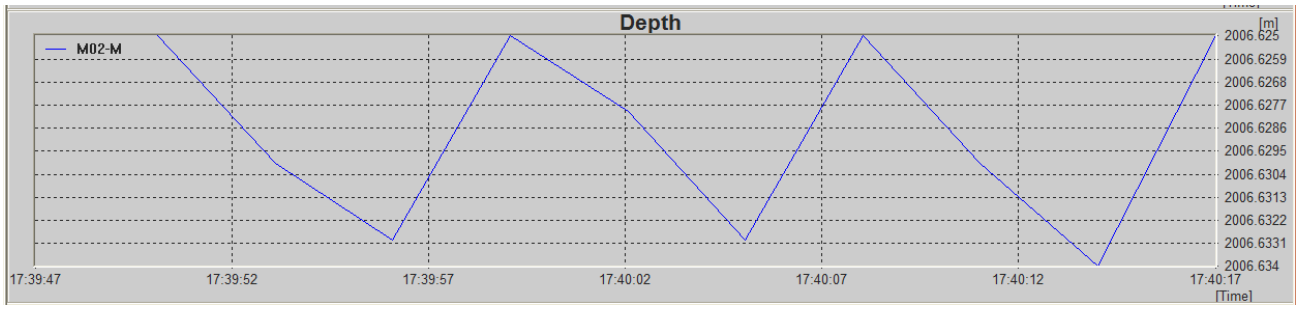


Figure E.22: Jitter 2000 meter

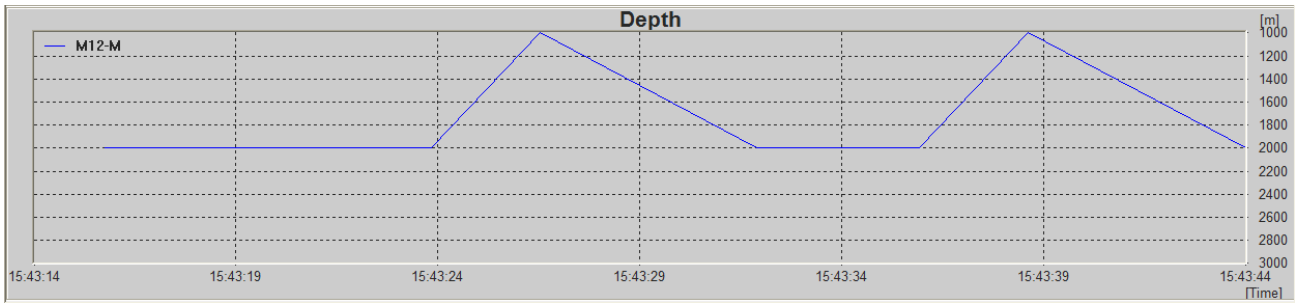


Figure E.23: Crosstalk 2000meters and 1000 meters

E.7 Results Noise

In Figure E.24 the power spectral density of the ambient noise added in the simulations is shown. Figure E.25 the power spectral density of the ambient noise during the sea trial is shown.

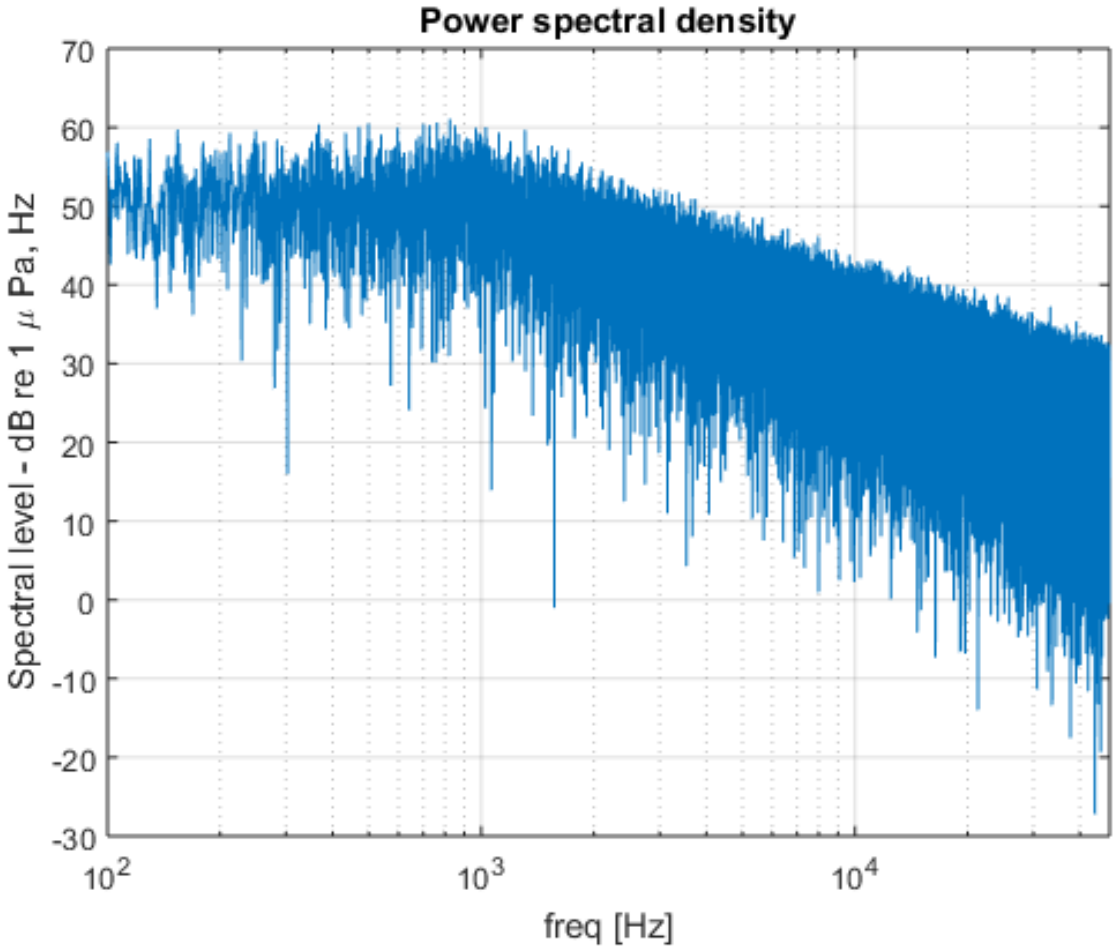


Figure E.24: Frequency response of the recorded noise at the simulator

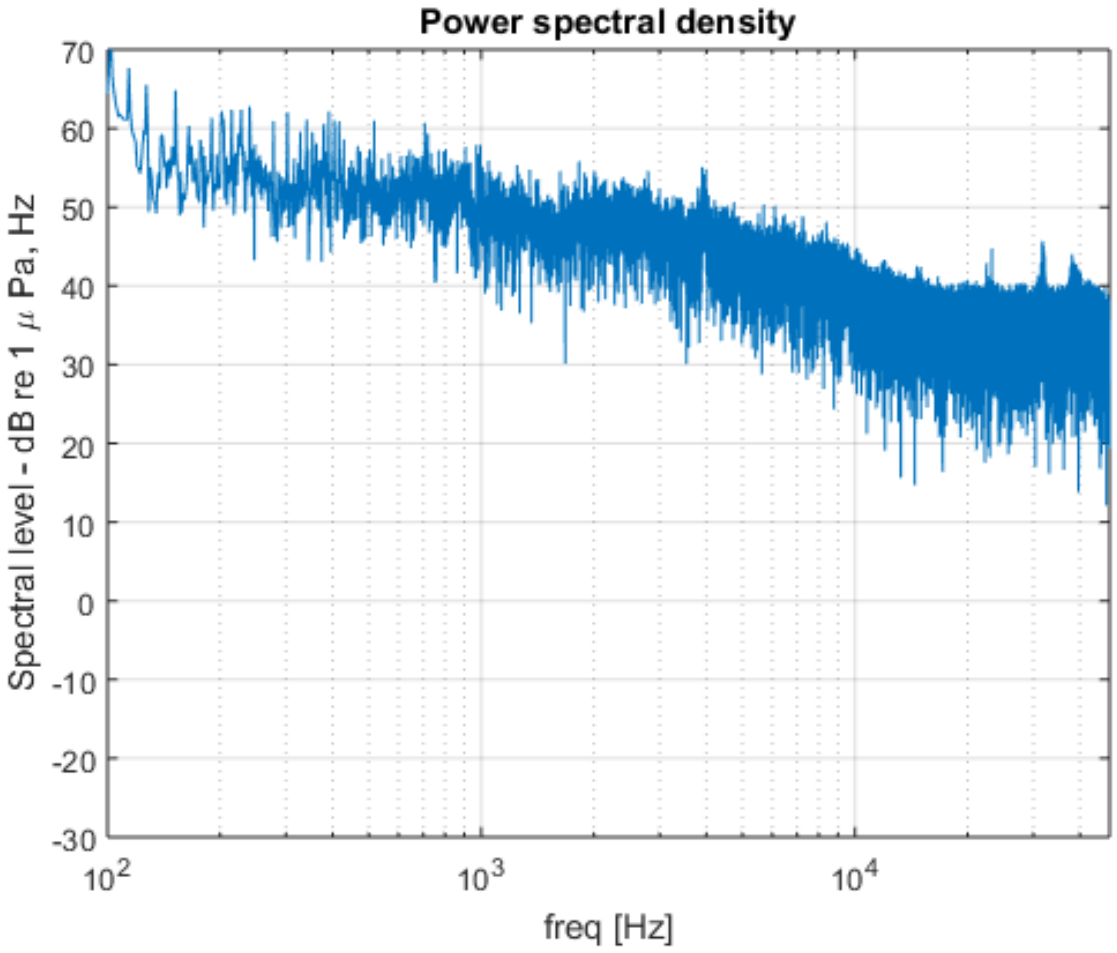


Figure E.25: Frequency response of the recorded noise at the simulator

Appendix F

Schematics

F.1 Block Scheme of El Interface

Utkast dempeledd, EL

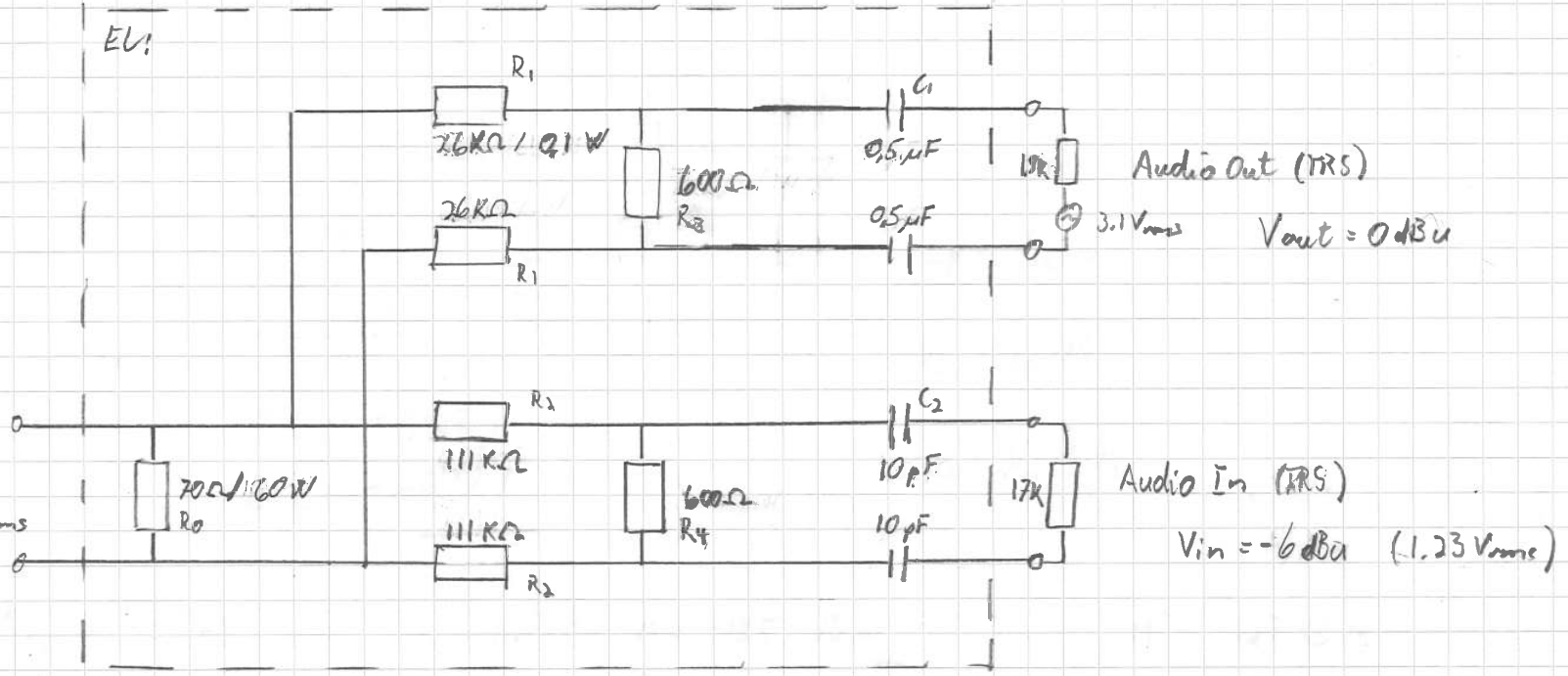
TH 2014.09.29

$Att_{UTB - Audio\ In} = 51\ dB$

$Att_{UTB - Audio\ Out} = 41\ dB$

$Att_{Audio\ Out - UTB} = 68\ dB$ → Resulting in hard limiter of e.g. > 132 dB re 1 µPa for HiPAP
OK for most scenario with reasonable noise levels

$Att_{Audio\ Out - Audio\ In} = 92\ dB$



ac-coupling cut off : 130 Hz

F.2 Transducer Data

KONGSBERG

Transduserdata til linksimulator

Til å simulere ein akustisk link der også transduserane blir modellert som ein del av linken har det blitt tatt fram transduserdata til å bruke i simulatoren.

Innhald

1 Dokumenthistorie

Dato:	Signatur:	Tekst:
2014-09-12	SSk	Første utkast.
2014-09-15	SSk	Utvida dokument til å ta med impedans, fase og retningsdiagram tilomnidireksjonal transduser.

2 Bakgrunn

Til å simulere ein akustisk link der også transduserane blir modellert som ein del av linken har det blitt tatt fram transduserdata til å bruke i simulatoren.

3 Generelt om datasetta

Datasetta i dette dokumentet er representative for kva ein kan forvente av yting til transduserar, men er ikkje data frå ein bestemt svingar eller type svingar.

4 Terminologi og definisjonar

Transduserane omtala i dette notatet er alle aksesymmetriske og symmetriaksen blir då ein referanse saman med punktet der symmetriaksen skjærer overflata til transduseren. Akustisk måling av frekvensrespons er gjort slik at den akustiske målinga refererar til eit punkt på symmetriaksen ein meter framfor der aksen går gjennom overflata til transduseren, sjå Figur 1.

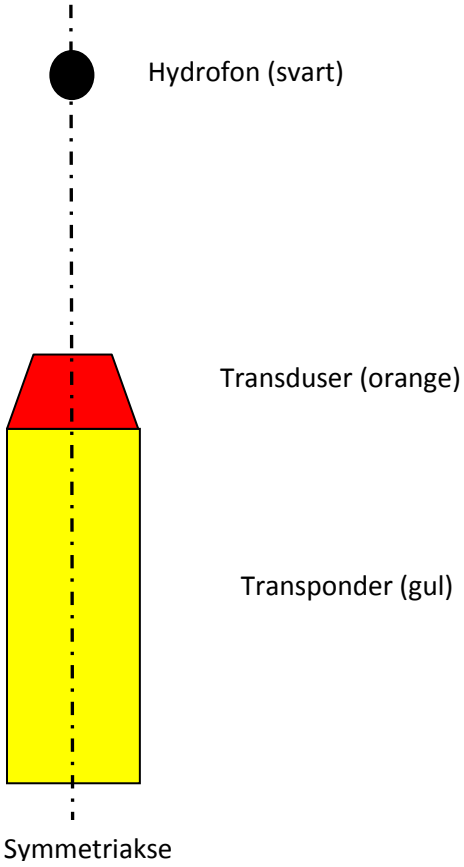
Verdiar for avgjeve lydtrykk relativt påtrykt spenning og teoretisk spenning over opne terminaler gjeve motatt lydtrykk er tatt opp ved måling. Samtidig er spenning over transduseren, straum tilført transduseren og fasen mellom desse målt og elektrisk impedans, Z, og fase til transduseren berekna.

Retningsdiagram er utført ved å rotere transduseren rundt punktet der symmetriaksen skjærer overflata til transduseren samtidig som hydrofonen blir halden i ro. Retningsdiagramma er slik gjort i eit plan som skjærer symmetriaksen der null grader referere til eit punkt på aksen. Sidan transduseren er aksesymmetrisk så betyr orientering av dette planet i rotasjon rundt aksen ikkje noko, alle orienteringar gjev same resultat. Retningsdiagramma er alle skalert slik at høgaste verdi er 0, rett fram for transduseren det frekvensresponsen er målt og avviket frå null etter som transduseren er rotert er vist.

5 Figur av transduser på transponder

I Figur 1 er det vist ei skisse av transduser montert på transponder med symmetriaksen til transduseren teikna inn. Framfor transduseren er plassering til hydrofon ved måling av frekvensrespons vist.

Figur 1, skisse av tansduser på transponder med symmetriakse til transduseren teikna inn.



6 Transduserdata

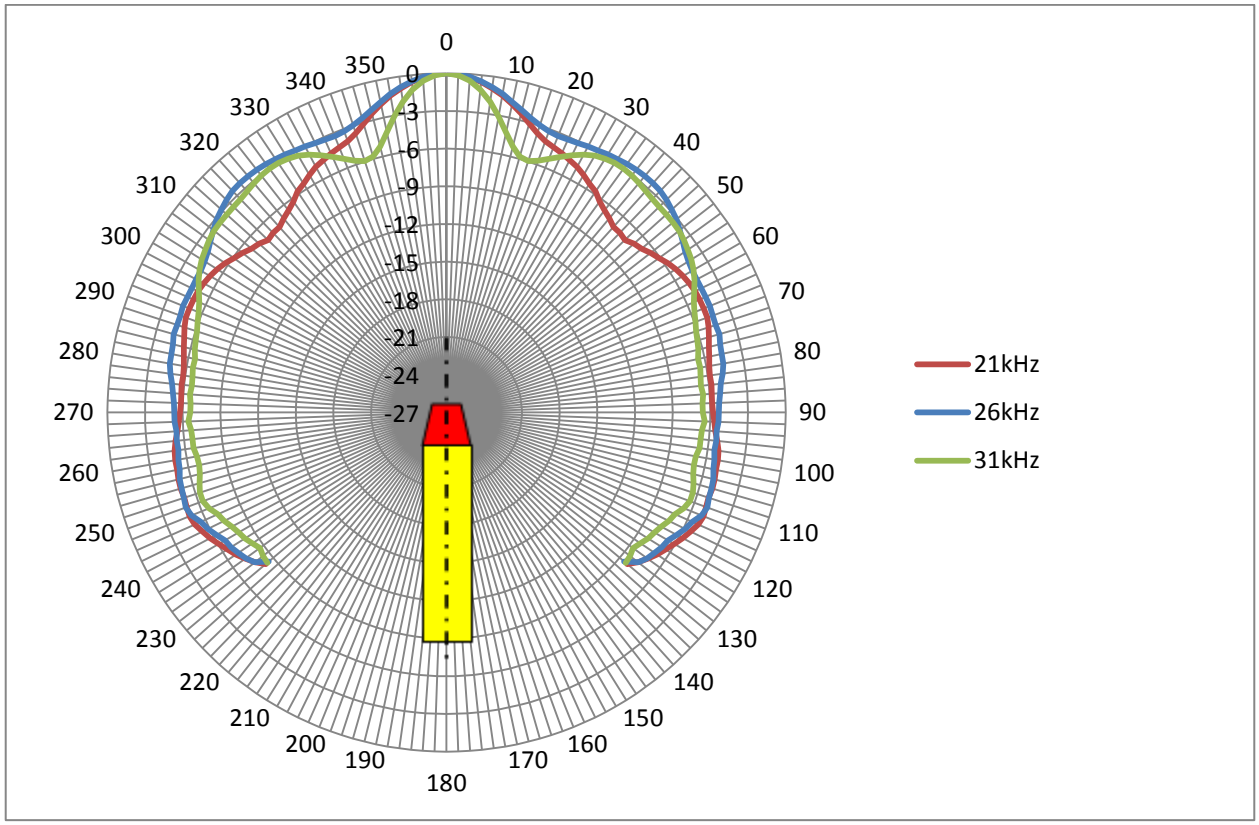
6.1 Omnidireksjonal melomfrekvens transduser

Første datasett er for ein transduser som sender lyd omnidireksjonalt i eit halvplan. Med dette meiner ein at lydeenergien blir fordelt mest mogeleg jamt frå symmetriaksen til ein possisjon 90 grader på symmetriaksen. I praksis er det ikkje oppnådd heilt jamn fordeling av lydeenergien, sjå retningsdiagram i Figur 2 for korleis lyden er fordelt. Data brukt til å teikne Figur 2 med er tabulert i Tabell 2. Tabell 1 viser frekvensrespons til transduseren rett framfor transduseren. Referer til Figur 1 og avsnitt 5 for plassering av målepunkt.

Tabell 1, frekvensrespons til kildenivå relativt påtrykt spenning og spenning relativt påtrykt lyd.

Frekvens [kHz]	Su [dB re. 1µPa/v @1m]	M [dB re. V/µPa]	Impedans, Z [ohm]	Fase [Grader]
20	144,941	-194,273	182,1228	70,49487
21	147,2077	-190,374	208,3186	53,16154
22	149,6318	-187,695	175,8382	30,87692
23	152,0987	-186,278	147,2009	11,61795
24	153,6259	-186,86	117,3575	-1,14615
25	154,6264	-188,173	92,07879	-5,45385
26	154,8205	-189,994	75,90442	-1,04103
27	155,0592	-190,211	75,29652	8,338461
28	154,7479	-189,709	85,89026	12,83077
29	154,0385	-189,335	101,095	10,48718
30	153,2708	-189,323	115,4839	1,807692
31	152,7505	-189,92	124,3816	-9,37179
32	152,6067	-190,629	133,9624	-17,5385

Figur 2, retningsdiagram til omnidirekgsjonal transdsuer på 21kHz, 26kHz og31kHz.



Tabell 2, tabulerte verdiar for retningsdiagram.

Vinkel	21kHz	26kHz	31kHz
0	0	0	0
2	-0,08363	-0,06048	-0,13173
4	-0,26155	-0,22577	-0,51833
6	-0,54345	-0,50167	-1,15256
8	-0,94369	-0,83732	-2,02649
10	-1,38333	-1,25196	-3,0706
12	-1,89083	-1,7353	-4,20214
14	-2,46982	-2,19643	-5,17827
16	-3,08494	-2,61292	-5,81595
18	-3,63518	-2,95363	-5,95399
20	-4,01286	-3,16661	-5,68161
22	-4,22137	-3,27845	-5,21964

24	-4,42857	-3,22881	-4,72101
26	-4,65363	-3,14762	-4,20899
28	-4,85036	-3,01637	-3,7406
30	-5,17179	-2,9047	-3,39268
32	-5,54065	-2,77815	-3,18286
34	-5,7894	-2,65833	-3,05667
36	-6,26821	-2,55589	-3,00631
38	-6,56863	-2,46393	-3,04089
40	-6,83458	-2,43286	-3,15363
42	-7,14333	-2,42982	-3,28524
44	-7,12071	-2,48833	-3,3897
46	-7,23643	-2,6497	-3,44375
48	-6,93065	-2,86923	-3,45137
50	-6,72321	-3,09976	-3,47012
52	-6,38244	-3,43649	-3,51214
54	-6,09923	-3,7703	-3,63768
56	-5,73857	-4,08565	-3,8422
58	-5,47387	-4,2931	-4,01792
60	-5,27202	-4,41304	-4,29708
62	-5,09726	-4,42685	-4,63994
64	-4,98214	-4,43649	-5,06905
66	-4,95417	-4,42012	-5,45256
68	-4,91268	-4,37702	-5,71815
70	-4,90244	-4,45488	-5,90994
72	-5,07012	-4,46429	-6,10929
74	-5,27006	-4,40042	-6,22524
76	-5,45857	-4,55042	-6,34006
78	-5,61512	-4,61417	-6,52827
80	-5,76071	-4,62214	-6,49935
82	-5,82649	-4,80536	-6,57298
84	-5,80054	-5,03863	-6,62494
86	-5,82554	-5,17054	-6,52167
88	-5,83917	-5,28054	-6,61577
90	-5,77976	-5,31655	-6,58958
92	-5,66643	-5,34286	-6,43685
94	-5,54208	-5,44976	-6,6256
96	-5,32232	-5,41762	-6,65923
98	-5,12935	-5,46107	-6,64744
100	-5,07589	-5,33625	-6,83952

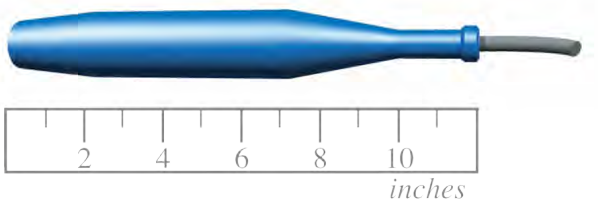
102	-5,07554	-5,27804	-6,86673
104	-5,00827	-5,09994	-6,76452
106	-4,98214	-5,06762	-6,53857
108	-5,05208	-5,02464	-6,36929
110	-5,0053	-4,90827	-6,3625
112	-4,94589	-5,10476	-6,63256
114	-5,06714	-5,59202	-7,08089
116	-5,38113	-5,86524	-7,3025
118	-5,69589	-6,25256	-7,64762
120	-6,05577	-6,64107	-7,89185
122	-6,42179	-6,77821	-8,0897
124	-6,7744	-7,0175	-8,3744
126	-7,19679	-7,25375	-8,64018
128	-7,64869	-7,62214	-8,44929
130	-8,22464	-8,53595	-8,37875

F.3 Hydrophone Data

Model ITC-8095

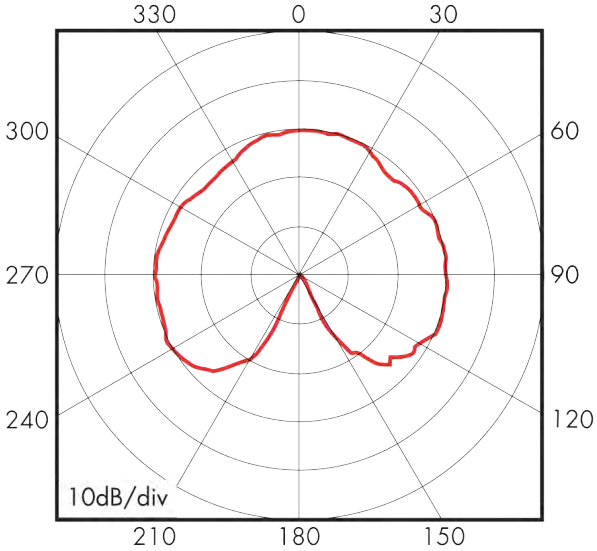
Special Purpose Preamplified Hydrophone

The Model ITC-8095 is a spherical, omnidirectional broadband hydrophone using the ITC-1089D sphere. It is supplied with a preamplifier.

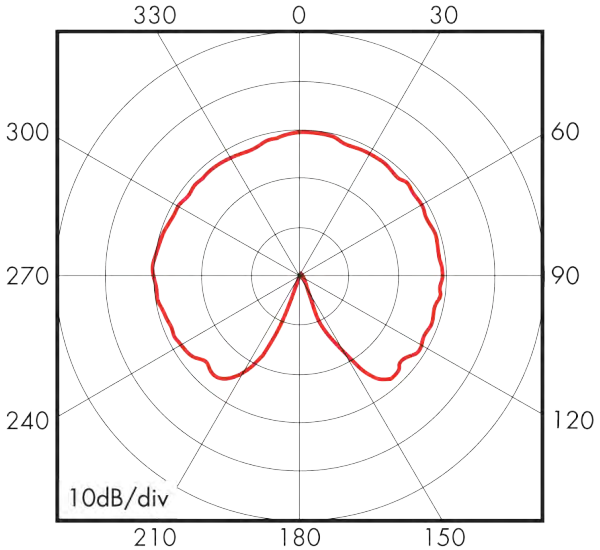


Specifications (Nominal)

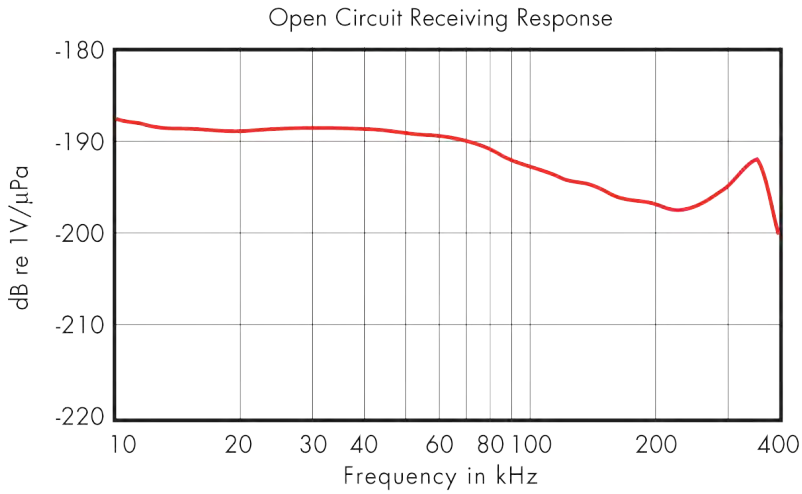
Midband OCV	-193 dB//1V/µPa
Midband Beam Pattern Shape	Omnidirectional
Usable Frequency Range	.002 - 300 kHz
Preamplifier Type	Single in/Single out, Voltage
Depth	1500 meters



Directivity Pattern at 300.0 kHz



Directivity Pattern at 100.0 kHz



869 Ward Drive, Santa Barbara, CA 93111
805.683.2575 • 805.967.8199 FAX
International Transducer Corporation
www.itc-transducers.com