2019/2020

FINAL YEAR PROJECT

Submitted in fulfillment of the requirements for the

ENGINEERING DEGREE FROM THE LEBANESE UNIVERSITY
FACULTY OF ENGINEERING – BRANCH II

Major: Electrical and Electronics Engineering

Prepared By:

Charbel MANSOUR

Environment-Specific Microphone Array-Based 3D Sound Source Localization System

Supervised by:

Dr. Karim YOUSSEF

Defended on 30th of July 2020 in front of the jury:

Dr. Carine KASSIS	**President**
Dr. Lise SAFATLY	**Member**
Dr. Maria EL ACHKAR	**Member**

Acknowledgment

I would like to express my gratitude and appreciation to all those who helped me and gave the possibility to complete this project. A special thanks to my supervisor Dr. Karim Youssef for his support. Even though the environment was a little bit challenging to complete this project during the pandemic, he was always ready to give the help required and to create a supportive learning environment that allowed me to learn a wide range of new skills. I would like to extend my gratitude to our chairperson Dr. Carine Kassis, and to the Electrical and Electronics Engineering Department at ULFG-II as a whole.

Abstract

In this work, a microphone array is used in combination with a machine-learning approach to estimate the position of a sound source in three different environments, and testing under various conditions. The sound source localization in this project is performed using features extracted from sound signals such as the time difference of arrival and the energy difference of sound signals acquired with the microphone array. The microphone array in this work consists of four microphones placed near the corners of a rectangular room and the estimated position is expressed in terms of coordinates in a three-dimensional Cartesian system. Extracted features are exploited by a deep artificial neural network trained with the backpropagation algorithm to output the sound source coordinates. The established localization system is evaluated in three different environments simulated to provide different acoustic conditions. Results shows that the localization is performed efficiently with an error of 0.019 m at it's best. They also show that using both energy and time of arrival differences leads to better results than using only one of them as input to the neural network.

Contents

List of Figures

List of Abbreviations

ANN	Artificial Neural Network
ED	Energy Difference
FFT	Fast Fourrier Transform
HRTF	Head Related Transfer Function
IR	Impulse Response
NN	Neural Network
RIR	Room Impulse Response
TDOA	Time Difference Of Arrival

Chapter 1

Introduction

As the name suggests, sound source localization means to determine where the sound of a source originates from. The determined sound source position is usually expressed using the direction of the sound signals and the distance between the sensor position and the source position. There are many different aspects, different paradigms and algorithms applied for sound source localization. For instance, the sound receivers can be binaural robot heads [1] or microphone arrays [2]. Data extracted from sound can be exploited in different ways such as neural network [3], probabilistic approaches [4], or other techniques and algorithms. The purpose of this project is the localization of a sound source based on an estimation made by the system with high accuracy. This position is estimated in 3D Cartesian coordinates in a fixed frame. Such a task can be enrolled in the machine listening domain, where software systems like Roomsim can be used to improve the feasibility and evaluation of approaches and algorithms. Different related concepts will be introduced in the following.

1.1 Machine Listening

"Listening puts us in the world" (Handel, 1989)[5].

Machine listening is a domain that basically covers devices that respond to specific sounds. In other terms, they are devices that perform certain tasks based on information obtained from sound signals. For example, machine listening tasks can be speech recognition, allowing to identify text pronounced orally by a speaker, or speaker recognition, allowing to identify the speaker himself. Such tasks usually requires others to improve their performances, such as dereverberation, reducing the negative effects of echoes in the signals, and de-noising, reducing the effects of unwanted sound sources. A lot of applications use machine listening specially when the image doesn't present enough of information about the environment. For example, in video feed you can not assume and predict the amount of reverberation that will occur in a room, but from a sound signal applied in this room we can extract this information. Also a lot of studies presented facts that an owl doesn't make any sound with the wings when flying but if we observe only by the video comparing the owl flight with other birds we can say that it's different, only when observing the sound signal of each bird we manage to reach a more realistic conclusion concerning this hypothesis.

1.2 RoomSim

The RoomSim program[6] is a software coded in Matlab that allows the simulation of acoustics in a room. The program uses the image source method [7] to reproduce the geometrical acoustics of a perfect rectangular parallelepiped room volume. It produces an impulse response from each omni-directional primary source to a directional receiver system that may be a single sensor, a sensor pair or a simulated head.

The Roomsim environment allows the user to specify certain parameters of the room he desires that affect the simulation of the environment made. Such parameters are:

- Humidity of the air in the room (affecting the air absorption of sound waves coefficient)
- Temperature of the air in the room

- Sampling frequency of the simulated sound signals
- Enclosure dimensions (Lx, Ly,Lz) defining the length, width and height of the simulated room respectively
- Air absorption of sound (on or off)
- Type of each of the six surfaces of the room, affecting it's absorptivity and reflectivity of the sound.
- Receiver type (Single sensor, HRTF)
- Receiver coordinates(x,y,z) in a coordinate system centered at one of the corners of the room
- Sensor separation
- Sensor directionality
- Multiple Sources
- Position of the sources specified as polar coordinates from the receiver
- Order of reflections
- Length of Impulse Response

These inputs can be done by submitting a text file or excel worksheet or even through a mat file previously prepared with all the requirements of the room, or even manually by the user each time the software requests a new parameter to complete the simulation.

The simulation displays a lot of factors and saves them in a mat file after the simulation is over.Some of these outputs are:

- Plot of surface sound absorption Vs frequency
- 3D display of the room showing the room surfaces of walls, ceiling and floor, as well as the receiver and the source.
- Plot of mean reverberation time Vs frequency

- Plot of IR vs time or sample number
- Plot of magnitude spectrum, FFT length selectable

1.3 Image Source Method

Sound propagation in a closed environment like a room, witnesses different reflections on its surfaces. Supposing that there is a sound source and a sound receiver, the receiver senses the sound coming directly from the source, the reflections of this sound from room surfaces, and the reflections of the reflections, and so on. In this context, the terms direct sound, early reflections and late reflections are used.

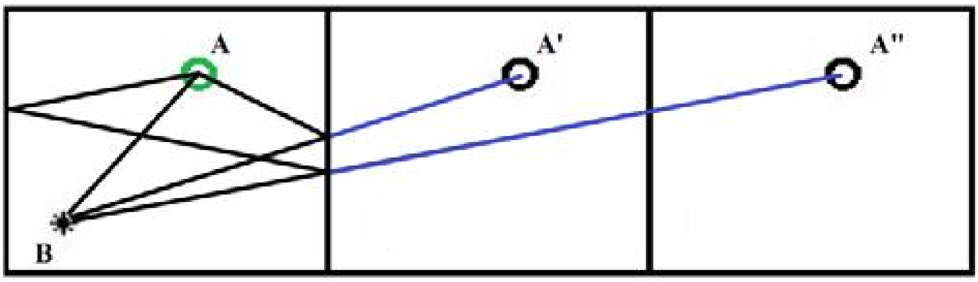


Figure 1.1: Top View of a Room With First and Second Order Image Room

As we can see in figure 1.1 there are three rectangles, the first rectangle represents the top view of a room with a source A and a receiver B that receives the direct sound signals emitted by A and all the reflected ones. Let's suppose the first rectangle is named R_0, the one next to it is R_1 and the last one is R_2. The line that represents the direct sound signal transmitted from the source A to the receiver B directly and without any reflection is the one that links A and B directly. The next target is to construct systematically the lines that represent the reflections reaching B, originating from the sound emitted by A. This is why R_1 and R_2 are used. R_1 is the rectangle adjacent to R_0 that represents the mirror image of the room, with respect to the wall at the right side, and R_2 similarly represents mirror image of R_1. A' is point located in R_1 symmetrical to A in R_0, and A" is a point located in R_2 identical to A in R_0 because it's symmetrical to A' in R_1. The straight lines connecting A' to B and A" to B represents the reflections reaching B from the right-side wall, originating from the direct sound and from the first reflection of the left-side wall. A' and A" can be seen as virtual sources, separated from B by specific distances and emitting sounds similar to the sound emitted by A, delayed in time as they are further from B, and modified in spectral content depending on the wall type, absorption and reflection properties. And seeing this problem this way helps us generating the reflected sounds and all the signals received by B. This method is similarly applied to the four reflecting surfaces seen in figure 1.1 (in a 3D schema there as six surfaces) and are used depending on the order of the reflection, The higher order reflection of sound signal can be found out by repeating the process explained earlier, to generate more images like R_3, R_4 and further on.

1.4 Time Delay

The time delay sound source position estimation method is based on computing the time difference of arrival (TDOA)[8]. Indeed, for sound receivers located at different positions in an environment, a sound wave emited by a source may not arrive to them at the same time. And this time difference of arrival depends on how close the source is to each receiver. Therefore, for the estimation of a sound source position , the time difference of arrival is main parameter that can be relied on. For example, if we place a source between two receivers having the source closer to one than the other, if we plot the two signals received by the receivers we can see the delay between the two signals because the sound signal will reach one receiver before the other. So by using n microphones we will have (n-1)! TDOA's. For example, with 4 microphones, 6 TDOA's can be computed: between the first and the second, the first and the third, and so on. The TDOA between every two microphones gives us information about the position of the source. So by taking into consideration the information collected from all the TDOA's, it is possible to estimate the position of the sound source. So the efficiency of such systems depends on the configuration of the microphones used. Also, among the main factors that affect the quality of information provided by the Time Difference of arrival, are the noises and the reverberations can be noted.

1.5 Energy Difference

Another method to estimate the sound source position is via the energy difference. The energy difference reflects the difference of the loudness between two or multiple sound signals received by different receivers sensing the sounds of the same source. Indeed, when the sound travels through the surrounding environments the strength is dissipated. Simply if you are standing right next to the sound source, the sound signals are very loud, but further from the source, the loudness of the sound signal will decrease and won't be that loud. For example with a honking car, when standing right in front of it, the honking sounds unbearable to a human listener but when we are distant from it the strength of this honk will dissipate and we will hear it with a lower loudness.

1.6 Project

This project is divided into two different parts, the first one is the creation of a database of sound signals corresponding to the simulation of sound receivers, sensing sounds originating from a sound source located at different positions in an environment. Such a database is useful for the second part of the project, and may also be made publicly available for other research teams to use. The second part of the project consists of an exploitation of the sound signals with a feature extraction step followed by a deep neural network that estimates the sound source position. We will be going into multiple tasks that we will explain later in order to create the database required for the learning process of the neural network and to estimate the position of the sound source. The features used to allow this learning and estimation are two, the first one is the time delay between the signals received by the sensors in

the room, and the second is the energy difference between the same signals. In this project the microphone array is composed of four distant microphones located near the corners of the room. The presence of four microphones allows the calculation of 6 TDOA's and 6 energy differences. This simulation is held in three different environments, an anechoic environment where no reverberation is present, an echoic environment where the surfaces have a very low absorption factor and thus have noticeable reflections, and the concrete unpainted environment which is the average environment between the extreme previous ones. We will be passing the data corresponding to each environment to the NN built based on the backpropagation algorithm to estimate the position of the source using the regression. The project is built based on two different software systems. For the database part, Matlab is used. And for the Neural Network part, the Python-based Tensorflow library is used[9].
The remaining chapters of the report are organized as follows: In chapter 2, we will be explaining the creation of the database, in chapter 3 the architecture of the artificial neural network used in the project is presented, in chapter 4 the results and the analysis of the results acquired from the tests made, and last but not least in chapter 5 the conclusion of the project is shown, with insights to future work.

Chapter 2

Database

As mentioned in the first chapter, this project estimates the position of a sound source based on a microphone array. This array is made out of 4 microphones distributed in the four corners of a room. In order to establish and evaluate the approach, a database providing sound signals in the same context of usage of the system is necessary. This database is established using the room acoustics simulation software Roomsim that provides room impulse responses (RIR) corresponding to different sound sensor positions and source locations. Each RIR is later convolved with a sound signal to create a new sound signal that reproduces the acquisition taking into account the sound content, the source and receiver positions and the room acoustics. The size of the room simulated to create the RIR database is $10 \times 5 \times 3m^3$. A coordinate system is attached to the room with Z being the vertical axis, the XY plane is horizontal and the origin is located at one of the floor corners. the 4 microphones are in the plane z=1,5 and placed near the corners, 0.5 m from the walls. That will leads to the four sensors positions, S_1(0.5,0.5,1.5) , S_2(9.5,0.5,1.5), S_3(9.5,4.5,1.5), S_4(0.5,4.5,1.5).

2.1 Database Creation Algorithm

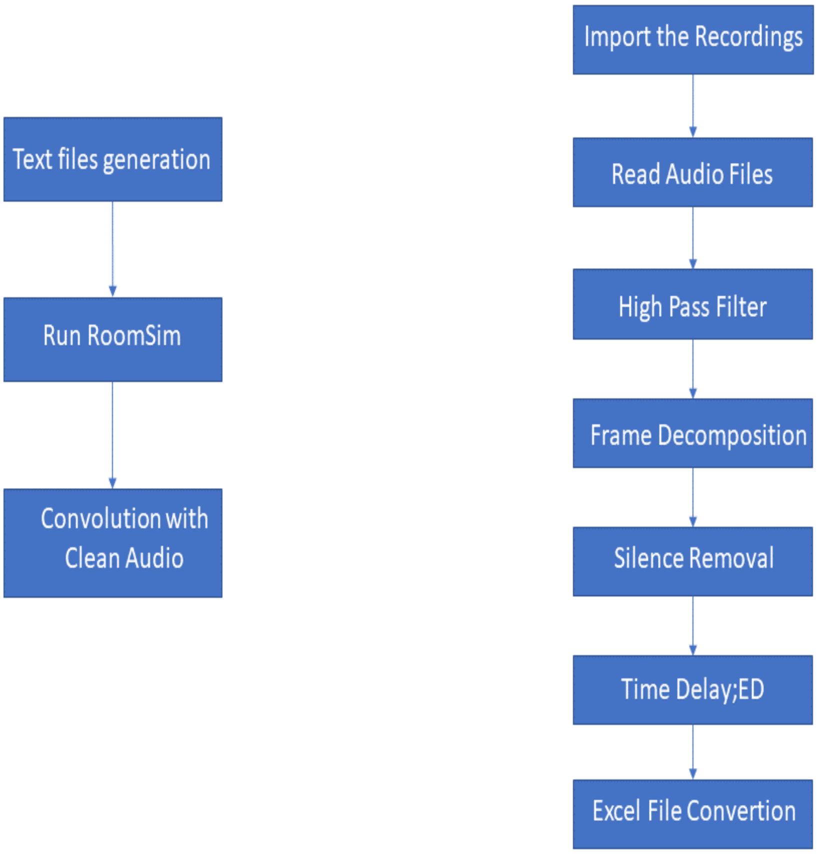


Figure 2.1: Database Creation Algorithm

The following steps make the algorithm followed to generate the database. It begins with the usage of Roomsim to generate audio files, and moves to the expoitation of these audio files to extract localization features from them.

1. Text files generation: generate a series of text files in which the user specifies the parameters already mentioned before of the room to run the simulation in RoomSim as mentioned in section 1.2

2. Run Roomsim: loads the selected parameters and generates corresponding RIRs and saves them in matfiles

3. Create recordings: having a clean audio recording, and a room impulse response corresponding to a room, a receiver position and a source location, a convolution between these two signals generates a new one. The new signal represents the clean signal that would have been emitted in the room, and sensed by the receiver, as if the source and the receiver were in their locations corresponding to the RIR. Thus, it embeds the effects of the positions as well as the room acoustics, and the information present in the original clean signal, in the new one.

   Stopping at the previous step would lead to a database of sound signals that can be exploited in different ways according to the desired algorithm and method of researchers using them. Thus, they can be made publicly available for research in sound source localization in similar contexts. The following steps show how signals were exploited in this work, to extract localization features from them, leading to an additional database of features corresponding to the different simulated rooms, source and receiver positions.

4. Read audio files: after the convolution, the resulting signals are saved in mat files. When reading the audio files each file is accessed and read using the audioread() function, that returns the sample amplitudes of the signal arranged in a vector.

5. High-pass filtering: there may be low-frequency noises s in the surrounding, it is better to apply a high pass filter on the audio signals, with a minimum order and a cutoff frequency of 30 Hz.

6. Frame decomposition: after filtering the audio signals, they are divided into multiple signals distributed on small fractions of time. In this case it's more efficient to observe the features from each frame which will allow it to be more flexible and achieve more accuracy and efficiency during the calculation of the time delay and the energy difference

between the signals received by the microphones. Note that the signals of the four microphones have the same length and they result in the same number of frames.

7. Silence removal: this steps helps to detect the silence from the actual speech in the audio signal and allows us to remove it and create a new audio signal free of silence. Indeed, silence is sensed the same way from any source position,therefore, it provides no information about the position and is harmful for the training of a sound source localization system.

8. Time delay and Energy difference: time delay and energy difference are for all time frames and each microphone pair as features required to the training and testing of the neural network. These features will compose the input training data later.

9. Excel file generation: the automatic generation of Excel files is taking place here, where each file is composed from the coordinates and the time delay and energy difference of each frame in a particular source position. Later on, these files are the ones considered as features database and will be loaded to the neural network for training and testing.

In the following paragraphs, each function and step are being explained in details to create the database made of the inputs and the outputs of the neural network and will be later in Python divided between training and testing data. The neural network will be explained in details in the next chapter.

2.2 Text Files Generation

The flowchart of this processing step is presented in appendices A.1.
A MATLAB function is built in a way to create multiple text files for all sensors, in a sort of way that each text file has the parameters specified for a certain sound source position. Since the room as previously mentioned

in the previous paragraphs has a size of $10 \times 5 \times 3m^3$ three for loops are executed. The first one for the x coordinate with x varying from 1 to 9 meters, the second one is the y coordinate with y varying from 1 to 4 meters, and the third loop is for the z coordinates varying from 1 to 2.5 meters. These three loops have a common step of 0.5 m. Therefore 17 values of x, 7 values of Y and 4 values of Z are present, this will leave us with 476 simulation positions for each sensor of the 4 microphones and for each environment of the three: anechoic, concrete unpainted, echoic. When reading from text files, the Roomsim reads the coordinates of the source based on the spherical base coordinates with respect to the sensor coordinates. So first, all we need to find the coordinates of the source position with respect to the sensor coordinates in Cartesian base, then we need to transform these Cartesian coordinates to spherical coordinates by computing the distance, the Azimuth and the elevation of the source with respect to the sensor. In each environment, and for each sensor, these text files are created and the only variable parameters between the text files of the same sensor are the ones expressing the position of the source. By moving to the other sensor the coordinates of the sensor must be altered by the user in Matlab before regenerating the text files, and of course when changing the environment, the user must change the absorption table, because as mentioned earlier each environment is specified with a special acoustic factor such as the the absorption factor that the software retrieves from the corresponding Excel file.

The most important parameters for all the text files are:

- Sampling frequency : 44100 Hz
- Room Temperature: 20° C
- Receiver type: one microphone
- Position of the sound source: Depending on the case being simulated
- Position of the receiver: Depending on the case being simulated

After the generation, each text file is named after the real coordinates of the source corresponding to it, Test-x-y-z. For example: Test-6-3.5-2 in the anechoic file in the sensor 2 file, corresponds to the source in anechoic environment at (6,3.5,2).

2.3 Room Impulse Response Generation

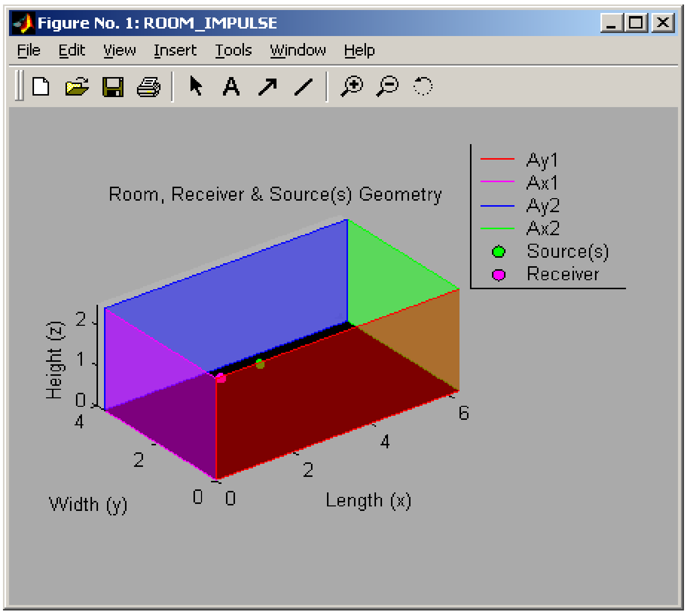


Figure 2.2: Geometric Representation Of a Room In RoomSim

The simulation code was modified to access the text files generated via three for loops based on the name, the for loops are the same as the one stated earlier and the outputs are saved in mat files and named like the corresponding text file. One of the parameters saved in the mat file is the RIR, this parameter is the most important for this study and the creation of the database. The pictures presented in the figures 2.3,2.4 and 2.5 show the difference of the IR between the different environments but for the same sensor position and for the same source position. By fixing these parameters we will be able to observe and compare the IR between the three environments. In the anechoic environment the we can notice the actual plot of the Impulse in the room with no repetition in the early future because all the sound signal is absorbed and none reflected as shown in the figure 2.2.

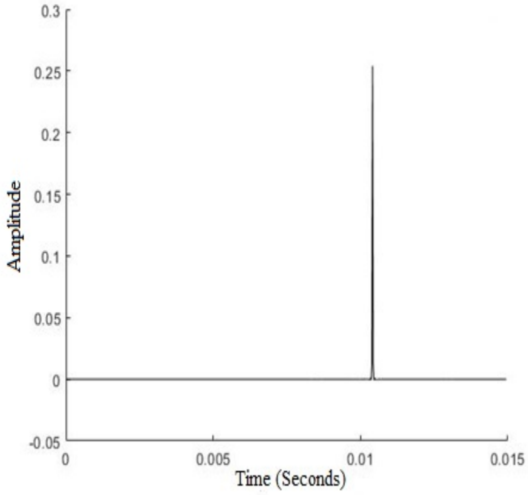


Figure 2.3: Impulse Response of an Anechoic Room

In the echoic environment represented in figure 2.3 where there is no absorption of sound at the surfaces, we notice that the IR doesn't vanish on a short period, the decision taken in this matter was to cut the signal at an impulse response duration of 0.4 seconds. The peaks we see correspond to the direct sound, early reflections from the walls and the late reflections. We can notice the Concrete unpainted in figure 2.4 is in between the previous two, there is echo in the plot of the IR but it ends gradually, if the echoic IR was left without any editing on the reverberation time, reflections would vanish in the same manner but take more time.

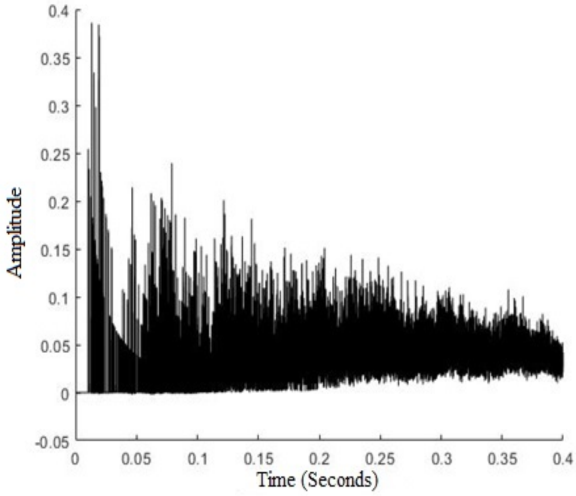


Figure 2.4: Impulse Response of an Echoic Room

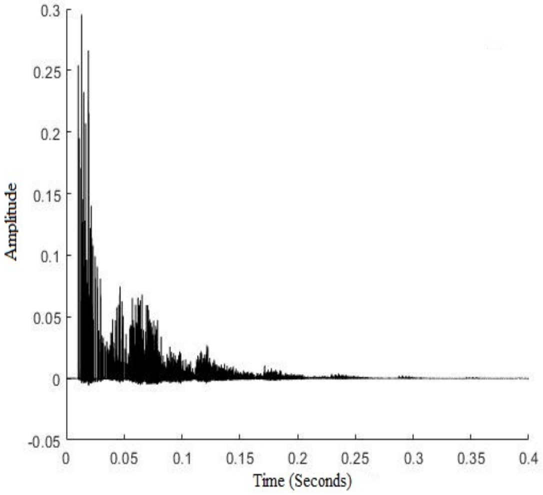


Figure 2.5: Impulse Response of an Concrete Unpainted Room

2.4 Sound Signal Generation

The period of the RIR is so small, specially in an anechoic environment. Not only that, but the RIR is a mean of calculations and does not correspond to real-life signals. So if we need to extract data from this signal to use it in the creation of the database, our database would be very small and not enough for the neural network to learn. That's why we need to create longer signals for the database. We download a clean audio speech signal that is about 50 seconds long. A Matlab function is created that access the RIR in mat files saved earlier from RoomSim and we convolves the RIR with clean audio obtained earlier. This way we can assume that we recorded live the audio signals in real room using real microphones because once we hear the result of this convolution it sounds like an audio signal will react in this specific environment. Since the RIRs for each sensor are saved separately, the Matlab function accesses, the same position in the 4 folders and saves the four results of the convolution in four cells in the same mat file and saves it under the name 'Position-x-y-z' with x, y and z being the coordinates of the source position.

2.5 Reading Audio Files

Once all the recordings of each source position and in each environment are saved in the same mat file there is no need to access multiple files to read the audio signals for a certain position. The audioread function is a built in matlab function that allows us to read audio signals either from .wav files or mat files and returns the sampling frequency and the samples of the signal and places them in a vector. Since there is a 4 microphones array that means we have 4 channels or 4 four different audio signals in each mat file we want to access to process it later, and each signal is stored in a cell in the mat file. Each cell is named after the sensor that corresponds to it, for example HC1 is for the first sensor at the position S_1 and so on.

2.6 Frame Decomposition

The flowchart of this processing step is presented in appendices A.2.
Basically in every project where there is some processing to be done to some data such as image or signal processing, it is very important to decompose and transform data from a large vector or matrix into smaller samples, this process

will allow us to be more flexible and to be more efficient in case of processing. Using this method we are able to extract more data for our neural network to train, plus by extracting more data we can monitor and analyze the differences causing different results in the neural network but using the same architecture. It also allow to track the dynamics in the studied signal. Applying some filters without losing or effecting a lot of data and affect our total results later on. In this project we transformed each audio signal which is vector of length sampling $Frequency \times DurationOfTheCleanAudio$ into a matrix of dimension $nframes \times FrameSamples$ where FrameSamples corresponds to a frame of this audio signal, and n frames is the number of frames deducted from our audio. This kind of frame decomposition mainly exists in machine listening application such as speech recognition and of course sound source localization. But in order to obtain more data and in case some group samples exists between two frames we decided to apply in overlapping percentage that allows us to have a common set of samples between consecutive frames. This way we will obtain larger amount of data that could be useful to the training of the neural network and minimize our losses, and have more efficient and accurate set of data. The chosen duration of the frames is related the human vocal tract that has a fixed configuration for about 30 ms and it takes a longer duration to change. Different studies take different frame duration, they go from a couple of milliseconds to 50 ms, the majority takes 20-30 ms durations [5]. In this project the period of each frame is one second. The reason we took it longer than usual is based on personal conclusion. Basically in the previous, the sound source localization systems the distance between the microphones in the microphone array is not larger than 20 to 50 cm because either the study is being held in a binaural case and the distance between the two microphones is the same as the distance between the actual ears, or in case of microphone arrays, the set of microphone are placed in the middle of the room. The distances between the microphones of the previous studies are small, which means that when calculating the time delay the major part of the information exists in the small duration frame. In our case the microphones are placed near the corners of the room in the same plane z=1.5, the distance between two sensors in our room can reach up to

8 meters, which is way bigger than the distance between the microphones of the previous researches. In a simple assumption, we noticed that for a distance between a pair of microphone about 20-50 cm, the frame length is about 20-50 ms. We conclude that for 8 meters that means 800 cm we should consider the frame length hundred of milliseconds, and after several trials, the duration of 1 s, proved to be a good choice. For this project we decompose the sound signals of each microphone imported from the mat files into frame, and with an overlapping percentage "ovpc=30%" so for a certain length of an audio signal, the sampling frequency "Fs=44100Hz" and for a frame period "FramePer=1 sec", we may compute the FrameLeng by using the following formula to round to the higher integer $ceil(FrampePer \times FreqSample)$, we compute the FrameLarg which is the number of frames by ceiling the result of this formula, the audiolength is the number of samples in the audio signal.

$$FrameLarg = R \times \left(\frac{AudioLength}{FrameLeng}\right) \times (\left(\frac{100}{ovpc}\right) - 1)$$

This way, we turn our on dimension audio signal into a matrix in which each colomun is considered as a frame.

2.7 Silence Removal

This function is dedicated to the detection of the silence in the audio signals or basically in the matrix since we will be applying this function to the frames. The flowchart of this processing step is presented in appendices A.3.

Like the name says, it detects the silence and removes it and will keep the part of the signal where there is voice activity. For the same position, the four audio signals that correspond to the four different sensors have the same number of frames. So in order to eliminate the silence from the audio as frames we need to classify frames as silence or voice frames. A frame that is classified as silence from any microphone signal is removed from all four microphone signals so the number of frames remain the same and the calculation of the time delay and the energy difference will be done properly. The function that we created needs the following inputs: the matrix of the frames and a threshold that the user specifies, this way by applying the formula

$$EnThres = MinEn + k \times (MaxEN - MinEN)$$

The variables in this formula are the $EnThres : EnergyThreshold, MinEn = MinimumEnergy, MaxEn = MaximumEnergy,k$ is the condition set by the user to filter the frames. The threshold energy is computed which is the separating factor that will classify each frame as silence or voice activity. So the energy of each frame is calculated as the sum of the squares of the samples and from these energies we compute the $EnThres$, and we compare the energy of the frames one by one to the energy threshold, if the energy of the frame is lower than the energy of the threshold the index of the frame is saved and the frame corresponding to this index will be deleted from the four signals. After the filtering we will obtain new matrices with less frames but with the voice activity in our signals.

2.8 Time Delay

The flowchart of this processing step is presented in appendices A.4.
The sound source localization in this project is based on Neural network estimation, this estimation in calculated based on two different approaches the time delay and the energy difference. In this paragraph, we will be explaining how to extract the data required for the first approach which is the time delay. The time delay or the time difference of arrival is computed between each pair of microphones, since our microphone array is made of four microphones, that means we need to compute six values for the delay, $Tab = Tb - Ta$ is the time delay between microphone a and microphone b. It is the subtraction of the time that the sound took to reach from the source to the microphone a and the time that the sound took to reach from the source to the microphone b. In our case we have $T12$, $T13$, $T14$, $T23$, $T24$, $T34$. To compute this feature we access the recordings already saved in the mat files after the convolution and we decompose the four audio signals into frame matrixes and to calculate the time delay between microphone a and b on a frame by frame basis. The peak of the cross-correlation at two corresponding frames, corresponds to the time delay between them.

2.9 Energy Difference

The flowchart of this processing step is presented in appendices A.5.
In this paragraph, we will be explaining the required algorithm to extract the feature required for the sound source localization from the second approach which is the energy difference. The energy difference is computed between each pair of microphones and since our microphone array is made out of four microphones, this will leave us with six energy differences, $E12$, $E13$, $E14$, $E23$, $E24$, $E34$. So to compute Eab which is the energy difference of two microphones a and b we use the following formula on a frame by frame basis:

$$Eab = 10log\left(\frac{Ea}{Eb}\right)$$

and the energy of each frame is the sum of its the samples squared,

$$Ef = \sum_{i=1}^{n} x_i^2$$

where Ef is the energy of the frame f, and x_i is the amplitude of it's i^{ith} sample. To compute the energy difference, we access the recordings saved in mat files after the convolution and after decomposing them into frames, we apply the formula of E_f on each frame on the four signals, this way we will have a vector of Ea and each element of this vector represents the energy of each frame of the signal corresponding to the microphone a. So to compute the energy difference of a pair of microphone we apply the formula Eab. This way we will create a vector Eab and each element in it is the difference F1 applied between the same frame in the two signals a and b. So in the end, after looping the energy difference over the pairs combination of the four signals, we leave it in vectors for later extraction into an excel file to present it as a database to the Neural Network later on.

2.10 Excel File Generation

The steps stated earlier are all done inside three for loops to access the recordings saved after the convolution in an consecutive way. The three loops are the same ones mentioned earlier in paragraph 2.2. We have x going from 1 to 9 and y from 1 to 4 and z from 1 to 2.5 with a 0.5 step, so we access the four signals in each mat file and we apply all the functions stated earlier from reading to audio until the energy difference computation. After that we will have six vectors of time delay and six vectors of energy difference, so we combine the following vectors into one big matrix with the first second and third column having the value of coordinates of the source x,y,z respectively. The next six columns are the ones dedicated for the time delay vectors computed earlier, and the last six columns are the one corresponding to the energy difference vectors. So overall, we will have a matrix 15x FrameLarg and each row is the row of a frame. We print this matrix into an excel file with the name Position-x-y-z.

Chapter 3

Neural Network

In this chapter we will be presenting a brief introduction about the neural network and later we will go into details explaining about the usage of a neural network in the project. They are a chain of calculations which endeavor to distinguish connections between data sets[6]. A neural network is a series of algorithms and mathematical equations that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refers to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input, so the network generates the best possible result without needing to redesign the output criteria[7]. Neural Network are data handling frameworks that are enlivened by the natural neural systems like a mind.

3.1 Neurons

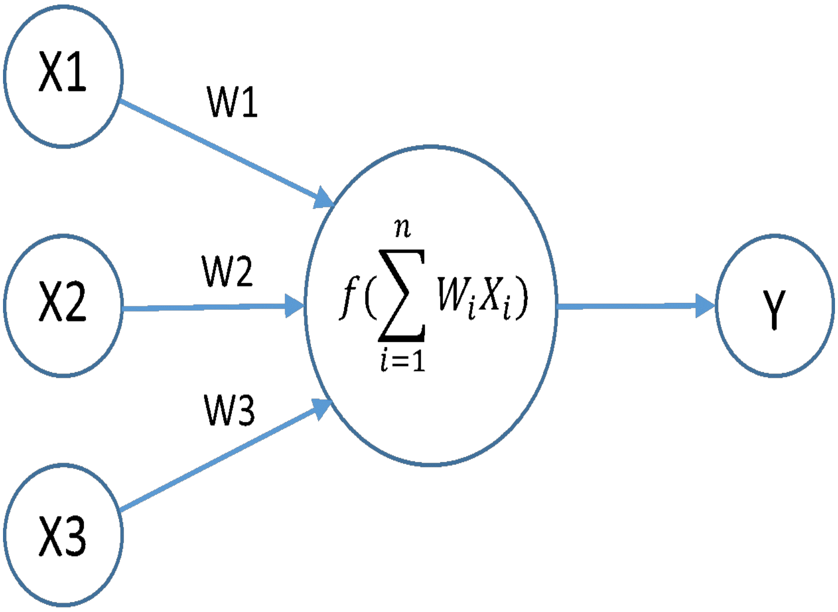


Figure 3.1: Representation of a Neuron

A Neuron is an important part when talking about Deep Learning and Neural Networks. In a Neural network, a Neuron has inputs as a vector and it processes it in a way to have an output. It computes the total net input which is the sum of the weights times the inputs the formula is the following:

$$TotalNetInput = \sum_{i=1}^{n} w_i x_i \text{ where n is the total number of inputs.}$$

After that the neuron compute the output by applying an activation function to the total net input.

$$Y = Output = f(TotalNetInput)$$

The output is sent later to the next neuron and will be interpreted as input, and the process mentioned earlier will repeat itself. In a neural network, neurons are arranged in multiple layers, the output from a layer is sent as an input to the next one until they exit from the output layer as an output to whole system.

3.2 Activation Functions

To generate an output from a neuron, and as stated in the previous paragraph, the formulas presented the activation function inside the neurons. Activation functions give a multiplex non-linear functional mapping between the net inputs and outputs of the neural network. If we don't apply the activation function or even if we apply a linear function, the output with respect to the input which is $\sum_{i=1}^{n} w_i x_i$. The activation function can be a non-linear function such as polynomials fucntions, that presents high degrees to help create complex mapping between the output and the inputs of the system. Some of the most used activation functions are the : Tanh(hyperbolic tangent), ReLU(Rectified Linear Unit), and the sigmoid. In this project, we used the sigmoid as an activation function for our neural network, it is characterized by the equation $\phi(z) = \frac{1}{1+e^{-z}}$. The curve of the sigmoid activation function is the following:

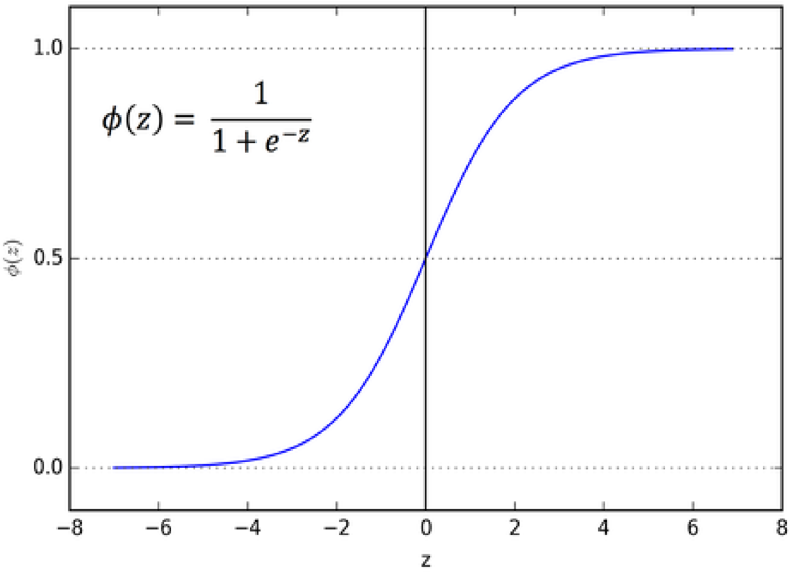


Figure 3.2: Curve of the activation function sigmoid

3.3 Topology of an Artificial Neural Network

A neural network is made of three types of layers, the input layer, the hidden layers and the output layer. Figure 3.3 shows an artificial neural network with one layer of each type. The input layer doesn't have any neurons in it but it's a type vector. The number of hidden layers allows the network to learn and adapt into more complex problems. And the output of the whole system is presented in the output layer either classification or a regression system. The regression is about predicting a quantity or estimate a certain number, but the classification predicts labels or as the name says it classifies the output into one of predefined classes in the output layer. All these associations in the Neural network have a main target to produce an output which will be fed forward on to the following neuron until it reach the output layer.

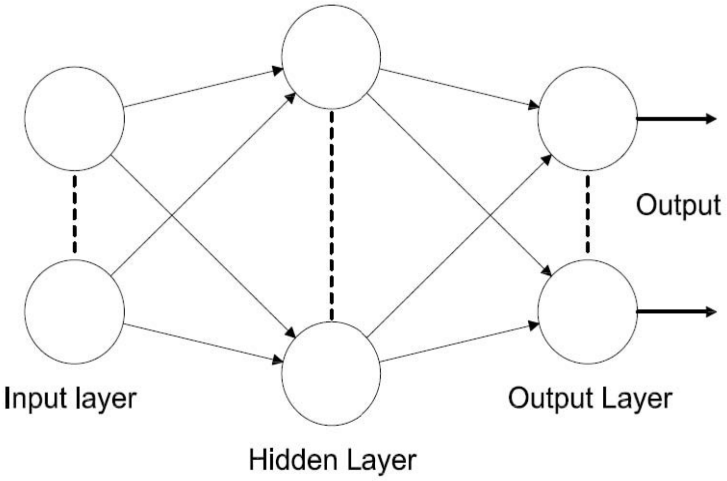


Figure 3.3: :Topology of an Artificial Neural Network With One Hidden Layer

3.4 Backpropagation Algorithm

The backpropagation algorithm is a widely used algorithm in machine learning that trains a feed-forward neural network. This algorithm uses the gradient of the neural network error against the weights to update it's parameters such as the weights, the bias and in the same time to minimize the error of the neural networks estimation. To start with the backpropagation algorithm, first of all, we need to compute the total error which is the sum off the error values between the estimated one by the network and the real output value presented by the training output data . This definition can be translated into the following equation:

$$TotalError = \sum \frac{1}{2}(RealOutput - EstimatedOutput)^2$$

Let's consider a single output neuron with a one input connection in order to simplify this algorithm. In order to compute the Gradient of the Total error against the weights, we need to calculate the partial derivative of the total error against it's weight $\frac{\delta TotalError}{\delta Weight}$. We can transform the previous partial derivative into simpler partial derivative multiplied and easier to compute such as

$$\frac{\delta TotalError}{\delta Weight} = \frac{\delta TotalError}{\delta Error} \times \frac{\delta Error}{\delta RealOutput} \times \frac{\delta RealOutput}{\delta TotalNetInput} \times \frac{\delta TotalNetInput}{\delta Weight}$$

Now we can compute each of this partial derivatives in the previous equation separated in order to get the final value of the gradient error against the weights and apply the modifications on the parameters of the neural network. In this neural network, we are only dealing with one single input and a single output estimation. And the total error is the sum of all the errors from the outputs, this will leave us with $TotalError = Error$, so basically we are trying to compute the partial derivative of an element against it self, so this will leave us with a value of 1.

$$\frac{\delta TotalError}{\delta Error} = 1$$

In the second partial derivative of this equation, we have

$$TotalError = \frac{1}{2}(RealOutput - EstimatedOutput)$$

So by applying it's partial derivative against the output, we will obtain

$$\frac{\delta TotalError}{\delta RealOutput} = (RealOutput - EstimatedOutput)$$

In this neural network the activation function is the sigmaoid function, or the equation of the sigmoid function is

$$f(x) = \frac{1}{1+e^{-x}}$$

and it's derivative againt it's input is

$$f'(x) = f(x) \times (1 - f(x))$$

this will leave us with the following value for the third partial derivative of the equation. As we already know, the input of the neural network is multiplied by the weights and summed together to obtain the Total Net Input which is the sum of the $Weights \times Inputs$, therefore the partial derivative of the total net input against the weights is the input. After solving for each individual partial derivative, we can multiply them and compute the partial differential of the Total Error with respect to the weights. And based on this value, we can update weights of the neural network $NewWeights = OldWeight - (LearningRate \times input)$. Learning rate has a value between 0 and 1. The learning rate is the speed that a neural network modifies the weights in order to reduce the error, and have a better estimation. Learning rate, activation function, number of neurons, number of hidden layers determines the speed, accuracy, in total controls the efficiency of the neural network.

3.5 The Neural Network in the project

The neural networks used in our project and as stated earlier, are feed-forward multi-layer perceptrons trained with back-propagation algorithm, build in python using a library called Tensorflow, to estimate the position of the sound source in the Cartesian coordinates system used by Roomsim to locate the sound source and the microphones, so first of all the system estimates x then y then z. This neural network has 6 inputs, which are the Time delay and the energy difference computed earlier and generated into excel files which consist the database. The Database used to train and test the Networks. Using TensorFlow, Keras, Sklearn and many other libraries in python the neural network of this project is built with one hidden layer a 0.0001 learning rate , 15 neurons, and one single output since we are solving this matter as a regression problem and not a classification. With 17 positions in the X coordinate and 7 positions in the Y coordinate and 4 in the Z coordinated, we have a total of 476 different positions in the room $10 \times 5 \times 3m^3$. The Data for each environment are stored in different folder, all the positions are imported from the excel sheets, put together in one big array of size approximately 15×34272, because in each position we have the data for 476 positions we obtain this big amount of data for our system. After importing all of these data, the rows in the array are shuffled randomly using a built in function in the NumPy library for the system to learn in s better way. The data is split between the outputs and the inputs in which the inputs are standardized in way to take a mean of 0 and a variance of 1 and in this manner the input data are all normalized. After the normalization, we split this imported data into four main parts the training input data, testing input data, training output data and testing output data.

Chapter 4

Evaluations and Results

The steps taken to train and test the neural networks are presented in figure 4.1. In this part, we considered that the positions that are classified as integer and the corresponding rows contains the required information for our model to learn are stacked as Training Data which contains the Training Output (The positions) and the training inputs(Time Delay and energy difference) and the remain positions are stacked as testing data, Testing Output(The Position) and the testing input(Time delay and energy Difference). For example, in the X coordinated the data corresponding to the positions x=1,2,3,4. . . 9 are classified as training data and the data corresponding to the positions x=1.5,2.5. . . .8.5 are classified as testing data, and the same principal is applied on the Y and Z coordinates. This will leave us with about 50-60 % of the data classified as learning overall, and from the learning 10 % of the data in the neural network, and with each iteration are used as validation data to monitor our system during the learning and watch our backs from the over-fitting our similar problems. The cross validation is the estimation of the results on an independent set of data, and the overfitting is when a model memorizes the data and learn all the details in the data set, this case will lead to an error in the classification or the estimation of the outputs on new data on the model. To verify the efficiency of this neural network, 10 tests were made on each combination of the parameters in the project, since we have 3 environments, 3 different approach and 3 coordinates, this will leave us with 27 different combinations. The parameters computed in these tests were used to evaluate the neural network:

- Error: $e_i = RealOutput - EstimatedOutput$
- Mean Squared Error $MSE = \frac{1}{n}\sum_{i=1}^{n} e_i^2$
- Mean Absolute Error $MAE = \frac{1}{n}\sum_{i=1}^{n} |e_i|$
- Mean Absoulte Percentage Error $MAPE = \frac{100}{n}\sum_{i=1}^{n} |\frac{e_i}{RealOutput}|$
- Accuracy $Acc = 1 - \frac{MAPE}{100}$

Where i is the testing example index, and n is the total number of test examples. In this chapter, we present the process of learning and the result of testing the system on data never seen before and we will be analyzing our different results based on the different environments, different approach in each of the coordinates.

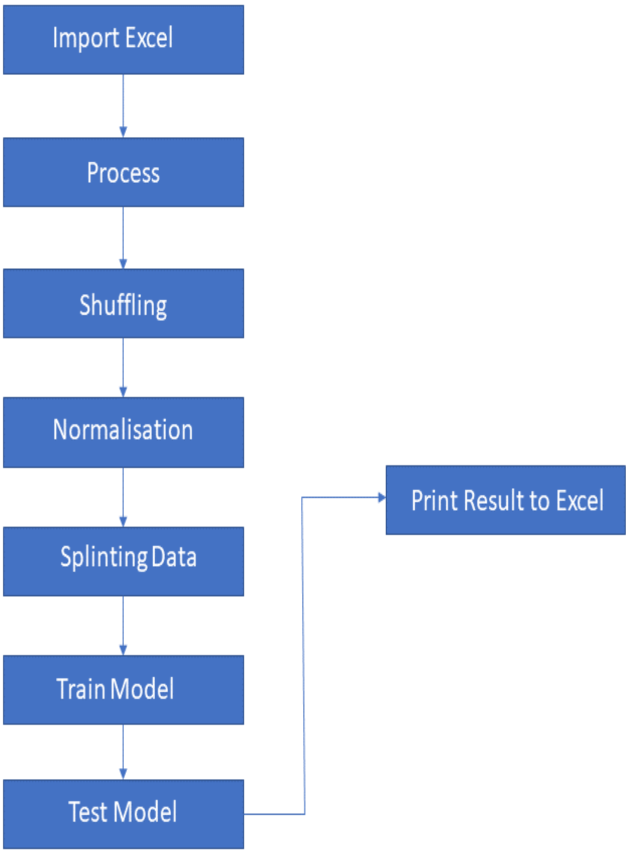


Figure 4.1: Neural Network Processing Algorithm

In the following sections, we will presenting some training and testing results, and saving all the required interpretation and analysis into one paragraph to summarize the entire process and collect all the necessary results.

4.1 Verification of the Calculated Features

In this section, we will be presenting the sound source localization based on the time delay approach in the three different environment. When importing the data into python, specially in the anechoic environment the data will

seem much realistic.
We will be comparing between the data of time delay obtained from the simulation and one computed by hand on one random source position (1,2,1) and in the center of the room(5,2.5,1.5) to see the error between the difference data which we should have theoretically and the results from the calculations. The comparison with the random position is only made in anechoic data because when reverberation exists, these data will be changed and will not be as accurate as the one in this case.

1. For the Random Source position S_5(1,2,1), the sensors are placed at the following coordinates , S_1(0.5,0.5,0.5),S_2(9.5,0.5,1.5),S_3(9.5,4.5,1.5)S_4(0.5,4.5,1.5). D_1, D_2 D_3, D_4 are the distances between the source position S5 and the sensors positions S_1, S_2,S_3,S_4 The distance from the sound source to the sensors are:
   D_1=1.6583 m
   D_2=8.6458 m
   D_3=8.8741 m
   D_4 = 2.5981 m

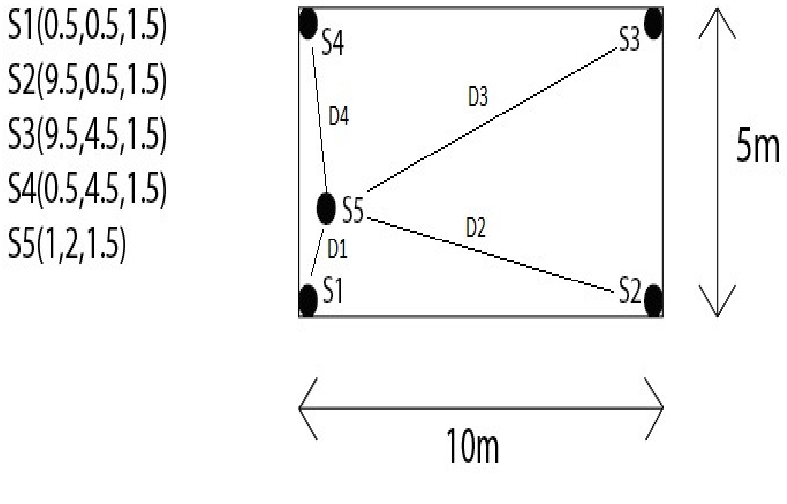


Figure 4.2: :Top View of the room for a specific sound source position

By assuming that the velocity of the sound is 340 m/s the average time for the sound to reach the sensors from the source are

$$TimeDelay = \left(\frac{Distance}{Sound speed}\right)$$

so this will leave us with :
$T1 = 4.87 \times 10^{-3}$ seconds
$T2 = 0.02542$ seconds
$T3 = 0.0261$ seconds
$T4 = 7.641 \times 10^{-3}$ seconds
So Tab=Tb-Ta so the feature of Tab are:

T12	T13	T14	T23	T24	T24
0.02055 s	0.02123 s	0.002771 s	6.8x10-3 s	-0.017779 s	-0.018459 s

Table 4.1: Table with Time Delay computed by hand

From the excel sheet of this position, we extract the features of one frame, and we notice that in an anechoic, environment the data in all the frames are similar for the same position not like the data in the other environments where the reverberation exists.
In the table below we can see the features from the excel sheet:

T12	T13	T14	T23	T24	T24
0.020385 s	0.021043 s	0.002744 s	0.000658 s	-0.01764 s	-0.0183 s

Table 4.2: Table with Time Delay extracted from the excel sheets

As we notice, the difference between each pair of corresponding values is very small. So in order to estimate the real difference, we tend to compute the error percentage between the two values by using the following formula:

$$Eab = Abs(Tab_{real} - Tab_{simulated})/Tab_{real}) \times 100$$

So we obtain the following table :

E12	E13	E14	E23	E24	E24
0.8%	0.88%	0.97%	3%	0.78%	0.86&

Table 4.3: Table of errors between the two approaches of time delay calculation

As we can see over here, in table 4.3, the error percentage is very small, so basically we can conclude that the data obtained from the simulation are accurate and close to reality so we can lean on this data to train and test our approach.

2. For the position in the center of the room, figure 4.3, the source is at equal distances from the four sensors so in reality the time delay and the energy difference must be zero, because the signal will arrive to the four sensors in the same time and same loudness; so if we check the excel sheet corresponding to this position (5,2.5,1.5) we shall see that in the three environments we will have an excel sheets full of zero.

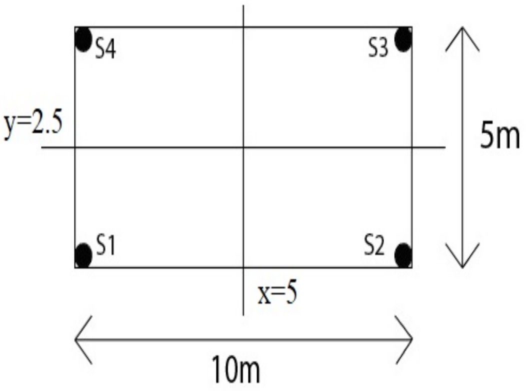


Figure 4.3: :Top View of the Room with thel symetric planes

3. If the source is on the plane of Y=2.5, figure 4.4, we will notice that the features of the difference between microphone 2 and 3 and the

microphones 1 and 4 will be zeros which is logical because at Y=2.5 we are at the plane cutting the room in half and the plane of symmetry of the room. Same logic at X=5, figure 4.5, we are at the other plane of symmetry and the features of the microphone 1 and 2 and the microphones 3 and 4 are zero.

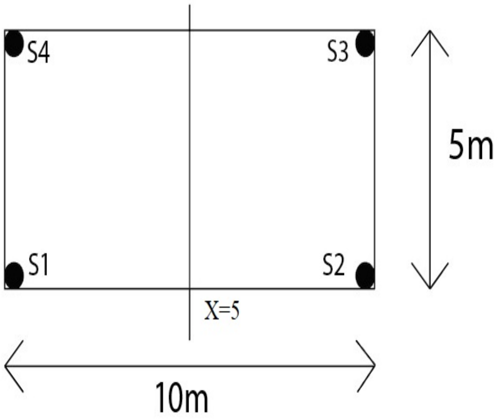


Figure 4.4: :Top View of the Room with the vertical symetric planes

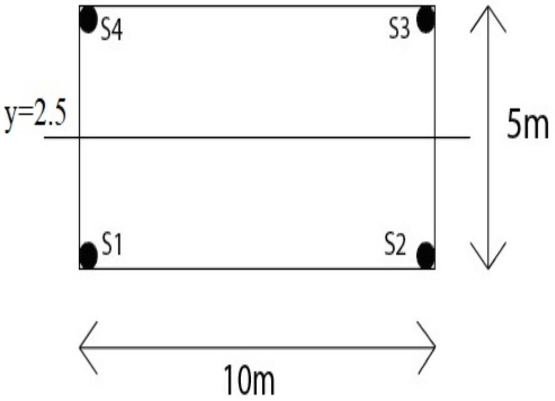


Figure 4.5: :Top View of the Room with the horizontal symetric planes

In this case, we can assume that the data obtained from the simulation in the three environments are accurate and very much close to the reality.

4.2 Time Delay Approach

Environment	Coordinates	Mean Absolute Error	Error Percentage
Anechoic	X	0.023	0.756
	Y	0.035	1.63
	Z	0.505	20.361
Concrete Unpainted	X	0.76	20.82
	Y	0.46	22.038
	Z	0.507	20.415
Echoic	X	1.83	55.82
	Y	0.581	27.53
	Z	0.51	20.36

Table 4.4: Time Delay Approach Test Results

In figures 4.6, 4.7, 4.8, we can see the learning curves of our neural network during the estimation of the X, Y, Z positions respectively in an anechoic environment and using the time delay approach. The Mean Sqaured error stablizes in the X and Y curve apprixmetly at 0 and the accuracy reaches about 100% which is better than the ones of the Z estimation that reaches an Accuracy of 50% and a Mean squared error value of approximetly 0.25 m^2. Concerning the 10 tests made on each coordinate, we can notice that the results in the learning rate on the tests. We notice that the Mean Absolute error and the error percentage are very low 0.03 meters and 1% on the estimation of Y, but the estimation the Z coordinate is high with a mean absolute error of 0.5 meters and an error percentage of 20%.

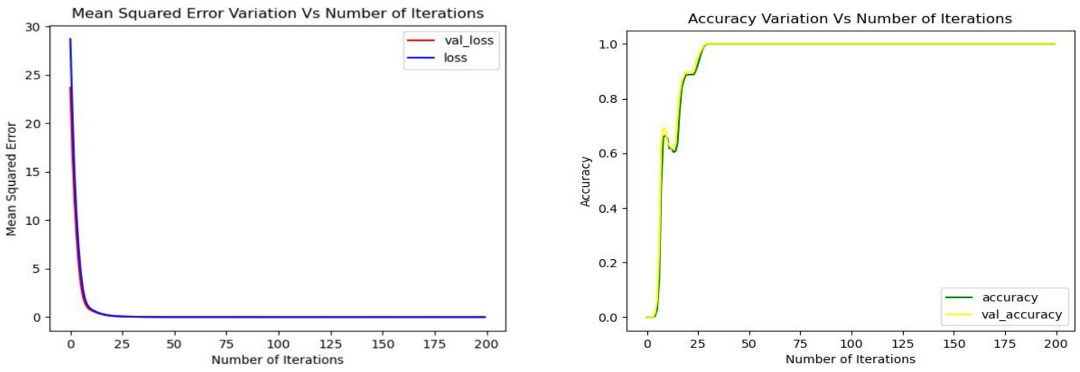


Figure 4.6: Left: MSE Vs Number of iterations X estimation in anechoic environment using the time delay approach. Right: Accuracy Vs Number of iterations X estimation in anechoic environment using the time delay approach

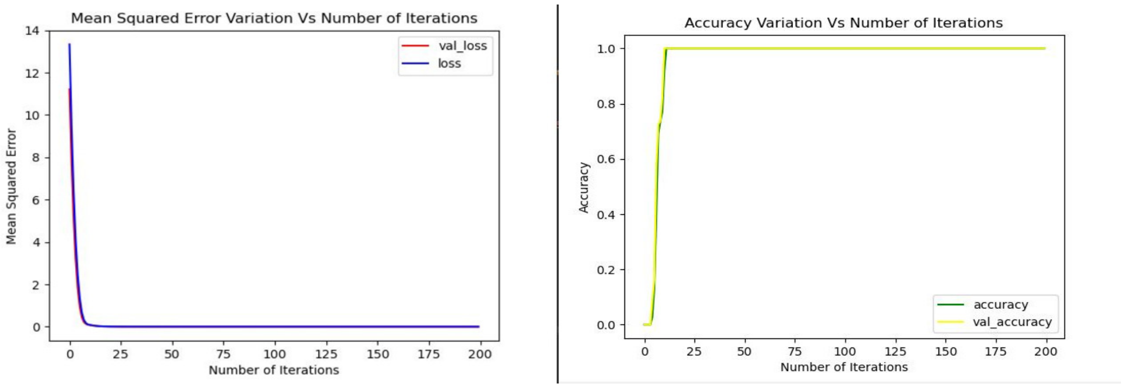


Figure 4.7: Left: MSE Vs number of iterations Y estimation in anechoic environment using the time delay approach. Right: Accuracy Vs Number of iterations Y estimation in anechoic environment using the time delay approach

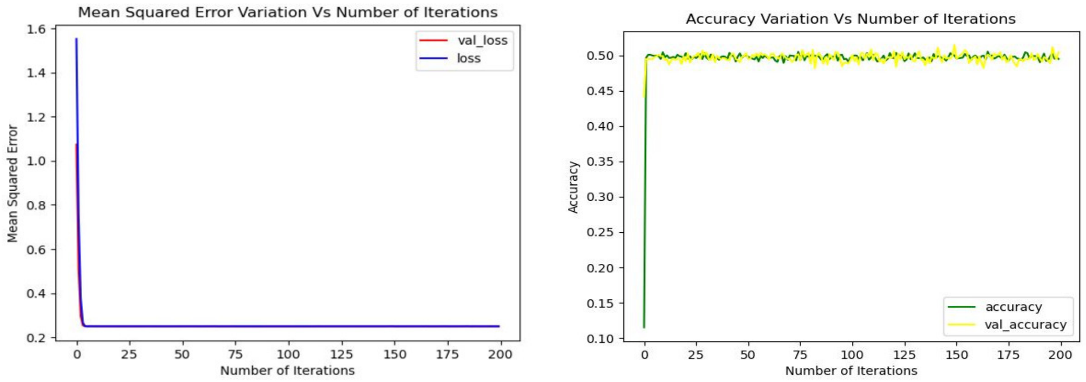


Figure 4.8: Left: MSE Vs Number of iterations Z estimation in anechoic environment using the time delay approach. Right: Accuracy Vs Number of iterations Z estimation in anechoic environment using the time delay approach

The figures 4.9, 4.10, 4.11 represents the learning curves of our neural network during the estimation of the X, Y, Z positions respectively in a concrete unpainted environment and using the time delay approach. The final results in the three coordinates are very similar, we notice that the mean squared error reaches a minimum of 0.2 meters and the accuracy reaches a maximum of 50%. These learning results reflects on the testing errors, as we can see in table 4.4, the mean absolute error of the estimation of the three coordinates is close, with a maximum difference of a 0.2 m, and the error percentages are close approximately 21%.

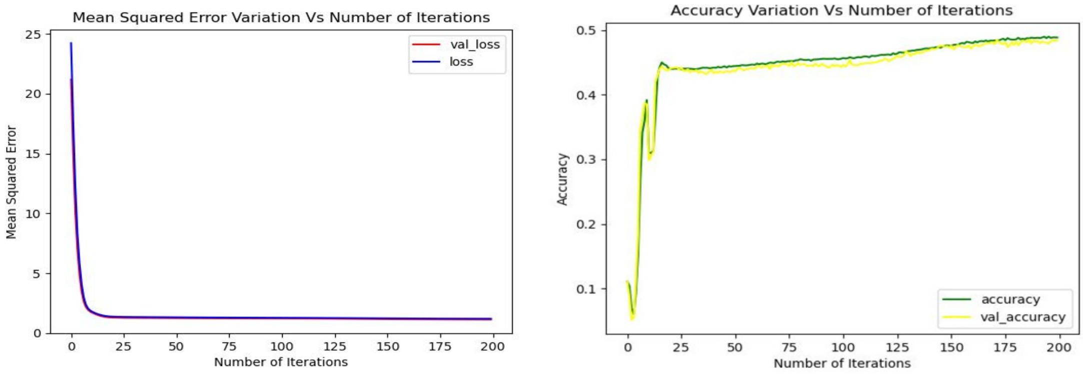


Figure 4.9: Left: MSE Vs Number Of iterations X estimation in concrete unpainted environment using the time delay approach. Right: Accuracy Vs Number of iterations X estimation in concrete unpainted environment using the time delay approach

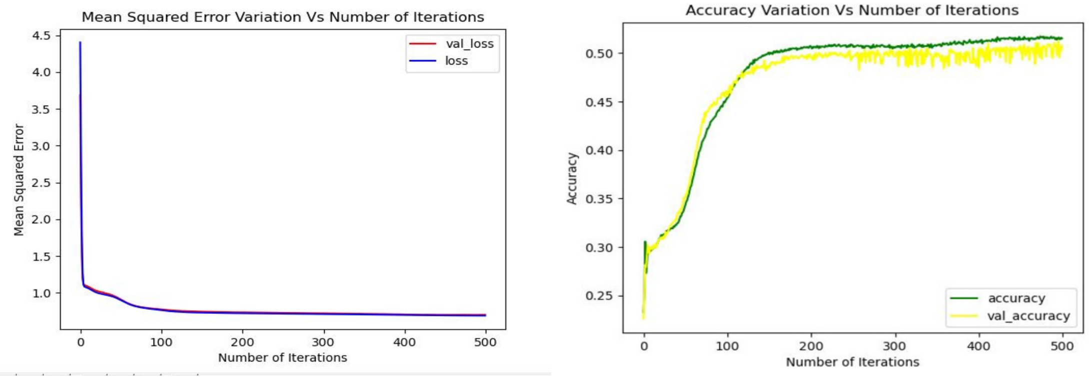


Figure 4.10: Left: MSE Vs Number Of iterations Y estimation in concrete unpainted environment using the time delay approach. Right: Accuracy Vs Number of iterations Y estimation in concrete unpainted environment using the time delay approach

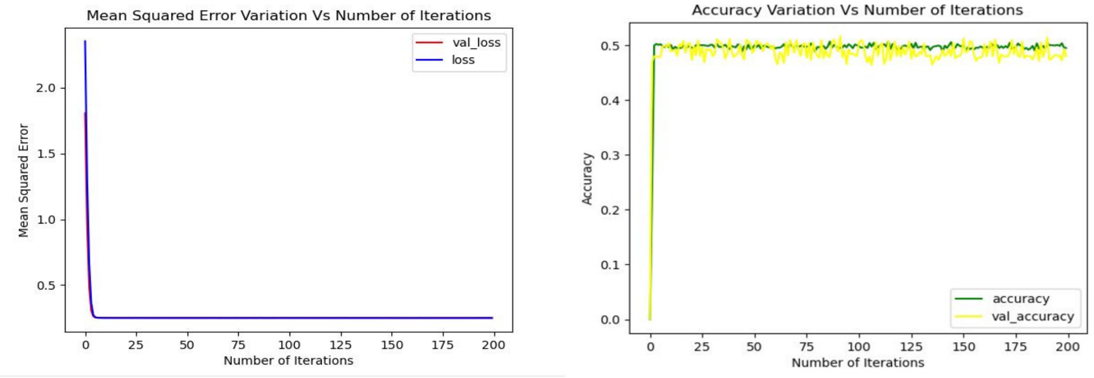


Figure 4.11: Left: MSE Vs Number Of iterations Z estimation in concrete unpainted environment using the time delay approach. Right: Accuracy Vs Number of iterations Z estimation in concrete unpainted environment using the time delay approach

The figures 4.12, 4.13, 4.14 represents the learning curves of our neural network during the estimation of the X, Y, Z positions respectively in a concrete unpainted environment and using the time delay approach. The results are very bad, the accuracy in the estimation of the X coordinate have a maximum rate of 20% and the mean squared error is 5 m^2, the results at the estimation of Y are better. We notice that the accuracy can go up to 25 % and the mean sqaured error is 1 m, but the interesting this is that the learning curves at the estimation of Z are very similar to the ones of the previous environments. Then the results of the testing on this neural network are better than the ones during the learning process, we notice that the mean absolute error during the estimation of Y and Z are close about 0.5 m, but the error percentage of the Y is higher than the Z with a rate of 27.53 %, but the testing on the estimation of the X gave results worse than the Y and Z estimation, the mean absolute error is 1.82 m and the error percentage is 55.82 %.

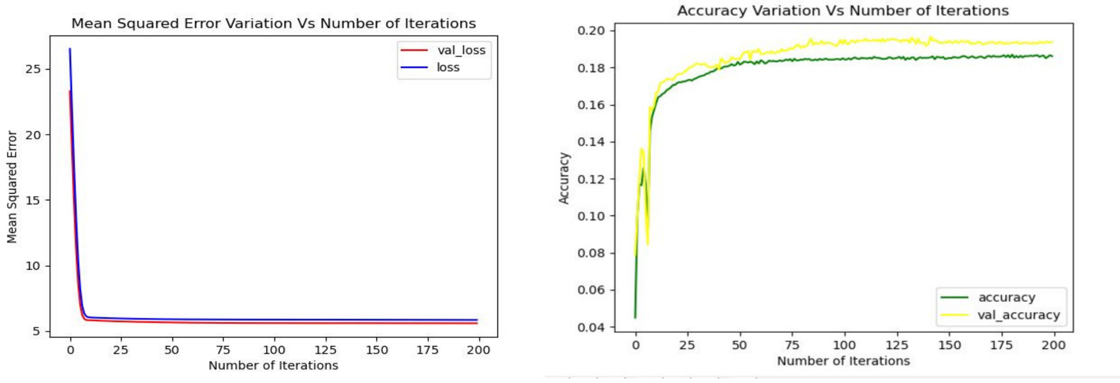


Figure 4.12: Left: MSE Vs Number of iterations X estimation in echoic environment using the time delay approach. Right: Accuracy Vs Number of iterations X estimation in echoic environment using the time delay approach

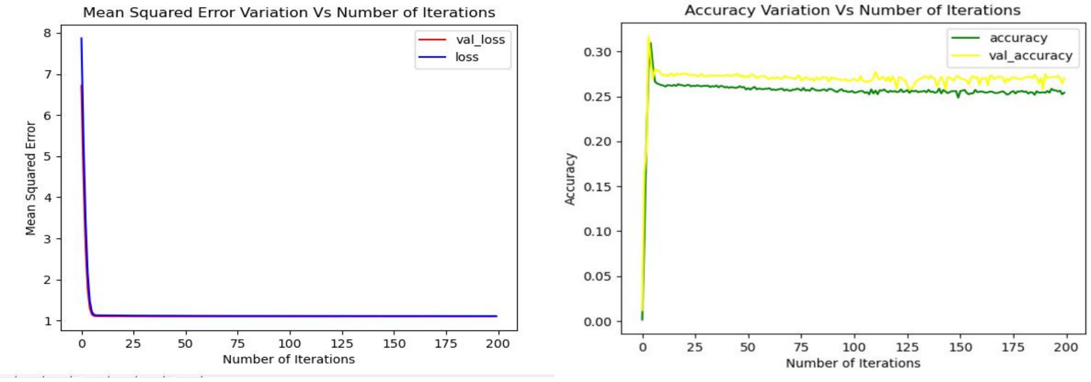


Figure 4.13: Left: MSE Vs Number of iterations Y estimation in echoic environment using the time delay approach. Right: Accuracy Vs Number of iterations Y estimation in echoic environment using the time delay approach

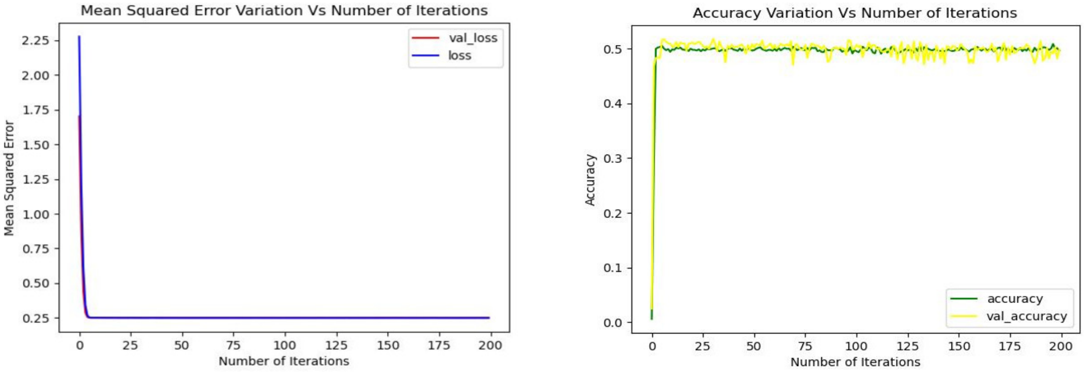


Figure 4.14: Left: MSE Vs Number of iterations Z estimation in echoic environment using the time delay approach. Right: Accuracy Vs Number of iterations Z estimation in echoic environment using the time delay approach

4.3 Energy Difference

Environment	Coordinates	Mean Absolute Error	Error Percentage
Anechoic	X	0.072	2.295
	Y	0.106	4.87
	Z	0.505	20.31
Concrete Unpainted	X	0.71	18.18
	Y	0.496	23.62
	Z	0.51	20.24
Echoic	X	1.141	31.70
	Y	0.551	26.31
	Z	0.504	20.31

Table 4.5: Energy Difference Approach Test Results

In figures 4.15, 4.16, 4.17, we can see the learning curves of our neural network during the estimation of the X, Y, Z positions respectively in an anechoic environment and using the energy difference approach. The learning curves are very much similar to the same conditions using the time delay approach in paragraph 4.2. The testing results are not better that the ones in the paragraph 4.2. We notice that the mean absolute error has a value of 0.072, 0.106 and 0.505 m and the error percentage has a value of 2.295, 4.87 and 20.31 % for the estimation of X,Y and Z respectively.

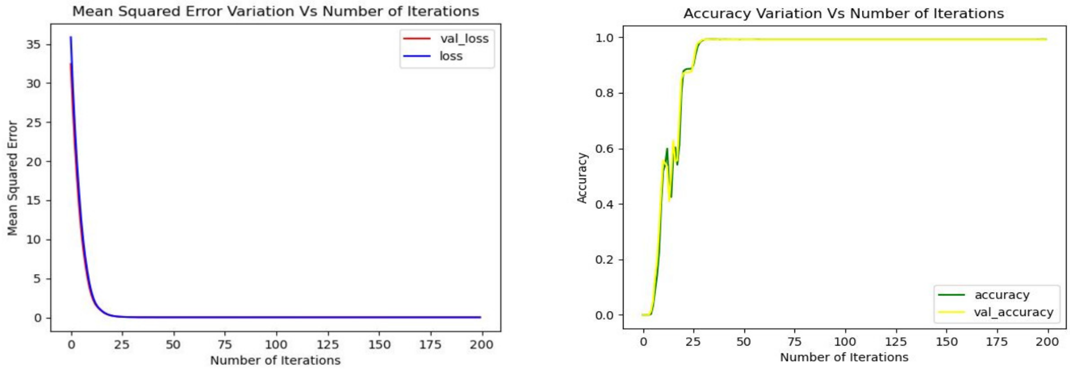


Figure 4.15: Left: MSE Vs Number of iterations X estimation in anechoic environment using the energy difference approach. Right: Accuracy Vs Number ofiIterations X estimation in anechoic environment using the energy difference approach

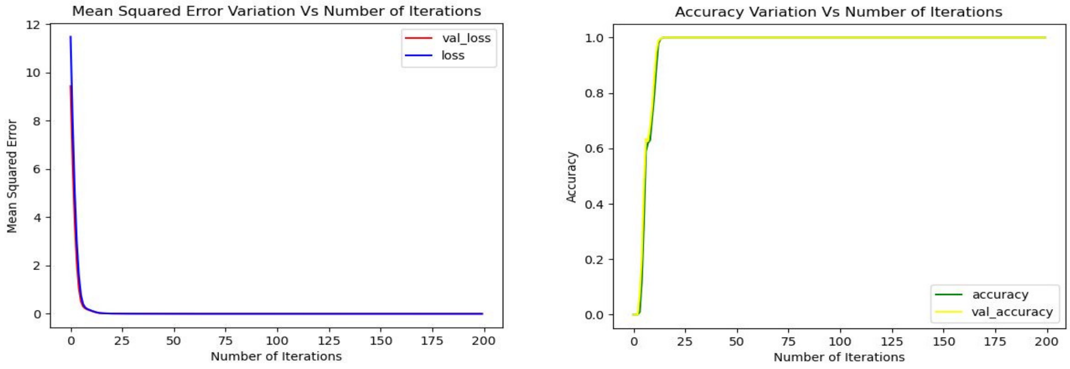


Figure 4.16: Left: MSE Vs Number of iterations Y estimation in anechoic environment using the energy difference approach. Right: Accuracy Vs Number of iterations Y estimation in anechoic environment using the energy difference approach

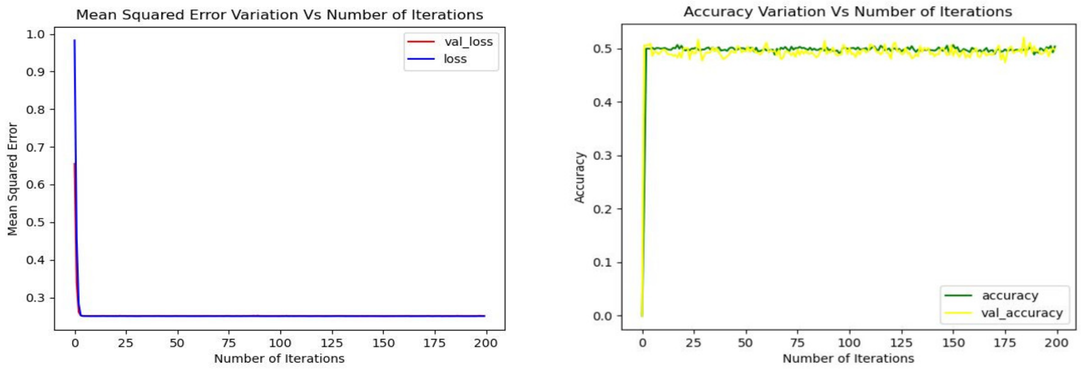


Figure 4.17: Left: MSE Vs Number of iterations Z estimation in anechoic environment using the energy difference approach. Right: Accuracy Vs Number of iterations Z estimation in anechoic environment using the energy difference approach

In figures 4.18, 4.19, 4.20, we can see the learning curves of our neural network during the estimation of the X, Y, Z positions respectively in an concrete unpainted environment and using the energy difference approach. The learning curves of the estimation of X and Z are similar to the learning curves in the same environment using the time delay approach, but the estimation of Y has a lower value of accuracy of 30 %and higher value of mean squared error of 1. The mean absolute error has a value of 0.71, 0.49 and 0.51 m, and an accuracy of 18.18, 23.62 and 20.24% respectively for the testing and the estimation of X, Y and Z in the concrete unpainted environment.

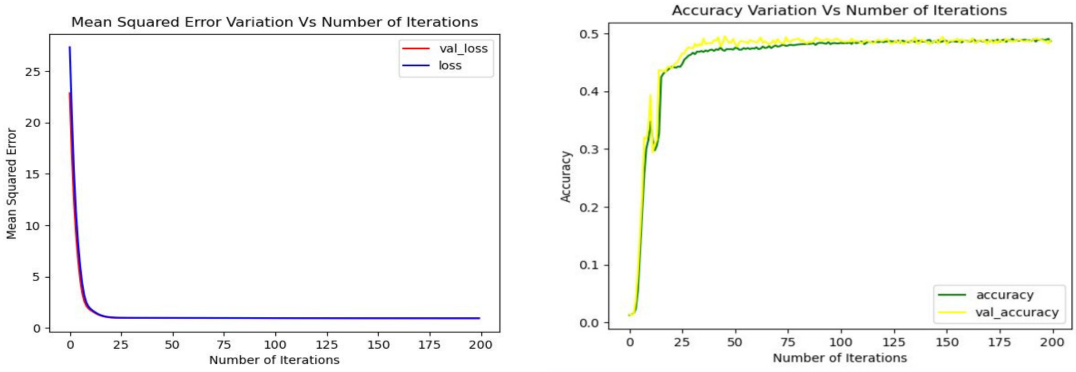


Figure 4.18: Left: MSE Vs Number of iterations X estimation in concrete unpainted environment using the energy difference approach. Right: Accuracy Vs Number of iterations X estimation in concrete unpainted environment using the energy difference approach

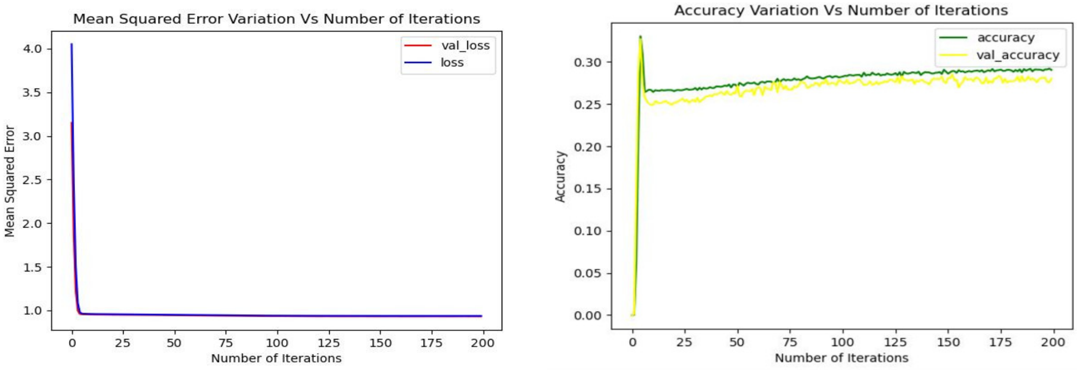


Figure 4.19: Left: MSE Vs Number of iterations Y estimation in concrete unpainted environment using the energy difference approach. Right: Accuracy Vs Number of iterations Y estimation in concrete unpainted environment using the energy difference approach

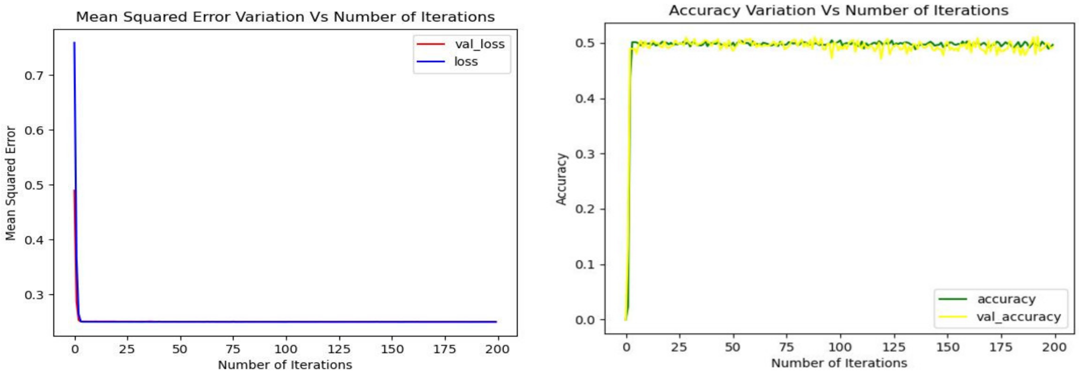


Figure 4.20: Left: MSE Vs Number of iterations Z estimation in concrete unpainted environment using the energy difference approach. Right: Accuracy Vs Number of iterations Z estimation in concrete unpainted environment using the energy difference approach

In figures 4.21, 4.22, 4.23, we can see the learning curves of our neural network during the estimation of the X, Y, Z positions respectively in an echoic environment and using the energy difference approach. The learning curves are better, the mean squared error curves stabilizes at a lower value that the previous study in the time delay approach and the accuracy hits higher values. The mean squared error stabilizes at a value of 1.5, 1.2 and 0.2 m^2, the accuracy reach 37, 30 and 50% during the learning for the estimation of X, Y and Z respectively. The results of the testing

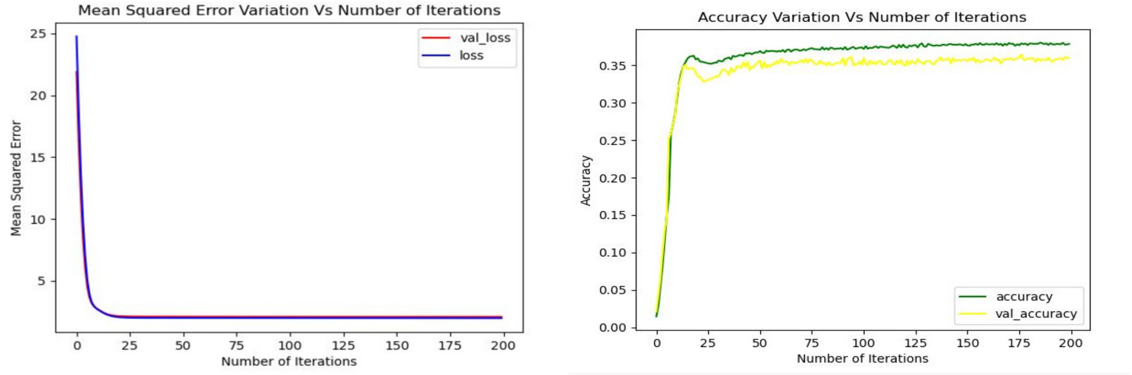


Figure 4.21: Left: MSE Vs Number of iterations X estimation in echoic environment using the energy difference approach. Right: Accuracy Vs Number of iterations X estimation in echoic environment using the energy difference approach

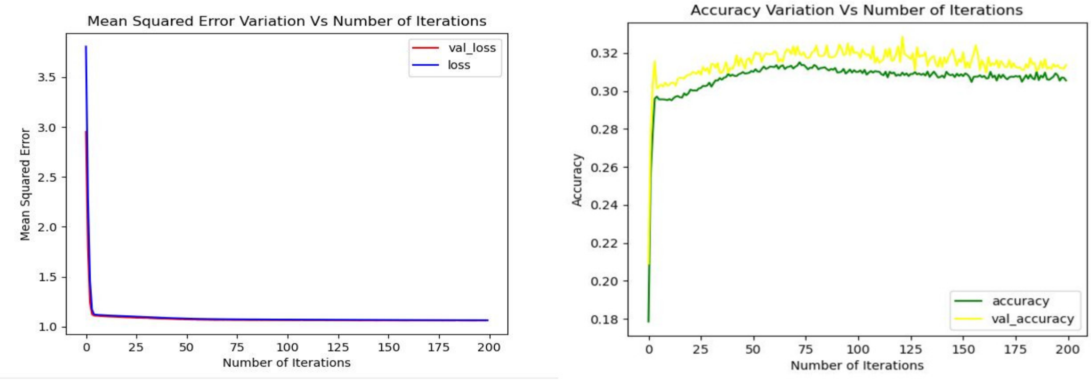


Figure 4.22: Left: MSE Vs Number of iterations Y estimation in echoic environment using the energy difference approach. Right: Accuracy Vs Number of iterations Y estimation in echoic environment using the energy difference approach

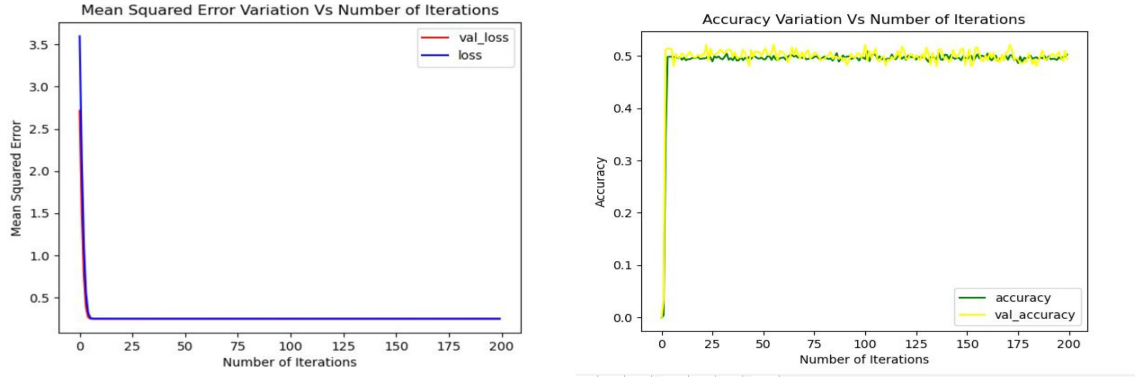


Figure 4.23: Left: MSE Vs Number of iterations Z estimation in echoic environment using the energy difference approach. Right: Accuracy Vs Number of iterations Z estimation in echoic environment using the energy difference approach

4.4 Time Delay and Energy Difference

Environment	Coordinates	Mean Absolute Error	Error Percentage
Anechoic	X	0.0197	0.64
	Y	0.048	2.22
	Z	0.506	20.39
Concrete Unpainted	X	0.585	15.46
	Y	0.471	22.33
	Z	0.504	20.31
Echoic	X	1.128	31.23
	Y	0.52	24.75
	Z	0.508	20.52

Table 4.6: Time Delay and Energy Difference Approach Test Results

In figures 4.24, 4.25, 4.26, we can see the learning curves of our neural network during the estimation of the X, Y, Z positions respectively in an anechoic environment and using the time delay and energy difference approach. The learning curves are very much similar to the same conditions using the time delay approach in paragraph 4.2. The testing results using this approach are better than the ones of the two previous approaches for the same conditions. The mean absolute error have values of 0.0197, 0.048 and 0.506 m. The values of the error percentage are 0.64, 2.22 and 20.39 % respectively for the estimation of X,Y and Z in the anechoic environment.

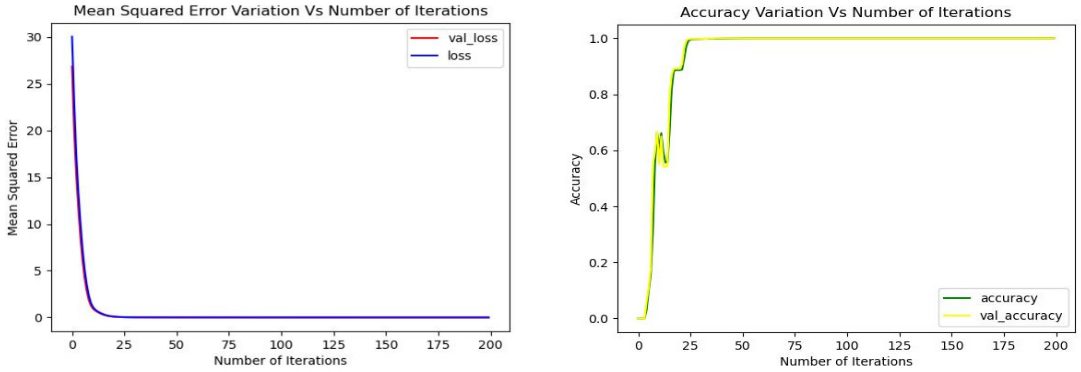


Figure 4.24: Left: MSE Vs Number of iterations X estimation in anechoic environment using the time delay and the energy difference approach. Right: Accuracy Vs Number of iterations X estimation in anechoic environment using the time delay and the energy difference approach

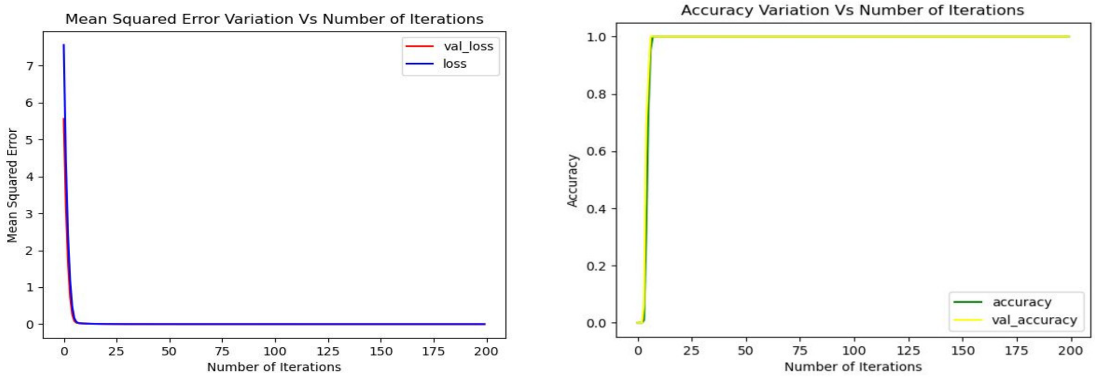


Figure 4.25: Left: MSE Vs Number of iterations Y estimation in anechoic environment using the time delay and the energy difference approach. Right: Accuracy Vs Number of iterations Y estimation in anechoic environment using the time delay and the energy difference approach

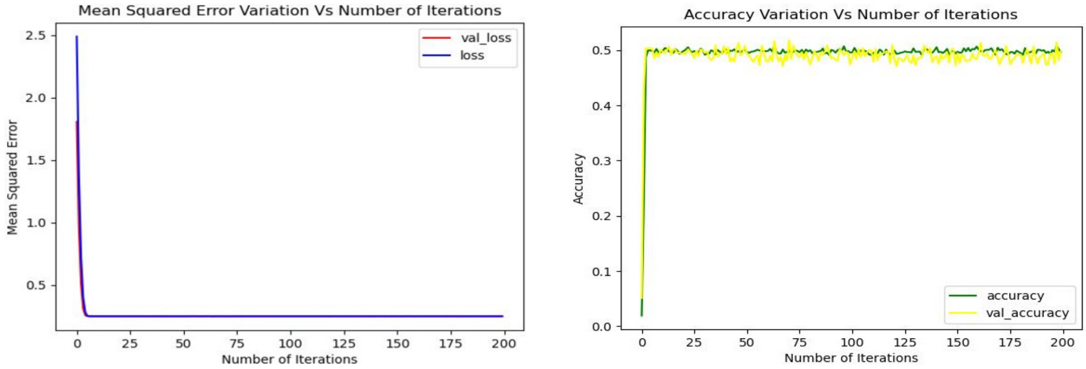


Figure 4.26: Left: MSE Vs Number of iterations Z estimation in anechoic environment using the time delay and the energy difference approach. Right: Accuracy Vs Number Of Iterations Z estimation anechoic environment using the time delay and the energy difference approach

In figures 4.27, 4.28, 4.29 we can see the learning curves of our neural network during the estimation of the X, Y, Z positions respectively in a concrete unpainted environment and using the time delay and energy difference approach. The learning curves are better than the curves in the previous approaches, the accuracy reaches about 60% in the estimation of X and Y and the mean squared error tends to be 0 in the estimation of X, and about 0.4 in the estimation of Y. The estimation is very much similar to the curves in the previous approaches. The testing results are better than the previous approaches, we notice the mean absolute error have values of 0.585, 0.471 and 0.504 m, and the error percentage 15.46, 22.33 and 20.3 % respectively for the estimation of X,Y and Z.

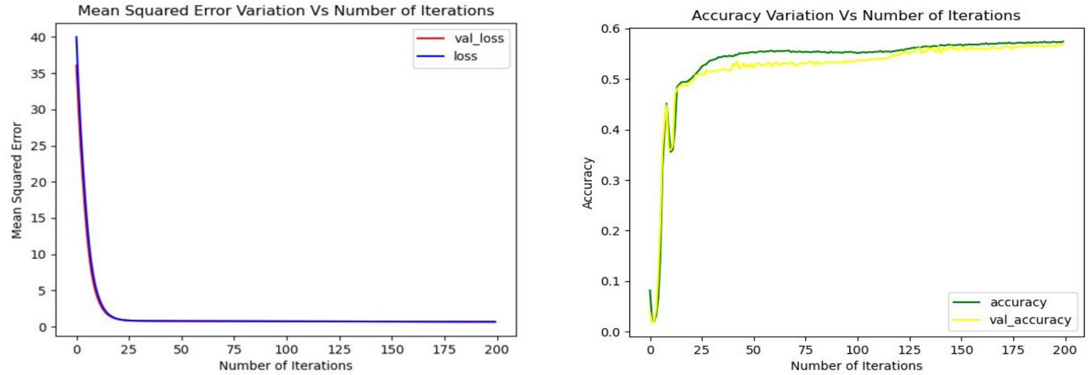


Figure 4.27: Left: MSE Vs Number of iterations X estimation in concrete unpainted environment using the time delay and the energy difference approach. Right: Accuracy Vs Number of iterations X estimation in concrete unpainted environment using the time delay and the energy difference approach

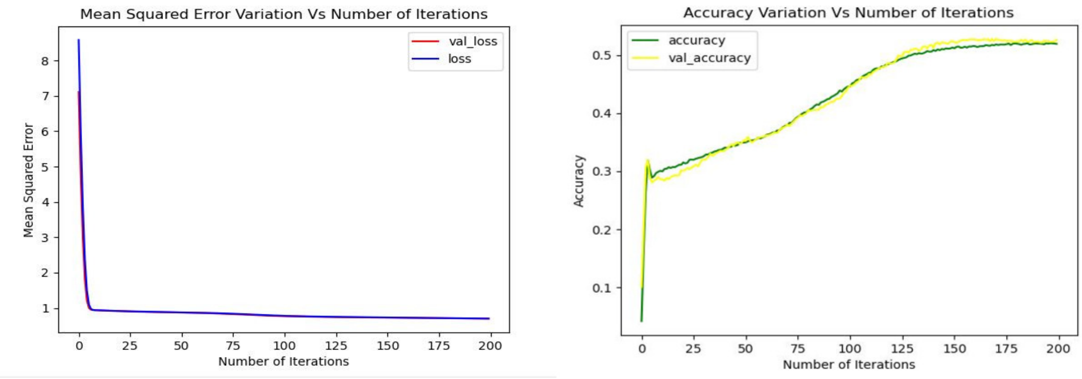


Figure 4.28: Left: MSE Vs Number of iterations Y estimation in concrete unpainted environment using the time delay and the energy difference approach. Right: Accuracy Vs Number of iterations Y estimation in concrete unpainted environment using the time delay and the energy difference approach

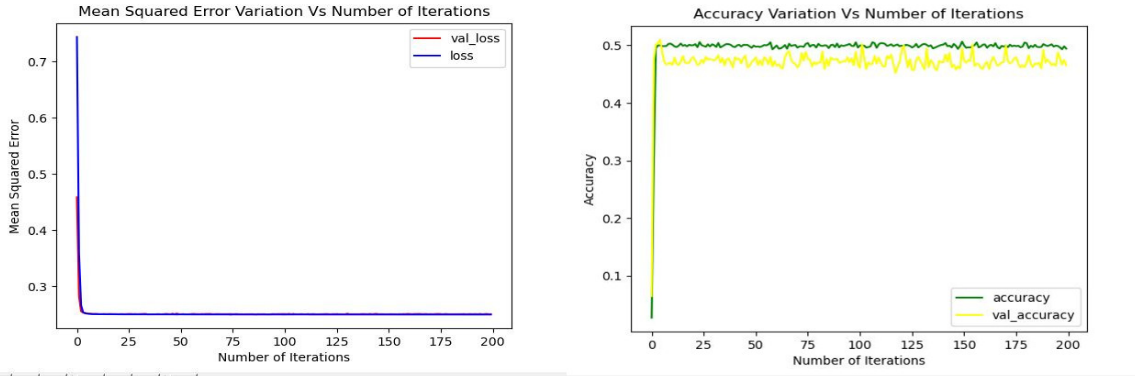


Figure 4.29: Left: MSE Vs Number of iterations Z estimation in concrete unpainted environment using the time delay and the energy difference approach . Right: Accuracy Vs Number of iterations Z estimation in concrete unpainted environment using the time delay and the energy difference approach

In figures 4.30, 4.31, 4.32, we can see the learning curves of our neural network during the estimation of the X, Y, Z positions respectively in a concrete unpainted environment and using the time delay and energy difference approach. The learning curves are similar to the curves of the previous approaches for the echoic environment. The testing results are better than the previous approaches, we notice the mean absolute error have values of 1.128, 0.52 and 0.508 m, and the error percentage 31.23, 24.75 and 20.52 % respectively for the estimation of X,Y and Z.

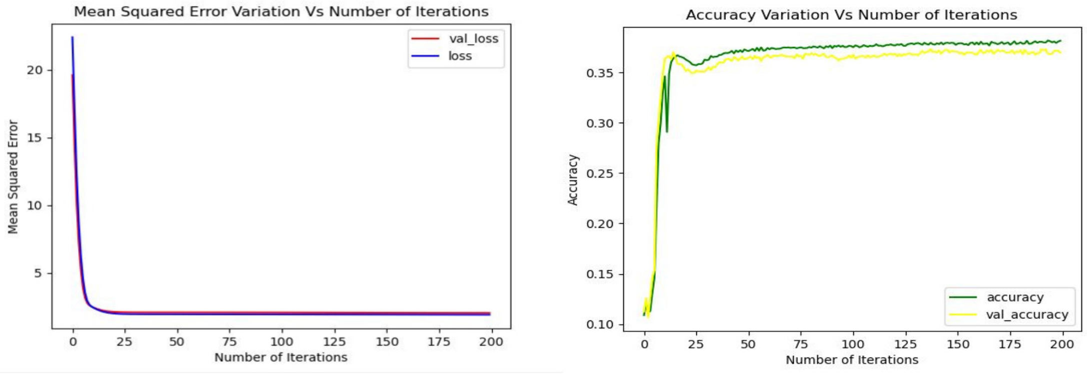


Figure 4.30: Left: MSE Vs Number of iterations X estimation in echoic environment using the time delay and the energy difference approach. Right: Accuracy Number of iterations X estimation in echoic environment using the time delay and the energy difference approach

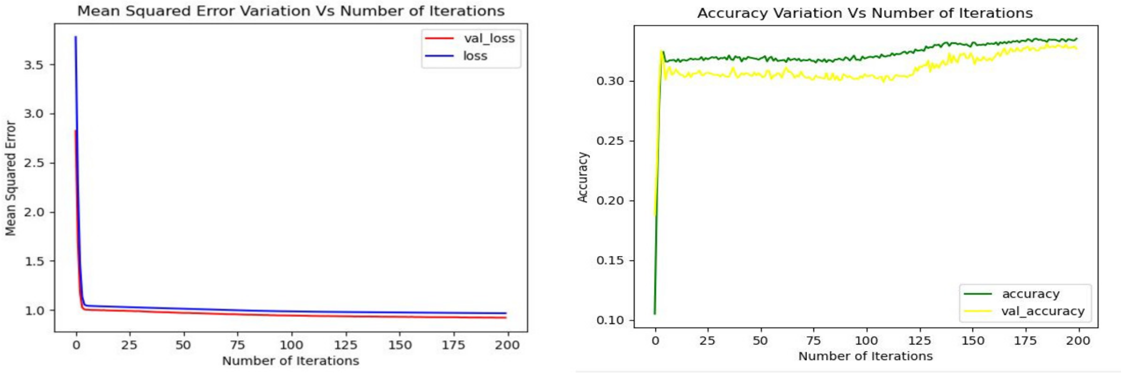


Figure 4.31: Left: MSE Vs Number of iterations Y estimation in echoic environment using the time delay and the energy difference approach. Right: Accuracy Vs Number of iterations Y estimation in echoic environment using the time delay and the energy difference approach

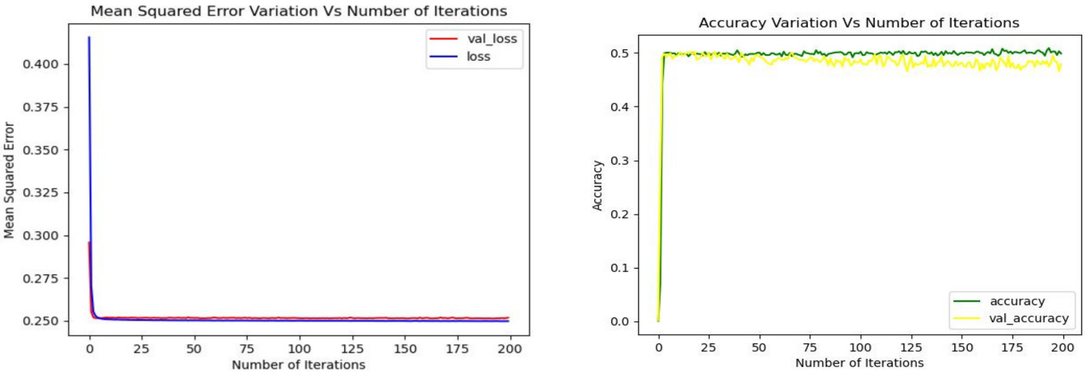


Figure 4.32: Left: MSE VsNumber of iterations Z estimation in echoic environment using the time delay and the energy difference approach. Right: Accuracy VsNumber of iterations Z estimation in echoic environment using the time delay and the energy difference approach

4.5 Analysis of the results

The testing of the neural network, and as mentioned earlier is made on positions that were never in the training data of the of the neural network, and it was testing on completely new dataset and totally new estimations, when testing on data that the neural network already trained on, we will be expecting to see lower error values. In the previous paragraphs 4.2, 4.3 and 4.4, the results from different approaches and in different environment and between the coordinates was presented and compared. We notice that the estimation of Z is the same, not varying when the parameters changed. This is because the training outputs of the Z coordinates are just two, so a lot of different data exists only for the two same output, this will lead us to a confused neural network and not being able to learn properly for the estimation of this output, plus the microphones are placed in the same plane z which will not allow the extraction of precise information about the elevation. The training positions of the X coordinate are 9, of Y are 4 and of Z there are only 2. And we cam notice in the anechoic environment, more training positions the better the results. But in the echoic environment this principle is reversed.

When discussing the environments, the only different parameter is the reverberation, the anechoic environment don't have any reverberation, the echoic environment has the highest rate of reverberation and we notice that the lowest accuracy and the highest errors exists in the echoic environment. So this means that the higher reverberation rate, the higher the value of the errors and the lower the accuracy, the harder for the neural network to learn in these conditions. The reverberation alters the signals and reduce the quality of the information extracted from these signals at the time difference.

Despite the quality of the acoustics, the neural network is capable of learning and estimation the position of a sound source. As if it were learning the acoustics indirectly and their influence on the position features. Discussing the different approaches, we notice that when using both features in our neural network we get the best results, rather than using only one feature. Indeed this allows the neural network to take advantage of a combination of information provided by the energy difference and time difference. This makes data richer, and training more efficient and generalization easier.

4.6 Noisy Data

In this paragraph, we will be training our model on the data coming from the anechoic environment and using the approach using both of the features extracted, the time delay and the energy difference since we obtained the best training and testing results. But the testing will be on positions the neural network did not use it to train, which we added white gaussian noise with a signal to noise equal to 20.

We can conclude that the noise have the same effect on the estimation of the neural network just like the reverberation. The more noise exists the worst the estimation of the neural network.

Coordinate	Mean Absolute Error	Error Percentage
X	2.88 m	86.95%
Y	0.66 m	28.14%
Z	0.64 m	30.63%

Table 4.7: Results from testing on noisy signal but training on clean signal

We can conclude that the noise have the same effect on the estimation of the neural network just like the reverberation. The more noise exists the worst the estimation of the neural network.

4.7 Simultaneous Sound Sources

After training the neural network on an anechoic environment with the time delay and energy difference approach, we tested the neural network by giving it the features extracted from an audio signal corresponding to two sources transmiting signals at the same time. The signals were recorded when the two sources are close, far or with a distance in between. The training of the neural network was on anechoic data with one source signal, but the testing took place on signals containing the mixed audio signals of two active sound sources in the room. The coordinates of the sources are $S(1, 1, 1.5), S_n(2, 2, 2), S_m(6, 3, 1.5), S_f(8.5, 4, 2.5)$. So when saying close the features were extracted from the signal mixing S and S_n. Medium in between means the features are extracted from the sources S and S_m, and far between the sources S and S_f. In the first paragraph, the two sources are transmitting the same audio signal, but in the second paragraph the sources are transmitting different signals.

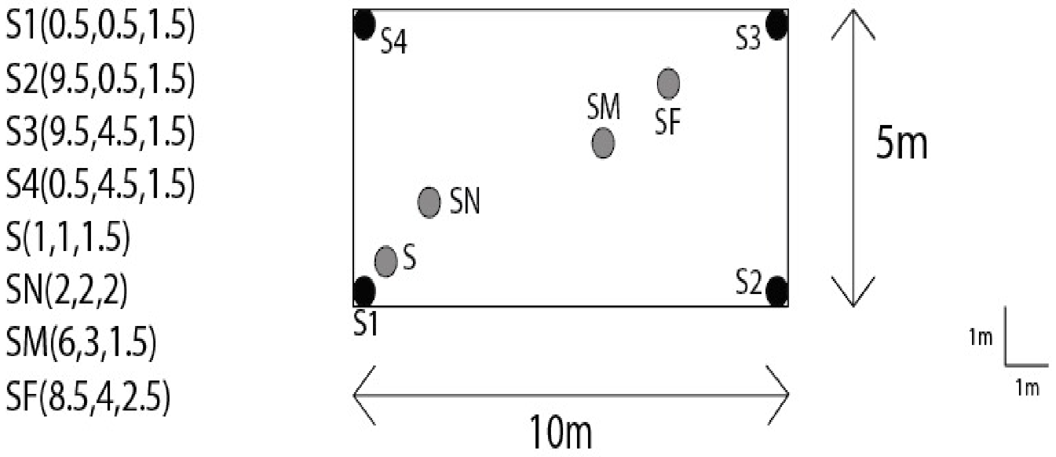


Figure 4.33: Multiple Sound Source positions in the room top view

4.7.1 Transmiting the same signal

Status	Coordinates	Mean Absolute Error(m)	Output Average(m)
Close	X	3.18	5.18
	Y	0.76	2.76
	Z	0.457	1.54
Medium Distance	X	1.65	7.66
	Y	0.35	2.65
	Z	0.45	1.05
Far	X	1.62	6.88
	Y	1.54	2.46
	Z	1.41	1.09

Table 4.8: Two Sources transmiting same signal tests

Distance	Frame Number	X	Y	Z
Close	1	6.13	3.11	1.59
	2	3.628	2.23	1.53
	3	4.86	2.55	1.48
	4	4.68	2.98	1.64
	5	5.83	3.05	1.64
Medium	1	7.79	2.62	1.02
	2	7.56	2.62	1.06
	3	7.72	2.66	1.00
	4	7.66	2.85	1.08
	5	7.62	2.63	1.05
Far	1	6.88	2.45	1.09
	2	6.92	2.45	1.08
	3	6.86	2.46	1.10
	4	6.88	2.46	1.08
	5	6.88	2.45	1.09

Table 4.9: Output testing on two sound sources transmitting same signal

As we notice, the mean absolute error of this estimation is high, and the estimated outputs are not close to any of the sound source positions. Since the sources are transmitting the same signal, when computing the time delay feature, the system will be confused and the estimation of this feature will not be exact, because the same signal is being received an additional one time, this can put us in a case pretty much close to an environment with a reverberation and an echo rate in the room. From the previous tests and learning curves we concluded the effect of this reverberation on the estimation of the system and how it affects it badly. In the table below, couple of frames estimation were presented to monitor, if the system is varying the estimation between the coordinates of the source or stabilizing on a mean value and varying around it. The estimation of the coordinates don't have a big variance, this means that the estimation is being made arround a mean value error and varies within the variation of the loudness of the system. Other approaches can be envisioned in order to enhance and improve the estimation and localization of two sources in the future work.

4.7.2 Transmiting different signals

Status	Coordinates	Mean Absolute Error	Output Average(m)
Close	X	2.231	4.23
	Y	1.10	3.10
	Z	1.167	0.83
Medium Distance	X	0.475	6.31
	Y	0.252	2.75
	Z	0.975	0.524
Far	X	3.26	5.23
	Y	1.849	2.15
	Z	0.811	1.68

Table 4.10: Two Sources transmiting different signal tests

The results in this case in not better than the previous one, we notice that the estimation error is higher. The results don't flow a clear pattern when speaking in terms of estimation between the coordinates, but when speaking in terms of X, we notice that the estimation error varies a lot between a distance and another. This act in monitoring is not a lot helpful in this case specially because the period of the frame which is 1 second. This frame period don't allow us to monitor the estimation of each frame because of the frame period. When the frame period is high just like this case, both of the sources will be active in all the frames and we won't be able to detect when one of them is off and see how the estimation of the neural network will act accordingly. In this case a post processing step could be helpful in order to refine the outputs and improve the results.

Distance	Frame Number	X	Y	Z
Close	1	4.18	3.12	0.85
	2	4.20	3.14	0.84
	3	4.30	3.11	0.84
	4	4.25	3.12	0.83
	5	4.32	3.12	0.83
Medium	1	5.77	2.94	0.40
	2	5.77	2.96	0.41
	3	6.26	2.69	0.43
	4	6.45	2.68	0.45
	5	6.52	2.69	0.47
Far	1	5.73	2.14	1.63
	2	5.85	2.21	1.55
	3	5.69	1.98	1.8
	4	5.66	2.01	1.78
	5	5.64	2.07	1.78

Table 4.11: Output testing on two sound sources transmitting different signal

4.8 Discussion

When speaking in terms of sound source localization, most of the previous papers present the determination of the sound source either in one dimension which is the direction of arrival. Or in two dimensions, estimations the azimuth and the elevation angle, because in such features and using the spherical coordinates it is difficult to estimate the distance. In this project the sound source localization is built in a 3D Cartesian coordinates using a neural network, would this of X, Y and Z be applicable without the help of the neural network and only based on the signals features? The features, the approaches and the tests in this project, presented pretty much all the problematics related to the sound source localization domain. From the reverberation (from different environments) to different coordinates system, the noise and the multiple sources [10]. This paper published in 2017, presented the indoor sound localization trying to estimate two coordinates and compute the this one which is the distance in high reverberation environments and achieved an error of 4 degrees in this the estimation of azimuth and elevation angles [11]. The following paper presented a comparison of two algorithm based on time delay, and presented their results. Most of the studies takes into consideration on feature to try the estimate the position of a source, either the time delay or the energy difference of the signal, and of course with a period of frame that do not pass 50 ms, in this project the two approaches were compared clearly with a bigger frame period and in the end a new approach was used, the one that presented the best of the two features by mixing it together and based on the results presented earlier, this approach was a lot more helpful in the localization of a sound source in the environments. In the following paper [12], a lot of sound source localization algorithms were introduced and reviewed, but the approach explained earlier in which we used all the features extracted is not mentioned, this could only mean that this method of solving problems might open new beginnings in the sound source localization domain.

Chapter 5

Conclusion

The purpose of this Project is the 3D localization of a sound source in a room in a specific environment. Using the RoomSim simulator, we created multiple room with different acoustics properties to determine in which the best results obtained and in three different approaches, time delay, energy difference and using both of them to localize the sound source while using a four microphone array and extracting from it the required information for the training of our neural network and testing our Network for the best results. As a conclusion for all the 27 tests presented earlier, the best result were in an anechoic environment using the time delay and the energy difference approach for the localization of the source in the learning and testing phase, and the results decreased gradually once going from and environment with less reverberation to one with more, and improved gradually once going from the time delay approach to the energy difference approach. And last but not least the time delay and energy difference approach. So based on these results we can say the reverberation caused by echoes in a room are very much important and affects the features extracted from the signals from the room. This project might be a little step forward in the research domain of a room acoustics or in a military application or even in human robot interaction. The creation of the database was not easy in a way or another and we pulled it off, maybe by uploading this database online, it can be useful to researchers seeking good performance in the machine learning domain. This approach helped solving a couple of problems but in return it presented and showed us a couple of difficulties such as the reverberation and the need to dereverberate the signal received in an efficient way, multiple sound source localization and estimation, and tracking of a moving sound source in an indoor environment. This project helped my Matlab Skills and python skills specially in what comes in the signal processing and machine learning applications. It was a big push and help to obtain the mindset of an engineer in terms of thinking planning, executing and managing the time and analyze the results in order to become a successful engineer.

Bibliography

[1] Monika Rychtarikova,Jan Wouters,Gerit Vermeir. Binaural Sound Source Localization in Real and Virtual Rooms.Journal of the Audio Engineering Society 4(57). 2012.

[2] Tao Song, Jing Chen, Daibing Zhang, Tianshu Qu, Xihong Wu.Acoustic Array Systems.Proceedings of the 22^{nd} *International Congress on Acoustics. Paper ICA2016-352.A sound source localization algorithm using microphone array with rigid body. 2016.*

[3] Ryu Takeda and Kazunori Komatani. Discriminative multiple sound source localization based on deep neural networks using independent location model.IEEE Spoken Language Technology Workshop (SLT).Date of Conference: 13-16 Dec. 2016.INSPEC Accession Number: 16657499.

[4] Hendrik Kayser, Jorn Anemuller. A discriminative learning approach to probabilistic acoustic source localization.Conference: 2014 14^{th} *International Workshop on Acoustic Signal Enhancement (IWAENC).*

[5] Dan Ellis. A history overview of Machine Listening.Presented at the Computational Audition Workshop, University College London, May 12-14, 2010.

[6] Douglas R. Campbell, Kalle J. Palomäki and Guy J. Brown. A MATLAB Simulation of "Shoebox" Room Acoustics for use in Research and Teaching.9th INTERNATIONAL CONGRESS ON ACOUSTICS MADRID, 2-7 SEPTEMBER 2007.

[7] Prarthan Mehta and Vaishalee Bhadradiya.Measurement of Room Impulse Response Using Image Source Method. International Conference on Electrical, Electronics and Mechatronics (ICEEM).2015.

[8] D. Surov, D. Ge, and R. Zhukov. Deep residual network for sound source localization in the time domain.Journal of Engineering and Applied Sciences, 2018, vol. 13, no. 13, P. 5096-5104. 2018.

[9] Martın Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: A system for large-scale machine learning.12th USENIX Symposium on Operating Systems Design and Implementation.2016.

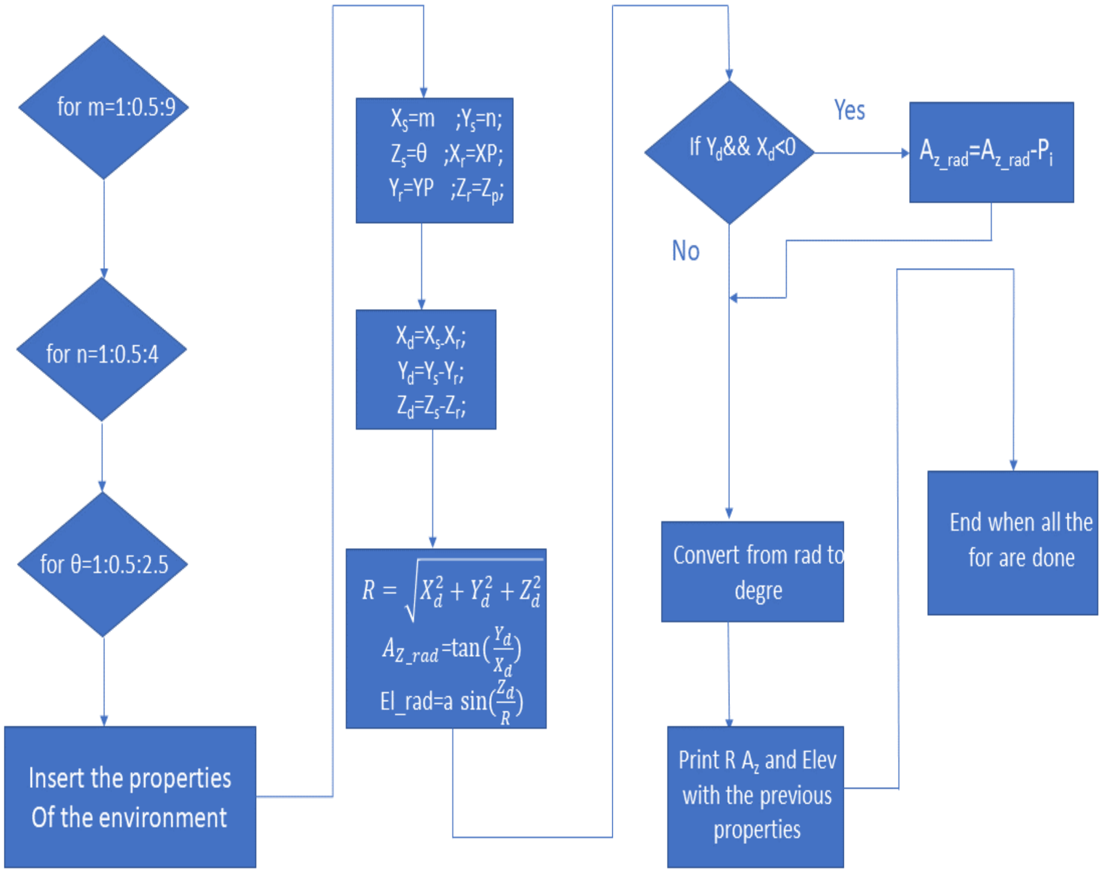
[10]Yingxiang Sun, Jiajia Chen, Chau Yuen, and Susanto Rahardja. Indoor Sound Source Localization with Probabilistic Neural Network .IEEE Transactions on Industrial Electronics, vol. 65, no. 8, pp. 6403-6413, Aug. 2018.

[11] Alessio Brutti, Maurizio Omologo, Piergiorgio Svaizer.Comparison between different sound source localization techniques based on a real data collection.Hands-Free Speech Communication and Microphone Arrays.2008.

[12] Maximo Cobos, Fabio Antonacci, Anastasios Alexandridis, Athanasios Mouchtaris, and Bowon Lee. Review Article A Survey of Sound Source Localization Methods in Wireless Acoustic Sensor Networks.Wireless Communications and Mobile Computing.2017.

Appendices

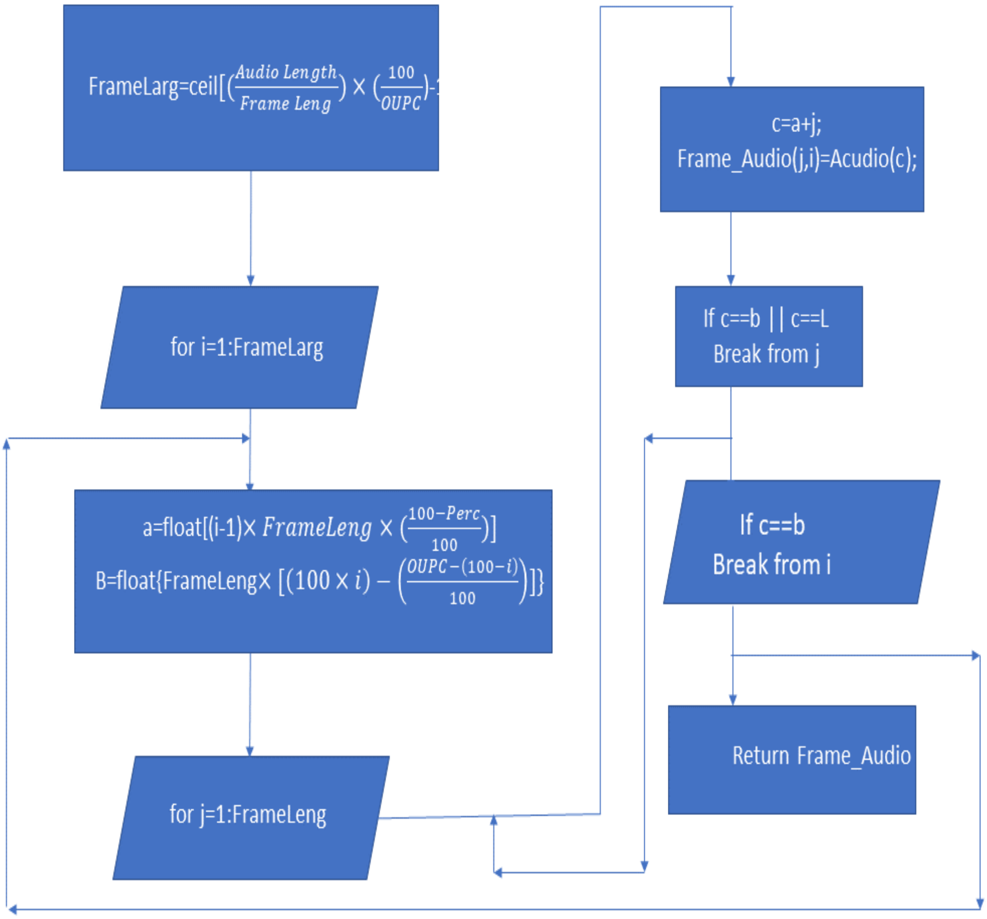
A.1 Textfile Generation



for m=1:0.5:9
for n=1:0.5:4
for θ=1:0.5:2.5
Insert the properties Of the environment
X_S=m ;Y_s=n; Z_s=θ ;X_r=XP; Y_r=YP ;Z_r=Z_p;
X_d=$X_{s-}X_r$; Y_d=Y_s-Y_r; Z_d=Z_s-Z_r;
$R = \sqrt{X_d^2 + Y_d^2 + Z_d^2}$
A_{Z_rad}=tan$(\frac{Y_d}{X_d})$
El_rad=a sin$(\frac{Z_d}{R})$
If Y_d&& X_d<0
Yes
No
A_{z_rad}=A_{z_rad}-P_i
Convert from rad to degre
Print R A_z and Elev with the previous properties
End when all the for are done

Figure A.1: Textfile Generation Flowchart

A.2 Frame Decomposition



FrameLarg=ceil[($\frac{Audio\ Length}{Frame\ Leng}$) × ($\frac{100}{OUPC}$)-
for i=1:FrameLarg
a=float[(i-1)× $FrameLeng$ × ($\frac{100-Perc}{100}$)]
B=float{FrameLeng× [$(100 \times i) - \left(\frac{OUPC-(100-i)}{100}\right)$]}
for j=1:FrameLeng
c=a+j;
Frame_Audio(j,i)=Acudio(c);
If c==b || c==L
Break from j
If c==b
Break from i
Return Frame_Audio

Figure A.2: Frame Decomposition Flowchart

A.3 Silence Removal

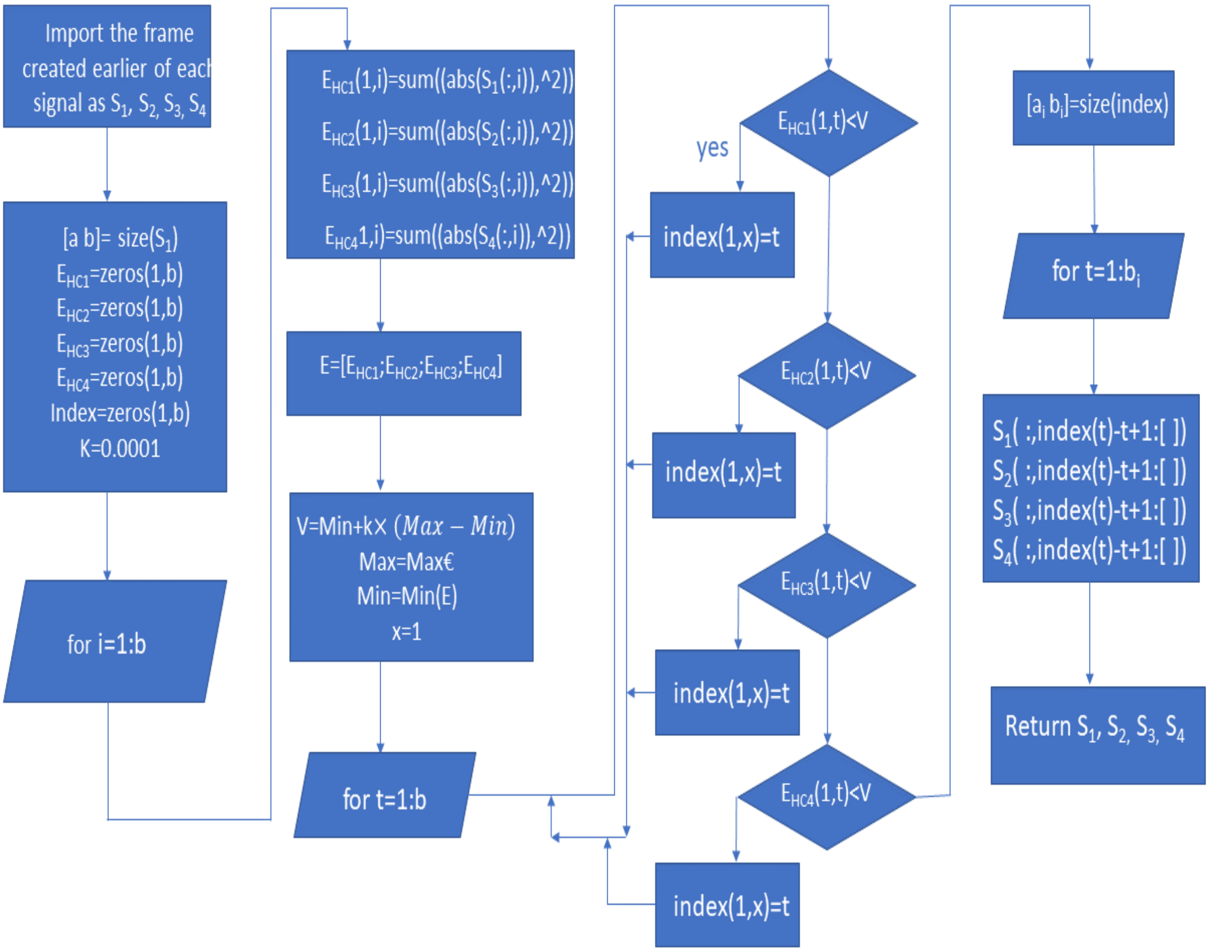

Import the frame created earlier of eac signal as S_1, S_2, S_3, S_4
[a b]= size(S_1)
E_{HC1}=zeros(1,b)
E_{HC2}=zeros(1,b)
E_{HC3}=zeros(1,b)
E_{HC4}=zeros(1,b)
Index=zeros(1,b)
K=0.0001
for i=1:b
E_{HC1}(1,i)=sum((abs(S_1(:,i)),^2))
E_{HC2}(1,i)=sum((abs(S_2(:,i)),^2))
E_{HC3}(1,i)=sum((abs(S_3(:,i)),^2))
E_{HC4}1,i)=sum((abs(S_4(:,i)),^2))
E=[E_{HC1};E_{HC2};E_{HC3};E_{HC4}]
V=Min+kx $(Max - Min)$
Max=Max€
Min=Min(E)
x=1
for t=1:b
E_{HC1}(1,t)<V
yes
index(1,x)=t
E_{HC2}(1,t)<V
index(1,x)=t
E_{HC3}(1,t)<V
index(1,x)=t
E_{HC4}(1,t)<V
index(1,x)=t
[a_i b_i]=size(index)
for t=1:b_i
S_1(:,index(t)-t+1:[ ])
S_2(:,index(t)-t+1:[ ])
S_3(:,index(t)-t+1:[ ])
S_4(:,index(t)-t+1:[ ])
Return S_1, S_2, S_3, S_4

Figure A.3: :Silence Removal Flowchart

A.4 Projet TDOA(Time Difference of Arrival

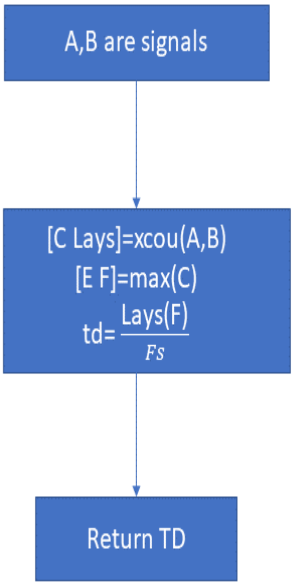


Figure A.4: Function TDOA Flowchart

A.5 Time Delay and Energy Difference Calculation

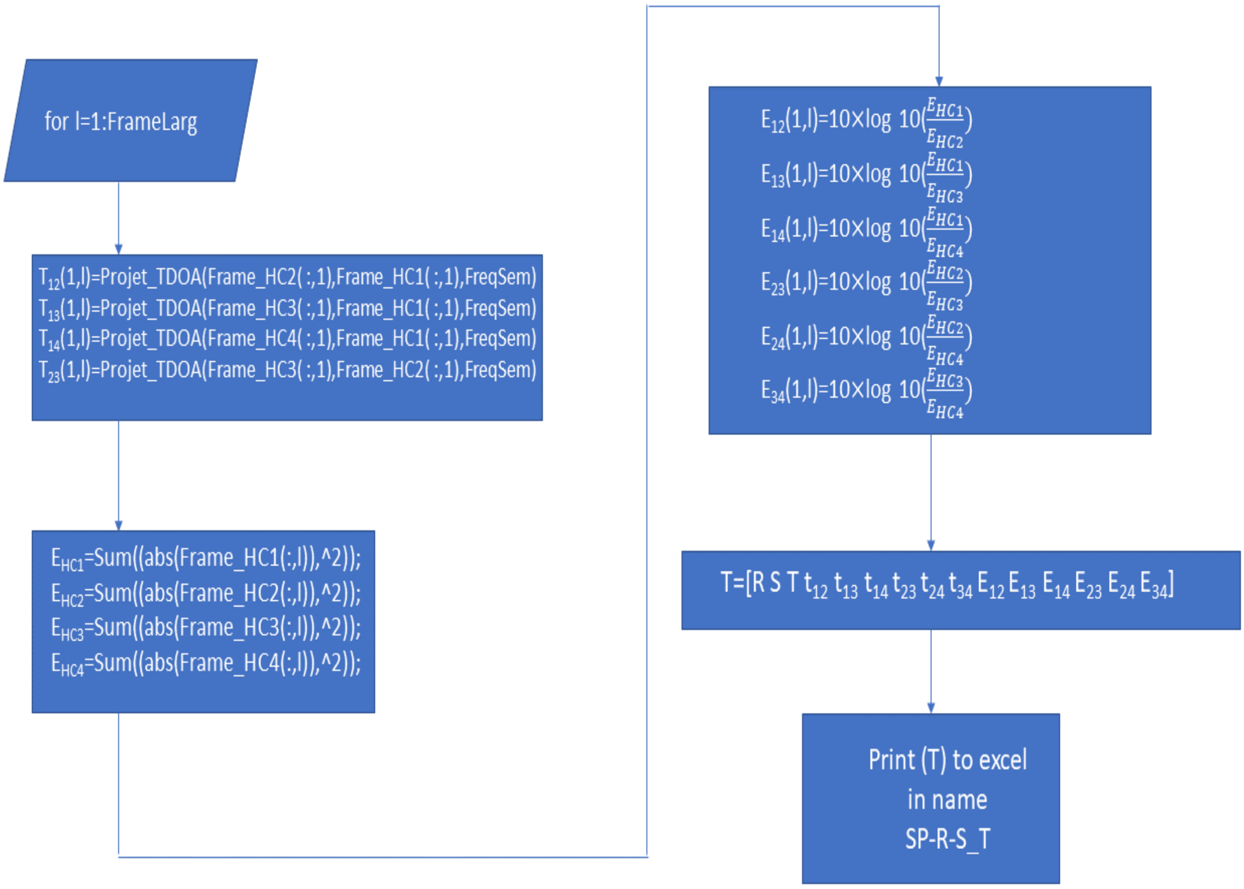

for l=1:FrameLarg
T_{12}(1,l)=Projet_TDOA(Frame_HC2(:,1),Frame_HC1(:,1),FreqSem)
T_{13}(1,l)=Projet_TDOA(Frame_HC3(:,1),Frame_HC1(:,1),FreqSem)
T_{14}(1,l)=Projet_TDOA(Frame_HC4(:,1),Frame_HC1(:,1),FreqSem)
T_{23}(1,l)=Projet_TDOA(Frame_HC3(:,1),Frame_HC2(:,1),FreqSem)
E_{HC1}=Sum((abs(Frame_HC1(:,l)),^2));
E_{HC2}=Sum((abs(Frame_HC2(:,l)),^2));
E_{HC3}=Sum((abs(Frame_HC3(:,l)),^2));
E_{HC4}=Sum((abs(Frame_HC4(:,l)),^2));
$E_{12}(1,l)=10\times\log 10(\frac{E_{HC1}}{E_{HC2}})$
$E_{13}(1,l)=10\times\log 10(\frac{E_{HC1}}{E_{HC3}})$
$E_{14}(1,l)=10\times\log 10(\frac{E_{HC1}}{E_{HC4}})$
$E_{23}(1,l)=10\times\log 10(\frac{E_{HC2}}{E_{HC3}})$
$E_{24}(1,l)=10\times\log 10(\frac{E_{HC2}}{E_{HC4}})$
$E_{34}(1,l)=10\times\log 10(\frac{E_{HC3}}{E_{HC4}})$
T=[R S T t_{12} t_{13} t_{14} t_{23} t_{24} t_{34} E_{12} E_{13} E_{14} E_{23} E_{24} E_{34}]
Print (T) to excel
in name
SP-R-S_T

Figure A.5: Time Delay and Energy Difference calculation Flowachart