Hellenic Mediterranean University
School of Music and Optoacoustic Technologies
Department of Music Technology and Acoustics

A Stereo Sonic Interaction System Triggered by Impact Sounds

M.Sc. Thesis
of
Achilles Kappis

Supervisor: Nikolaos Stefanakis,
Assistant Professor

April 16, 2022

A Stereo Sonic Interaction System Triggered by Impact Sounds

Achilles Kappis
mta2@edu.hmu.gr

April 16, 2022

Acknowledgements

There is a large number of people to whom I owe my gratitude. First and foremost I would like to thank my supervisor Nikolaos Stefanakis for his inexhaustible support, the freedom he provided me with to work on the Thesis topic and the gentle pressure he applied when the circumstances dictated it. Without his motivation the fulfillment of this Thesis wouldn't had been possible. I found ample support in every step of my course through this Masters degree from people who I cannot but wholeheartedly thank. Special thanks belong to Stella Paschalidou who was the instigator of this journey and a strong supporter throughout its duration, Chrysoula Alexandraki for her motivating presence in times of personal recession and Eythimios Bakarezos for his enlightening conversations and unending support up to this very moment.

I am blessed with wonderful friends and relatives who have been on my side long before the departure to this venture. They all contributed equally and each one in their own unique way. Thanks from the depths of my heart go to my twin brother Aris, my young brother Dimitris and our mother Margarita. Their constant support in my every move and decision has been invaluable. I could not but include to this list my beloved friend Michalis Terzakis, who helped in various parts of this work and supported me with long "light" and enlightening conversations. Last but definitely not least I have to thank my good friends and "pressure release valves", Lambros and Maria for all the long coffee and beer moments and the kind reminders of the need to maintain life regulation and balance as much as possible.

I apologise to all those I may have missed. This work is attributed to them as much as is to those mentioned above and myself. Their contribution, direct or indirect, has been the driving force for the fulfillment of this Thesis.

Abstract

Sonic interaction is an active research field with applications in video games, smart homes and music performance. It allows for both tangible and intangible control through a wide variety of electroacoustic transducers that lend themselves to use in such systems. Traditionally, sonic interaction systems rely on machine learning methods to recognise the gestural stimuli, and thus require a large database for the training phase which sometimes may burden the end user. In this Thesis, the problem of sonic interaction is studied from the perspective of a system that has a stereo audio capturing system, e.g. two microphones. Particularly, a system comprised of an energy detector and a Time-Difference-of-Arrival (TDoA) estimator is implemented and tested, assuming a user that produces impulsive gestures, at different locations, as stimuli. The Thesis presents the theoretical background of several energy detectors and TDoA estimators and presents simulation and real-data results obtained under different noise conditions and physical setups. A sonic interaction system with a complete signal processing pipeline is implemented to demonstrate the control of a virtual digital musical instrument with its technical aspects showing excellent results.

Περίληψη

Τα συστήματα ηχητικής διάδρασης αποτελούν ένα ενεργό πεδίο έρευνας με εφαρμογές στα βιντεοπαιχνίδια, τα «έξυπνα σπίτια» και την μουσική εκτέλεση. Η διάδραση μέσω ήχου επιτρέπει απτή και άυλη διάδραση με την χρήση πλήθους ηλεκτροακουστικών μετατροπέων που μπορούν να χρησιμοποιηθούν σε τέτοιου είδους συστήματα. Συνήθως, τέτοιου τύπου διαδραστικά συστήματα βασίζονται σε μεθόδους μηχανικής μάθησης για την αναγνώριση των ήχων που παράγονται από το χρήστη, επομένως υπάρχει η ανάγκη εκπαίδευσης του συστήματος πριν από τη χρήση. Σε αυτή την διπλωματική εργασία, το πρόβλημα της ηχητικής διάδρασης εξετάζεται από την σκοπιά ενός συστήματος δύο μικροφώνων. Συγκεκριμένα, υλοποιήθηκε και δοκιμάστηκε ένα σύστημα αποτελούμενο από έναν ανιχνευτή ενέργειας και έναν εκτιμητή διαφοράς χρόνου άφιξης ανάμεσα στα δύο μικρόφωνα, θεωρώντας πως ο χρήστης θα παράγει κρουστικούς ήχους, σε διάφορες περιοχές, σαν μέσο διέγερσης. Παρουσιάζεται το θεωρητικό υπόβαθρο για διάφορους ενεργειακούς ανιχνευτές και εκτιμητές διαφοράς χρόνου άφιξης καθώς και αποτελέσματα προσομοιώσεων και πραγματικών μετρήσεων όπως αυτά προέκυψαν για διάφορες συνθήκες θορύβου και διατάξεις μικροφώνων. Με βάση κάποιες από αυτές τις τεχνικές υλοποιήθηκε εν τέλει ένα ολοκληρωμένο σύστημα ηχητικής διάδρασης για τον χειρονομιακό έλεγχο ενός ψηφιακού μουσικού οργάνου, με πολύ καλά αποτελέσματα στο τεχνικό κομμάτι.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Human-Computer Interaction

Human-Computer Interaction is a field that has drawn significant attention the last years despite the fact that it is a necessity emerged with the advent of computers. The fist appearance of the term is dated back in 1976 [6] and the field is a cross-disciplinary intersection of computer science, design, media and behavioral and cognitive sciences. Since the first days, it has evolved with the help of the significant progress made in electronics, both in the analog and digital domains. The latest processing units (CPU) used in today's personal computers allow for the operation of millions of calculations in a fraction of a second, dramatically increasing the capabilities of a system both in the amount of data to be processed as well as the complexity of the processing [7].

Emergent fields such as *Computer Vision* and *Machine Learning* have led to the implementation of many interactive systems capable of recognising and producing perceptually high level visual and acoustic events and stimuli [8]. Such systems are the topic of ongoing research and most of them are very demanding on processing power [9]. Nevertheless, those new systems find applications in many fields with diverse needs and specific features such as education [10, 11], medical sciences, entertainment [12, 13] and finance [14].

Most state-of-the-art systems are based on some kind of machine learning algorithm running in the back-end, in order to reach educated decisions on classification problems with the classes being emotions, gestures or any other high level cognitive feature and then provide the appropriate feedback. Even if the system is able to respond in real time, which often is not the case, most, if not all, systems require training in order to

"learn" their corresponding tasks [15]. Even if a large enough database is available to train the system, fine-tuning is a necessity and even then, the system will be able to recognise and react to specific events.

It is still very difficult to create generic interactive systems of low or even medium scalable complexity, able to be used on personal computers or smaller Microprocessor Units (MPU) for embedded applications. Lately some manufacturers have released the first Neural Processing Units (NPU), which are MPUs optimised for the acceleration of machine learning algorithms [16]. This may bring more machine learning and more complex interactive systems to the public but this is something to be witnessed in the near future.

There has been a constant trend to increase the complexity of the HCI systems in order to either achieve better results in regard to accuracy, or push the boundaries of current state-of-the-art methods in order to broaden the applicability of the system. The requirements an HCI system must meet, although may vary based on the use case, are well defined. "Almost" real-time response is most often a prerequisite for such a system, where certain tolerances are acceptable for some applications.

A constantly increasing demand for computational power is evident. Since a barrier has been reached on CPU speed, the solution is sought on different ways to increase the available computational capacity of the hardware [7]. Most often the simplification of the underlying algorithms or the decrease in the data used in a system constitute the ultimate solution for the system designers, resorting to it only when there is no other way to overcome the inability to perform all needed computations under the physical constraints (many times the most important constraint is the response time).

Despite the flood of machine learning algorithms in the recent literature, classical and empirical methods still find use in HCI systems [17, 18]. Many of these methods use statistical Digital Signal Processing techniques to perform detection of signals and estimation of other parameters such as pitch, spectral density, coherency and temporal characteristics [1, 19]. In [20] one can find many DSP algorithms to perform analysis and processing of audio signals resulting from deterministic formulations. Many such algorithms have found extensive use in HCI systems created for various applications ranging from artistic interactive installations [21, 22] and experimental musical hyper-instruments [23, 24] to control of smart appliances [17] and mobile devices [25].

It is the purpose of this Thesis to explore the applicability of such classical DSP techniques of relatively low complexity in HCI systems. In the current work, statistical and deterministic DSP methods are evaluated

for the purpose of creating a generic sonic interaction control system. The implementation of the complete pipeline is not designed with a specific use case in mind allowing for use in a broad spectrum of applications. For more information on the implemented system see *Section 1.3* below.

1.2 Applications

Many HCI systems have already been presented. This section contains suggestions relevant to the possible applications in which the tools implemented in this Thesis may find use. Many of the proposed fields are almost sterile of HCI systems based on sonic gestures.

1.2.1 Music performance

Interactive systems have found extensive applicability in the musical and entertainment industry [13]. Both commercial [26] and experimental [27] systems have been implemented to enhance the expressiveness of digital musical instruments or augment the capabilities of "conventional" ones, called hyper-instruments [23, 24].

Many systems use as input percussive sounds or other impulsive acoustic events [18, 28–31], as is the case of the current work. Contrary to the system of this Thesis, most of the implementations encountered in the literature use machine learning techniques. This requires that the system has to go through a training phase before deployment. Furthermore, incorrect training may degrade the accuracy or limit the applicability of the system to specific stimuli.

Although there is a plethora of research work on the different ways to control sound (only a small fraction of the available literature is cited here) and the mapping of features or control signals to audio parameters, only a small number is concerned with the use of sound as the control signal.

1.2.2 Video games

The video games industry is one of the biggest markets worldwide. A vast amount of console, computer and nowadays mobile phone games exist with many of them being simulators of real life events, such as racing or sports. Many different multi-modal controllers have been developed by companies but, to our knowledge, none of the commercial products utilise audio as a means to create control signals.

The only exception found is *Scream Flappy* [32] (and *Flappy Voice* [33] for iOS) and its clone *Eigth Note* [34], which are mobile phone games where the user controls the movement of a bird with their voice.

It seems that the control of game aspects with acoustic gestures is still a relatively unexplored topic. Relying only on sound to control a game may be difficult, especially when the degrees of freedom of the player's movement increase, or there is a large dictionary of actions to be controlled. Nevertheless, using sound in conjunction with other gestures could possibly increase the expressiveness of a control system.

An excellent example is Kinect [35], a motion sensing input device developed by Microsoft®. It contains a three color channel (Red Green Blue - RGB) camera, infrared depth sensor and a microphone. Despite the broad range of possible applications sensor fusion of the available data could provide, there is but a few proposals utilising the audio capabilities of the device [36, 37]. The next generation of Kinect is Azure Kinect DK which contains higher resolution RGB camera and depth sensor and a microphone array of seven channels in a circular configuration for 360^o audio capture [38]. Despite the superior technical characteristics of this device compared to its predecessor most research has been on topics benefited by the image sensors. Some use of the microphone array of this device has been proposed but in medical applications, not related to this work [39].

1.2.3 Smart appliances

Natural language processing, speech-to-text technology and machine learning have led to the introduction of the so called smart assistants like Alexa® (from Amazon®) and Siri® (from Apple®). These virtual assistants can be controlled with voice commands and respond verbally or through text. These complex systems are the product of years of development and huge training databases have been used to bring them to their current state.

Simpler systems can be used to control various appliances such as the one presented in [17]. In this work, the playlist of an MP3 player is controlled with finger snaps through the microphones present on a pair of headphones.

There is a lack of acoustical interfaces in the literature for use in similar applications. This may be partly justified due to the fact that it is not very intuitive to control appliances with acoustical gestures, but this could have also been the case for tangible control before the first touch-screens made their appearance. This field seems to be broadly open to experimentation. Possible applications include remote and intangible control of

home appliances, home automation, reduction of energy consumption and increase in security with detection of abnormal situations [40, 41].

1.3 Motivation

It is clear at this point that HCI systems find a wide range of applications. Such a system is present whenever there is the need to interact with a computer or a machine. Many of them constitute extremely complex systems that are developed and tuned for specific purpose. Furthermore, we have seen that various fields lending themselves to the development of interactive control systems are still vastly unexplored. In most applications presented in *Section 1.2* the time constraints of the systems are very strict with possible exception being the control of smart appliances and homes, where the constraints are relaxed to a certain degree.

This work aims at trying to tackle those issues. The sought system must satisfy the time constraints and be generic enough to be utilised in as many of the presented fields as possible. The first requirement leads to the realisation of a low complexity system so that it may be used on as many machines as possible regardless of processing capacity[1]. Ideally, the algorithms will be simple enough to adhere to the time constraints but easily parallelised and scalable.

The second requirement dictates the use of simple acoustic events easily produced with means available to the common person in their everyday life as triggers to the system. Moreover, the physical part should be simple. Simplicity in both the hardware and software parts of the system will allow for easy integration in existing systems. A simple system could be transported with ease, used on the fly for artistic or other purpose or be installed without the need of specialised equipment under special conditions.

An acoustic gesture controlled system possesses many positive traits against systems that use visual cues to recognise human gestures. First and foremost audio signals are one dimensional as opposed to the video streams that contain many pixels, often in three color channels, for each frame with 25 or 30 frames per second (this is considered the lowest standard in many applications). If the high complexity of computer vision or image processing algorithms is also taken into account, the computation power needed can be of orders of magnitude less in acoustic systems. Another important characteristic of audio is the speed of acquisition. Sam-

[1] This refers to current CPUs and MPUs but not necessarily expensive, high performance hardware.

pling rates are very high compared to the refresh rates of most widely used cameras[2]. The use of overlapping windows and small frame sizes can result in significantly shorter latency in acoustic systems.

Speech signals have been extensively used as the means to control interactive systems with impulsive sounds being widely neglected for this purpose. Speech has evolved to convey high level information and is one of the most complex audio signals with highly variable temporal and spectral characteristics. On the contrary, impact sounds are short duration acoustic events, well localised in time, making them ideal for high temporal accuracy and fast control of machines. Moreover, impulse sounds can be detected in noisy environments without the need of sophisticated algorithms. Both speech and impulsive sounds can be easily produced in a variety of situations (e.g. clap, fingersnap, tapping on a solid surface). Impulsive acoustic events compose a very generic class of sounds which the vast majority of people can generate making them ideal for people with disabilities and for mute people.

An aspect that is oftentimes overlooked is the affordability of a system. Arguably, there is a plethora of cheap visual input devices in the market that could possibly be used in machine control systems, but the simplicity of acoustic transducers makes them even cheaper. Furthermore, in acoustic interaction systems reciprocal transducers can be used, which is not the case with visual devices, allowing for control with non-specialised devices such as headphones (used as microphones) or piezoelectric transducers.

Finally, while there are several works in the scientific literature that focus on the recognition of impact sounds based on their sonic characteristics, in this Thesis, the focus is put on an implementation that exploits spatial information, i.e., the location of impact. Such an approach has the advantage that it may be implemented without the need of machine learning and as a consequence, without the need of a training phase.

1.4 Structure of the Thesis

The Thesis begins in *Chapter 2* with a general overview of the work given in *Section 2.1*. The rest of the chapter provides all the necessary theoretical background to follow the practical implementation and experiments that were conducted. The basis of the implementation of the proposed

[2]High speed cameras do exist with refresh rates that can reach even 2 million frames per second [42] but these are not considered here as they constitute highly specialised equipment for use in rare and specific applications (most often very demanding motion capture systems).

sonic interaction system is layed here, divided into sections corresponding to the tasks its subsystems have to perform.

In *Chapter 3* all information regarding the evaluation of the implemented algorithms is presented. The apparatus that was used, information on the environment where the experiments were conducted and evaluation results of the implemented algorithms are all found in this part of the text. The results are divided into sections in a manner similar to that of the previous chapter. Each section is concluded with a brief summary of the results to allow a quick review of the most important findings.

In *Chapter 4* the complete interaction system is evaluated in a scenario portraying the use of the system as a digital instrument controller used in a double layer orchestration with each layer representing a different instrument. The evaluation presents quantitative results whenever possible but qualitative observations are also made on performance and musical articulation aspects.

Finally, *Chapter 5* summarises the work done in this Thesis pointing out the most important aspects and results. Based on the insight provided in this chapter, proposals for improvements and future work are also presented.

Chapter 2

Methodology

The purpose of this work is to investigate signal detection and direction of arrival estimation algorithms that could lead to the implementation of a real time system able to perform both tasks under strict time constraints. The system will act as a sonic interaction interface with which intangible control will become possible.

The main source for interaction with the system are impulsive gestures produced by the user. The existence of a gesture in the input signal is detected at an initial stage by a *Signal Detector*. The angular position of the impact gesture is recognized in real time by the system using Time-Difference-of-Arrival estimation techniques. Such a system can serve along several different interactive applications, such as

- In music performance to control virtual instruments or other synthesis algorithms.
- Video games as an alternative way to control a game which also allows for multi-user control.
- Smart homes for the control of different home appliances based on the location of the user in the house.

2.1 General

The system must accomplish two given tasks. One is to detect the existence of the impulsive sounds in the input signals and the second is the estimation of its angle of incidence. The first falls into the field of *Signal Detection* and the second in that of *Direction-of-Arrival* (DoA) *Estimation*.

For the signal detection task, the investigated algorithms fall within the category of *Binary Hypothesis Testing* were an algorithm, termed *Detector*, makes a decision of whether a signal of interest is present in the

input or not. All evaluated algorithms use the energy of the signal to make a decision, thus they are termed *Energy Detectors*. The detectors evaluated in this Thesis are, an energy detector based on the formulation of the signal and noise as Gaussian distributed random variables [1, 40], a variation of this detector with adaptive threshold [1, 43, 44], a detector that makes decisions based on the variance of the input signal [2] and an empirically derived double-threshold energy detector.

For the task of estimating the angle of incidence, the algorithms investigated are the Generalised Cross Correlation (GCC) and its variations, Roth, SCoT, PhaT, Eckart and ML_{GCC} [5], Thresholded PhaT [45] and PhaT β [46] which constitute variations of the PhaT filter of the GCC and two algorithms of the Maximum Likelihood (ML) family, Conditional Maximum Likelihood (CML) and Unconditional Maximum Likelihood (UML) [4, 47–50].

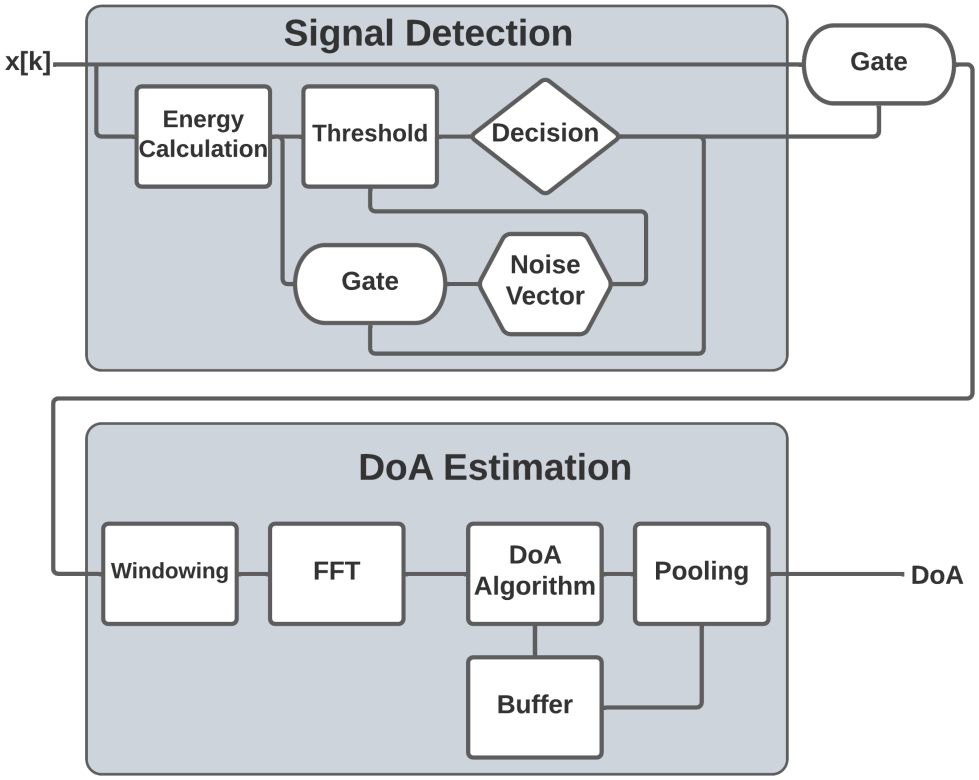


Figure 2.1: Block diagram of the system implemented in this work.

The block diagram of the implemented system is shown in *Figure 2.1*. The steps of the evaluation of each subsystem are shown below in a hier-

archical structure revealing the way tests were performed. The list does not resemble the sequence of the tests.

- Signal Detection
  - Simulated experiments
    - Signal-to-Noise Ratio
      - Thresholds
  - Real data experiments
    - Signal-to-Noise Ratio
      - Zero False Positive Rate
      - Zero False Negative Rate
    - Source type detection
  - Execution speed
- Direction-of-Arrival estimation
  - Simulated experiments
    - Inter-element distance of array elements
    - Variations of GCC algorithms
  - Real data experiments
    - Inter-element distance of array elements
    - Distance of source from the array
    - Windowing functions
    - Signal-to-Noise Ratio
    - Pooling method
    - Spatial aliasing
  - Execution speed

2.2 Signal Detection

The task of detecting the presence of a signal in a time series falls under the broad field of *Binary Hypothesis Testing*, where one of two hypotheses, signal being present or not, has to be decided. The most appropriate approach for this task is the one where the energy of the incoming time series is used to reach the decision. The detectors based on this test statistic are termed *Energy Detectors*. This class is used in this work and presented below.

2.2.1 Binary Hypothesis Testing

The approach followed in this work regarding the signal detection task is centred around binary hypothesis testing schemes. The formulation of the hypothesis tests will be based upon *Gaussian* distributed signals and noise, although the formulation with different Probability Density Functions (PDF) for either noise, signal or both is straight forward.

There are two hypotheses to be tested. One is whether the data contain only noise and the other is that the data contain noise plus the acoustic event of interest. The two hypotheses can be summarised as [41]

$$x(t) = hs(t) + n(t) \tag{2.1}$$

where $x(t)$ denotes the input data, $s(t)$ is the signal of interest, $n(t)$ noise and h is equal to 0 in case the signal is absent and 1 in case the signal is present. The hypothesis for $h = 0$ is denoted with $\mathcal{H}_0$ and termed *Null hypothesis* while for $h = 1$ denoted with $\mathcal{H}_1$ and termed *Alternative hypothesis*.

The algorithms that will decide whether the signal is present or not are called *Detectors*. The simplest example can be formulated if a single realisation of a random variable is considered with distribution $\mathcal{N}(0,1)$ under hypothesis $\mathcal{H}_0$ and $\mathcal{N}(1,1)$ under hypothesis $\mathcal{H}_1$, where $\mathcal{N}(\mu, \sigma^2)$ denotes *Normal* PDF with mean μ and variance σ^2. *Figure 2.2* illustrates this case.

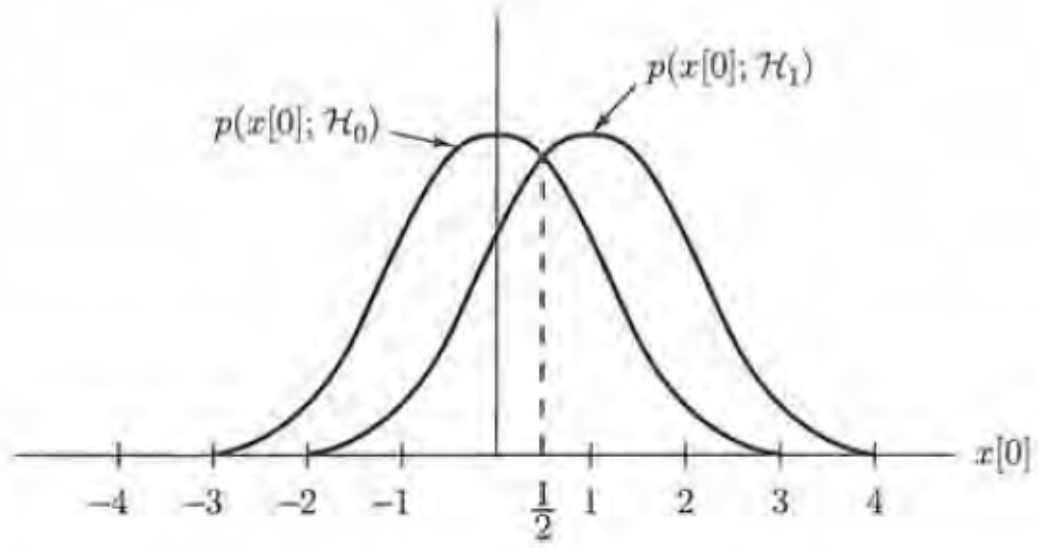


Figure 2.2: Illustration of the PDFs of two hypotheses distributed as $\mathcal{H}_0$: $\mathcal{N}(1,0)$ and $\mathcal{H}_1$: $\mathcal{N}(1,1)$. Taken from [1].

In order to decide upon one of the two hypotheses, a test statistic must be used and compared to a threshold value. It turns out that in this simple case the decision must be taken upon the mean of the two distributions and a reasonable threshold is, as shown in *Figure 2.2*, $\gamma = \frac{1}{2}$ where γ denotes the threshold value. This can be seen if one compares the probability of the received value under the two hypotheses [1]

$$p(x[0]; \mathcal{H}_0) \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\lessgtr}} p(x[0]; \mathcal{H}_1) \tag{2.2}$$

Equation (2.2) can be restated, including vector quantities as [1]

$$L(\mathbf{x}) = \frac{p(\mathbf{x}; \mathcal{H}_1)}{p(\mathbf{x}; \mathcal{H}_0)} > \gamma \tag{2.3}$$

which is called the *Likelihood Ratio Test* (LRT) and $L(\mathbf{x})$ denotes the likelihood ratio of the two PDFs.

The decision of the threshold value can introduce two types of errors. One happens if we decide $\mathcal{H}_1$ when $\mathcal{H}_0$ is true and is termed *Type I* error, *False Positive* (FP) or *False Alarm* (FA). The other is when we decide $\mathcal{H}_0$ while $\mathcal{H}_1$ is true and is termed *Type II* error, *False Negative* (FN) or *Miss*.

In general there are four different probabilities associated with the detection scheme. Those are the probability of correct detection denoted by p_D, false alarm denoted with p_{FA}, miss denoted with p_M and the probability of correctly deciding the absence of a signal denoted with p_{TN} (from *True Negative* (TN)). Below are shown the definitions of the four aforementioned probabilities [51] and *Figure 2.3* depicts the case used so far for an arbitrary threshold where p_{FA} and p_D can also be seen.

$$p_D = \int_{\gamma}^{\infty} p(\mathbf{x}; \mathcal{H}_1)\, d\mathbf{x} \tag{2.4}$$

$$p_{FA} = \int_{\gamma}^{\infty} p(\mathbf{x}; \mathcal{H}_0)\, d\mathbf{x} \tag{2.5}$$

$$p_M = \int_{-\infty}^{\gamma} p(\mathbf{x}; \mathcal{H}_1)\, d\mathbf{x} \tag{2.6}$$

$$p_{TN} = \int_{-\infty}^{\gamma} p(\mathbf{x}; \mathcal{H}_0)\, d\mathbf{x} \tag{2.7}$$

The probabilities of equations (2.4) to (2.7) represent Cumulative Distribution Functions (CDF) or their complements, alternatively called "right-tail" distributions.

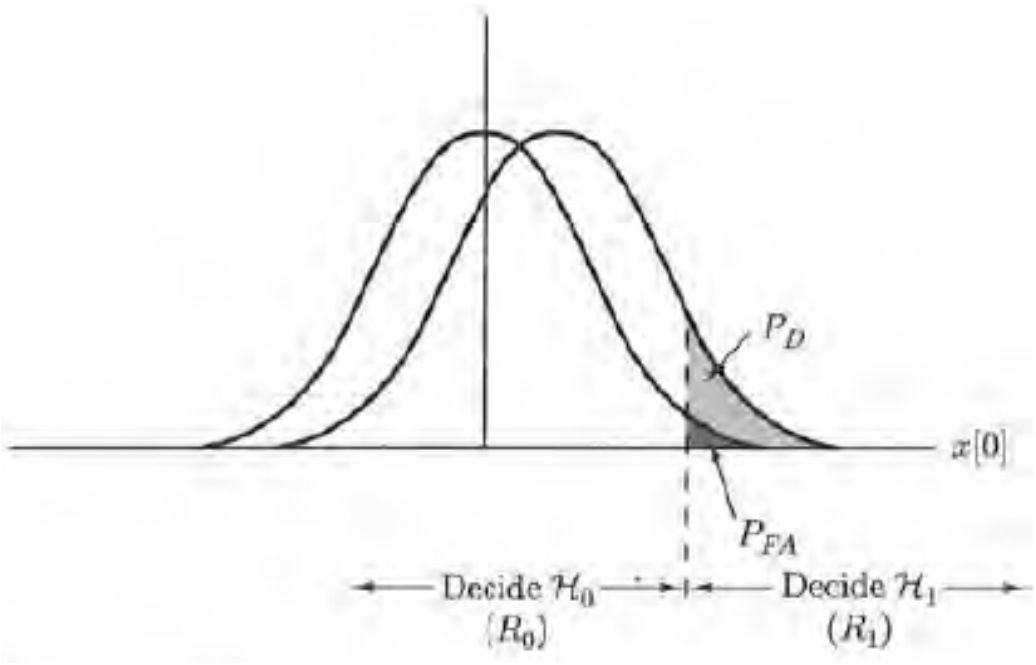


Figure 2.3: Illustration of binary hypothesis testing with two PDFs distributed as $\mathcal{H}_0 : \mathcal{N}(1,0)$ and $\mathcal{H}_1 : \mathcal{N}(1,1)$ for arbitrary threshold. Taken from [1].

2.2.2 Energy Detector

In this Thesis, the statistic used in the hypothesis testing is the energy of the signal, which for the discrete case is given by [19]

$$e = \sum_{n=0}^{N-1} x^2[n] \tag{2.8}$$

with N denoting the number of samples used for the calculation. The detectors implemented on this basis are called *Energy Detectors*. Two main approaches are used in this work. The first considers a theoretical formulation where both the noise and signal are considered to be Independent and Identically Distributed (IID) with Gaussian PDFs and is termed *Gaussian Detector*. The second uses the standard deviation of the energy of the data to reach an educated decision and in this work is termed *Variance Detector*. Both are presented below.

Gaussian Energy Detector

This formulation of the energy detector assumes IID noise and signal with Gaussian distributions. Using equation (2.8), the LRT leads to [1, 51]

$$\sum_{n=0}^{N-1} x^2[n] > \gamma \tag{2.9}$$

For a zero-mean Gaussian PDF this is like comparing the variance of the signal to a threshold. It is intuitive to assume that when there is only noise in the data the variance will be σ_n^2 and when the signal of interest is also present the variance will be $\sigma_n^2 + \sigma_s^2$, with σ_n^2 denoting the variance of the noise and σ_s^2 the variance of the signal.

Noting that the sum of squares of Gaussian distributed random variables has a Chi-squared PDF [52] given by [1]

$$\chi_v^2 = \begin{cases} \frac{1}{2^{\frac{v}{2}}\Gamma\left(\frac{v}{2}\right)} x^{\frac{v}{2}-1} e^{-\frac{1}{2}x} & x > 0 \\ 0 & x < 0 \end{cases} \tag{2.10}$$

where v denotes the degrees of freedom assumed to be a positive integer, in our case equal to N and $\Gamma(u)$ is the Gamma function, defined as [1]

$$\Gamma(u) = \int_0^\infty t^{u-1} e^{-t} dt \tag{2.11}$$

Hence, for the Gaussian energy detector, using equation (2.5) we have the probability of false alarm to be given by [1]

$$p_{FA} = Pr\left\{\sum_{n=0}^{N-1} x^2[n] > \gamma; \mathcal{H}_0\right\} = Q_{\chi_N^2}\left(\frac{\gamma}{\sigma_n^2}\right) \tag{2.12}$$

and similarly for the probability of detection [1]

$$p_D = Pr\left\{\sum_{n=0}^{N-1} x^2[n] > \gamma; \mathcal{H}_1\right\} = Q_{\chi_N^2}\left(\frac{\gamma}{\sigma_n^2 + \sigma_s^2}\right) \tag{2.13}$$

where in both equations $Pr\{\cdot\}$ denotes probability and $Q_{\chi_N^2}$ denotes the right-tail probability function (complement of the CDF) of the χ_v^2 function with N degrees of freedom.

Expressing the Signal-to-Noise Ratio (SNR) as $\frac{\sigma_s^2}{\sigma_n^2}$ and using that as the argument in equation (2.13) we get [1]

$$p_D = Q_{\chi_N^2}\left(\frac{\gamma/\sigma_n^2}{\sigma_s^2/\sigma_n^2 + 1}\right) = Q_{\chi_N^2}\left(\frac{\gamma'}{SNR + 1}\right) \tag{2.14}$$

with γ' denoting a different threshold value. Equation (2.14) shows that the probability of detection depends on the SNR and *Figure 2.4* depicts p_D against SNR for various probabilities of false alarm.

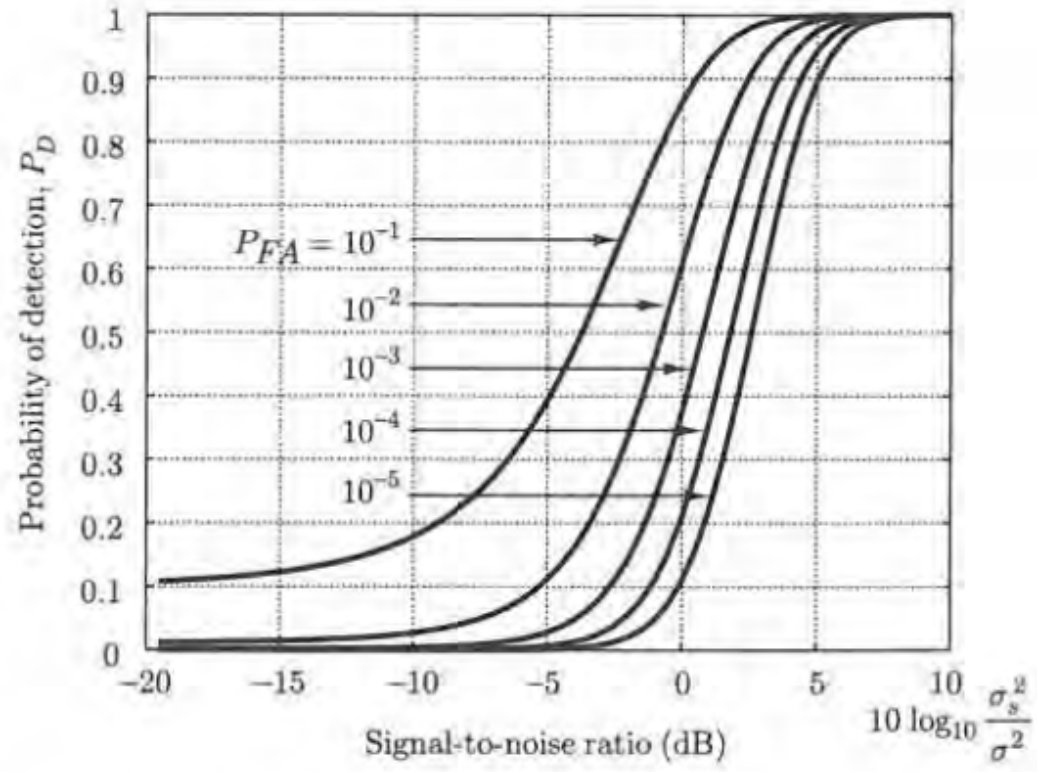


Figure 2.4: Probability of detection against SNR for various probabilities of false alarm for the Gaussian energy detector. Taken from [1].

Constant False Alarm Rate

It can be seen from equation (2.12) that the probability of false alarm depends solely on σ_n^2. If the variance of the noise can be estimated, then using the inverse right tail probability function of a Chi-squared PDF one can calculate the threshold that will achieve a specified probability of false alarm as [1, 53]

$$\frac{\gamma}{\sigma_n^2} = Q^{-1}_{\chi^2_N}(p_{FA}) \implies \gamma = \sigma_n^2 Q^{-1}_{\chi^2_N}(p_{FA}) \tag{2.15}$$

This adaptive threshold technique is termed *Constant False Alarm Rate* (CFAR) because the threshold varies in such a way as to achieve a constant probability of false alarm.

Variance Detector

This energy detector follows a more practical approach. The formulation presented here follows that of [2] and is based on the same assumption that the energy of the data is greater when the signal is present.

According to [2], the procedure can be described in the following five steps

- Estimation of the signal energy for each consecutive non-overlapping block of samples.
- Windowing of the obtained energy sequence to include only its more recent elements.
- Normalisation of the windowed energy sequence.
- Determination of the variance of the resulting normalised sequence.
- Application of a threshold on the variance.

The estimation of energy is done with use of equation (2.8) for each incoming frame of audio samples. Next, the normalisation step, which plays a fundamental role in this algorithm is performed like

$$e_{norm} = \frac{e(j) - \min[e(j)]}{\max\{e(j) - \min[e(j)]\}}, \qquad j = 0, 1, \ldots L-1 \tag{2.16}$$

where j is the index of audio frames, L is the number of total frames used for the detection process, $\min[\cdot]$ and $\max[\cdot]$ declare the minimum and maximum value of all used energy values.

This normalisation step is the most important part of this detection scheme. When the input samples contain constant or almost constant noise, the values of e_{norm} are spread between 0 and 1 with a considerable standard deviation of roughly equal to 0.3 [2]. When a pulse occurs, the last energy sample $e(j)$ will reach 1 but the rest of the samples will cluster close to 0. This will lower the value of standard deviation by a considerable margin. *Figure 2.5* depicts this exact behavior for $L = 20$ and a threshold of 0.15. An important detail regarding this algorithm is that the last value (the newly added value corresponding to the energy of the "current" frame) is not used for the calculation of the standard deviation of the normalised energy vector.

The probabilities of false alarm and miss detection of this algorithm are shown in *Figure 2.6* for various values of the standard deviation threshold. As is expected, the false alarm probability does not depend on the SNR. On the contrary, the probability of missed detection does depend on the strength of the audio event compared to that of the noise. Nevertheless, it

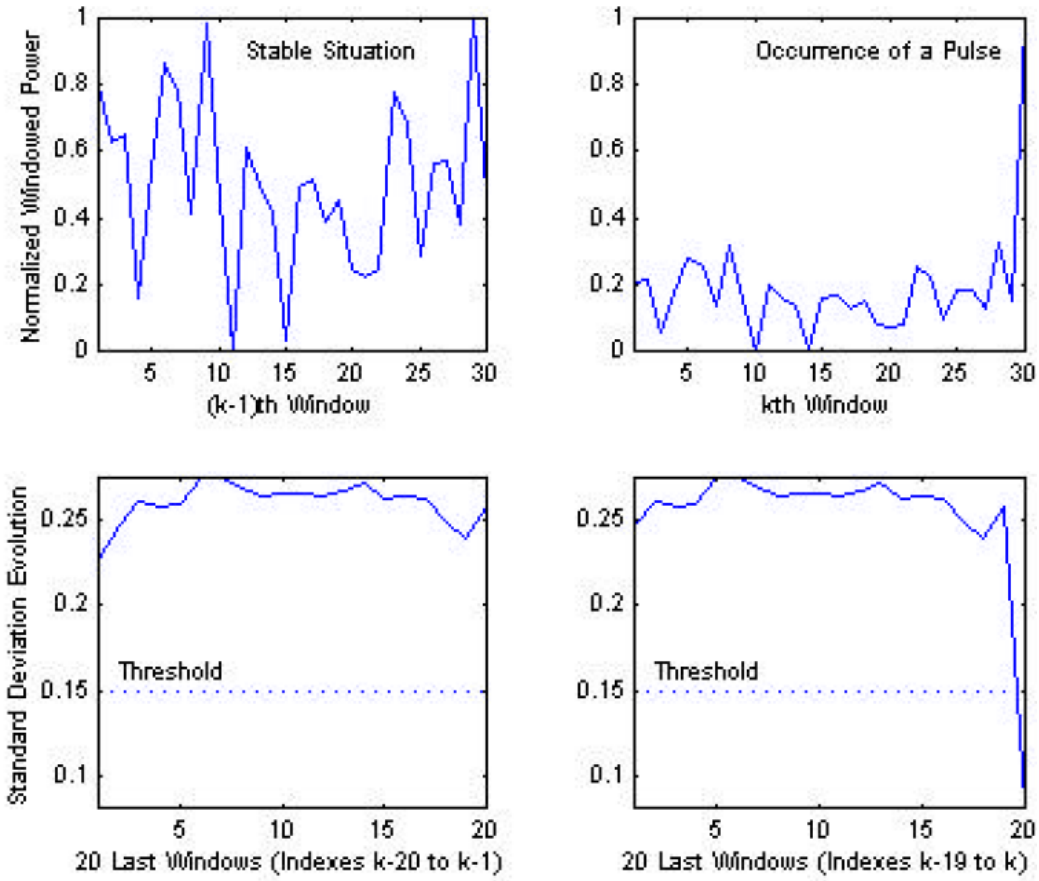


Figure 2.5: Standard deviation thresholding of the energy of input samples. Taken from [2].

seems that the detector can jointly achieve quite low probabilities for both false alarm and missed detection.

2.3 Direction-of-Arrival Estimation

In order to estimate the angle of an acoustic event, relative to a position in space, an acoustic sensor array can be used. In this work, the simplest case of a two-microphone array is set up.

Two broad categories of algorithms which utilise the structure of the array to extract angular information are investigated. Both use the TDoA principle, either directly or indirectly. In the first class of algorithms the Cross Correlation (CC) function between each sensor's recorded signal is calculated and from that a delay estimate is extracted. The second family

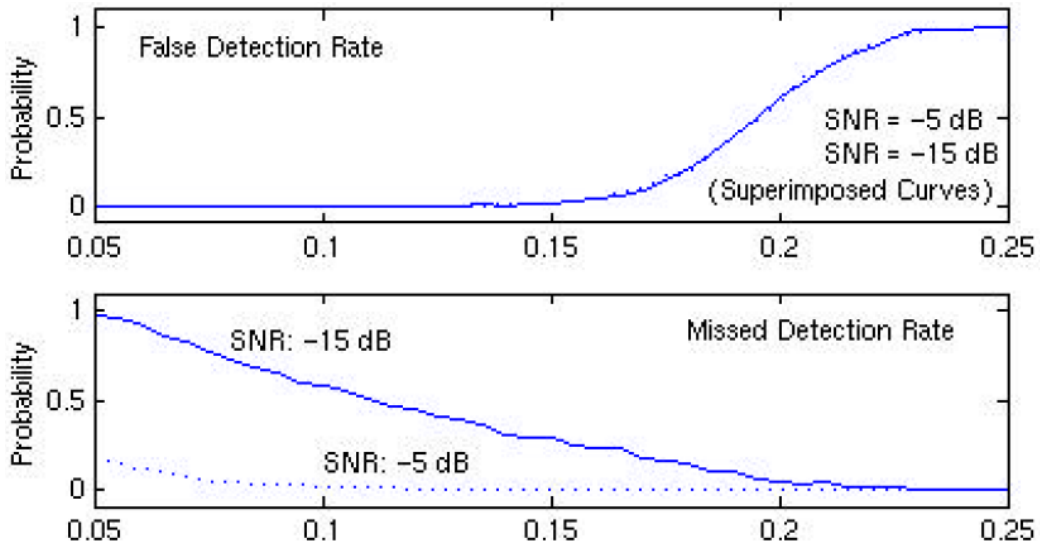


Figure 2.6: Probability of false alarm and missed detection of the variance detector for various standard deviation thresholds. Taken from [2].

of algorithms falls under the broad category of maximum likelihood estimation. These algorithms use the information related to the structure of the array in order to estimate the DoA by finding the one, out of a predefined set, that provides the best match with the data.

2.3.1 Uniform Linear Arrays

The microphone array used in this work falls under the category of Uniform Linear Arrays (ULA). The sensors are situated on a line and the distance between them is constant. This categorisation is trivial for a two element array since is the most natural way of arranging[1].

A setup with two sensors is shown in *Figure 2.7*. The inter-element distance is d, the angle of incidence is ϑ, the output of the two microphones is $y_1(k)$ and $y_2(k)$ respectively and the source $s(k)$ is considered to be located in the far-field. The latter is an assumption made in order to simplify the formulation of the algorithms. According to [54] this assumption holds adequately when

$$|r| > \frac{2L^2}{\lambda} \tag{2.17}$$

[1]The formulation could also follow that of a Uniform Circular Array (UCA) with identical results.

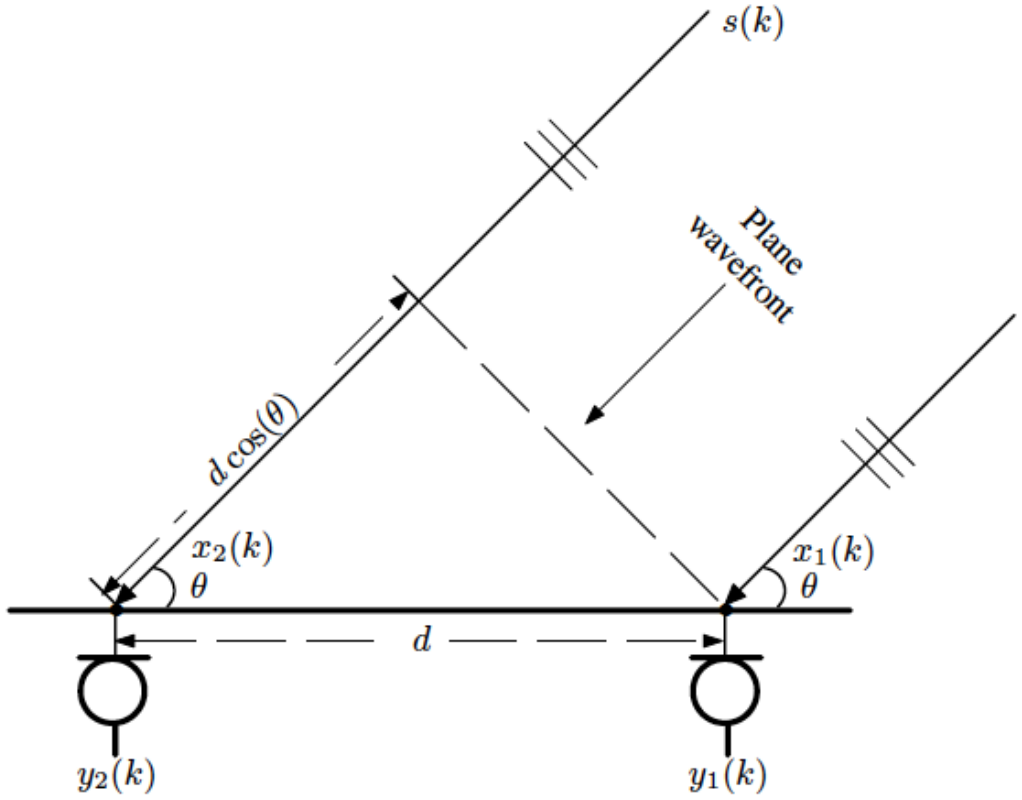


Figure 2.7: Illustration of a Uniform Linear Array. The microphones are assumed to be identical, $s(k)$ is the source which is located in the far-field, the angle of incidence is ∂ and the distance between the sensors is d. Taken from [3].

with r being the distance of the source from the array, L, the *effective length* of the array (see below) and $\jmath$ the wavelength corresponding to the frequency of interest.

Angle of incidence

In *Figure 2.7* is easy to see that the distance from the source to the two microphones is different, relating the signals $x_1(k)$ and $x_2(k)$ that denote the input to the elements, through a delay [3, 55]. Assuming plane wave propagation (far-field approximation) the distance of the wavefront the moment it impinges on the first sensor until it reaches the second microphone is given by

$$d_r = d\cos(\vartheta) \tag{2.18}$$

with d_r denoting the distance travelled by the plane wave. Furthermore, assuming isotropic medium, the distance between two points is expressed as

$$d_r = ct \tag{2.19}$$

where c is the speed of sound, considered to be equal to $343m/s$ throughout this work.

Solving equation (2.19) for t and using equation (2.18) to express d_r we get for the delay τ between the moments the wave reaches the two sensors

$$\tau = \frac{d\cos(\vartheta)}{c} \tag{2.20}$$

If the angle ϑ is constrained in the range $[0^o, 180^o]$ it can be uniquely determined as

$$\vartheta = \cos^{-1}\left(\frac{c\tau}{d}\right) \tag{2.21}$$

Thus, once the TDoA, is known, the angle of incidence can be determined [3, 55].

Angle Resolution

The angle resolution of the array is not constant throughout the range of angles. In order to see how the resolution is affected by the angle of incidence we solve equation (2.20) for $\cos(\vartheta)$ and differentiate both sides to get

$$\begin{gathered} d\left[\cos(\vartheta)\right] = d\left(\frac{c\tau}{d_m}\right) \implies -\sin(\vartheta)\, d\vartheta = \frac{c\, d\tau}{d_m} \implies \\ \implies d\vartheta = -\frac{c\, d\tau}{d_m \sin(\vartheta)} \implies |d\vartheta| = \frac{c\, d\tau}{d_m |sin(\vartheta)|} \end{gathered} \tag{2.22}$$

where d_m denotes the inter-element distance for clarity. We see from expression (2.22) that the absolute angle resolution becomes coarser as the angle of incidence ϑ goes from broadside (90^o) towards the end-fire (0^o or 180^o). The relation is not linear and shows a dependence on the sine of the angle.

These results are also verified from the calculation of the beam-width of an array. Following the derivation in [4], one can calculate the Half Power Beam Width (HPBW) for a uniform linear array as

$$\partial_H = \cos^{-1}\left(\cos(\partial) - 0.450\frac{\lambda}{Nd}\right) + \cos^{-1}\left(\cos(\partial) + 0.450\frac{\lambda}{Nd}\right) \tag{2.23}$$

for $0^o \leq \partial \leq 180^o$. ∂_H is the HPBW corresponding to $|d\partial|$ of equation (2.22), λ the wavelength, N the number of elements in the array and d the inter-element distance. For some ∂, one of the two terms of the right hand side of equation (2.23) will become 0. This point is referred to as the *scan limit* [4].

As can be seen in equation (2.23), the quantity in the denominator inside the parentheses is the number of elements in the array multiplied by the inter-element distance. This is referred to as the *effective length* of the array and is not the same as its *physical length*, which is $(N-1)d$. The *effective length* of a discrete sensor array is equal to the length of the continuous aperture that it samples and is a quantity of general interest in the field of array processing [4, 55].

The broadening of the beam width with increasing angle resembles a reduction of the array's *effective length* [4]. *Figure 2.8* shows an illustration of this analogy for an arbitrary angle of incidence.

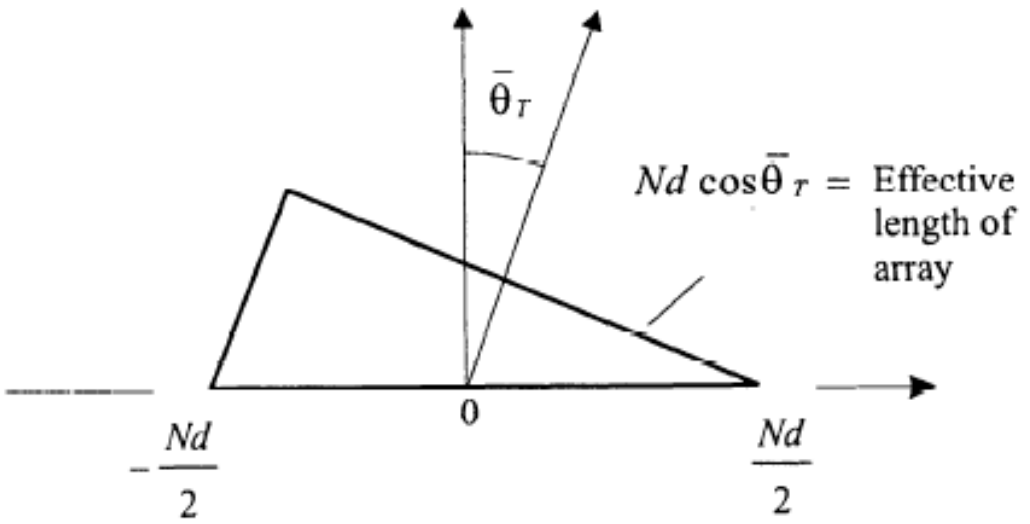


Figure 2.8: Illustration of the reduction of the *effective length* of an array for random angle of incidence, other than 90^o. Taken from [4].

Another important observation to be made is the dependence of the angle resolution on the time differential. For analogue systems this is

most probably limited by the step response of the electronics involved. For a digital system, like the one implemented in this work, the limiting value is the sampling period T_s. From equation (2.22) results that the relation of the angle resolution is proportional to T_s, meaning that an increase of the sampling frequency $f_s = \frac{1}{T_s}$ of the system will lead to better angle resolution.

Spatial aliasing

A discrete sensor array samples the sound field in the same manner an analogue signal is sampled to be digitised, but in the spatial dimensions. In direct analogy to the temporal sampling theorem, spatial aliasing can also occur when the following condition is violated [55]

$$f_s = \frac{1}{d} \geq 2f_{\max} \tag{2.24}$$

where in this case f_s is the spatial sampling frequency in units of $\left[\frac{samples}{m}\right]$ and $f_{\max}$ is the highest spatial frequency component of the signal for which

$$f_{\max} = \frac{1}{\lambda_{\min}} \tag{2.25}$$

is true. Inserting expression (2.25) into expression (2.24) we get the *spatial sampling theorem*

$$d < \frac{\lambda_{\min}}{2} \tag{2.26}$$

The highest frequency $f_{\max}$ allowed without spatial aliasing occurring is also given by [3]

$$f_{\max} = \frac{c}{2d} \tag{2.27}$$

2.3.2 Cross correlation methods

The family of the methods estimating the delay via the cross correlation function of the two microphone signals are found in the literature as the Generalised Cross Correlation (GCC) methods.

The cross correlation of two deterministic signals $y_1(t)$ and $y_2(t)$ is defined as [56]

$$r_{y_1 y_2}[\tau] = \int_{-\infty}^{\infty} y_1(t)\, y_2(n-\tau)\, dt, \qquad -\infty < \tau < \infty \tag{2.28}$$

with τ denoting a time shift (lag) by which the two functions are being offset. Similarly, for two functions being realisations of zero mean stochastic processes, the cross correlation function is given by [3, 19]

$$r_{y_1 y_2}(\tau) = E[y_1(\tau)\, y_2(t-\tau)] \tag{2.29}$$

and $E[\,\cdot\,]$ denotes the expectation operator.

The cross correlation function can be calculated using the linearity properties of the Fourier transform and the fact that cross correlation in the time domain is equivalent to multiplication in the frequency domain [19, 56]. For deterministic signals it is given by

$$r_{y_1 y_2}(\tau) = \mathcal{F}^{-1}\left\{Y_1(f)\,\overline{Y_2}(f)\right\} \tag{2.30}$$

with $\mathcal{F}^{-1}$ denoting the inverse Fourier transform, Y_1 and Y_2 the Fourier transform of the signals y_1 and y_2 respectively and $\overline{[\,\cdot\,]}$ complex conjugation. Similarly, for signals resulting from stochastic processes, one gets

$$r_{y_1 y_2}(\tau) = \mathcal{F}^{-1}\left\{E\left[Y_1(f)\,\overline{Y_2}(f)\right]\right\} \tag{2.31}$$

In both equation (2.30) and (2.31), the quantity $Y_1(f)\,\overline{Y_2(f)}$ is the Fourier transform of the cross correlation and is termed *cross spectrum*. *Figure 2.9* shows the cross spectrum and resulting cross correlation function for a random vector drawn from zero mean Gaussian distribution with variance $\sigma = 1$ and its replica delayed by 8 samples.

The delay between the two signals is estimated from the cross correlation function as the argument τ that maximises it, given by [3, 5, 57]

$$\hat{\tau} = \arg\max_{\tau} r_{y_1 y_2}(\tau) \tag{2.32}$$

where $\hat{\cdot}$ means that the value constitutes an estimate of the underlying quantity.

Generalised cross correlation

The cross correlation function is given by equation (2.28) for deterministic signals and (2.29) for realisations of stochastic processes. Explicitly defining the weighted cross spectrum, the generalisation of those equations is given by [3, 5, 55, 57]

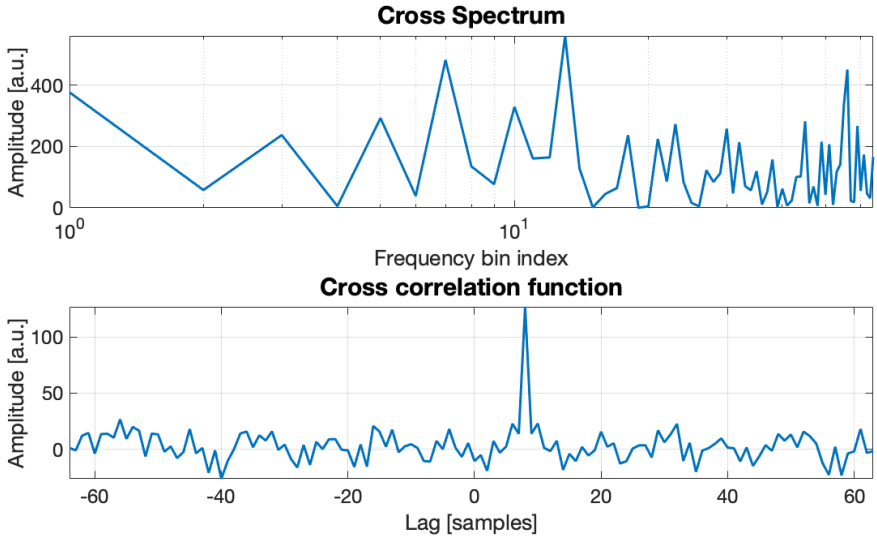


Figure 2.9: Cross spectrum and the resulting cross correlation function for a random vector drawn from a Gaussian distribution and its delayed by 8 samples replica.

$$\begin{aligned} r_{y_1 y_2}(\tau) &= \mathcal{F}^{-1}\left\{\int_{-\infty}^{\infty} \psi(f)\, Y_1(f)\, \overline{Y_2}(f)\, df\right\} \implies \\ \implies r_{y_1 y_2}(\tau) &= \int_{-\infty}^{\infty} \psi(f)\, Y_1(f)\, \overline{Y_2}(f)\, e^{j2\pi f t} df \end{aligned} \tag{2.33}$$

with $\psi(f)$ denoting the spectrum weighting function and j being the imaginary unit. For functions resulting from stochastic processes the expectation over the spectra has to be taken leading to [3, 19]

$$\begin{aligned} r_{y_1 y_2}(\tau) &= \mathcal{F}^{-1}\left\{\int_{-\infty}^{\infty} \psi(f)\, E\left[Y_1(f)\, \overline{Y_2}(f)\right] df\right\} \implies \\ \implies r_{y_1 y_2}(\tau) &= \int_{-\infty}^{\infty} \psi(f)\, E\left[Y_1(f)\, \overline{Y_2}(f)\right] e^{j2\pi f t} df \end{aligned} \tag{2.34}$$

Various choices of the weighting function $\psi(f)$ found in the literature lead to different GCC methods [3, 5]. Some of the most well known weighting functions are summarised in *Table 2.1*. One note to make is that whenever the function results from a stochastic process expectations have to be taken.

Table 2.1: Frequency weighting functions leading to different GCC algorithms. Y_1 and Y_2 are the signal functions and γ denotes the coherence function. Taken from [5].

Algorithm	**Weighting function** $\psi(f)$
CC	1
Roth	$\frac{1}{\lvert Y_1(f)\rvert^2}$
SCoT	$\frac{1}{\sqrt{\lvert Y_1(f)\rvert^2 \lvert Y_2(f)\rvert^2}}$
PhaT	$\frac{1}{\lvert Y_1(f)\overline{Y_2}(f)\rvert}$
Eckart	$\left\lvert Y_1(f)\,\overline{Y_2}(f)\right\rvert \cdot \left\{\left[\lvert Y_1(f)\rvert^2 - \left\lvert Y_1(f)\,\overline{Y_2}(f)\right\rvert\right] \cdot \left[\lvert Y_2(f)\rvert^2 - \left\lvert Y_1(f)\,\overline{Y_2}(f)\right\rvert\right]\right\}$
ML_{GCC}	$\frac{\lvert\gamma(f)\rvert^2}{\lvert Y_1(f)\overline{Y_2}(f)\rvert\left[1-\lvert\gamma(f)\rvert^2\right]}$

Something that requires a little more attention is the *Coherence Function* used for the calculation of the ML_{GCC} weighting. It is given by [58]

$$\gamma_{y_1 y_2}(f) = \frac{Y_1(f)\, Y_2(f)}{\sqrt{\lvert Y_1(f)\rvert^2 \lvert Y_2(f)\rvert^2}} \tag{2.35}$$

where $\lvert\cdot\rvert$ denotes the magnitude of the quantity of interest. It is worth noting that in the calculation of the ML_{GCC} weighting, the square of the quantity given by equation (2.35) is used. This is found in the literature as the *Squared Coherence Function* [58].

2.3.3 Maximum likelihood methods

Maximum Likelihood is a well known and widely used approach to many engineering problems. There are various formulations for the estimation of the angle of incidence found in the literature. In this work only the so called *Conditional Maximum Likelihood* and *Unconditional Maximum Likelihood* methods are discussed.

Steering vector

Following the derivation of [55], using the illustration of *Figure 2.7* as a starting point and expressing the sound field of a monochromatic plane wave at a position in space $\mathbf{x} = [x, y, z]^T$, where $[\,\cdot\,]^T$ denotes transposition, we can write for the position of a sensor [4, 55]

$$x_m(t) = s(t)\, e^{j(\omega t - \mathbf{k}\cdot\mathbf{x}_m)} + \mathbf{n}_m, \qquad m = 1, 2, \ldots, N \tag{2.36}$$

where $x_m(t)$ is the signal at the microphone m, ω the angular frequency for which $\omega = 2\pi f$ is true, $\mathbf{k}$ is the wavenumber in three dimensions and $\mathbf{n}_m$ is the noise at each microphone, which represents both spatially white noise and the noise of the electronics. The noise is considered to be uncorrelated for all sensors. The $\cdot$ symbol denotes the inner product.

For the wavenumber we know that [4, 55]

$$\mathbf{k} = -k\mathbf{u} = -\frac{\omega}{c}\mathbf{u} \tag{2.37}$$

with $\mathbf{u}$ a unit vector pointing from the array reference position to the source. Substituting equation (2.37) into equation (2.36) and separating the exponent into temporal and spatial parts we get

$$x_m(t) = s(t)\, e^{j\omega \frac{\mathbf{u}\mathbf{x}_m}{c}} e^{j\omega t} + \mathbf{n}_m, \qquad m = 1, 2, \ldots, N \tag{2.38}$$

Writing equation (2.38) into its vector form we end up with

$$\mathbf{x}(t) = \begin{bmatrix} e^{j\omega \frac{\mathbf{u}\cdot\mathbf{x}_1}{c}} \\ e^{j\omega \frac{\mathbf{u}\cdot\mathbf{x}_2}{c}} \\ \vdots \\ e^{j\omega \frac{\mathbf{u}\cdot\mathbf{x}_N}{c}} \end{bmatrix} s(t)\, e^{j\omega t} + \begin{bmatrix} \mathbf{n}_1(t) \\ \mathbf{n}_2(t) \\ \vdots \\ \mathbf{n}_N(t) \end{bmatrix} = \mathbf{a}(\mathbf{u})\, s(t)\, e^{j\omega t} + \mathbf{n}(t) \tag{2.39}$$

The vector $\mathbf{a}(\mathbf{u}) = \left[e^{j\omega \frac{\mathbf{u}\cdot\mathbf{x}_1}{c}}, e^{j\omega \frac{\mathbf{u}\cdot\mathbf{x}_2}{c}}, \ldots, e^{j\omega \frac{\mathbf{u}\cdot\mathbf{x}_N}{c}}\right]^T$ is called the *array manifold* or the *steering vector* and contains only spatial information.

ULAs have zero resolution on the vertical direction (perpendicular to their axis). Thus, the unit vector $\mathbf{u}$ can be expressed in polar coordinates for two dimensional space as [55]

$$\mathbf{u} = \begin{bmatrix} \cos(\partial) \\ \sin(\partial) \end{bmatrix} \tag{2.40}$$

Using one of the microphones' position as reference we can express the position of the rest (assuming they are located on the x axis) as

$$x_m = \begin{bmatrix} (m-1)\, d \\ 0 \end{bmatrix}, \qquad m = 1, 2, \ldots, N \tag{2.41}$$

Using equation (2.41) to express the inner product in the exponents of the steering vector we get [4, 55]

$$\begin{aligned} \mathbf{a}(\omega, \partial) &= \left[1, e^{j\omega \frac{d\cos(\partial)}{c}}, \ldots, e^{j\omega \frac{(N-1)d\cos(\partial)}{c}}\right]^T \implies \\ \implies \mathbf{a}(\omega, \partial) &= \left[1, e^{jkd\cos(\partial)}, \ldots, e^{jk(N-1)d\cos(\partial)}\right]^T \end{aligned} \tag{2.42}$$

In general, for ULAs the steering vector has Vandermonde structure and it also depends on the array reference position [4, 55]. The array manifold plays an important role in the ML techniques.

Narrowband maximum likelihood

Maximum likelihood is a statistical estimation algorithm. The signals are treated from a statistical perspective being described by their underlying PDFs. The most convenient choice is often the Gaussian distribution for its well defined and easily manipulated characteristics.

The narrowband model of the maximum likelihood formulation for the direction of arrival estimation is given by [4, 47–50, 59]

$$p_{\mathbf{x}|\partial}(\mathbf{x}) = \frac{1}{det[\pi \mathbf{R}_{\mathbf{x}}]} e^{-\left(\mathbf{x}^H - \mu_x^H\right)\mathbf{R}_{\mathbf{x}}{}^{-1}(\mathbf{x}-\mu_x)} \tag{2.43}$$

where $\mathbf{x}$ is an $Nx1$ complex Gaussian random variable, ∂ the angle(s) vector to be estimated, $\mathbf{R}_{\mathbf{x}}$ is the covariance matrix of the random vector $\mathbf{x}$, $\mu_{\mathbf{x}}$ the vector containing the means of the random variable, $[\,\cdot\,]^{-1}$ denotes inversion of a matrix, $[\,\cdot\,]^H$ Hermitian conjugation and $det[\,\cdot\,]$ the determinant. In this work, only one direction is sought for any given time so ∂ is a scalar. The covariance matrix is given by [4, 19]

$$\mathbf{R}_{\mathbf{x}} = E\left[\mathbf{x}\mathbf{x}^H\right] \tag{2.44}$$

Equation (2.43) gives the PDF of the data of a single snapshot for an array with N elements. Assuming that successive snapshots are statistically independent, their joint PDF is the product of the respective PDFs as given by equation (2.43). For K snapshots this is [4, 48]

$$p_{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_K|\partial} = \prod_{k=1}^{K} \frac{1}{det[\pi \mathbf{R}_{\mathbf{x}}]} e^{-\left(\mathbf{x}_k^H - \mu_x^H\right)\mathbf{R}_{\mathbf{x}}{}^{-1}(\mathbf{x}_k-\mu_x)} \tag{2.45}$$

The ML estimate of the DoA is the angle which maximises the PDF of equation (2.45). In order to find the extremum of the PDF is convenient to calculate its logarithm. Since the logarithm is a monotonic function, the maximum will be located at the same argument (angle). The logarithm of the PDF is termed log-likelihood function and is given by [4, 48, 59]

$$l(\partial) = \ln p_{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_K|\partial} = -\left[\ln det[\mathbf{R}_{\mathbf{x}}] + \frac{1}{K}\sum_{k=1}^{K} \mathbf{x}_k^H \mathbf{R}^{-1} \mathbf{x}_k\right] \tag{2.46}$$

where terms that do not depend on ∂ have been dropped. An equivalent form of the above expression is [4, 48–50]

$$l(\partial) = -\left\{\ln det\left[\mathbf{R_x}\right] + tr\left[\mathbf{R_x}^{-1}\mathbf{C_x}\right]\right\} \tag{2.47}$$

with $tr[\,\cdot\,]$ denoting the trace of a matrix and $\mathbf{C_x}$ the sample correlation matrix of the data given by [4, 19]

$$\mathbf{C_x} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{x}_k\mathbf{x}_k^H \tag{2.48}$$

It is instructive to say that for zero-mean stochastic processes the correlation and covariance matrices are equivalent [4, 60].

For the maximisation of the log-likelihood function, one has to set the gradient to zero and solve for the argument values. Using equation (2.39) to expand the covariance matrix in equation (2.47) we end up with the maximisation problem [4, 48, 49, 59]

$$f(\partial)_{CML} = \arg\max_{\partial}\{tr\left[\mathbf{P_A}\mathbf{C_x}\right]\} \tag{2.49}$$

where $\mathbf{P_A}$ is the projection matrix onto the range of a matrix A being composed of the steering vectors corresponding to the angles being searched and is given by [48]

$$A = \left[\mathbf{a}_{\mathbf{u}_1}, \mathbf{a}_{\mathbf{u}_2}, \ldots\right] \tag{2.50}$$

and $\mathbf{P_A}$ is given by [4, 48, 59]

$$\mathbf{P_A} = \mathbf{A}\left(\mathbf{A}^H\mathbf{A}\right)^{-1}\mathbf{A}^H \tag{2.51}$$

We see that the log-likelihood function depends on ∂ in a non-linear manner, making the peak finding process very computationally expensive. Many algorithms have been proposed that try to find the global maximum of the multivariate log-likelihood. The task is difficult since not only speed has to be achieved but the algorithm must be able to avoid being stuck in local maxima.

Miller et al [61] have proposed an *Expectation Maximisation* (EM) method for narrowband only signals, Stoica and Gershman [62] presented an algorithm that used the data to create a search grid which covers only a small part of the search range, thus achieving good speed but at the same time good results. A genetic algorithm is presented in [63] that seems to be very accurate.

Broadband maximum likelihood

When estimating the angle of incidence of broadband signals, one has to consider the dependence of the array manifold on the frequency of the source. Thus, the estimate must include all array manifolds corresponding to the frequencies contained in the signal. The formulation of equation (2.49) is valid for each frequency of interest and following the derivations in [4] and [47] we can state the broadband estimation problem as

$$f_{CML}(\partial) = \arg\max_{\partial} \left\{ \sum_{i=1}^{M} \ln\left(tr\left[\mathbf{P}_{\mathbf{A_i}}^{\perp} \mathbf{C}_{\mathbf{x_i}} \right] \right) \right\} \tag{2.52}$$

where $P_{A_i}^{\perp}$ denotes the projection matrix onto the *null space* of the steering vector for the ith frequency given by [4, 47, 48]

$$\mathbf{P_A}^{\perp} = \mathbf{I} - \mathbf{P_A} \tag{2.53}$$

where with $\mathbf{I}$ is denoted the identity matrix.

The combination of the information is done via the sum of the logarithms of the projected correlation matrices onto the corresponding steering vector ranges.

Unconditional maximum likelihood

The formulation presented so far assumes the estimated parameter to be deterministic. This formulation results in the so called *Conditional Maximum Likelihood* while if the parameter of interest (the angle of incidence in this case) is modeled as the realisation of a stochastic process with an underlying PDF, then the formulation is termed *Unconditional Maximum Likelihood*.

The maximisation problem in this case for a narrowband source is [4, 48]

$$f_{UML} = \arg\max_{\partial} \left\{ -\ln\det\left[\mathbf{P_A} \mathbf{C_x} \mathbf{P_A} + \frac{tr\left[\mathbf{P_A}^{\perp} \mathbf{C_x} \right] \mathbf{P_A}^{\perp}}{N - D} \right] \right\} \tag{2.54}$$

where D is the number of sources to be estimated and N the number of sensors.

Similarly, for a broadband source, the maximisation problem becomes [4, 47]

$$f_{UML} = \arg\max_{\partial} \left\{ -H_S - (N - D) \sum_{i=1}^{M} \ln tr\left[\mathbf{P_{A_i}}^{\perp} \mathbf{C_{x_i}} \right] \right\} \tag{2.55}$$

where H_S is [4, 47]

$$H_S = \sum_{i=1}^{M} \ln \det \left[\mathbf{P}_{\mathbf{A_i}} \mathbf{C}_{\mathbf{x_i}} \mathbf{P}_{\mathbf{A_i}} + \mathbf{P}^{\perp}_{\mathbf{A_i}} \right] \tag{2.56}$$

For uncorrelated sources, the statistical performance of the two formulations (CML and UML) is similar but when the correlation of the sources increases UML provides significantly better results [50]. Another important difference between the two formulations is that UML is efficient and can achieve the minimum possible variance for $t \to \infty$ or $SNR \to \infty$ while the CML cannot unless $N \to \infty$.

Chapter 3

Evaluation

This chapter provides information on the apparatus used in this work, the experimental setup, implementation details and the results of the evaluations. The tests are divided into single parameter comparisons to ease the process of reaching meaningful conclusions. For each test case, relevant metrics are shown, along with more information when deemed instructive and comments to provide insight on the interpretation of the results.

3.1 Apparatus

3.1.1 Hardware

This work, from acquisition and processing to algorithm implementation and evaluation was performed on an Apple® 13-inch MacBook Pro Mid-2012 with Dual-Core Intel® Core i5 processor clocked at 2.5*GHz* and 12 *GB* of DDR3 RAM memory clocked at 1333*MHz*. The OS version was macOS® Catalina 10.15.7.

Sound signals were acquired with a pair of condenser lavaliere microphones, UH1 150 L by dB Technologies® (model is discontinued). A Focusrite® Scarlett 2i2 2nd Gen [64] was used as an external audio interface for the amplification and digitisation of the audio signals. The same interface provided the needed 48*V "phantom power"* required for the microphones to function properly.

A ruler and a protractor were used to measure distance and angle relative to the array's centre.

3.1.2 Software

REAPER® [65] was used for the acquisition of the data. The acquisition parameters affecting the audio performance were the sampling rate, which was set to 44.1*KHz* and the requested block size set to 8192 samples. The version of the software was v.6.45/OSX64-clang rev64818. In addition to REAPER®, Mathworks MATLAB® was used for the processing of the data to create the database used in the evaluation of the implemented algorithms. MATLAB's version is R2018a (9.4.0.813654), 64-bit (maci64).

3.2 Setup

The experimental setup is quite simplistic and is comprised of the two lavaliere microphones placed on top of a wooden table in the middle of a common, parallelepiped room of dimensions 3.0*m* *x* 3.0*m* *x* 2.8*m* [Length x Width x Height]. The laptop and the external audio interface were not placed on the same table.

The microphones were positioned at a very short distance from the table, in the order of ~ 3*mm*. This arrangement was realised in attempt to minimise the appearance of discrete reflections in the recorded signals. Although the signals are not "reflection-free", for this distance from the boundary a rough estimate of the delay of a reflection can be calculated with the help of *Figure 3.1* below.

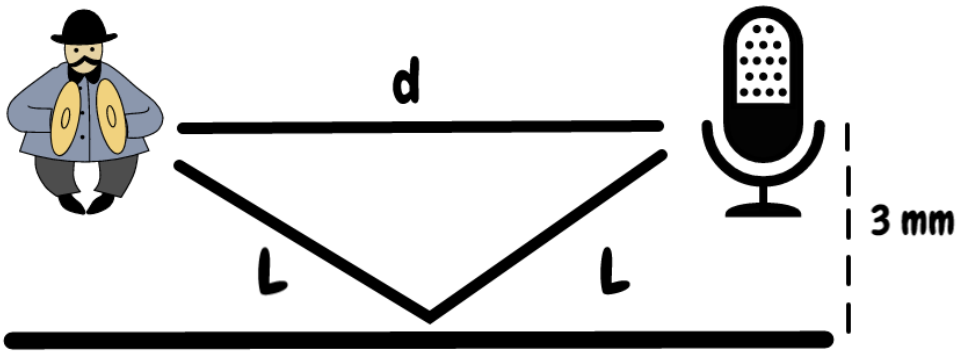


Figure 3.1: Approximate setup of source and receiver for the calculation of reflections. d is the direct path length between source and receiver and L half the path distance of the (specularly) reflected sound.

In the figure, s is the source, r the receiver, d their distance, which is the path of the direct sound too and L the half path of the reflected sound. One can calculate L from the right angled triangle formed by r, L and $\frac{d}{2}$. Using, $d = 0.2m$, which is the shortest value of distance of the source from the array used in this work we get

$$L = \sqrt{\left(\frac{d}{2}\right)^2 + (3 \cdot 10^{-3}m)^2} \approx 100.045 \cdot 10^{-3}m$$

and the full path of the reflection is

$$2L \approx 200.9 \cdot 10^{-3}m$$

The difference between the direct path and the path of the reflected sound is

$$2L - d \approx 0.9 \cdot 10^{-3}m$$

Assuming the speed of sound is $c = 343m/s$, we get

$$t = \frac{x}{c} \approx 2.62 \cdot 10^{-6}s$$

Using the sampling rate used in this work, $f_s = 44.1kHz$, we can get the delay in samples. This is

$$t_s = t \cdot f_s \approx 0.0116$$

This setup will experience the first reflection from the table, which is supposed to also be the strongest, after approximately 0.0116 samples. Using the same approach, one calculates the delay of the reflection in samples for the greatest distance of source from the array in this work, $d = 0.4m$, to get $t_s \approx 0.0058$ samples.

It is important to note though that the setup of *Figure 3.1* is a worst case scenario and it does not necessarily reflect the real setup. Thus, the numbers presented here serve only as a *worst-case* indications of the order of magnitude of the delays the reflected signals experience.

According to the results above, there will be no discrete reflection in the recorded signals from the table. However, this may very well influence the results of the estimated directions of incidence. The way the reflection will affect the result is not trivial to calculate. Most probably, it will act as a secondary source with high coherence to the direct signal introducing systematic errors in the estimated values. In the time domain, this could be seen as broadening of the signal's impulse, weakening the temporal localisation of the peak.

3.3 Reverberation time of room

To provide the ability to compare this work against that found in the available literature, the Reverberation Time (RT) of the room the experiments took place was calculated. The background noise was not measured due to lack of appropriate equipment.

For the reverberation time estimation, a NEXO PS10 [66] loudspeaker driven by a NEXO NXAMP4x4 [67] amplifier was used as source and an omnidirection condenser Beyerdynamic® MM-1 [68] microphone was used for the acquisition.

The measurements of the Room's Impulse Response (RIR) were performed with the method of logarithmically swept sine [69, 70]. The position of measurement was the centre of the microphone array.

Since the available apparatus didn't qualify for measurements of reverberation time in accordance with ISO 3382-2:2008 [71], an approach proposed by Papadakis and Stavroulakis [72] was followed. Multiple measurements were performed with the loudspeaker directed at angles corresponding to the placement of the drivers of a dodechahedron loudspeaker. All impulse responses were averaged to produce a mean response which subsequently was used to estimate the reverberation time, calculated with Schröeder's method [73].

Results of the room's reverberation time estimation are presented in *Table 3.1* for octaves with central frequencies from 125*Hz* to 8*KHz*.

Table 3.1: Reverberation time of the room where the experiments took place, measured at the centre of the array.

Octave [Hz]	**RT [s]**
125	0.76
250	0.88
500	0.62
1000	0.63
2000	0.55
4000	0.47
8000	0.60

Measured RT values do not qualify the room as "reverberant". Nevertheless, common domestic rooms are rarely qualified as such especially when compared to rooms and halls of larger dimensions.

3.4 Signal Detection

This section presents the results of the experiments run with the detection algorithms. It is divided into three subsections, one with the results of the experiments performed with simulated data, the second with real recordings and the last presents the execution times of the implemented algorithms.

In the first two parts, the algorithms are evaluated against diverse SNR conditions for various thresholds. In the simulated experiments the thresholds correspond to specific values fixed for each algorithm. In the experiments with real recordings, the threshold of each algorithm is set such that all algorithms present zero *False Positive Rate* (FPR) and *False Negative Rate* (FNR) and are compared under those constraints.

The metrics used to compare the algorithms had to convert the binary result of the task to a continuous value. They are *Accuracy*, *Precision*, *Sensitivity* (or *Recall*), *Specificity* and *F-Score*. These metrics provide an estimate of the performance of the binary test as a percentage. Their definitions are presented in *Appendix B*.

Algorithms

The algorithms implemented follow the derivations presented in *Section 2.2* but there are minor alterations which are described here. The changes concern the *Variance Detector* and the *CFAR* variation of the *Gaussian Detector*.

The *Variance Detector* is implemented exactly as described in [2] and presented in the methodology section of the current work with the sole exception being the update of the noise energy buffer. In the experiments run with simulated data two different update schemes are tested. One follows the original work presented in [2] where the noise energy vector is constantly updated. The other scheme updates the vector only when no acoustic event is detected, in order to avoid including frames containing high energy due to the impulsive event being present. In preliminary tests it was found that the detection of impulsive acoustic events present in consecutive frames is improved if the noise energy buffer is not updated when an acoustic event is detected. It is also shown in [2] that the constant update of the noise buffer degrades the detection of closely spaced impulsive acoustic events. The duration of decreased sensitivity lasts for the period of the noise buffer containing an impulsive event energy, which is the duration of the whole buffer in the best case. In this work, the scheme

that updates the noise vector only in the absence of an acoustic event[1] is termed "discard" (since it discards the energy when an acoustic event is present). The scheme that constantly updates the noise vector is termed "hold".

For the *CFAR Gaussian Detector*, the noise buffer is not updated in the case an acoustic event is detected. Since this detector uses only the noise of past frames to update the threshold value, it is more appropriate to use frames containing only noise for the adaptation process. Similarly to the *Variance Detector*, the inclusion of a signal frame to the noise vector is based on the decision of the algorithm and not on the true label of the frame.

For both aforementioned algorithms, the duration of noise used is roughly $1s$ (344 frames are used which correspond to $0.9985s$). The initial noise buffer is populated with the same algorithm used to create the noise added to the signals, which is *Additive White Gaussian Noise* (AWGN), and is updated with the value of the current frame if no impulse is detected.

3.4.1 Simulated experiments

The signal detection algorithms were initially trialed on a simulated experimental framework. A Monte Carlo simulation was performed where the three algorithms are evaluated for different SNR conditions with specific, fixed, thresholds. The SNR conditions used for the simulations are $0dB$, $10dB$ and $20dB$ and the noise added is AWGN.

The simulated impulses constitute white Gaussian noise with amplitude envelope generated as a *Beta* probability distribution function with values $\alpha = 2$ and $\beta = 5$. The amplitude of the impulses and the noise are adjusted to achieve the desired SNR conditions. All signals involved in the creation of an artificial signal are shown in *Figure 3.2* along with the final impulse embedded in noise for $SNR = 10dB$.

The probability of an impulse being present in a frame is 25% drawn from a uniform distribution. A total of 50000 frames were generated with 12551 of them containing an impulse. Each impulse was displaced so as to not be positioned in the same part of each frame. The exact same signal frames are used with all algorithms.

[1] This, of course, refers to the frames that the algorithm labels as positive and not the true positives, since those are unknown to the algorithm.

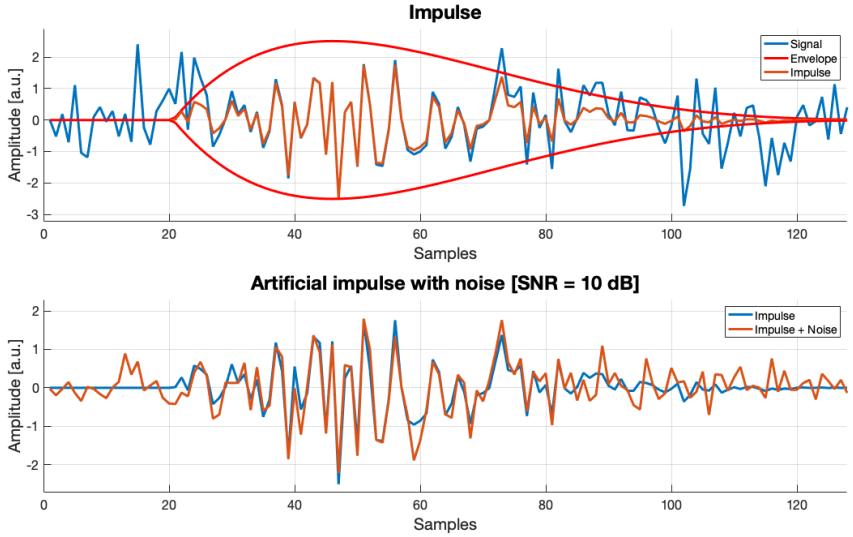


Figure 3.2: Creation of an artificial impulse for the evaluation of the detection algorithms. On the top are shown the underlying Gaussian noise signal, the envelope and the final impulse. On the bottom plot, both the "clean" impulse and the same signal embedded in noise are shown for $SNR = 10dB$.

Thresholds

Due to the way the detectors are formulated there is no direct way to compare the thresholds for all of them. The *Gaussian Detector* and its variation, *CFAR*, are formulated on the assumption of both signal and noise being random variables with Gaussian PDF, while the *Variance Detector* is formulated on a more practical approach to the detection problem. This led to the issue of having to use thresholds chosen on different criteria. The thresholds chosen for the *Gaussian Detector* and *CFAR* are calculated for specified probabilities of false alarm ranging from 10^{-1} to 10^{-4} with a halving step.

On the contrary, the thresholds of the *Variance Detector* are chosen based on the proposed values in the literature. The optimal value as given in [2] is 0.15, which depends on various factors, such as the duration of the noise window and the frame size. There is no way to calculate the optimal value so seven different values in the range $[0.1, 0.25]$ were tested.

Simulation results

Table 3.2 shows the results of the Monte Carlo simulations with each subtable corresponding to a different SNR condition.

There are some clear trends observed in the results. First and foremost, it seems that the two Gaussian detectors show similar detection capacity. This is of course to be expected since they are based on the same assumptions on the underlying distributions of the acoustic event and noise signals. Additionally, the fact that the simulated noise characteristics (variance is the most important factor here) are constant for all impulses, for each SNR condition, renders the *CFAR* formulation somewhat redundant. The metric values for this algorithm are consistently a little lower than the formulation with constant threshold. This may be attributed to the fact that *CFAR* uses about $1s$ of noise signal to adapt the threshold, which may be inadequate in order to achieve statistically good estimates of the noise characteristics, especially compared to the duration of the whole database (which is roughly $145s$).

As is also expected, both detectors show increased efficiency in all metrics for increasing SNR. Even for low SNR conditions though, they can achieve jointly quite high values of Precision and Specificity, showing that the FPR can be kept low, which is quite important for this work.

The scheme of the *Variance Detector* for which the noise vector is updated only in the absence of an acoustic event shows exceptionally good results for medium and high SNR conditions. It is the only algorithm that can achieve unity Precision for all levels of noise and can even achieve unity Accuracy for medium and high SNR. Despite the fact that for the case of $SNR = 0dB$ the best achieved Sensitivity of the algorithm is very low (0.0001), it is still the only algorithm that managed to keep the FPR equal to zero with Precision and Specificity jointly equal to one. For the purpose of this work, the minimisation of spontaneous misdetections is weighted heavier than the ability to detect all events.

On the contrary, when the noise vector is constantly updated, the *Variance Detector* shows consistently bad results with many false positive labels and low Sensitivity. This most probably is attributed to the fact that, on average, there are about 80 impulses in the duration of the noise vector[2] and most of them are missed (for details see *Section 3.4 - Algorithms*).

[2]Since there is 25% probability for an impulse to be present, there is, on average, one every four frames. The duration, in frames, of the noise vector is 344 which corresponds to ≈ 86 impulses for the whole duration.

Table 3.2: Results of simulated experiments for the evaluation of the *Signal Detection* algorithms. a) $SNR = 0dB$, b) $SNR = 10dB$, c) $SNR = 20dB$.

Gauss Threshold: $p_{FA} = 10^{-1}$ | **Variance Threshold**: **0.1**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9259	0.7721	1.0000	0.9011	0.8714
Variance [hold]	0.7490	—	0.0000	1.0000	—
Variance [discard]	0.7490	1.0000	0.0001	1.0000	0.7925
CFAR	0.8686	0.6563	1.0000	0.8245	0.4499
Gauss Threshold: $p_{FA} = 5 \cdot 10^{-2}$ | **Variance Threshold**: **0.12**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9634	0.8727	1.0000	0.9511	0.9321
Variance [hold]	0.7487	0.2903	0.0007	0.9994	0.0014
Variance [discard]	0.7495	1.0000	0.0022	1.0000	0.0043
CFAR	0.9260	0.7724	1.0000	0.9012	0.8716
Gauss Threshold: $p_{FA} = 10^{-2}$ | **Variance Threshold**: **0.15**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9930	0.9732	0.9998	0.9908	0.9863
Variance [hold]	0.7485	0.3077	0.0016	0.9988	0.0032
Variance [discard]	0.7515	0.5628	0.0453	0.9882	0.0839
CFAR	0.9818	0.9323	1.0000	0.9756	0.9649
Gauss Threshold: $p_{FA} = 5 \cdot 10^{-3}$ | **Variance Threshold**: **0.18**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9962	0.9855	0.9996	0.9951	0.9925
Variance [hold]	0.7479	0.2903	0.0029	0.9977	0.0057
Variance [discard]	0.2510	0.2510	1.0000	0.0000	0.4013
CFAR	0.9903	0.9629	0.9998	0.9871	0.9810
Gauss Threshold: $p_{FA} = 10^{-3}$ | **Variance Threshold**: **0.2**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9987	0.9967	0.9982	0.9989	0.9975
Variance [hold]	0.7449	0.2423	0.0076	0.9921	0.0147
Variance [discard]	0.2510	0.2510	1.0000	0.0000	0.4013
CFAR	0.9971	0.9893	0.9994	0.9964	0.9943
Gauss Threshold: $p_{FA} = 5 \cdot 10^{-4}$ | **Variance Threshold**: **0.22**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9986	0.9978	0.9967	0.9993	0.9972
Variance [hold]	0.7264	0.2455	0.0434	0.9553	0.0738
Variance [discard]	0.2510	0.2510	1.0000	0.0000	0.4013
CFAR	0.9982	0.9942	0.9988	0.9981	0.9965
Gauss Threshold: $p_{FA} = 10^{-4}$ | **Variance Threshold**: **0.25**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9974	0.9995	0.9903	0.9998	0.9949
Variance [hold]	0.4324	0.2452	0.6066	0.3741	0.3492
Variance [discard]	0.2510	0.2510	1.0000	0.0000	0.4013
CFAR	0.9986	0.9981	0.9962	0.9994	0.9971

(a) $SNR = 0dB$.

Like the other two detectors, the detection efficiency improves with increasing SNR. Furthermore, the sensitivity to the threshold value seems to be improving too. For low SNR the algorithm "gets stuck" to extreme values for a rather low threshold, while for high SNR this behavior is not observed for any of the tested values. These observations concern the "discard" scheme as the "hold" scheme shows results that do not change irrespective of SNR conditions and threshold values.

3.4.2 Experiments with recorded data

This section shows the evaluation results obtained with real recorded signals. The algorithms are tested under the constraints of the limiting thresholds to achieve zero FPR, which corresponds to unity Precision and zero FNR corresponding to Sensitivity value of one. The evaluation is performed for various SNR conditions. The recorded signals come from various sources and at the final part of this evaluation the algorithms are tested against each source type separately.

Acquisition of acoustic event recordings

Monophonic signals were recorded, corresponding to impulsive acoustic events created with hand claps, finger snaps, drumsticks and cutlery. The total number of frames in the database that was created is 741271 from which 63538 include an impulsive acoustic event. The ratio of frames with an acoustic event over those containing only noise is about 0.094 representing 8.57% of the total frames.

The peak amplitude range of the acoustic events is about $22dB$ with the maximum being $0dB_{FS}$[3]. The corresponding RMS values for those signals are $-5dB_{FS}$ for the maximum and $-30dB_{FS}$ for the minimum. The values of the energies (sum of squared sample amplitudes) in dimensionless (arbitrary) units are $1.4881 \cdot 10^{-4}$ and 10.3995 respectively, resulting in a ratio of approximately $6.9886 \cdot 10^{4}$.

Similar to the simulated data experiments, artificial AWGN is added in order to evaluate the algorithms. Four cases are distinguished here, with the first one being the original recordings containing only the noise of the recording equipment and the environment. The other three cases have artificial AWGN added with amplitude RMS values roughly equal to $-10dB_{FS}$, $-20dB_{FS}$ and $-30dB_{FS}$.

[3]The abbreviation *FS* stands for *Full Scale* and is a declaration of (logarithmic) distance from the maximum value which is $0dB_{FS}$

Gauss Threshold: $p_{FA} = 10^{-1}$ | **Variance Threshold**: **0.1**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9250	0.7699	1.0000	0.8998	0.8700
Variance [hold]	0.7489	0.2857	0.0003	0.9997	0.0006
Variance [discard]	1.0000	1.0000	1.0000	1.0000	1.0000
CFAR	0.9006	0.7163	1.0000	0.8673	0.8347
Gauss Threshold: $p_{FA} = 5 \cdot 10^{-2}$ | **Variance Threshold**: **0.12**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9622	0.8691	1.0000	0.9495	0.9300
Variance [hold]	0.7489	0.3750	0.0005	0.9997	0.0010
Variance [discard]	1.0000	1.0000	1.0000	1.0000	1.0000
CFAR	0.9478	0.8277	1.0000	0.9303	0.9058
Gauss Threshold: $p_{FA} = 10^{-2}$ | **Variance Threshold**: **0.15**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9927	0.9716	1.0000	0.9902	0.9863
Variance [hold]	0.7489	0.3913	0.0007	0.9996	0.0014
Variance [discard]	0.9953	0.9816	1.0000	0.9937	0.9280
CFAR	0.9884	0.9558	1.0000	0.9845	0.9840
Gauss Threshold: $p_{FA} = 5 \cdot 10^{-3}$ | **Variance Threshold**: **0.18**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9962	0.9852	1.0000	0.9950	0.9925
Variance [hold]	0.7487	0.3182	0.0011	0.9992	0.0022
Variance [discard]	0.9128	0.7422	1.0000	0.8836	0.8520
CFAR	0.9957	0.9831	1.0000	0.9942	0.9915
Gauss Threshold: $p_{FA} = 10^{-3}$ | **Variance Threshold**: **0.2**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9994	0.9977	1.0000	0.9992	0.9988
Variance [hold]	0.7485	0.3000	0.0014	0.9989	0.0029
Variance [discard]	0.2510	0.2510	1.0000	0.0000	0.4013
CFAR	0.9992	0.9970	1.0000	0.9990	0.9985
Gauss Threshold: $p_{FA} = 5 \cdot 10^{-4}$ | **Variance Threshold**: **0.22**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9996	0.9986	1.0000	0.9995	0.9993
Variance [hold]	0.7484	0.2933	0.0018	0.9986	0.0035
Variance [discard]	0.2510	0.2510	1.0000	0.0000	0.4013
CFAR	0.9996	0.9985	1.0000	0.9995	0.9992
Gauss Threshold: $p_{FA} = 10^{-4}$ | **Variance Threshold**: **0.25**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9999	0.9996	1.0000	0.9999	0.9998
Variance [hold]	0.7481	0.2830	0.0024	0.9980	0.0047
Variance [discard]	0.2510	0.2510	1.0000	0.0000	0.4013
CFAR	0.9999	0.9996	1.0000	0.9999	0.9998

(b) $SNR = 10dB$.

Gauss Threshold: $p_{FA} = 10^{-1}$ | **Variance Threshold**: **0.1**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9245	0.7688	1.0000	0.8992	0.8693
Variance [hold]	0.7489	0.3333	0.0002	0.9998	0.0005
Variance [discard]	1.0000	1.0000	1.0000	1.0000	1.0000
CFAR	0.9196	0.7574	1.0000	0.8926	0.8619
Gauss Threshold: $p_{FA} = 5 \cdot 10^{-2}$ | **Variance Threshold**: **0.12**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9626	0.8703	1.0000	0.9500	0.9306
Variance [hold]	0.7489	0.3333	0.0004	0.9997	0.0008
Variance [discard]	1.0000	1.0000	1.0000	1.0000	1.0000
CFAR	0.9595	0.8610	1.0000	0.9459	0.9253
Gauss Threshold: $p_{FA} = 10^{-2}$ | **Variance Threshold**: **0.15**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9930	0.9730	1.0000	0.9907	0.9863
Variance [hold]	0.7489	0.3913	0.0007	0.9996	0.0014
Variance [discard]	0.9610	0.8656	1.0000	0.9480	0.0038
CFAR	0.9919	0.9686	1.0000	0.9891	0.9850
Gauss Threshold: $p_{FA} = 5 \cdot 10^{-3}$ | **Variance Threshold**: **0.18**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9962	0.9855	0.9996	0.9951	0.9925
Variance [hold]	0.7487	0.3171	0.0010	0.9993	0.0021
Variance [discard]	0.7479	0.3060	0.0033	0.9975	0.0065
CFAR	0.9958	0.9839	0.9997	0.9945	0.9917
Gauss Threshold: $p_{FA} = 10^{-3}$ | **Variance Threshold**: **0.2**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9987	0.9967	0.9982	0.9989	0.9975
Variance [hold]	0.7485	0.2857	0.0013	0.9989	0.0025
Variance [discard]	0.7446	0.2380	0.0079	0.9915	0.0153
CFAR	0.9986	0.9963	0.9982	0.9988	0.9973
Gauss Threshold: $p_{FA} = 5 \cdot 10^{-4}$ | **Variance Threshold**: **0.22**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9986	0.9978	0.9967	0.9993	0.9972
Variance [hold]	0.7484	0.2985	0.0016	0.9987	0.0032
Variance [discard]	0.7265	0.2457	0.0433	0.9554	0.0737
CFAR	0.9986	0.9975	0.9971	0.9992	0.9973
Gauss Threshold: $p_{FA} = 10^{-4}$ | **Variance Threshold**: **0.25**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9974	0.9995	0.9903	0.9998	0.9949
Variance [hold]	0.7482	0.2796	0.0021	0.9982	0.0041
Variance [discard]	0.4324	0.2452	0.6066	0.3741	0.3492
CFAR	0.9976	0.9994	0.9910	0.9998	0.9952

(c) $SNR = 20dB$.

Algorithms

At this stage, four algorithms are tested. From the two schemes of the *Variance Detector*, only that for which the noise energy vector is updated in the absence of an acoustic event is used.

In addition, an empirical double-threshold algorithm is evaluated. In this algorithm, the RMS value of the recorded signal is used instead of its energy. The two threshold conditions are given by

$$x_{RMS} > \gamma_{RMS} \tag{3.1}$$

$$\frac{x_{RMS}}{x_{smooth}} > \gamma_{ratio} \tag{3.2}$$

where x_{RMS} is the RMS value of the current signal frame and x_{smooth} is given by

$$x_{smooth}[k] = 0.8 \cdot x_{smooth}[k-1] + 0.2 \cdot x_{RMS}[k] \tag{3.3}$$

with k being the frame index. The x_{smooth} represents a weighted moving average filtering process that smooths the RMS values of the input frames, also termed *leaky integrator*. Such a filter has been used in similar detection schemes [17]. In the experiments the first "true negative" frame is used to initialise the value of x_{smooth}.

The noise vector duration for the *Variance Detector* in these experiments is 30 frames long and for the *CFAR Detector* 344, which correspond to $\approx 87ms$ and $\approx 998ms$ respectively.

Maximum Precision

In this experiment, the threshold of the algorithms is chosen such that they achieve unity Precision. The chosen threshold corresponds to the limiting value up to four significant digits. For the *Empirical* detector, first γ_{RMS} is set to achieve the best possible Precision and next γ_{ratio} is tweaked to attain zero FPR.

Attaining maximum Precision is equivalent to ensuring zero false positive labels. Under this constraint the most important metric to watch out for is Sensitivity, which declares the number of identified acoustic events out of the total number of true positive ones. Equivalently, one could aim for high F-Score values, since this is the (harmonic) mean of the two metrics (Precision and Sensitivity) and will be high when both values are high simultaneously. The results of this experiment are shown in *Table 3.3* for all four SNRs.

Table 3.3: Evaluation metrics of the detection algorithms with real recorded signals. The values correspond to thresholds chosen to achieve maximum (unity) Precision.

Noise: $-\infty$					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9711	1.0000	0.6629	1.0000	0.7973
Variance	0.9387	1.0000	0.2848	1.0000	0.4433
CFAR	0.9542	0.6517	0.9994	0.9499	0.7890
Empirical	0.9711	1.0000	0.6630	1.0000	0.7974
Noise: -30 **dB$_{FS}$**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9729	1.0000	0.6843	1.0000	0.8125
Variance	0.9455	1.0000	0.3641	1.0000	0.5338
CFAR	0.9843	0.8521	0.9878	0.9839	0.9149
Empirical	0.9243	1.0000	0.1168	1.0000	0.2092
Noise: -20 **dB$_{FS}$**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9302	1.0000	0.1862	1.0000	0.3139
Variance	0.9275	1.0000	0.1536	1.0000	0.2663
CFAR	0.9300	1.0000	0.1834	1.0000	0.3100
Empirical	0.9303	1.0000	0.1863	1.0000	0.3141
Noise: -10 **dB$_{FS}$**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9151	1.0000	0.0095	1.0000	0.0188
Variance	0.9148	1.0000	0.0060	1.0000	0.0120
CFAR	0.9300	1.0000	0.1834	1.0000	0.3100
Empirical	0.9148	1.0000	0.0058	1.0000	0.0116

One note to make, before going on to discuss the results, is that for the cases of $SNR = \infty$ and $SNR = 30dB_{FS}$, the *CFAR* algorithm could not achieve maximum Precision due to limited numerical capacity. The algorithm calculating the χ^2_N inverse CDF returned infinity for all chosen false alarm rates before Precision could reach unity. Thus, the results are presented as are for the maximum possible Precision value attained.

The metrics show that the Gaussian detectors have achieved the best scores. For high SNR conditions the *Gaussian Detector* has achieved very good results, with a rather high percentage of impulses being detected (evident in the Sensitivity value). Judging from the two lowest SNR conditions,

where the *CFAR* formulation has performed better, it is reasonable to assume that, had it been numerically possible to calculate the appropriate threshold, it would have performed equally well, or even better.

The *Variance* and *Empirical* detectors performed comparably well, with the former providing marginally better results than the latter in low SNR conditions. For very high SNR, the *Variance Detector* seems to be unable to perform very well. Most probably, the reason behind that is that the standard deviation of the noise does not change significantly when an acoustic event is presented. The difference of noise and impulse energy is way too large, effectively rendering the changes in standard deviation rather small to be easily distinguished by the detector.

All algorithms managed to jointly achieve unity Precision and Specificity values, effectively labeling correctly all frames without an acoustic event while at the same time providing no false positive labels. Nevertheless, the fraction of correctly labeled acoustic events is quite small for medium and low SNR conditions. This, of course, may be attributed to the fact that many of the acoustic events are of low energy, which as already mentioned is the basis of the formulation of all detectors.

As a final remark, the Sensitivity falls with decreasing SNR, which is to be expected. The higher the noise energy is, the less acoustic events will be distinguishable since their energy will be comparable or even lower than that of noise.

Maximum Sensitivity

The same approach as before was followed in order to achieve maximum Sensitivity. As already mentioned, this is equivalent to minimum FNR, or making sure that all acoustic events were detected. In this case, the metrics that will highlight the best method are Precision and Specificity. Both are interrelated since the former declares the ratio of identified events over those that are falsely identified, while the latter shows how many of the frames without an acoustic event were identified correctly. *Table 3.4* shows the results for all four SNRs.

The metrics reveal the inability of the algorithms to detect low energy acoustic events in noise. It is easily seen that for medium to low SNRs ($SNR = -20dB_{FS}$ and $SNR = -10dB_{FS}$), in order to detect all the acoustic events the Specificity value reaches zero. Some algorithms achieve 0.0001 value of Specificity but the number is very small to be of any practical significance.

From the trial run with the original recordings (no AWGN) it seems that all but the *Variance Detector* show quite good characteristics with

Table 3.4: Evaluation metrics of the detection algorithms with real recorded signals. The values correspond to thresholds chosen to achieve maximum (unity) Sensitivity.

Noise: $-\infty$					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.8874	0.4323	1.0000	0.8769	0.6036
Variance	0.0987	0.0868	1.0000	0.0142	0.1598
CFAR	0.8866	0.4305	1.0000	0.8760	0.6018
Empirical	0.8873	0.4321	1.0000	0.8768	0.6034
Noise: -30 **dB$_{FS}$**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.7296	0.2407	1.0000	0.7042	0.3880
Variance	0.0900	0.0861	1.0000	0.0047	0.1585
CFAR	0.7397	0.2477	1.0000	0.7153	0.3971
Empirical	0.7283	0.2398	1.0000	0.7029	0.3869
Noise: -20 **dB$_{FS}$**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.0858	0.0857	1.0000	0.0001	0.1579
Variance	0.0858	0.0857	1.0000	0.0001	0.1579
CFAR	0.0858	0.0857	1.0000	0.0000	0.1579
Empirical	0.0858	0.0857	1.0000	0.0001	0.1579
Noise: -10 **dB$_{FS}$**					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.0857	0.0857	1.0000	0.0000	0.1579
Variance	0.0858	0.0857	1.0000	0.0001	0.1579
CFAR	0.0857	0.0857	1.0000	0.0000	0.1579
Empirical	0.0857	0.0857	1.0000	0.0000	0.1579

reasonably high Specificity and F-Score values. Precision is not very satisfactory, showing that there were more false positive than true positive labelled frames but this is most probably attributed to the constrained Sensitivity value. On the contrary, the *Variance Detector* does not provide good results in this case. All metrics are very low, showing complete inability to achieve perfect acoustic event detection with low false positive or false negative rates.

Since all detectors use the energy of the signal to reach the decision, it is expected to get many false positive labels for low SNR if detection of very low energy acoustic events is to be achieved.

Detection of each source type

In this section, the algorithms are evaluated separately for each source type. The constraints are the same as in the previous evaluations, minimum FPR and FNR and there is no artificial noise added to the recordings. The thresholds used are those of the previous sections for each constraint.

In order to get a more clear view of the diversity of the signals, *Figure 3.3* shows the time-domain representation of a random acoustic event for each source type. It is easy to observe the variety of onsets as well as the amplitude envelopes of the events. A consequence of the latter is that the energy content of the signals will vary quite drastically[4].

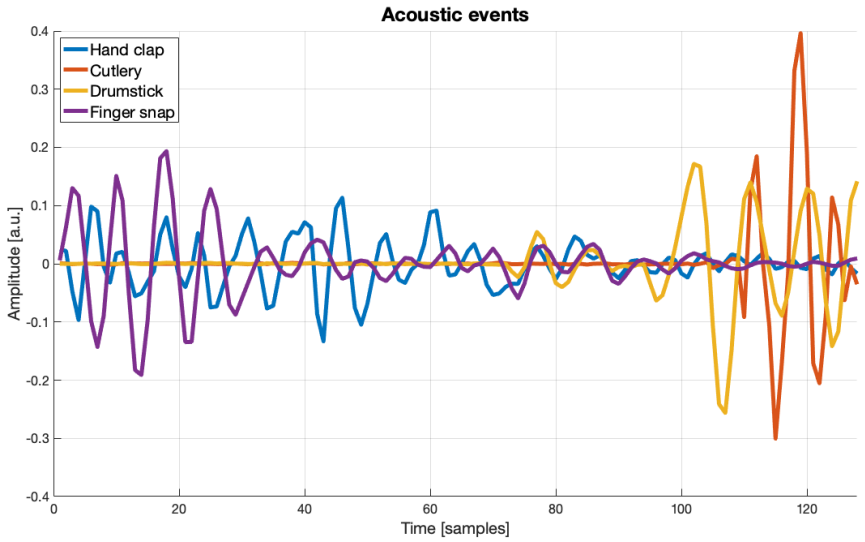


Figure 3.3: Time-domain representation of four random acoustic events, one of each source category.

Table 3.5 shows the results attained with the threshold used to get maximum Precision. Notable is the inability of *CFAR* to achieve unity Precision as is the case for this SNR condition in the previous evaluations. Interestingly, the *Gaussian* and *Empirical* detectors achieve identical results, showing very similar efficiency in all source types.

No trends are distinguishable in the results. The values of the metrics show quite good consistency with the greatest deviation present in the Sensitivity value, which is more prominent for the *Variance Detector*. Similar

[4]The energy is calculated with the use of equation (2.8), which calculates the squares of the samples. Thus, the energy content increases rapidly with increasing sample values.

Table 3.5: Evaluation metrics of the detection algorithms with real recorded signals. The values correspond to thresholds chosen to achieve maximum (unity) Precision for each source type.

Claps					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9741	1.0000	0.7033	1.0000	0.8258
Variance	0.9563	1.0000	0.4998	1.0000	0.6665
CFAR	0.9413	0.5985	0.9997	0.9357	0.7487
Empirical	0.9741	1.0000	0.7033	1.0000	0.8258
Cutlery					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9800	1.0000	0.8084	1.0000	0.8940
Variance	0.9742	1.0000	0.7523	1.0000	0.8586
CFAR	0.9589	0.7173	0.9997	0.9541	0.8353
Empirical	0.9800	1.0000	0.8084	1.0000	0.8940
Drumsticks					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9784	1.0000	0.8093	1.0000	0.8946
Variance	0.9693	1.0000	0.7288	1.0000	0.8430
CFAR	0.9428	0.6642	1.0000	0.9355	0.7982
Empirical	0.9784	1.0000	0.8093	1.0000	0.8946
Finger snaps					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9701	1.0000	0.6399	1.0000	0.7804
Variance	0.9342	1.0000	0.2073	1.0000	0.3434
CFAR	0.9552	0.6499	0.9990	0.9513	0.7875
Empirical	0.9701	1.0000	0.6400	1.0000	0.7805

is the variation in the F-Score for the same algorithm but this is expected, since this metric is dependent on the Sensitivity.

Similarly, *Table 3.6* shows the results obtained with the thresholds achieving maximum Sensitivity. This experiment reveals some interesting results. In general, the metric values show enough consistency for all source types and algorithms with the exception of the *Variance* and *CFAR* detectors for the "cutlery" and "finger snaps" categories. For the latter source, the aforementioned detectors show complete inability to correctly label all acoustic events without producing a very large number of false positives. Comparing to the results of *Table 3.5*, the same detectors provide

the lowest Sensitivity value for this source type, which gives away a possible relation between the two cases.

Table 3.6: Evaluation metrics of the detection algorithms with real recorded signals. The values correspond to thresholds chosen to achieve maximum (unity) Sensitivity for each source type.

Claps					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9298	0.5546	1.0000	0.9231	0.7134
Variance	0.9028	0.4736	1.0000	0.8935	0.6428
CFAR	0.8884	0.4393	1.0000	0.8777	0.6105
Empirical	0.9297	0.5545	1.0000	0.9230	0.7134
Cutlery					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9524	0.6864	1.0000	0.9468	0.8140
Variance	0.1771	0.1125	1.0000	0.0813	0.2022
CFAR	0.9247	0.5805	1.0000	0.9159	0.7346
Empirical	0.9524	0.6864	1.0000	0.9468	0.8140
Drumsticks					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.9292	0.6151	1.0000	0.9202	0.7617
Variance	0.9112	0.5603	1.0000	0.8999	0.7182
CFAR	0.8940	0.5163	1.0000	0.8805	0.6810
Empirical	0.9292	0.6150	1.0000	0.9201	0.7616
Finger snaps					
Detector	**Accuracy**	**Precision**	**Sensitivity**	**Specificity**	**F-Score**
Gaussian	0.8792	0.4075	1.0000	0.8683	0.5791
Variance	0.0981	0.0843	1.0000	0.0164	0.1555
CFAR	0.0831	0.0831	1.0000	0.0000	0.1534
Empirical	0.8791	0.4073	1.0000	0.8682	0.5788

What is common in both those algorithms is the dependence of the decision on the past values of the input signals. Their inability to correctly distinguish between noise and signal may be directly related to this feature. Either the onset of the amplitude envelope of the acoustic events may be long, or the temporally neighbouring noise may be comparable to the impulse in amplitude. The second possibility is rather interesting and the most probable. Compared to the *Gaussian Detector*, *CFAR*, although based on the same formulation may be affected negatively if an acoustic

event is missed resulting in increase of the (assumed) noise energy, effectively decreasing the detection sensitivity of the algorithm. The same principle applies to the *Variance Detector* since it has an inherent adaptive thresholding mechanism, albeit indirect.

3.4.3 Running times

The four algorithms used in the evaluation with recorded data are also subjected to "speed" tests. It is expected that all algorithms will have different running times based on the specific environment they are being executed. The same algorithm will exhibit significantly lower running time when implemented in a compiled programming language such as C/C++, or run on an embedded configuration such as a microprocessor or Field Programmable Gate Array (FPGA), compared to an implementation in a scripting language such as Python or MATLAB®, which is the case in this work.

Nevertheless, the tests serve as an indicative, qualitative comparison between the algorithms. Since the result of the detection process does not affect the running time of the algorithms, they are tested with noise signals drawn from a Gaussian PDF. In order to try and compensate the possible optimisations resulting from repetitive execution of the same code (which, of course, can be part of an optimisation scheme, such as cache misses optimisation, in some implementation) for each iteration, all four algorithms are run and timed consecutively.

The versions of the algorithms tested are identical to those evaluated with the recorded signals. For the *Variance* and *CFAR* detectors the process of updating the noise vector is included in the resulting times.

The algorithms were run one million times each. The input is one frame of signal samples, so the process of calculating the energy of the frame is included in the resulting time values. The metrics used to evaluate the results of this experiment are the *mean*, *median* and *mode*. The results are presented in *Table 3.7*. Additionally, the percentage of the duration of a frame occupied for the processing is shown in the last column to provide some clear indication on the usage of the available time for the detection task. The percentage is calculated with the mean value in order to indicate what part of the available time this process takes on average.

The results show clearly that the fastest algorithm is the *Gaussian* and the one with the longest running time is its variant *CFAR*. It is obvious that the heaviest burden is on the calculation of the threshold, happening each frame. The other two algorithms, *Variance* and *Empirical* lay between the two Gaussian detectors but a lot closer to the lower margin. With

Table 3.7: Metrics of the running times of the four detection algorithms.

Detector	**Mean** [μs]	**Median** [μs]	**Mode** [μs]	**% of frame duration**
Gaussian	6.8114	5.3450	5.2520	0.2347
Variance	29.2104	24.0980	23.6550	1.0064
CFAR	362.9615	320.9170	308.7030	12.5052
Empirical	16.1810	12.3770	12.2290	0.5575

the exception of the *CFAR Detector*, all other algorithms use only up to 1% of the available time, showing that they are appropriate for the task at hand, leaving an abundance of processing time for the DoA estimation algorithms.

3.4.4 Summary

Four different algorithms were evaluation for the task of *Signal Detection*. Three of them are *Energy Detectors* and one is an *Empirical Detector* using the RMS value of the incoming signal.

At an initial stage, the three energy detectors were evaluated on a theoretical framework with a simple Monte Carlo simulation. The results provided insight both on the validity of the implementations as well as the expected detection efficiency of the algorithms. During the simulations the appropriate threshold ranges to be re-evaluated with the recorded signals was also investigated. At this stage, the best results were obtained with the *Variance Detector* which achieved perfect score on every metric.

The four algorithms were evaluated with real, recorded signals with $22dB$ variation in their peak amplitude and about $25dB$ in their RMS values. The algorithms were evaluated under two constraints, zero FPR and zero FNR. The latter showed that no algorithm is capable of detecting all acoustic events without introducing large false positive labelled frames. Similarly, when no false positives were forced, a small fraction of the acoustic events were detected. This is largely attributed to the fact that many of the acoustic events' energy is comparable to the energy of the noise.

Finally, the running times of the algorithms were tested. Most of them use only a small fraction of the available time leaving ample for the DoA estimation task. The sole exception is the adaptive Gaussian detector which uses up to almost 12% of the duration of a frame due to the threshold adaptation process.

3.5 Direction-of-Arrival

This section presents the results of the tests performed to evaluate the direction of arrival estimation algorithms. In all cases the estimated angles of the tested algorithms are shown in addition to the corresponding metrics.

The metrics used for the evaluations of DoA estimation are the *Root Mean Square Error* (RMSE), *Mean Absolute Error* (MAE), variance (σ^2) and *Percentage Error* (PE). The latter is specified for some criterion of error. In this work deviation from the true value of more than 5^o qualifies the estimate as erroneous. For definitions of the error metrics see *Appendix B*.

3.5.1 Simulated experiments

The theoretical evaluation of the algorithms served as an initial investigation step to evaluate the performance of the implementations of this work before moving on to perform tests with real, recorded signals.

The simulations are useful for verifying that the different DoA estimation methods are correctly implemented as well as verifying that they are appropriate for the use case considered in this Thesis.

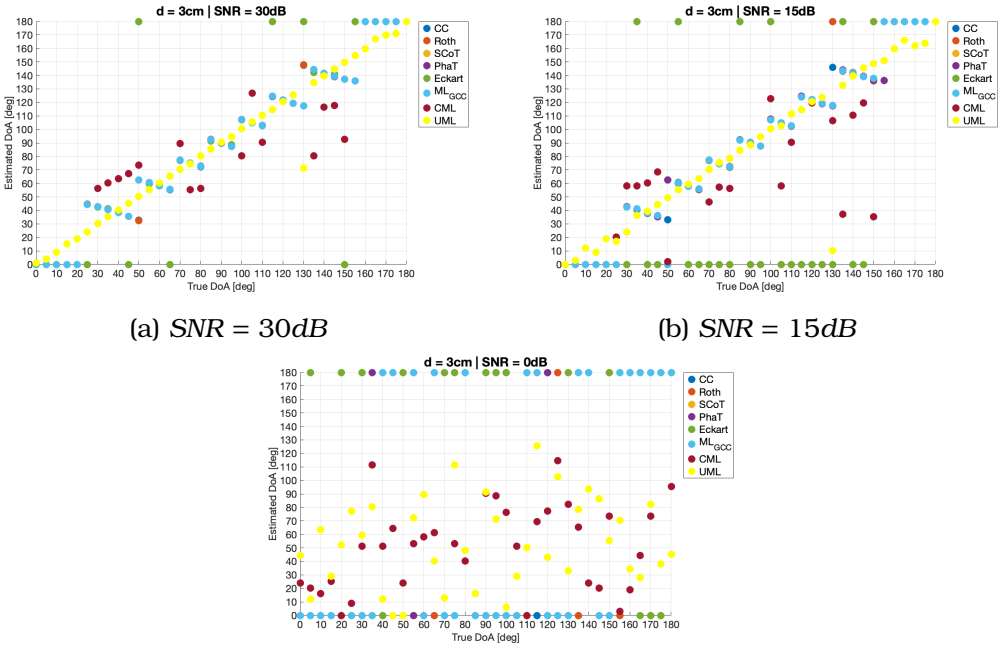
At this stage of the evaluation a rather simplistic Monte Carlo simulation was performed. The generated signals are noise signals drawn from a Gaussian distribution. Each sensor's input is contaminated with IID AWGN. To make a fair comparison, the same signal was used for the evaluation of all algorithms in each iteration.

Simulations were performed for three SNR conditions ranging from $30dB$ to $0dB$ with a step of $15dB$ and angles ranging from 0^o to 180^o with a step of 5^o. Additionally, three inter-element distances were simulated that coincide with those used during the measurements, $3cm$, $5cm$ and $10cm$. 1000 iterations were performed for each angle per SNR value per inter-element distance. The frame size of the signal is kept constant at 128 samples and no processing was applied to the signals prior to being "fed" to the algorithms.

As a rule of thumb, Monte Carlo methods require many iterations to achieve a good estimate within specified confidence intervals [1]. The theoretical evaluation of the implemented algorithms is out of the scope of this work though and the number of iterations chosen is adequate to provide insight on the trends to be expected in the evaluation with real recorded signals.

The outcomes of the theoretical evaluation tests are presented separately for the various microphone array setups.

Inter-element distance of 3 cm



(a) $SNR = 30dB$ (b) $SNR = 15dB$

(c) $SNR = 0dB$

Figure 3.4: Mode of estimated angles of 1000 trials per angle, calculated with all tested algorithms for an array with inter-element distance $d = 3cm$.

Figures 3.4a to 3.4c show the results of the trials for simulated inter-element distance $d = 3cm$. The figures show the mode value for each angle of incidence per method and evaluation metrics are summarised in *Table 3.8*.

The metrics show clear evidence of complete inability of the *Eckart* algorithm to estimate the angle of arrival, even for high SNR. The rest of the GCC algorithms show moderate results for high and medium SNR and all algorithms fail for low SNR conditions overall. The maximum likelihood algorithms show a mixed behavior with the UML being superior in all aspects both compared to CML and the GCC algorithms. Nevertheless, they too fail under dire SNR conditions.

What the metrics are unable to show can be easily observed in the figures. For the GCC algorithms, the estimated angles seem to be clustered around certain values, with the extreme angles close to either 0^o or 180^o showing the most severe behaviour. The same pattern is not visible for the ML family. The behavior has to do with the formulation of the GCC algorithms, which, as stated in *Section 2.3.1* has resolution limitations re-

Table 3.8: Evaluation metrics for simulated inter-element distance of $3cm$. (a) SNR = $30dB$, (b) SNR = $15dB$, (c) SNR = $0dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross Correlation	10.5247	8.4706	19.9407	70.1811
Roth	11.4161	8.7303	44.3267	69.6108
SCoT	10.2288	8.2872	17.3619	69.9541
PhaT	10.2288	8.2872	17.3619	69.9541
Eckart	48.8326	31.1358	1489.1657	76.6838
ML_{GCC}	10.3168	8.3265	22.9681	69.6243
CML	18.9269	12.2603	91.8526	46.8468
UML	6.4282	1.6010	16.7684	5.4054

(a) SNR = $30dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross Correlation	8.9089	6.5606	86.9353	68.2324
Roth	21.0632	12.2806	396.7252	68.0459
SCoT	12.6640	8.9941	89.6960	67.3054
PhaT	12.6640	8.9941	89.6960	67.3054
Eckart	61.7937	45.9542	2004.3283	86.9595
ML_{GCC}	13.4376	9.2632	113.3618	67.0351
CML	30.2255	17.5651	428.5745	52.2523
UML	14.5784	4.6998	141.9391	16.2162

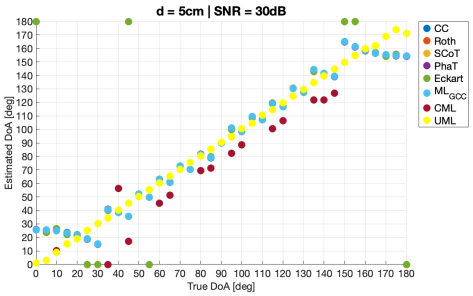
(b) SNR = $15dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross Correlation	63.5822	48.4729	2062.9680	88.7027
Roth	68.4861	55.1921	2088.5384	93.3324
SCoT	65.6437	51.3703	2077.7854	91.0568
PhaT	65.6437	51.3703	2077.7854	91.0568
Eckart	69.9680	57.2693	2047.1695	94.5081
ML_{GCC}	67.1285	53.2063	2102.1124	92.0081
CML	64.0979	48.9300	1134.9904	86.4865
UML	64.9317	55.1603	1213.5546	96.3964

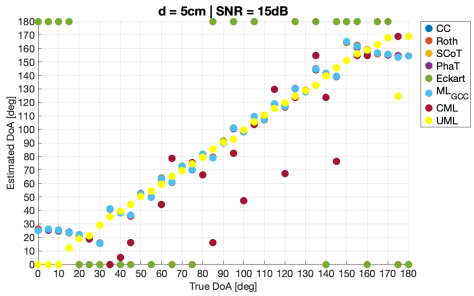
(c) SNR = $0dB$.

lated to inter-element distance and angle of incidence. The ML algorithms, based on different formulation, do not seem to suffer from this problem.

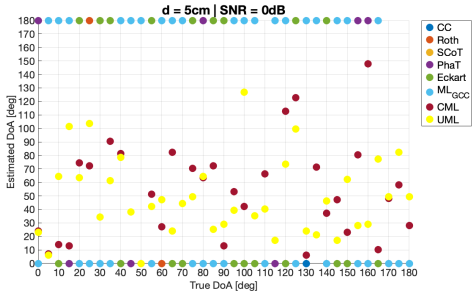
Inter-element distance of 5 cm



(a) $SNR = 30dB$



(b) $SNR = 15dB$



(c) $SNR = 0dB$

Figure 3.5: Mode of the estimated angles of 1000 trials per angle, calculated with all tested algorithms for an array with inter-element distance $d = 5cm$.

Similar to the previous test case, the results of the simulation are shown in figures 3.5a to 3.5c. Again, the mode values of the estimated angles are shown and the inter-element distance is $d = 5cm$.

The clustering of the estimated values for the GCC family of algorithms is distinct in the figures, albeit this time on a finer grid. Additionally, extreme angles of incidence seem to never return an estimate close to the edges of the estimated range. The maximum likelihood algorithms do not suffer from such an issue as mentioned in the previous section too. Similar to the simulation with inter-element distance $d = 3cm$, *Eckart* algorithm shows inability to perform the task of estimation.

The values of the metrics for this run of the simulation are shown in *Table 3.9* below. Similar trends to the setup with inter-element distance of $3cm$ are visible here too. All algorithms seem to have some minor improvement in all values, except maybe for the case of $SNR = 0dB$, where again the results seem to imply complete inability to estimate the directions of arrival.

Table 3.9: Evaluation metrics of all implemented algorithms for simulated inter-element distance of $5cm$. (a) SNR = $30dB$, (b) SNR = $15dB$, (c) SNR = $0dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross Correlation	10.1630	7.5496	0.1959	53.8973
Roth	10.9681	7.7152	18.2259	53.3486
SCoT	10.1301	7.5252	0.8352	53.6351
PhaT	10.1301	7.5252	0.8352	53.6351
Eckart	45.7350	28.1443	1281.9876	68.9405
ML_{GCC}	10.1542	7.5416	2.0501	53.4459
CML	9.8912	6.3561	45.7077	40.5405
UML	1.6589	0.7952	1.1964	2.7027

(a) SNR = $30dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross Correlation	10.1963	5.3251	0.6939	52.9000
Roth	18.6569	10.1848	212.7032	54.6838
SCoT	10.2135	7.5661	10.9680	52.0649
PhaT	10.2135	7.5661	10.9680	52.0649
Eckart	59.3098	43.5775	1809.6389	84.2865
ML_{GCC}	10.5368	7.6611	17.6262	52.0459
CML	25.6392	14.0732	383.6564	45.0450
UML	7.3269	2.4506	34.8425	9.0090

(b) SNR = $15dB$.

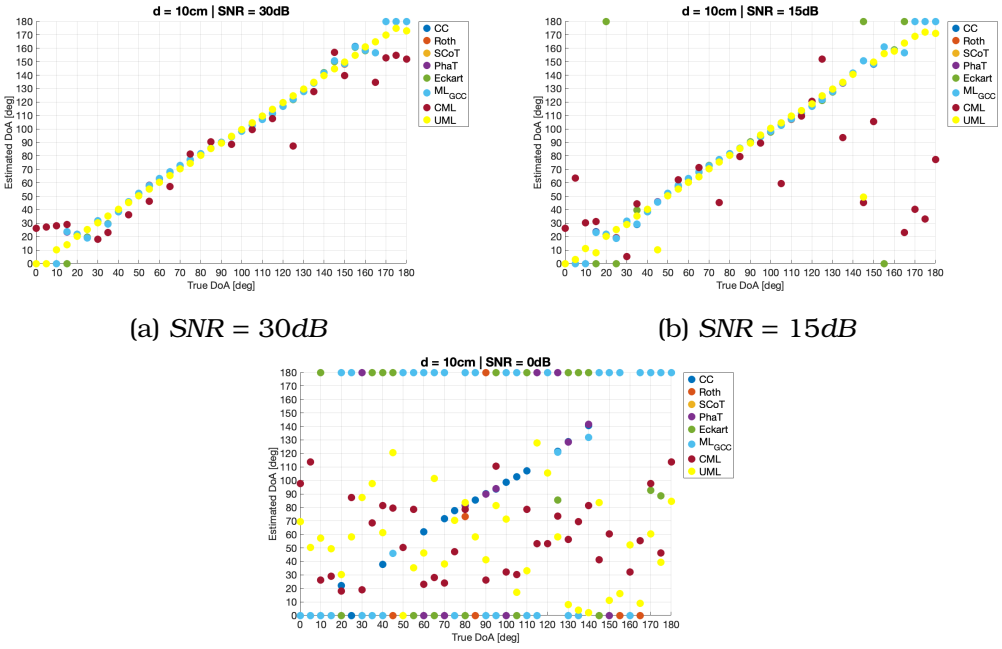
Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross Correlation	47.7713	46.2117	1838.7928	87.6486
Roth	66.8914	53.8445	1833.9549	92.8514
SCoT	64.6291	50.7887	1857.5709	90.2757
PhaT	64.6291	50.7887	1857.5709	90.2757
Eckart	68.8111	56.4535	1853.3666	94.4432
ML_{GCC}	65.6913	52.1846	1854.5860	91.3432
CML	66.5406	51.1732	1371.1932	87.3874
UML	66.0183	53.4607	1284.5029	92.7928

(c) SNR = $0dB$.

Once more, the UML algorithm outperforms the rest by a considerable margin showing even better results than before. Stability seems to be

better for all tested algorithms as the variance has gone down by almost a tenfold in many cases.

Inter-element distance of 10 cm



(a) $SNR = 30dB$ (b) $SNR = 15dB$

(c) $SNR = 0dB$

Figure 3.6: Mode of the estimated angles of 1000 trials per angle, calculated with all tested algorithms for an array with inter-element distance $d = 10cm$.

In the final run of the Monte Carlo simulation, an array with inter-element distance $d = 10cm$ is simulated. The results of this run are shown in figures 3.6a to 3.6c. The mode of the estimates for each angle are shown for all implemented algorithms. The corresponding statistical metrics are given in *Table 3.10*.

The inability of the *Eckart* algorithm to perform to task is visible in this test case too. Regarding the rest of the GCC algorithms, they seem to perform especially well, compared to the other two setups. Of course this is not the case for the low SNR condition, where once more all algorithms failed to provide any correct estimates.

The clustering of the values is not observable any more, except at the extremes of the estimated range. It seems that the further away from

Table 3.10: Evaluation metrics of all implemented algorithms for simulated inter-element distance of $10cm$. (a) SNR = $30dB$, (b) SNR = $15dB$, (c) SNR = $0dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross Correlation	4.2773	3.2916	1.4852	21.6189
Roth	5.7789	3.5236	20.0367	22.0405
SCoT	4.2246	3.2663	1.1026	21.6108
PhaT	4.2246	3.2663	1.1026	21.6108
Eckart	50.5769	32.0599	1462.4589	60.4730
ML_{GCC}	4.2488	3.2773	2.1124	21.5622
CML	11.9995	8.1841	34.2838	59.4595
UML	1.1461	0.5506	0.3948	0.9009

(a) SNR = $30dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross Correlation	4.5933	3.3343	6.8137	21.3216
Roth	17.9594	6.8887	296.3677	26.0027
SCoT	4.3691	3.3336	6.6414	21.1676
PhaT	4.3691	3.3336	6.6414	21.1676
Eckart	61.1456	45.6191	1753.5205	80.7108
ML_{GCC}	5.4804	3.4961	19.5094	21.1568
CML	46.3966	26.5147	535.7379	59.4595
UML	19.0614	4.5045	160.4748	8.1081

(b) SNR = $15dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross Correlation	63.6991	49.6785	1750.8391	86.4946
Roth	66.3726	53.8065	1694.6832	92.5622
SCoT	65.0056	51.6522	1735.4461	89.3811
PhaT	65.0056	51.6522	1735.4461	89.3811
Eckart	67.4666	55.2952	1724.3777	94.3595
ML_{GCC}	65.7849	52.8180	1729.9016	90.8405
CML	70.4573	59.3211	1132.6400	95.4955
UML	75.2300	62.5104	900.8465	95.4955

(c) SNR = $0dB$.

the broadside of the array (at 90^o) the source is situated, the worse the

estimates become. Nevertheless, the accuracy does not drop significantly until the angle of incidence reaches the edges of the range.

The UML algorithm seems to provide the best results for high SNR conditions in this case too. In medium SNR conditions though there seems to be rise of the RMSE to a rather high value. Joint inspection of RMSE and MAE though indicate that those values may constitute outliers with the mean value of the errors being close to that of all other algorithms. This may be a result of the low number of iterations performed for the Monte Carlo simulation.

Summary

In all three simulated setups *Eckart* didn't provide any meaningful results for any SNR condition, thus it is dropped from further testing.

Visible in the metrics' values is the tendency of the estimates to become more accurate with increasing inter-element distance. Additionally, the clustering of the estimates seems to happen on an increasingly finer grid which monotonically follows the increment of the inter-element distance. Both trends are interrelated, it seems that the mode values, which represent the estimates with the highest frequency, are situated on the grid which can introduce a rather large and systematic error (bias). The finer the grid, the smaller the systematic error of the estimates.

One more note to make is the decreasing accuracy of all algorithms with increasing angle, in reference to the broadside. It is especially visible at the extremes of the search range, and more prominent for small inter-element distances. These observations are supported by the theory covered in *Section 2.3.1*.

From all algorithms the one with consistently the best performance is UML, which in addition to being able to accurately estimate the angle of incidence for all tested inter-element distances, does not suffer from the clustering behaviour observed with the GCC methods. It is not clear whether any of the two ML algorithms provides similar results to GCC_{ML} (which is derived on the same basis [5]). It is not easy to reach any conclusions since the latter provides an estimate of the cross correlation function, which does suffer from decreasing angle resolution, in contrast to the other two ML methods. Thus, comparison is not possible on this basis.

3.5.2 Variations of GCC algorithms

Two proposed algorithms providing slight modifications to the GCC family are investigated at this stage. The first includes an additional step

where the prominence of the resultant cross correlation function is tested. If the value exceeds a threshold the estimate is considered reliable, while on the opposite case it is discarded.

The second algorithm introduces a modification of the PhaT frequency weighting filter where the weighting parameter provides an implementation with characteristics of both CC and PhaT weightings of varying degree.

The simulations for the evaluation of those algorithms are performed for an array with inter-element distance $d = 10cm$ for the same three SNR conditions used so far.

Thresholded PhaT

This algorithm is proposed by Jeon et al. [45] to decrease the number of wrong estimates provided by PhaT. The proposed approach can be used with any algorithm within the family of GCC since the added step is not related to the unique implementation of PhaT. Nevertheless, in this work it is tested only with PhaT like proposed by the authors of the original article.

The cross correlation function is tested for its main peak's prominence. If this is higher than a threshold value the estimate is considered valid, otherwise is discarded. The non-linear prominence test is of the form

$$\frac{1}{K^2}\left(\max\left\{r_{y_1y_2}(\tau)\right\} - \min\left\{r_{y_1y_2}(\tau)\right\}\right) > \eta \tag{3.4}$$

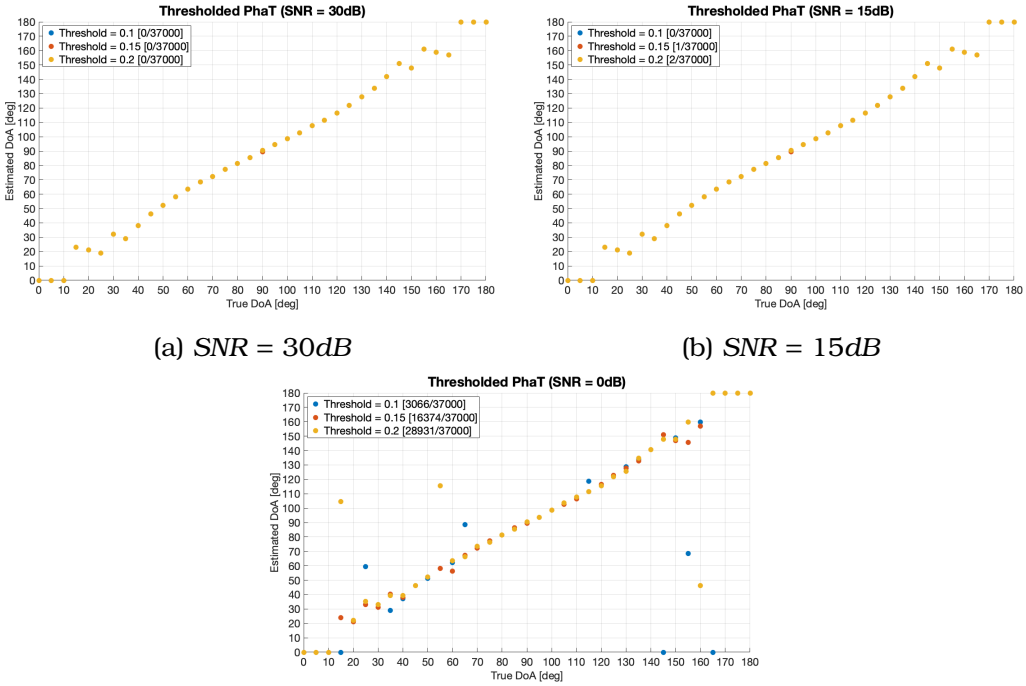
where η is the threshold and K the total lags in the cross correlation function (its length). The values of threshold tested in this work are 0.10, 0.15 and 0.20 which are 0.05 below, at, and 0.05 above the proposed value.

The results of the simulations are shown in figures 3.7a to 3.7c for the three SNRs. Each figure shows the mode of all trials for each angle per parameter value. The numbers in the brackets express the number of DoAs rejected by the algorithm out of the total estimates.

It is apparent in *Figure 3.7c* that the algorithm does not provide much immunity against low SNR conditions. This, of course, is to be expected since the algorithm does nothing to alter the shape of the calculated cross correlation function. The only additional safeguard provided is the rejection of unreliable estimates.

The results of the corresponding metrics are summarised in *Table 3.11*. All metrics are taken over non-rejected estimates only. This may degrade the statistical significance of the metrics but it wouldn't make sense to calculate values over practically non-existent estimates.

The metrics indicate good estimation efficiency for high to medium SNRs ($30dB$ and $15dB$). On the contrary, for low SNR conditions, the



(a) $SNR = 30dB$ (b) $SNR = 15dB$

(c) $SNR = 0dB$

Figure 3.7: Mode of the estimated angles of 1000 trials per angle, calculated with the Thresholded PhaT algorithm for an array with inter-element distance $d = 10cm$. The parameter values are 0.1, 0.15 and 0.2.

algorithm does not provide good results. Even by rejecting the estimates with low prominence the angles seem to be off by a considerable margin most of the time. In *Figure 3.7c*, the mode of the evaluated angles shows good agreement with the true values but since there is no indication as to how many times the presented values appeared in the simulation, it is more appropriate to trust in the metrics.

The algorithm doesn't seem to be able to provide any improvements to the accuracy or susceptibility to noise for the basic GCC PhaT algorithm in a statistical sense, but it may provide the ability to reject estimations with inherent high uncertainty, even at medium SNR.

PhaT β

This variant of PhaT is proposed by Ramamurthy et al. [74] and verified to admit increased robustness to noise [46, 75] and reverberation [76]. The contribution of the algorithms is the alteration of the frequency filter of PhaT. The weighting function becomes

Table 3.11: Evaluation metrics for the parameter of the Thresholded PhaT algorithm. (a) SNR = 30dB, (b) SNR = 15dB, (c) SNR = 0dB.

Prominence threshold	**RMSE**	**MAE**	**Variance**	**PE**
0.1	4.2219	3.2640	0.9300	21.6162
0.15	4.2196	3.2628	0.9123	21.6135
0.2	4.2231	3.2647	1.0441	21.6188

(a) SNR = 30dB.

Prominence threshold	**RMSE**	**MAE**	**Variance**	**PE**
0.1	4.4019	3.3354	6.9844	21.1432
0.15	4.4189	3.3383	7.5040	21.1649
0.2	4.3730	3.3323	6.8544	21.1428

(b) SNR = 15dB.

Prominence threshold	**RMSE**	**MAE**	**Variance**	**PE**
0.1	64.4748	51.1443	1702.5032	89.0495
0.15	64.4054	50.9999	1719.7554	88.4754
0.2	62.3275	48.5991	1656.1960	86.6444

(c) SNR = 0dB.

$$\psi(f) = \frac{1}{\left|Y_1(f)\,\overline{Y_2}(f)\right|^{\beta}} \tag{3.5}$$

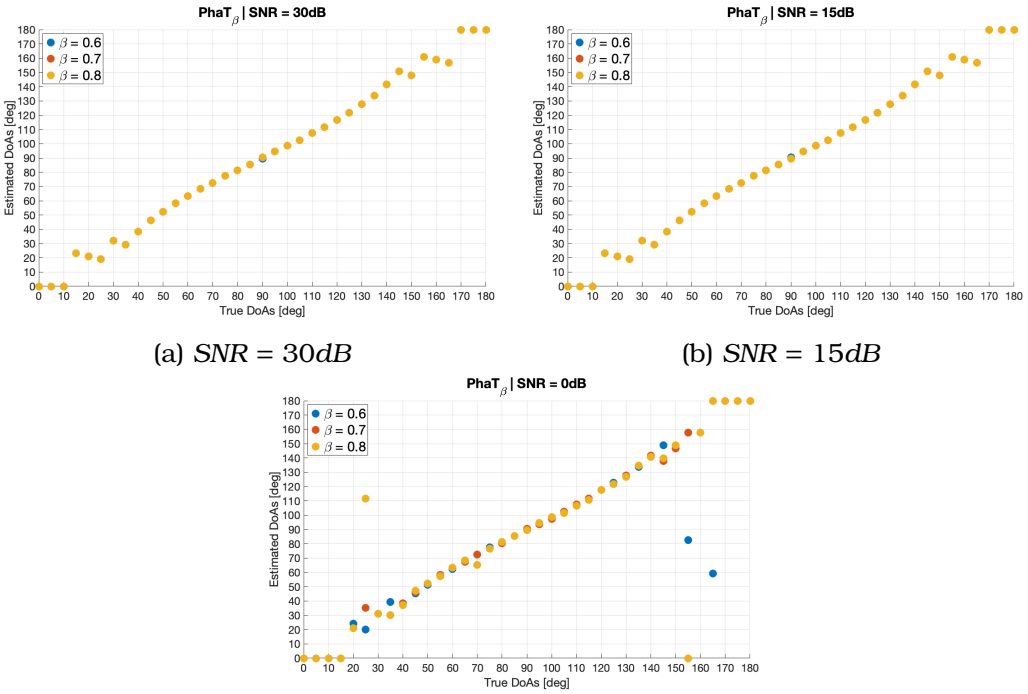
where β is the parameter controlling the algorithm and assumes values in the range $[0, 1]$. It is seen that when $\beta = 0$ the algorithm corresponds to the standard CC and for $\beta = 1$ to the generic PhaT.

The implementation is trialed for three values of the β parameter, 0.6, 0.7 and 0.8 corresponding to 0.1 below, at, and 0.1 above the proposed value in [46] for the case of $SNR = 0dB$.

The mode values of the estimated angles for each SNR condition per parameter value are presented in figures 3.8a to 3.8c and the metrics in *Table 3.12*.

Similar trends compared to the other variants of PhaT (including the generic formulation) are clear regarding the inability of the variant to improve robustness under low SNR conditions. All estimates seem to be wrong in the 0dB SNR case.

Examining the statistical metrics it can easily be deduced that the results for high and medium SNRs are improved compared to the generic



(a) $SNR = 30dB$

(b) $SNR = 15dB$

(c) $SNR = 0dB$

Figure 3.8: Mode of the estimated angles of 1000 trials per angle, calculated with the Phat β algorithm for three SNRs. The results correspond to and array setup with inter-element distance $d = 10cm$ and the β parameter values are 0.6, 0.7 and 0.8.

implementation of PhaT. The estimates seem to be quite stable for those conditions and in certain cases even lower the variance of the base PhaT algorithm by a small margin. Nevertheless, for low SNR, there doesn't seem to be any improvement at all.

3.5.3 Experiments with recorded data

The algorithms evaluated with the Monte Carlo simulations are now tested with real recorded signals. The tests are divided down to single parameter evaluations to isolate the effects each parameter has on the estimation task.

Both physical setup and algorithmic parameters were tested. Parameters relevant to the physical setup are the inter-element distance and the distance of the source from the array. Algorithmic parameter tests comprise the inclusion of windowing functions in the signal chain, the behavior

Table 3.12: Evaluation metrics for the parameter of PhaT β algorithm. (a) SNR = 30dB, (b) SNR = 15dB, (c) SNR = 0dB.

β **value**	**RMSE**	**MAE**	**Variance**	**PE**
0.6	4.2159	3.2645	0.1474	21.6216
0.7	4.2171	3.2647	0.2334	21.6216
0.8	4.2175	3.2645	0.3294	21.6216

(a) SNR = 30dB.

β **value**	**RMSE**	**MAE**	**Variance**	**PE**
0.6	4.3016	3.3007	4.4547	21.3703
0.7	4.3042	3.3002	4.6119	21.2892
0.8	4.3185	3.3075	5.3055	21.3000

(b) SNR = 15dB.

β **value**	**RMSE**	**MAE**	**Variance**	**PE**
0.6	63.9339	50.1549	1736.0379	87.2973
0.7	64.3786	50.6706	1734.0077	87.9649
0.8	64.1856	50.5634	1718.8745	88.0919

(c) SNR = 0dB.

of the algorithms under noisy conditions, the exclusion of aliased frequencies from the estimation process and the use of more than one successive frames of data.

All GCC algorithms but *Eckart* are evaluated with the inclusion of PhaT$_\beta$ presented in *Section 3.5.2*. The ML algorithms, despite showing great potential, were dropped from further evaluation due to their very long running times. For more information, see *Section 3.5.4*.

Acquisition of acoustic event recordings

The acquisition process includes the recording of acoustic events in stereo channels, where each channel corresponds to a specific set of setup parameters. At least twenty repetitions of the excitation signal where recorder for each set of parameters. For all measurement setups, the angles of incidence ranged from 0^o to 180^o with a step of 10^o. All possible combinations were recorded resulting in a database of 3496 elements.

Finger snaps were used as the main type of impulsive sonic gesture to trigger the system. Using a signal that is easy to replicate, but at the same time provide adequate variability to approach realistic conditions

was of utmost importance for the current work. Thus, a signal a person can easily produce was chosen as the best candidate. It is important to mention that the generated repetitions were performed by a person and are not reproductions of a prerecorded signal.

A different signal, with good characteristics, taken into consideration was an electric discharge produced by a high voltage pulse generator. It was rejected early on, due to the fact that it does not resemble an easily produced "every-day" signal.

The recorder signals were cropped into one second audio blocks and were exported as *.wav* audio files. The onset of the acoustic event was placed randomly at a position close to the middle of the block. The resulted *.wav* files form the elements of the database used in the evaluation process of the DoA estimation algorithms. In all test cases the frame size is kept constant at 128 samples.

In all figures containing results of the evaluation with recorded signals the x axis does not represent a continuum of angle values. The vertical lines cluster the estimates with all points situated in between two lines representing the results for a true angle equal to the value of the left edge of the interval.

Inter-element distance

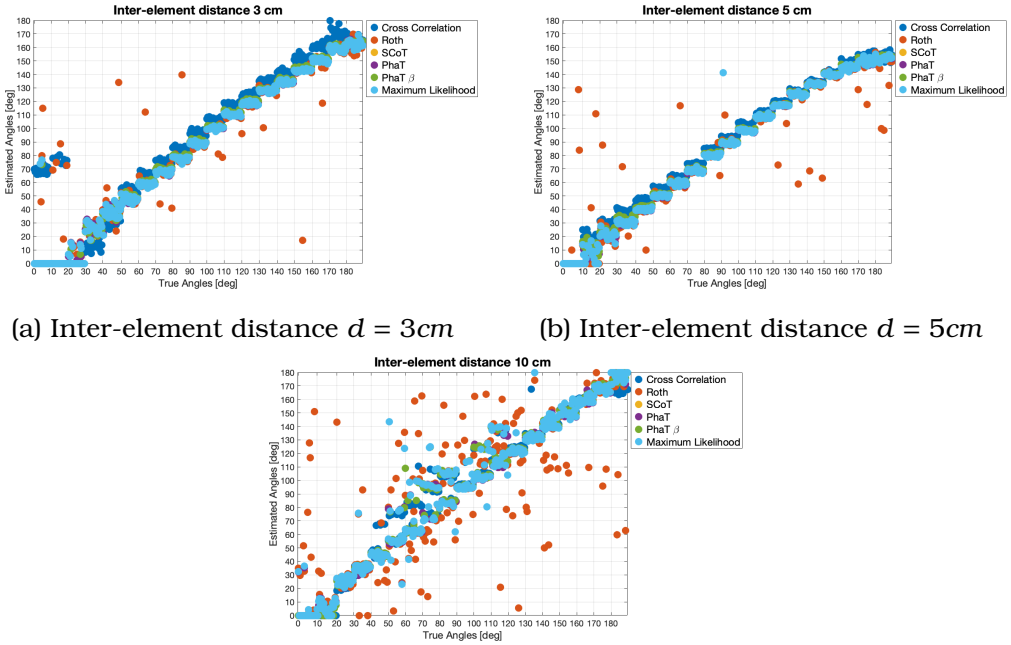
The inter-element distance is one of the main design parameters of the array, its effect on the estimation task is calculated theoretically in section 2.3.1 and is partially tested in *Section 3.5.1*. In brief, it affects both the angle resolution and the "alias-free" frequency bandwidth. These two factors counter each other, with shorter inter-element distances providing coarser resolution but greater bandwidth free of spatial aliasing artifacts, thus more robust estimates. On the contrary, greater inter-element distances allow for more precise estimates but are prone to aliasing artifacts.

Here the parameter is put to the test with recorded signals. Three inter-element distances that coincide with those used in the theoretical evaluation, $3cm$, $5cm$ and $10cm$ were tested. The data used correspond to recordings made at a distance from the array $l = 40cm$.

The resulting estimates of all algorithms are presented in figures 3.9a to 3.9c. The results seems to be better than those obtained by the Monte Carlo simulations. The clustering of the estimates is not visible and seems that most of the algorithms, with the exception of *Roth*, provide reasonably good estimates.

What seems to be in good agreement with the simulations is the reduction of angular resolution and the increase of erroneous estimates at the

extremes of the search range. Furthermore, the setup with inter-element distance of $10cm$ seems to give results with greater variance, even for angles close to 90^o. The statistical metrics of *Table 3.13* reveal that indeed this is the case for this setup.



(a) Inter-element distance $d = 3cm$ (b) Inter-element distance $d = 5cm$

(c) Inter-element distance $d = 10cm$

Figure 3.9: Estimated angles of incidence of recorded signals. The source was located $40cm$ from the array centre and the inter-element distances are (a) $d = 3cm$, (b) $5cm$ and (c) $10cm$.

A slight increase, compared to the simulations, of the RMSE and MAE for the $3cm$ and $5cm$ inter-element distances and decrease of the same metrics for the $10cm$ distance is observed. Joint inspection of the metrics and figures leads to the conclusion that in big part, those values depend on the errors introduced at extreme angles of incidence, at least for the two shortest inter-element distances. All algorithms fail to estimate the angle correctly there.

It may be instructive to compare the algorithms on a reduced search range from 20^o to 160^o. The relevant metrics can be seen in *Table 3.14*.

It is clear that there is increase in performance at $3cm$ and $5cm$ distances. On the contrary, it seems that further increasing the inter-element distance deteriorates the results. Comparison of the results of tables 3.13, 3.14 and inspection of *Figure 3.9c* show that the range of angles most er-

Table 3.13: Evaluation metrics for the inter-element distance parameter of the setup. (a) d = 3cm, (b) d = 5cm, (c) d = 10cm.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	20.1026	11.5898	90.9780	68.0519
Roth	16.1670	7.9308	194.0848	43.6364
SCoT	8.5248	5.4759	18.8750	39.7403
PhaT	8.5248	5.4759	18.8750	39.7403
PhaT_{β} [$\beta = 0.7$]	5.1211	8.1518	15.7279	37.4026
GCC_{ML}	9.3848	5.7288	31.6654	40.2597

(a) d = 3cm.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	9.3375	6.9196	3.6529	52.2078
Roth	17.3907	8.4814	171.4017	36.1039
SCoT	9.3803	5.4587	2.0034	30.1299
PhaT	9.3803	5.4587	2.0034	30.1299
PhaT_{β} [$\beta = 0.7$]	5.3098	2.0253	3.6596	30.1299
GCC_{ML}	9.7090	5.6576	9.4940	31.4286

(b) d = 5cm.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	11.1272	7.5228	37.5183	42.0779
Roth	28.6216	15.6002	651.1683	51.9481
SCoT	9.7162	5.7109	50.7186	29.6104
PhaT	9.7162	5.7109	50.7186	29.6104
PhaT_{β} [$\beta = 0.7$]	10.5180	6.2684	38.9283	34.5455
GCC_{ML}	13.0542	6.9829	106.2606	33.5065

(c) d = 10cm.

rors occur lie in the range [60^o, 130^o] where the array is expected to exhibit increased performance.

Distance of source from the array

All algorithms implemented in this work are formulated with the assumption of far-field radiation (incidence of plane waves). For a linear array, this assumption holds if the condition of equation (2.17) holds [54]

Table 3.14: Statistical metrics for the inter-element distance parameter of the setup. The values correspond to the results obtained for evaluation in range $[20^o, 160^o]$. (a) d = $3cm$, (b) d = $5cm$, (c) d = $10cm$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	8.6816	6.9509	4.3401	64.9180
Roth	12.8536	5.9388	120.8738	34.4262
SCoT	6.2539	4.1651	5.5150	30.1639
PhaT	6.2539	4.1651	5.5150	30.1639
PhaT_{β} [$\beta = 0.7$]	6.1615	3.8211	2.8564	27.2131
GCC_{ML}	6.2669	4.2189	5.8922	30.4918

(a) d = $3cm$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	6.4320	5.4724	1.7499	48.0132
Roth	11.3799	5.3918	85.2934	27.4834
SCoT	5.4056	3.4686	1.0678	23.5099
PhaT	5.4056	3.4686	1.0678	23.5099
PhaT_{β} [$\beta = 0.7$]	5.2557	3.4855	1.1198	23.1788
GCC_{ML}	6.1676	3.6927	9.9860	24.5033

(b) d = $5cm$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	11.8094	7.8829	46.9838	41.6667
Roth	26.0319	15.4115	555.2201	57.0000
SCoT	10.4736	6.4416	56.2021	38.0000
PhaT	10.4736	6.4416	56.2021	38.0000
PhaT_{β} [$\beta = 0.7$]	11.6942	7.3093	48.0781	38.3333
GCC_{ML}	14.3762	8.0434	127.1528	41.6667

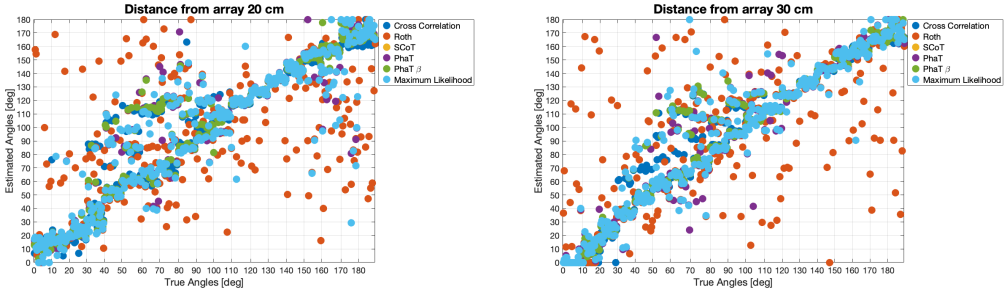
(c) d = $10cm$.

$$|r| > \frac{2L^2}{\lambda}$$

As can be deduced from this expression, the validity of the far-field approximation is frequency dependent. Since the acoustic events in this work are broad-band signals the condition will not hold for all frequencies in the spectrum. For the investigation of the effect in the estimation the distance of the source to the array has, three source-array distances were

tested, $l = 20cm$, $l = 30cm$ and $l = 40cm$. The array used for the recordings has inter-element distance $d = 10cm$.

The resulting estimates are shown in figures 3.10a and 3.10b for two of the tested distances. The results of the third distance coincide with those of *Figure 3.9c* and are not replicated here. Similarly, the corresponding metrics for the two distances are shown in *Table 3.15* while the metrics of the distance $l = 40cm$ can be seen in *Table 3.13*.



(a) $l = 20cm$. (b) $l = 30cm$.

Figure 3.10: Estimated angles of incidence of recorded signals. The inter-element distance of the array is $10cm$ and the distance of the source from the array (a) $l = 20cm$ and (b) $l = 30cm$.

A general tendency to decrease variance with increasing distance from the array is visible in the figures. Nevertheless, it is not easy to extract quantitative results. The metrics on the other hand do provide better insight and the increase in estimation performance is easy to detect, with all metric values improving the further away the source is from the array.

The findings support the assumption that the further the source is from the array, the better the far-field approximation holds and for a larger part of the spectrum. Since the formulation of the algorithms is based on such an approximation, it is expected to get better results for reasonably large distances.

Windowing functions

Up to this stage, the algorithms were evaluated without the application of a window. The next step tests the algorithms with four windowing functions being applied to the signals prior to being used for the estimation. The windows tested are the *Blackman*, *Gaussian*, *Hann* and *Kaiser*. The definitions of the windowing functions are given in *Appendix C*. The a parameter for the *Gaussian* window is set to 2.5 and for the *Kaiser* window to 3.

Table 3.15: Evaluation metrics for the tests made with varying distance of the source from the array. (a) $l = 20cm$ and (b) $l = 30cm$

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	22.0871	13.8428	177.1712	64.1026
Roth	43.2564	28.0080	1134.6689	79.2308
SCoT	21.8892	13.9362	259.6774	68.4615
PhaT	21.8892	13.9362	259.6774	68.4615
PhaT_{β} [$\beta = 0.7$]	19.4290	12.2884	148.8020	66.4103
GCC_{ML}	24.5952	15.6096	353.2052	71.5385

(a) $l = 20cm$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	16.8346	10.5436	84.7828	52.3077
Roth	38.3157	23.0457	1080.2894	64.1026
SCoT	15.9815	9.5843	179.4448	53.8462
PhaT	15.9815	9.5843	179.4448	53.8462
PhaT_{β} [$\beta = 0.7$]	14.9708	8.6616	101.5528	46.4103
GCC_{ML}	17.5466	10.2607	205.0957	55.3846

(b) $l = 30cm$.

The evaluation is performed on the data set corresponding to an array setup with inter-element distance $d = 10cm$ and distance of the source from the array $l = 40cm$.

The results of windowing the data prior to being fed to the GCC PhaT algorithm are shown in *Figure 3.11*. In addition to the windowing functions mentioned above, the *Rectangular* window is also shown for completeness. This function does not affect the data in any way and is the one used in all prior experiments. The metrics resulting from the data of the windowed signals are presented in *Table 3.16*.

Both the figure and the metrics show small deviations in the results provided by each window function. The *Rectangular* window seems to be a good choice with only the *Kaiser* outperforming it in some of the metric values. Nevertheless, the differences between all windows seem to be rather small, performance-wise.

The task of estimating the angle of arrival is benefited from the inclusion of all available data and this is a possible explanation for good performance of the *Rectangular* window, which does not discard any samples. The most probable explanation for *Kaiser*'s good performance may

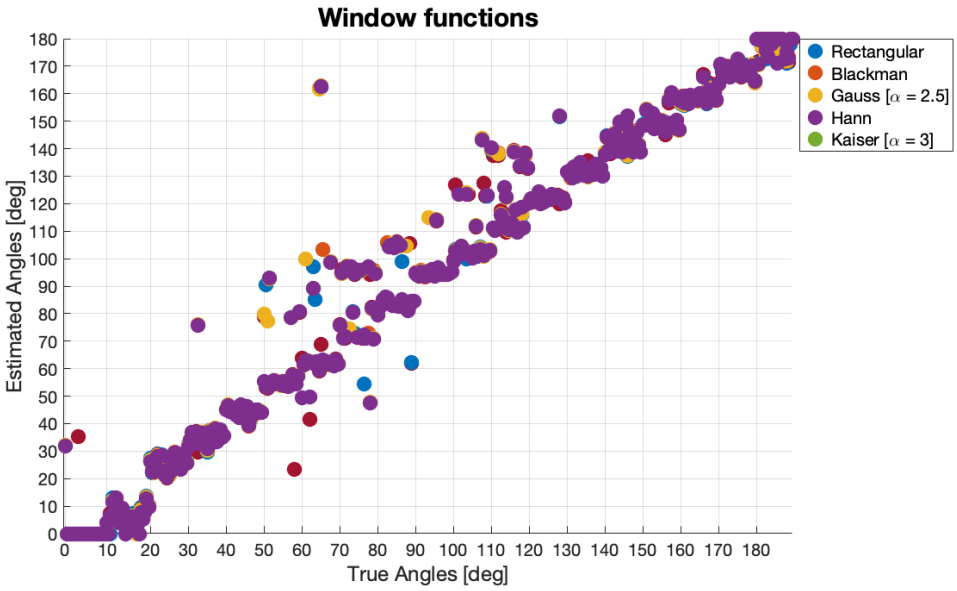


Figure 3.11: Resulted DoA estimates with the use of GCC PhaT on windowed data.

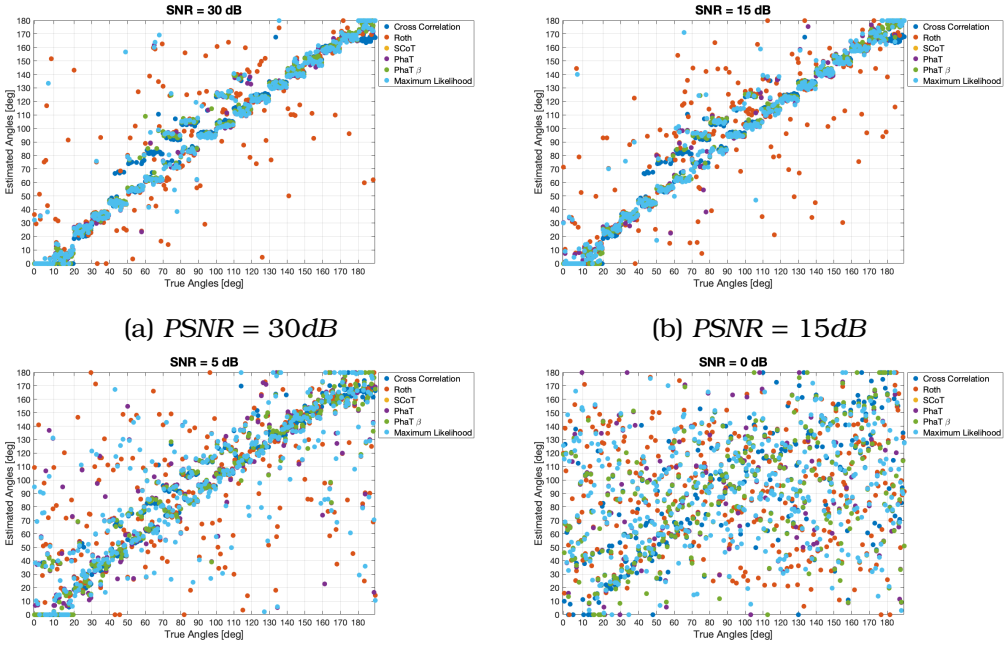
Table 3.16: Metric values for the windowing functions evaluation performed with the PhaT algorithm.

Window function	**RMSE**	**MAE**	**Variance**	**PE**
Rectangular	9.7162	5.7109	50.7186	29.6104
Blackman	12.9722	6.6735	108.5161	32.2078
Gauss [a = 2.5]	12.7185	6.4724	103.9669	31.4286
Hann	12.4699	6.3204	102.5226	31.1688
Kaiser [a = 3]	10.7221	5.6275	77.0343	28.8312

be given by the fact that, approximating the *Discrete-Prolate-Spheroidal-Sequence* (DPSS) window, it maximises the energy concentration on its main lobe (in the frequency-domain) and reduces information loss at the edges of the time frame [77].

Signal-to-Noise Ratio

Due to inability to add artificial background noise during the recordings, AWGN was added at the evaluation stage. Calculating the energy of only the signal portion of each audio block of the database is not easy since the acoustic events do not have the same duration.



(a) $PSNR = 30dB$ (b) $PSNR = 15dB$

(c) $PSNR = 5dB$ (d) $PSNR = 0dB$

Figure 3.12: Estimated angles of arrival of recorded signals. The inter-element distance of the array is $d = 10cm$ and the distance from the source $l = 40cm$. The PSNR conditions are (a) $PSNR = 30dB$, (b) $PSNR = 15dB$, (c) $PSNR = 5dB$, (d) $PSNR = 0dB$

The approach followed here is that the SNR condition imposed on the recorded signals resembles a "Peak SNR" (PSNR) where the peak value of the added noise has the specified relation to the signal's peak. From the two channels, the one with the highest peak was used to calculate the PSNR, resulting in the worst case for the given value. Additionally, both noise channels' amplitude was regulated jointly, making sure their joint maximum value provides the PSNR condition of interest. This however does not guarantee that the noise and signal peaks will be found in the same channel.

The values of PSNR used in these tests are $30dB$, $15dB$, $5dB$ and $0dB$. The elements of the database used are those corresponding to an array with inter-element distance $d = 10cm$ and distance of the source from the array $l = 40cm$.

Table 3.17: Evaluation metrics for the tests made with varying PSNR conditions. (a) $PSNR = 30dB$, (b) $PSNR = 15dB$, (c) $PSNR = 5dB$ and (d) $PSNR = 0dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	11.0128	7.4807	36.1527	42.0779
Roth	27.8156	15.0094	621.8993	50.3896
SCoT	11.0033	5.9561	73.4232	29.3506
PhaT	11.0033	5.9561	73.4232	29.3506
$PhaT_{\beta}$ [$\beta = 0.7$]	10.4437	6.2177	38.5382	34.5455
GCC_{ML}	16.7859	7.4903	213.1907	31.6883

(a) PSNR = $30dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	10.9374	7.4219	37.1663	42.3377
Roth	28.3725	16.7909	626.3096	54.5455
SCoT	12.3895	6.7969	93.3486	34.0260
PhaT	12.3895	6.7969	93.3486	34.0260
$PhaT_{\beta}$ [$\beta = 0.7$]	9.8102	5.9695	31.7641	32.9870
GCC_{ML}	15.5797	7.7213	168.6089	33.2468

(b) PSNR = $15dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	13.5046	9.1000	80.7864	49.3506
Roth	43.2024	28.3495	1238.5906	75.3247
SCoT	28.9614	17.1874	582.3508	67.0130
PhaT	28.9614	17.1874	582.3508	67.0130
$PhaT_{\beta}$ [$\beta = 0.7$]	16.6377	11.1924	163.6575	61.2987
GCC_{ML}	36.9059	23.2640	874.6932	72.2078

(c) PSNR = $5dB$.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	46.2900	32.4604	1146.8449	80.0000
Roth	66.9957	54.4508	1586.5478	92.7273
SCoT	61.1790	47.6337	1473.0283	91.1688
PhaT	61.1790	47.6337	1473.0283	91.1688
$PhaT_{\beta}$ [$\beta = 0.7$]	53.2768	39.5236	1304.8108	86.4935
GCC_{ML}	61.7302	48.3815	1467.6235	92.2078

(d) PSNR = $0dB$.

The estimated angles under the tested PSNR conditions are shown in figures 3.12a to 3.12d and the corresponding evaluation metrics in *Table 3.17*. Note that the results of the estimated angles for the case where no artificial noise is added (corresponding to $PSNR = \infty$) are shown in *Figure 3.9c* and the respective metrics in *Table 3.13*.

The figures and the metrics show similar results to those acquired from the Monte Carlo simulations. It seems that as the SNR (PSNR in this case) conditions deteriorate, so do the estimates with the case of $PSNR = 0dB$ showing no sign of correct estimates. Up to the case with $PSNR = 5dB$, it seems that some of the algorithms provide reasonably well results. More specifically, the Cross Correlation and PhaT$_\beta$ methods seem to be able to estimate the angle of arrival with a rather small MAE for such noisy conditions. The similar behavior of the algorithms is of course attributed to the fact that the β parameter regulates the PhaT$_\beta$ algorithm between the PhaT version and the "simple" Cross Correlation, which in this case shows better results than PhaT. Thus, PhaT$_\beta$, stands in between the two algorithms.

A remark to be made is that the crest factor of the noise is possibly smaller than that of the signal's, as the latter is impulsive in nature. The implication this may have is that the respective RMS values may correspond to lower SNR conditions than what the PSNR values indicate.

Pooling methods

The pooling method refers to the way the data of two successive frames are combined in order to extract a better estimate. It makes sense to be used only when the frames are tagged with positively identified acoustic events. Nevertheless, for the purpose of evaluating the methods two frames were used without first going through the detection step.

This method of integrating information of more than one frame is proposed by Blandin et al. [57]. The cross correlation functions of the two frames are calculated independently and then combined into one. The way they are combined is either with a *max* or a *sum* pooling function. The naming implies the way the lag values of both correlation functions are mapped to one value for the corresponding lag. The *max* method calculates the final correlation function as the maximum value of the two for each lag like

$$r_{y_1 y_2}^{\max}(\tau) = \max_{\tau} \sum_{f=1}^{M} r_{y_1 y_2}(t, \tau, f) \tag{3.6}$$

while the *sum* method adds the values together giving

$$r_{y_1 y_2}^{sum}(\tau) = \sum_{t=1}^{T} \sum_{f=1}^{M} r_{y_1 y_2}(t, \tau, f) \tag{3.7}$$

where in both expressions (3.6) and (3.7), the dependence of the cross correlation function on time is made explicit with t denoting the frame.

The set of data used for the evaluation of the two methods correspond to an array with inter-element distance $d = 10cm$ and distance of source from the array $l = 40cm$. The evaluation was performed with the use of only the PhaT algorithm. The estimated angles of the trials are shown in *Figure 3.13* and the metrics in *Table 3.18*.

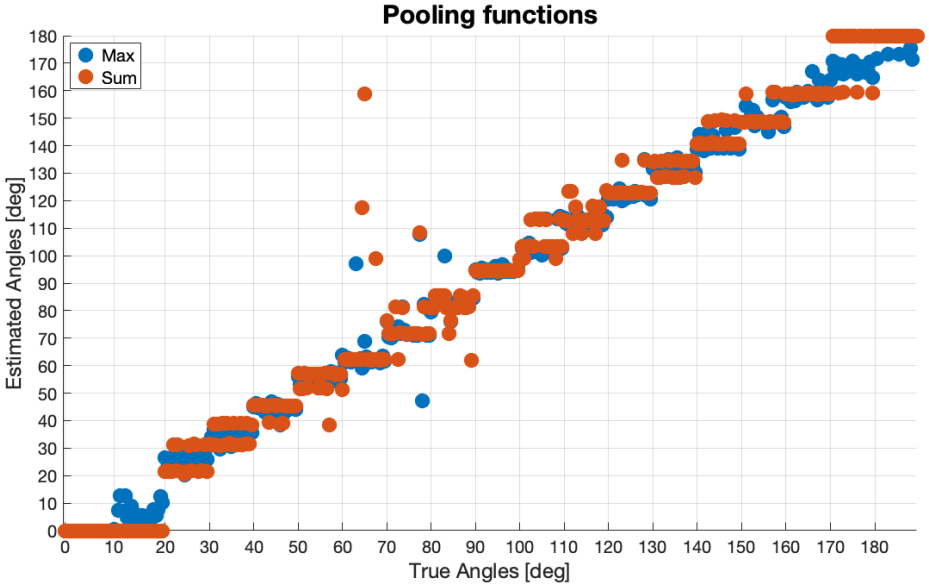


Figure 3.13: Estimated DoAs using two different pooling methods to combine information of two successive frames. The inter-element distance is $d = 10cm$, the distance of the source from the array centre $l = 40cm$ and the PhaT algorithm is used.

Table 3.18: Evaluation metrics for the tests of the pooling functions for the DoA estimation using two successive frames.

Pooling function	**RMSE**	**MAE**	**Variance**	**PE**
Max	5.7096	3.6505	20.2535	22.3377
Sum	8.6288	4.7945	50.2051	34.0260

It is clear that the results obtained with the *max* function are superior to those of *sum* in pretty much every aspect. It seems that the results obtained with the *sum* pooling function introduce a systematic error at the extremes of the scanning range while the *max* function manages to emend the errors up to a certain level resulting in better estimates (in a statistical manner) even at extreme angles of incidence.

What proves to be even more interesting, is the comparison of these results with those acquired for the PhaT algorithm with the use of only one frame of data. The latter are presented in tables 3.13 and 3.14. It seems that the additional information presented to the system increased the overall estimation efficiency by a considerable margin, halving the metric values in some cases. The metrics with the inclusion of the second frame are better, even for the case of using one frame but excluding the extreme angles from the evaluation. This proves to a certain degree that, at least PhaT, provides some efficiency in the data and it is possible to improve the estimation with the introduction of additional information.

Spatial Aliasing

As presented in *Section 2.3.1*, spatial aliasing can occur at frequencies higher than a value dependent on the array geometry. According to [3] the frequency of spatial aliasing (spatial Nyquist frequency) is given by equation (2.27)

$$f_c = \frac{c}{2d}$$

In order to investigate the effect spatial aliasing has on the estimated DoAs, the GCC algorithms have been slightly modified to calculate the cross correlation function with data reaching up to the maximum non-aliased frequency.

As this parameter depends on the geometry of the array, the evaluation was performed for all inter-element distances. The distance of the source from the array was kept constant to $l = 40cm$.

Figure 3.14 illustrates an example of the cross correlation function calculated with and without spatial aliasing allowed in the estimation. The algorithm used is PhaT, the array inter-element distance $d = 5cm$, the distance of the source from the array $l = 20cm$ and the true angle of incidence $\theta = 60^o$. The upper frequency limit for this setup corresponds to

$$f_{max} = \frac{c}{2d} = 3430Hz$$

With a frame size of 128 samples and sampling frequency of $44.1kHz$ the bin with the highest index has central frequency of

$$\lfloor \frac{3430Hz \cdot 128}{44100Hz} \rfloor \cdot \frac{44100Hz}{128} \approx 9 \cdot 344.53Hz \approx 3100.77Hz$$

where $\lfloor \cdot \rfloor$ denotes the *floor* function. It is instructive to note here that from the 64 frequency bins holding unique information, only 9 will be used. The information that is utilised constitutes only a small fraction of the available data.

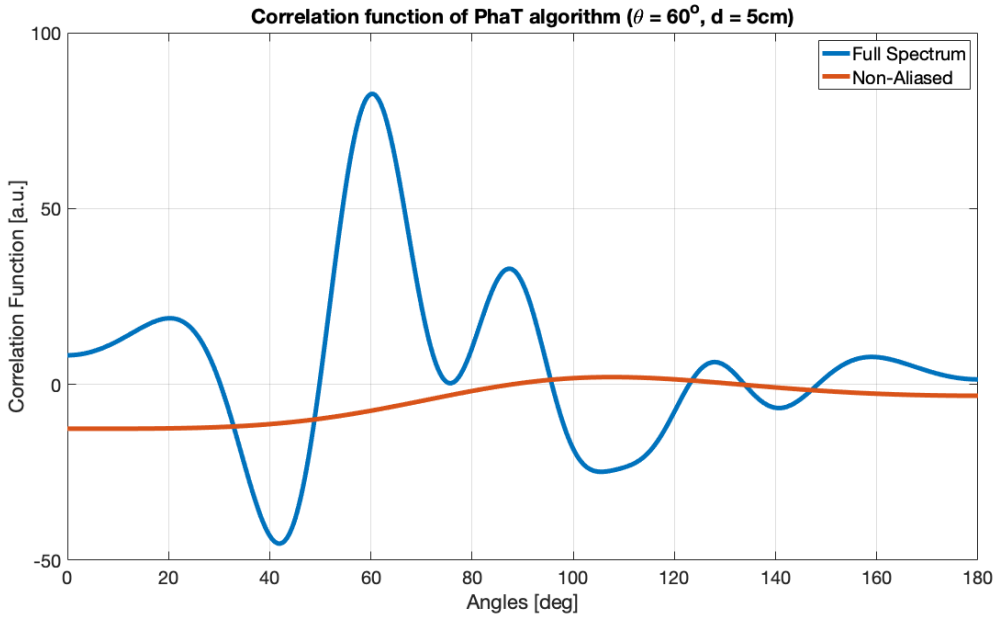
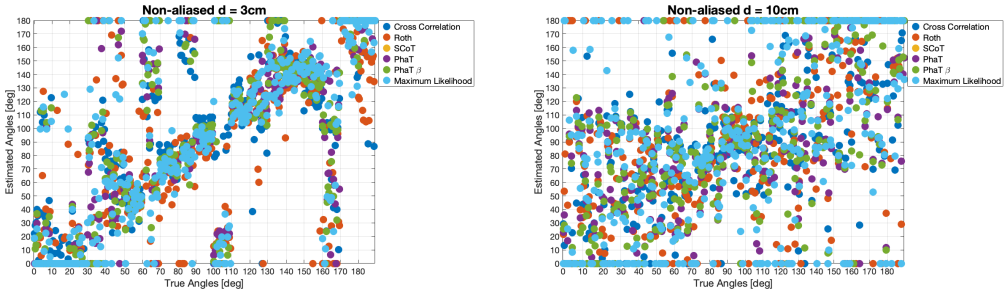


Figure 3.14: Estimated cross correlation functions with and without spatial aliasing allowed. Inter-element distance is $d = 5cm$, distance from source $l = 20cm$ and the true angle is $\partial = 60^o$. The algorithm used is PhaT.

In figures 3.15a and 3.15b the estimates shown are those corresponding to the exclusion of frequency components which incur spatial aliasing for inter-element distances $d = 3cm$ and $d = 10cm$. The metrics for all three inter-element distances are shown in *Table 3.19*. The corresponding "aliased" results are shown in figures 3.9a and 3.9c and their metrics in *Table 3.13*.

As it is obvious both in the figures and the statistical metrics, the exclusion of spatial aliased frequencies deteriorates the estimates' accuracy.



(a) Inter-element $d = 3cm$. (b) Inter-element $d = 10cm$.

Figure 3.15: DoA estimates resulting from the exclusion of spatial aliasing frequencies. Distance of source from the array is $l = 40cm$ and inter-element distances (a) $d = 3cm$ and (b) $d = 10cm$.

The results get worse with increasing inter-element distance. This is expected since the spatial aliasing frequency threshold is lowered when the distance d gets longer, resulting in progressively less information taking part in the estimation process.

Comparing the results of the non-aliased estimates with those that allowed for aliasing frequencies, in all cases the "aliased estimates" outperform the "non-aliased" and the greater the inter-element distance of the array becomes, the greater the difference is between the two.

3.5.4 Running times

The DoA algorithms are also tested for their speed of execution. The *Eckart* algorithm is not subjected to speed tests as it was dropped early form the evaluation process. The results provide indicative running times and it is expected to find implementations with optimised speed designed in high level compiled programming languages or low level execution environments.

Like in the tests performed with the detection algorithms, the result does not affect in any way the execution time of the algorithms, so the input data were noise signals drawn from a Gaussian PDF. Similar execution order to the detection measurements was implemented in an attempt to partially compensate possible optimisations resulting from repetitive code execution.

The algorithms were executed one million times each and the metrics are the *mean*, *median* and *mode* of the resulting timed runs, presented in *Table 3.20*. The final column shows what percentage of the total du-

Table 3.19: Evaluation metrics for the tests of DoA estimation made with frequencies resulting in spatial aliasing being excluded. (a) d = 3cm, (b) d = 5cm, (c) d = 10cm.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	31.1448	12.0877	974.8916	4.1558
Roth	51.1231	31.7997	1227.5627	78.4416
SCoT	44.1040	25.1833	1038.1375	68.0519
PhaT	44.1040	25.1833	1038.1375	68.0519
$PhaT_{\beta}$ [$\beta = 0.7$]	43.9471	27.3717	1119.5906	76.3636
GCC_{ML}	44.2045	25.8544	1228.7777	71.4286

(a) d = 3cm.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	50.2870	35.5035	1192.8383	87.0130
Roth	61.5969	42.2786	2267.6611	86.2338
SCoT	51.7913	34.7275	1578.7870	88.3117
PhaT	51.7913	34.7275	1578.7870	88.3117
$PhaT_{\beta}$ [$\beta = 0.7$]	46.8385	32.2751	1161.6908	85.1948
GCC_{ML}	55.7079	37.7301	1806.7426	86.4935

(b) d = 5cm.

Algorithm	**RMSE**	**MAE**	**Variance**	**PE**
Cross correlation	50.3316	36.7441	2055.8594	87.5325
Roth	59.3264	42.7932	2773.0030	88.0519
SCoT	54.5737	39.7111	2438.3607	87.2727
PhaT	54.5737	39.7111	2438.3607	87.2727
$PhaT_{\beta}$ [$\beta = 0.7$]	51.8698	37.5069	2239.5497	85.4545
GCC_{ML}	61.9745	45.3274	2899.5704	87.7922

(c) d = 10cm.

ration of a frame is occupied for the estimation process, calculated with the use of the mean value of the first column. For a system to be implementable under hard real time constraints the sum of these values with the corresponding ones of the detection algorithms must sum up to 100 at maximum.

The results show that all GCC algorithms occupy up to 75% of the available time at maximum, with the lower value being achieved by the

Table 3.20: Metrics of the running times of the four detection algorithms.

Algorithm	**Mean** [ms]	**Median** [ms]	**Mode** [ms]	**% of frame duration**
CC	2.1681	1.9647	1.9647	74.6976
Roth	1.8795	1.7684	1.7470	64.7563
SCoT	1.8517	1.7410	1.7260	63.7960
PhaT	1.8867	1.7553	1.7404	65.0027
PhaT_{β}	1.8523	1.7436	1.7340	63.8190
GCC_{ML}	1.9150	1.7847	1.7811	65.9769
CML	11.9022 [s]	11.4507 [s]	11.2553 [s]	410066.5161
UML	12.1465 [s]	11.7092 [s]	11.5719 [s]	418486.0633

$Phat_{\beta}$ algorithm on average but with only very small difference from all other algorithms. On the contrary, the ML algorithms show very long execution times reaching the order of $\sim$ 12 seconds. Despite their superior estimation capabilities, they are deemed inappropriate for use in a system running under hard real time constraints.

3.5.5 Summary

This section holds the results of all tests and trials performed for the evaluation of the TDoA estimation algorithms. Initially a short Monte Carlo simulation was run to test all the algorithms mentioned in *Section 3.5*. Some proved to be inappropriate for the task and were rejected at this stage.

In addition to performing quality tests on the algorithmic implementations used later in the evaluations with recorded signals, this stage provided information on the expected accuracy of the algorithms and other characteristics worth of attention. Most important is the reduction of resolution of all GCC algorithms with decreasing inter-element distance and increasing angle of incidence.

Implementation parameters of some variations of the PhaT algorithm were also evaluated. Finally, the superiority of the maximum likelihood formulation of the problem of DoA estimation was also confirmed with the UML algorithm providing exceptional results.

The remained algorithms were tested with recorded signals from the database created for this work. The tested parameters are a mix of physical setup parameters, inter-element distance and distance of source from the array and algorithmic variations such as the use of windowing functions, use of successive frames for the estimation task, the inclusion of noise and

rejection of spatial aliasing.

Of those parameters tested, the use of *Kaiser* windowing function and the inclusion of additional information on the estimation process provided consistently good results. The effect of inter-element distance seems to be a trade-off between angle resolution and excessive spatial aliasing, with medium distances of the order $d \approx 5cm$ providing good results. The findings suggest that the far-field approximation conditions also play an important role on the performance of the algorithms. The quality of the estimates is increasing for increasing distance of the source from the array, and the results become more consistent with decreased variance.

The last section provides information on the running times of the algorithms with those of the GCC family showing reasonably good results occupying at maximum three quarters of the available processing time. The ML algorithms, although very efficient in the estimation are very computationally expensive which is prohibitive for a real time system.

Chapter 4

System Implementation

This chapter holds the evaluation of the complete pipeline as depicted in *Figure 2.1*. The test case is the shown system acting as a digital virtual instrument controller. The orchestration includes two virtual instruments, a drum set and a piano, both controlled by the implemented sonic interaction system. The performance was conducted in two passes, one for each instrument.

An amateur percussion performer played both digital instruments making use of only the implemented system. A modern rock musical piece was chosen to be played and the basic drums track and "melodic-line" were performed while the user was listening to both the backing track of the piece as well as the sonic result of their actions.

The evaluation of the system is done both quantitatively and qualitatively. Quantitative evaluation differs from that presented in *Chapter 3*. For the detection task, metrics such as missed detection and false positive rates are used. The DoA estimation is evaluated solely based on the correctly estimated angles of arrival.

Quantitative results concern the impact the interface has on factors affecting the musical performance such as musical articulation and expressiveness.

4.1 Implementation

Setup and system parameters

The process is implemented entirely in MATLAB® using Audio System Toolbox© to acquire the microphone inputs in real-time. The sampling frequency is $44.1kHz$ and the frame size 128 samples resulting in overall latency of $\sim 2.9025ms$ for a single frame. No overlap and windowing are

used in this implementation as initial trials showed no improvement of the system in any aspect. In order to ensure both adequate angle resolution and estimation accuracy, inter-element distance is chosen to be $d = 7cm$.

Detection

CFAR is used for the detection process with data being provided by the first channel of the streamed signals. The first $1s$ (344 frames corresponding to $\sim 1s$) of the performance is used for the acquisition of the noise vector used in the threshold adaptation process.

Since said algorithm was unable to provide zero false positive rate in the evaluation stage, the need for threshold scaling is identified. Thus, a scaling factor was also implemented to affect the threshold after the adaptation process. The optimal value of the probability of false alarm (used for the adaptation of the threshold value) and scaling factor found to be 10^{-16} and $2.5 \cdot 10^2$ respectively. These values allow for optimisation of the system behavior under varying environmental conditions and different hardware, but in this implementation are kept constant throughout the performance.

Direction-of-Arrival

DoA estimation is performed with *PhaT* since it provided consistently good results in the evaluation. The DoAs are clustered in 20^o sectors with the central one covering the $[-10^o, 10^o)$[1] range. This led to an non-uniformly sectioned performance line (for more information see the *Performance* section below) but it was straight forward to be implemented from an algorithmic point of view. Variable angle sectors could lead to constant line segments if this is preferred in some applications. Five sectors were used in total using angles from -50^o to $+50^o$. All angles outside the edge values are clustered to the outer sectors. The microphone setup along with the sectors is shown in *Figure 4.1*.

It is shown in *Section 3.5.3* that using two frames with a pooling function to reach a DoA estimate significantly increases the performance of the system. On the other hand, the inclusion of more information inherently introduces additional latency. Whether this is important depends on many factors such as the tempo of a musical piece[2], the genre and play style just

[1]The range is closed on the low end and open on the upper end. Since each angle could belong to only one interval it was arbitrarily chosen to include the upper end of each interval to the next and the lower to the current.

[2]The same latency value may result in significantly greater rhythmic value offset for

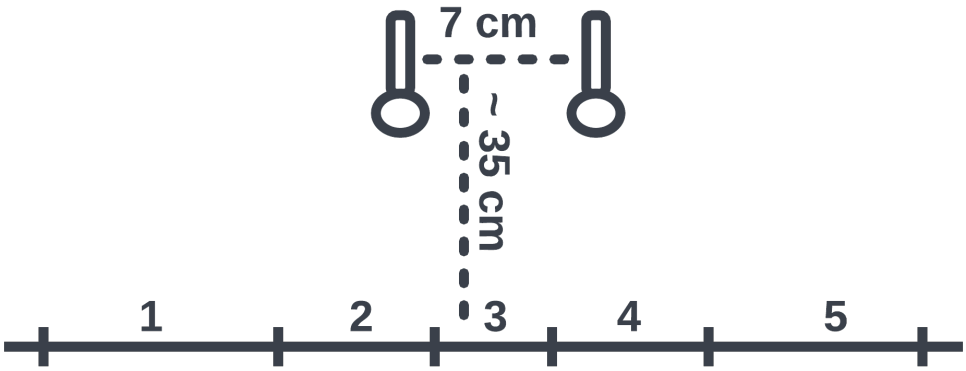


Figure 4.1: Physical arrangement of the complete HCI system implemented as a digital virtual instrument controller. The sectors are arbitrarily numbered from left to right.

to name a few [78]. In a study aiming to investigate the perception and production of temporal intervals, Ivry and Hazeltine show that the variance in production depends on the square of the duration of the interval [79]. The experiments were conducted with the participants being asked to tap the intervals, which is very close to the paradigm used in this interaction system (impulsive excitation) and the values presented are of the order of $15ms - 25ms$. They are larger by a considerable margin compared to the imposed latency of the system $\sim 5.8050ms$, when pooling is used (the duration of two frames).

It is not clear if latency and musical articulation are related or how and under which conditions one can affect the other. This is not the topic of this Thesis and will not be considered further, but it is evident that the timing variation due to the performer can outscore the latency introduced by the system under all conditions considered in our experiments. Thus it was decided to use pooling in this test case. From the two pooling functions, *max* is used due to achieving better scores in the evaluation stage.

Performance

The implemented sonic interaction system is used as a controller of two virtual musical instruments. The player struck two pieces of cutlery against each other to create the impulses that triggered the system. The

faster tempos.

sonic gestures where performed above a straight line[3] parallel to the array's axis at approximately 35*cm* from the array (measured as the length of the line forming right angles with both axes).

The fact that constant angle intervals are used in conjunction with this setup leads to the creation of uneven sectors. The physical arrangement can be seen in *Figure 4.1* where, further away from the array centre, greater line segments correspond to equal angle intervals. This may, or may not be ideal for specific performances but it suffices to demonstrate the use of the implemented interface.

A modern rock musical piece is chosen to be performed, titled "Seven Nation Army" from a band named "The White Stripes". Only the drums and the melody line with a piano sound are played in this evaluation, both by the same performer, an amateur percussionist. The two instruments are played in two "takes" several minutes apart. The "sound engine" is a very simple version of a polyphonic sampler able to play back five different samples with at least five "voices" being active concurrently. All audio samples are taken from *freesound.org* [80] and are used as a means to demonstrate the applicability of the control system rather than to create a uniform sonic feeling.

Both performances were done while the player was listening to the original song as well as the sounds generated from their performance. The duration of the demonstration piece is 1*min* and 25*s*.

4.2 Evaluation

This section holds the results of the performance. Both qualitative and quantitative results are presented. The former concern the technical aspects of the system with the most important being the detection rate, the ability to avoid false positive labels and the correct estimation of the angle of incidence.

The quantitative evaluation concerns the performance quality as a whole, based on more artistic criteria, such as the temporal variability and whether this seems to be affected by the system, ease of gesture generation and expressiveness.

[3] In this experiment the straight line was actually drawn to be able to evaluate the system in a qualitative way. In real-life performances the line may be imaginary or conducted on a circular line or area.

4.2.1 Quantitative evaluation

Detection

As mentioned in *Section 4.1*, the detection threshold was optimised to achieve zero FPR because it would be unacceptable for such a system to generate sound "spontaneously". On the contrary, missing a hit, although not ideal, has less severe consequences in a musical performance.

Figure 4.2 depicts a random sonic gesture, where it can be seen that the energy of an impulse may sustain for several hundred samples. The main, most prominent peak, which is of main interest to us, is well localised in time at the initial part of the sound. Being able to detect this peak and reject the rest of the impulse energy is crucial. Even more, it is highly beneficial to detect the initial peak and the first frame that follows to allow the use of pooling during the DoA estimation process. The system tuned with the values presented in the first part of *Section 4.1* achieved such a "two-frame" detection in many cases.

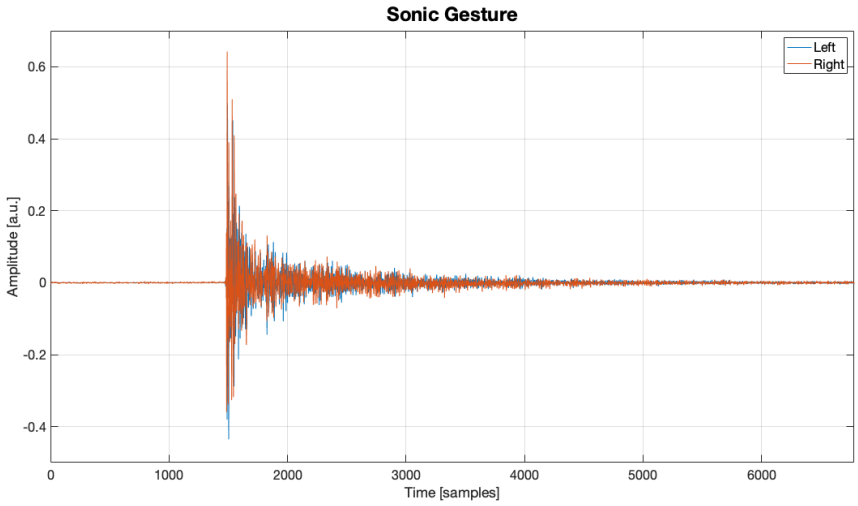


Figure 4.2: Time-domain representation of a random sonic gesture generated at the evaluation stage of the sonic interaction system.

Detection results are shown in *Table 4.1*. The values are shown for each instrument separately and for the performance of both jointly. All columns of the table contain results of "single-frame" detection, while the last column shows the cases where a "double-frame" detection occurred.

The results show perfect detection rate with a reasonably good "double-frame" detection rate. The latter could possibly improve with further tuning of the probability of false alarm and threshold scaling factor parameters.

Table 4.1: Detection results of the implemented sonic interaction system.

Inst.	**Detect**	**Miss**	**False Pos.**	**False Neg.**	**Double Detect** [%]
Drums	1.0000	0.0000	0.0000	0.0000	41.22
Piano	1.0000	0.0000	0.0000	0.0000	32.54
Total	1.0000	0.0000	0.0000	0.0000	36.59

Such good results were possible due to the very high SNR. Although in linear scale, *Figure 4.2* reveals a large difference between the noise and signal's peak. Such good conditions allow for a rather high threshold value providing good immunity to false alarms, which is the case here.

DoA estimation

In the DoA estimation task, use of rather broad angle sectors decreased the susceptibility to erroneous results. The choice of inter-element distance provided a good compromise between angle resolution and robustness against aliasing effects. Due to use of broad sectors a smaller distance could had been possibly used but as can be seen below, the chosen one provided excellent results.

All samples are triggered correctly with no hits being estimated to belong to wrong sections. The results of the estimation are shown in *Figure 4.3* where each category corresponds to a sample being triggered. The ordinate shows the deviation from the centre of each sector, in degrees, as a percentage.

Our findings from this performance come to good agreement with those from the evaluation of the algorithms. The estimates at the centre of the array (angles close to 90^o) are consistently closer to the centre of the sector. The further away from the broadside of the array we move, the more the mean estimate deviates from the central angle of the sector. The variance of all angle estimates seems to be very similar regardless of the true value of the central angle.

4.2.2 Qualitative evaluation

There seems to be lack of proper tools to reliably reach any conclusions regarding the quality of the implemented system on musical performance. It is for sure that the focus of this evaluation is on high level concepts such as expressivity and ease of use. It is very hard to separate the effect of the interface on these concepts as they are related to other factors such as the audio engine quality and the performer's skills. Nevertheless, the following

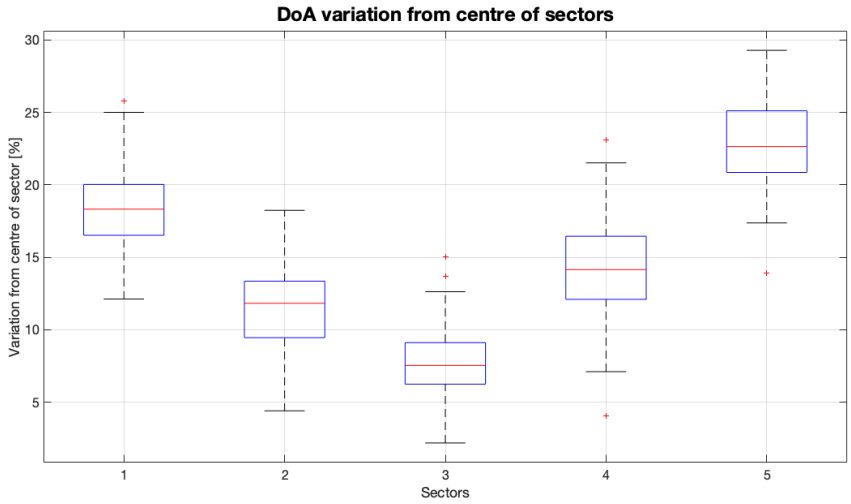


Figure 4.3: Variation of DoA estimates from the centre of each sector of the sonic interaction performance setup. The central mark indicates the median, the edges of the box the 25*th* and 75*th* percentiles. The whiskers extend to the most extreme values not considered outliers and the stars are the outlier values.

discussion can provide insight on possible improvements of the interface to allow easier use and better musical articulation.

The evaluation was performed in cooperation with the performer. They were asked to provide personal feedback on anything they felt that could be improved or was of very good quality.

The focus of the user was the absence of any kind of variability of the generated sound. This is partly attributed to the interface and partly to the audio engine. More samples could had been used and triggered quasi-randomly for each instrument. Since the latter is not the topic of this Thesis, it is not investigated further.

In many commercial drum machines/samplers, the "amplitude"[4] of the control can be used to trigger different samples corresponding to timbre variations resulting from execution with specific dynamics. The implemented interface does not provide any information related to "strength", thus such a feature is not available with this system. An alternative to bypass this issue is to use different sectors to represent various dynamics.

On the positive side of the feedback lies the ease of use of the system

[4]The term may refer to the parameter value controlling the amplitude of a generated sound.

due to its highly intuitive interface. It is instinctively easy for a person to create impulsive sounds to control a musical instrument, especially when it comes to percussive instruments like those used in this experiment.

A very important note made by the user was the fast response of the system. They stated that they didn't realise there was any kind of lag between the sonic gesture and the generated sound giving them the feeling they were performing in real-time a very responsive instrument.

The system setup was performed by the author of this work and the user didn't have the chance to go through the process. After verbally describing the procedure they seemed very satisfied with how easy it would be to perform the task themselves.

4.3 Summary

A complete sonic interaction system was implemented as a proof-of-concept using some of the algorithms evaluated in *Chapter 3* and the findings led to the optimisation of the system achieving perfect scores both on the detection and estimation tasks. Very high SNR conditions helped in both tasks and the formation of wide sectors provided high immunity to erroneous estimates.

The interface was used to control two virtual instruments to create a simple, short excerpt from a modern Rock song. It was performed by an inexperienced amateur percussion player with good results. There was a notable lack of variability of the generated sounds which is partially attributed to the interface because of the limited number of controls and range.

The user defined the interface as very easy to learn with a very mild learning curve. The control scheme of the interface is extremely intuitive to the average person due to use of very familiar sonic gestures. Moreover, setting up such a system is very easy and can easily become a "plug-and-play" solution.

Chapter 5

Conclusions and future work

5.1 Thesis summary

The focus of this Thesis was the implementation of a sonic interaction system triggered by impulsive acoustic gestures. The proposed implementation is formulated in terms of two steps; the detection of the impulsive acoustic events and the estimation of the Direction of Arrival with a pair of microphones.

Various microphone array setups and algorithms were evaluated with simulated signals and real recordings. All experiments were conducted for a wide range of Signal to Noise Ratio conditions with noise being added artificially. In most cases, simulation results came to good agreement with results obtained from real recordings.

One detection algorithm achieved perfect detection rate with the simulated signals but its superiority was not confirmed in the case of real data recordings. Similarly, a Maximum Likelihood algorithm tested for the DoA estimation provided very good results but was dropped from further evaluation since the processing time was many orders of magnitude larger than what is required to achieve real-time operation.

Additional algorithmic improvements were tested for both tasks. Threshold adaptation was implemented for the signal detection and found to provide marginally better results compared to using a constant threshold value. DoA estimation was shown to be greatly benefited from the use of information from successive frames. Interestingly enough, the inclusion of spatially aliased frequencies in the estimation process seemed to greatly improve the results.

The sonic interaction system was realised to act as the controller of a virtual digital instrument. The system showed excellent results on both

detection and estimation tasks. Moreover, the hard time constraints necessary for live performance were met and no frame overflows occurred. Possible improvements were identified, most of which are related to performance articulation.

5.2 Future work

The current work is focused on the implementation and technical evaluation of a sonic interaction system. This realisation forms an initial attempt to create a fully functional algorithmic pipeline. There are many aspects, either directly or indirectly related to the interface, that could be improved.

From a purely technical viewpoint, it is a matter of engineering to combine optimal solutions to each subtask into a complete system with superior performance characteristics. All parts of the system could be improved and many more features can be added. Many sophisticated algorithms exist that could achieve better performance overall for both detection and DoA estimation. It is a matter of algorithmic optimisation to achieve short processing times in order to fulfil the time constraints.

Modern, state of the art Bayesian methods have been implemented that can achieve extremely good results with high immunity to low SNR [1, 51, 81]. Additionally, Maximum Likelihood was shown to provide good results for the DoA estimation problem. Maximum likelihood algorithms showed promising behavior and the investigation of alternative formulations able to meet the time constraints could prove to be a great improvement. Implementations on high level programming languages such as C/C++ could benefit from various optimisation techniques and parallelisation schemes.

As already stated, sonic interaction is a multidisciplinary field and treating such systems from a pure technical viewpoint won't yield optimal results. Most probably, improvements that have the greatest impact come from this perspective. An important improvement in the context of controlling a musical application would be the ability of the system to detect more than one acoustic gestures simultaneously. This feature would allow for more expressive performances and control of polyphonic instruments.

Information provided by the interface is limited to the angle of incidence of an acoustic gesture. Additional information, such as the (peak) amplitude of the event could allow for more complex control schemes. In video games the intensity of the gesture could result in moves with greater extend, in musical performances a direct mapping to the amplitude of the

generated sound could be done and when used to control appliances such as the one presented in [17], gesture intensity could adjust the volume.

Finally, a broad extension of the system would be to allow for a larger dictionary of sonic gestures. Granular acoustic events could be an interesting addition to this dictionary. This type of triggering signal could prove to be useful in most fields where HCI systems find use. In musical performance sustained sounds could be generated with the spatio-temporal evolution of the gesture providing continuous update of some audio generation parameter. In video games, smooth control of movement could be achieved with additional acceleration controlled sensitivity. Using the system presented in [17] as an example, such a gesture could be mapped to the change of reproduction position.

The possible upgrades of this, or any other HCI system, are limited only by the ability of the designer to overcome technical difficulties and find ways to convey and map information from one domain to another. Experimentation is highly encouraged, only good results can come out of it!

Appendix A

List of abbreviations

AWGN Additive White Gaussian Noise
CC Cross Correlation
CDF Cumulative Distribution Function
CFAR Constant False Alarm Rate
CML Conditional Maximum Likelihood
CPU Central Processing Unit
DFT Discrete Fourier Transform
DPSS Discrete Prolate Spheroidal Sequence
DSP Digital Signal Processing
DoA Direction of Arrival
EM Expectation Maximisation
FA False Alarm
FFT Fast Fourier Transform
FN False Negative
FNR False Negative Rate
FP False Positive
FPGA Field Programmable Gate Array
FPR False Positive Rate
FS Full Scale
GCC Generalised Cross Correlation
HCI Human Computer Interaction
HPBW Half Power Beam Width
IID Independent Identically Distributed
LRT Likelihood Ratio Test
MAE Mean Absolute Error
ML Maximum Likelihood
MPU Micro-Processor Unit
OS Operating System

PDF Probability Density Function
PE Percentage Error
PSNR Peak Signal to Noise Ratio
PhaT Phase Transform
RAM Random Access Memory
RGB Red Green Blue
RIR Room Impulse Response
RMSE Root Mean Square Error
RT Reverberation Time
SCoT Smoothed Coherence Transform
SNR Signal to Noise Ratio
TDoA Time Difference of Arrival
TN True Negative
TP True Positive
UCA Uniform Circular Array
ULA Uniform Linear Array
UML Unconditional Maximum Likelihood

Appendix B

Evaluation metrics

All definitions here are presented without proof. They can be found in the literature, with references provided in the corresponding sections below.

B.1 Signal detection

The metrics used in the evaluation of the signal detection algorithms are the *Accuracy*, *Precision*, *Sensitivity* (or *Recall*), *Specificity* and *F-Score*. The abbreviations used are

- *TP*: True positive
- *TN*: True negative
- *FP*: False positive
- *FN*: False negative

and refer to the state of the results of the algorithms compared to the ground truth (or gold standard). All definitions of this section are taken from [82].

B.1.1 Accuracy

Accuracy is the most intuitive metric used in the evaluation of the detection algorithms. It is the sum of all correctly labelled results over the total number of trials. It is given by

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{B.1}$$

and shows the overall correctly decided cases over the total cases tested.

B.1.2 Precision

Precision is the ratio of correctly labelled positive results over the total cases that were labelled positive. It shows the fraction of the positive results that are correctly labelled and is given by

$$Precision = \frac{TP}{TP + FP} \tag{B.2}$$

B.1.3 Sensitivity (Recall)

Sensitivity is the metric that shows from the total of true positive cases how many were labelled correctly. It is the fraction of results labelled positive over the true positives and false negatives and is given by

$$Sensitivity = \frac{TP}{TP + FN} \tag{B.3}$$

B.1.4 Specificity

Specificity is the counterpart of *Sensitivity*. It shows how many cases were correctly labelled negatively out of the total true negative. It is given by

$$Specificity = \frac{TN}{TN + FP} \tag{B.4}$$

B.1.5 F-Score

F-Score is the (harmonic) mean of *Precision* and *Sensitivity*. *F-Score* is high when there is a balance between *Precision* and *Sensitivity*. For example if *Precision* is 0 and *Sensitivity* is 1, or the opposite, *F-Score* is 0. It is given by

$$F - Score = 2 \cdot \frac{Precision \cdot Sensitivity}{Precision + Sensitivity} \tag{B.5}$$

As can be seen, *F-Score* is low when one of the other two metrics involved is improved at the expense of the other.

B.2 Direction of arrival estimation

The angle of incidence estimation algorithms are evaluated using the *Root Mean Square Error* (RMSE), *Mean Absolute Error* (MAE), *Variance* and

Percentage Error (PE). The definitions come from different sources, stated at each subsection.

B.2.1 Root Mean Square Error

The *Root-Mean-Square Error* (RMSE) is given by the following formula [83]

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1}\left(\hat{\vartheta}_i - \vartheta\right)^2} \tag{B.6}$$

with N denoting the number of values used for the calculation of the error metric, $\hat{\vartheta}_i$ the estimate of the parameter of interest, ϑ the true value of the parameter and i the index of the estimated value.

It can be seen that the errors are weighted in a quadratic way with larger deviations having more impact on the metric, thus being quite sensitive to outliers [83].

B.2.2 Mean Absolute Error

The *Mean Absolute Error* is a more intuitive metric than RMSE. It is the average value of the error, without including the information of the direction of deviation (positive or negative error). It is calculated as [83]

$$MAE = \frac{1}{N}\sum_{i=0}^{N-1}\left|\left(\hat{\vartheta}_i - \vartheta\right)\right| \tag{B.7}$$

where N is the number of values used for the calculation of the metric, $\hat{\vartheta}_i$ the estimate of the parameter, ϑ the true value of the parameter and i the index of the estimated value.

This metric is less sensitive to outliers than the RMSE and leads to a better interpretation of the behavior of an estimator on the average [83].

B.2.3 Variance

For a random variable, the variance is the second central moment of the parameter, is denoted with σ^2 and given by [19]

$$\sigma^2 = E\left[(x[n] - \mu_x)^2\right] \tag{B.8}$$

where $x[n]$ denotes a realisation of the random variable x, μ_x is the mean of the random variable realisations and $E[\cdot]$ denotes the expectation operator.

Similarly, for a deterministic variable, or for the estimation of the variance from a number of measurements the following formula can be used [19]

$$\sigma^2 = \frac{1}{N-1}\sum_{n=0}^{N-1}(x[n]-\mu_x)^2 \tag{B.9}$$

with N being the number of samples used. Equation (B.9) provides an unbiased estimator for the sample variance. If in the same equation the sum is divided by N instead, the estimate is biased unless $N \to \infty$ and is called the *"empirical estimate"* [60]. The mean value is given by [60]

$$\mu_x = \frac{1}{N}\sum_{n=0}^{N-1}x[n] \tag{B.10}$$

B.2.4 Percentage Error

The *Percentage Error* (PE) is an easily interpretable metric and provides a good overview of the behavior of an estimator. It is similar to the *Accuracy* used in the evaluation of the detection algorithms. For its calculation a criterion is used in order to designate the estimate as erroneous and each estimate is compared to the criterion. The sum of the erroneous estimates over the total number of trials provides the value of the metric. It can be calculated as

$$PE = \left[\sum_{i=0}^{N-1}\left(\left|\hat{\partial}_i - \partial\right| > c\right)\right]\cdot 100 \tag{B.11}$$

where N denotes the total number of samples used for the calculation of the metric, $\hat{\partial}_i$ is the estimates of the parameter of interest, ∂ the true value of the parameter and c the criterion value. The binary function $[\cdot > \cdot]$ is given by

$$f(x,c) = \begin{cases} 1 & x > c \\ 0 & else \end{cases} \tag{B.12}$$

with x denoting the left operand and c the right.

Appendix C

Window functions

This appendix contains the definitions of the windowing functions used in the experiments of the DoA estimation. The way windows are applied to frames of audio samples is through an element-wise multiplication between the window function and the samples. All definitions are taken from [56].

C.1 Rectangular

The rectangular window is the "default" window when no processing is applied to the audio frame. It does not affect the data and is given by

$$w[n] = 1 \tag{C.1}$$

where n denotes the sample index.

C.2 Blackman

The form of the equation resulting in the *Blackman* windowing function is

$$w[n] = a_0 - a_1 \cos\left(\frac{2\pi n}{N}\right) + a_2 \cos\left(\frac{4\pi n}{N}\right) \tag{C.2}$$

with n denoting the sample index, a_0, a_1 and a_2 coefficients and N the total number of samples in a frame.

There are two different versions of the *Blackman* window, the **exact** and an approximation to the exact function. Most often the term *"Blackman window"* refers to the approximate one. The coefficients of the exact window are $a_0 = 7938/18608 \approx 0.42659$, $a_1 = 9240/18608 \approx 0.49656$,

and $a_2 = 1430/18608 \approx 0.076849$ while for the approximate, the values become $a_0 = 0.42$, $a_1 = 0.5$, and $a_2 = 0.08$ giving $a = 0.16$.

C.3 Gaussian

The Gaussian window has some very interesting and convenient properties. Its Fourier transform is also Gaussian and its logarithm produces a parabola, which is exploited in frequency estimation to perform nearly exact quadratic interpolation [84]. The function is of the form

$$w[n] = e^{-\frac{1}{2}\left(\frac{n-N/2}{\sigma N/2}\right)^2}, \qquad \sigma \leq 0.5 \tag{C.3}$$

where N denotes the number of samples in a frame, n the sample index and the standard deviation of the function is $\sigma N/2$ sampling periods.

An alternative formulation, which is the one used in this work is given by

$$w[n] = e^{\frac{1}{2}\left(a\frac{n}{(N-1)/2}\right)^2} = e^{-n^2/2\sigma^2} \tag{C.4}$$

with a being a parameter inversely proportional to the standard deviation σ and the sample index running from $-(N-1)/2$ to $(N-1)/2$. The exact correspondence of the a parameter to the standard deviation is $\sigma = (N-1)/(2a)$.

C.4 Hann

The Hann window is one of the most well known windows. The edges just touch zero and has good frequency characteristics with the sidelobes rolling-off at about $18dB$ per octave. The Hann function is

$$w[n] = 0.5\left[1 - \cos\left(\frac{2\pi n}{N}\right)\right] = \sin^2\left(\frac{\pi n}{N}\right) \tag{C.5}$$

with n denoting the sample index and N the number of samples in a frame.

C.5 Kaiser

The Kaiser window, sometimes refer to as *Kaiser-Bessel* is an approximation to the DPSS window using Bessel functions. Two very similar formulations of the function are

$$w[n] = \frac{I_0\left(\pi a \sqrt{1 - \left(\frac{2n}{N} - 1\right)^2}\right)}{I_0(\pi a)}, \qquad 0 \le n \le N \tag{C.6}$$

$$w[n] = \frac{I_0\left(\pi a \sqrt{1 - \left(\frac{2n}{N}\right)^2}\right)}{I_0(\pi a)}, \qquad -N/2 \le n \le N/2 \tag{C.7}$$

where I_0 is the zeroth order modified Bessel function of the first kind and a is a parameter that determines the trade-off between main lobe width and sidelobe levels. The main lobe width is given by $2\sqrt{1 + a^2}$ in units of Discrete Fourier Transform (DFT) bins.

Bibliography

[1] S. M. Kay, "Fundamentals of statistical signal processing, volume 2: Detection theory. 1998."

[2] A. Dufaux, "Detection and recognition of impulsive sound signals [ph. d. thesis]," *University of Neuchatel, Neuchatel, Switzerland*, 2001.

[3] J. Benesty, J. Chen, and Y. Huang, *Microphone array signal processing*. Springer Science & Business Media, 2008, vol. 1.

[4] H. L. Van Trees, *Optimum array processing: Part IV of detection, estimation, and modulation theory*. John Wiley & Sons, 2004.

[5] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE transactions on acoustics, speech, and signal processing*, vol. 24, no. 4, pp. 320–327, 1976.

[6] J. H. Carlisle, "Evaluating the impact of office automation on top management communication," in *Proceedings of the June 7-10, 1976, national computer conference and exposition*, 1976, pp. 611–616.

[7] M. McCool, J. Reinders, and A. Robison, *Structured parallel programming: patterns for efficient computation*. Elsevier, 2012.

[8] M. Jeon, R. Fiebrink, E. A. Edmonds, and D. Herath, "From rituals to magic: Interactive art and hci of the past, present, and future," *International Journal of Human-Computer Studies*, vol. 131, pp. 108–119, 2019.

[9] K. Berggren, Q. Xia, K. K. Likharev, D. B. Strukov, H. Jiang, T. Mikolajick, D. Querlioz, M. Salinga, J. R. Erickson, S. Pi *et al.*, "Roadmap on emerging hardware and technology for machine learning," *Nanotechnology*, vol. 32, no. 1, p. 012002, 2020.

[10] D. Morris and R. Fiebrink, "Using machine learning to support pedagogy in the arts," *Personal and ubiquitous computing*, vol. 17, no. 8, pp. 1631–1635, 2013.

[11] R. B. Shapiro, R. Fiebrink, and P. Norvig, "How machine learning impacts the undergraduate computing curriculum," *Communications of the ACM*, vol. 61, no. 11, pp. 27–29, 2018.

[12] K. E. Wolf and R. Fiebrink, "Personalised interactive sonification of musical performance data," *Journal on Multimodal User Interfaces*, vol. 13, no. 3, pp. 245–265, 2019.

[13] R. Fiebrink, "Machine learning education for artists, musicians, and other creative practitioners," *ACM Transactions on Computing Education (TOCE)*, vol. 19, no. 4, pp. 1–32, 2019.

[14] R. Fiebrink and M. Gillies, "Introduction to the special issue on human-centered machine learning," pp. 1–7, 2018.

[15] F. Camastra and A. Vinciarelli, *Machine learning for audio, image and video analysis: theory and applications*. Springer, 2015.

[16] ARM, "Latest npu adds to arm's ai platform performance, applicability, and efficiency." [Online]. Available: https://www.arm.com/company/news/2020/10/latest-npu-adds-to-arm-ai-platform-performance

[17] S. Vesa and T. Lokki, "An eyes-free user interface controlled by finger snaps," in *Proceedings of the 8th International Conference on Digital Audio Effects (DAFx-05)*, 2005, pp. 262–265.

[18] M. S. Puckette, M. S. P. Ucsd, T. Apel *et al.*, "Real-time audio analysis tools for pd and msp," 1998.

[19] V. Ingle, S. Kogon, and D. Manolakis, *Statisical and adaptive signal processing*. Artech, 2005.

[20] M. Müller, *Fundamentals of music processing: Audio, analysis, algorithms, applications*. Springer, 2015, vol. 5.

[21] B. Lartigue, "Passifolia - lightscape / soundscape," May 2021. [Online]. Available: https://www.creativeapplications.net/maxmsp/passifolia-lightscape-soundscape/

[22] F. Visnjic, "Halo – science of nature through the eyes and ears of a technological sublime," Jun 2021. [Online]. Available: https://www.creativeapplications.net/maxmsp/halo-science-of-nature-through-the-eyes-and-ears-of-a-technological-sublime/

[23] K. A. Beilharz, J. Jakovich, and S. Ferguson, "Hyper-shaku (border-crossing): Towards the multi-modal gesture-controlled hyper-instrument." in *NIME*, 2006, pp. 352–357.

[24] F. Rocha and J. Malloch, "The hyper-kalimba: developing an augmented instrument from a performer's perspective," in *Proc. Int. Conf. Sound and Music Computing (SMC)*, 2009.

[25] G. Marentakis and S. A. Brewster, "A study on gestural interaction with a 3d audio display," in *International Conference on Mobile Human-Computer Interaction*. Springer, 2004, pp. 180–191.

[26] "Olympia noise co." [Online]. Available: https://www.olympianoiseco.com/#

[27] "Mimu | home." [Online]. Available: https://mimugloves.com/

[28] G. Weinberg and S. Driscoll, "Toward robotic musicianship," *Computer Music Journal*, pp. 28–45, 2006.

[29] E. D. Battenberg, *Techniques for machine understanding of live drum performances*. University of California, Berkeley, 2012.

[30] N. Stefanakis, Y. Mastorakis, and A. Mouchtaris, "Instantaneous detection and classification of impact sound: Turning simple objects into powerful musical control interfaces." in *ICMC*, 2014.

[31] N. Stefanakis and A. Mouchtaris, "A multi-sensor approach for real-time detection and classification of impact sounds," in *2015 23rd European Signal Processing Conference (EUSIPCO)*. IEEE, 2015, pp. 2038–2042.

[32] "Scream flappy - control with your voice - apps on google play." [Online]. Available: https://play.google.com/store/apps/details?id=com.afkarstudios.flappyscreamgo&hl=en&gl=US

[33] S. Suri, "Flappy voice," Apr 2017. [Online]. Available: https://apps.apple.com/am/app/flappy-voice/id1219755200

[34] "Eighth note for android - apk download," Mar 2017. [Online]. Available: https://apkpure.com/eighth-note/com.gamefree.eighth

[35] "Kinect," Feb 2022. [Online]. Available: https://en.wikipedia.org/wiki/Kinect

[36] G. Galatas, G. Potamianos, and F. Makedon, "Audio-visual speech recognition incorporating facial depth information captured by the kinect," in *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*. IEEE, 2012, pp. 2714–2717.

[37] ——, "Audio-visual speech recognition using depth information from the kinect in noisy video conditions," in *Proceedings of the 5th International Conference on PErvasive Technologies Related to Assistive Environments*, 2012, pp. 1–4.

[38] "Azure kinect dk – develop ai models: Microsoft azure." [Online]. Available: https://azure.microsoft.com/en-us/services/kinect-dk/#overview

[39] F. Nonaka, S. Kawashiri, and A. Kawakami, "Next-generation rheumatoid arthritis specialized telemedicine enabled by iot and ai," *Impact*, vol. 2021, no. 8, pp. 58–60, 2021.

[40] Y. Arslan, "A new approach to real time impulsive sound detection for surveillance applications," *arXiv preprint arXiv:1906.06586*, 2019.

[41] T. Ahmed, M. Uppal, and A. Muhammad, "Improving efficiency and reliability of gunshot detection systems," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 513–517.

[42] "Photron sa-z." [Online]. Available: https://www.mctcameras.com/product/photron-fastcam-sa-z/

[43] F. Gini, A. Farina, and M. Greco, "Selected list of references on radar signal processing," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 1, pp. 329–359, 2001.

[44] M. Skolnik, *Radar Handbook 3rd ed. McGraw-Hill*, 2008.

[45] H. Jeon, S. Kim, L.-Y. Kim, H.-Y. Lee, and H. Yoon, "Reliability measure for sound source localization," *IEICE Electronics Express*, vol. 5, no. 6, pp. 192–197, 2008.

[46] J. M. Villadangos, J. Ureña, J. J. García-Domínguez, A. Jiménez-Martín, Á. Hernández, and M. Pérez-Rubio, "Dynamic adjustment of weighted gcc-phat for position estimation in an ultrasonic local positioning system," *Sensors*, vol. 21, no. 21, p. 7051, 2021.

[47] P. M. Schultheiss and H. Messer, "Optimal and suboptimal broadband source location estimation," *IEEE transactions on signal processing*, vol. 41, no. 9, pp. 2752–2763, 1993.

[48] M. Li, Y. Lu, and B. He, "Array signal processing for maximum likelihood direction-of-arrival estimation," *Journal of Electrical and Electronic Systems*, vol. 3, no. 1, p. 117, 2013.

[49] P. Stoica and K. C. Sharman, "Maximum likelihood methods for direction-of-arrival estimation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 7, pp. 1132–1143, 1990.

[50] P. Stoica and A. Nehorai, "Performance study of conditional and unconditional direction-of-arrival estimation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 10, pp. 1783–1795, 1990.

[51] H. L. Van Trees, *Detection, estimation, and modulation theory, part I: detection, estimation, and linear modulation theory*. John Wiley & Sons, 2004.

[52] H. Urkowitz, "Energy detection of unknown deterministic signals," *Proceedings of the IEEE*, vol. 55, no. 4, pp. 523–531, 1967.

[53] J. Moragues, L. Vergara, J. Gosalbez, T. Machmer, A. Swerdlow, and K. Kroschel, "Background noise suppression for acoustic localization by means of an adaptive energy detection approach," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2008, pp. 2421–2424.

[54] B. D. Steinberg, *Principles of aperture and array system design: Including random and adaptive arrays*, 1976.

[55] M. R. Bai, J.-G. Ih, and J. Benesty, *Acoustic array systems: theory, implementation, and application*. John Wiley & Sons, 2013.

[56] J. G. Proakis, *Digital signal processing: principles algorithms and applications*. Pearson Education India, 2001.

[57] C. Blandin, A. Ozerov, and E. Vincent, "Multi-source tdoa estimation in reverberant audio using angular spectra and clustering," *Signal Processing*, vol. 92, no. 8, pp. 1950–1960, 2012.

[58] G. C. Carter, "Coherence and time delay estimation," *Proceedings of the IEEE*, vol. 75, no. 2, pp. 236–255, 1987.

[59] B.-h. Wang, Y.-l. Wang, H. Chen, and Y. Guo, "Generalized maximum likelihood algorithm for direction-of-arrival estimation of coherent sources," *Frontiers of Electrical and Electronic Engineering in China*, vol. 1, no. 1, pp. 42–47, 2006.

[60] A. Papoulis and S. U. Pillai, *Probability, random variables, and stochastic processes*. Tata McGraw-Hill Education, 2002.

[61] M. I. Miller and D. R. Fuhrmann, "Maximum-likelihood narrow-band direction finding and the em algorithm," *IEEE transactions on acoustics, speech, and signal processing*, vol. 38, no. 9, pp. 1560–1577, 1990.

[62] P. Stoica and A. B. Gershman, "Maximum-likelihood doa estimation by data-supported grid search," *IEEE Signal Processing Letters*, vol. 6, no. 10, pp. 273–275, 1999.

[63] K. Sharman and G. McClurkin, "Genetic algorithms for maximum likelihood parameter estimation," in *International Conference on Acoustics, Speech, and Signal Processing,*. IEEE, 1989, pp. 2716–2719.

[64] "Scarlett 2i2." [Online]. Available: https://focusrite.com/en/usb-audio-interface/scarlett/scarlett-2i2

[65] "Reaper | audio production without limits." [Online]. Available: http://www.reaper.fm/

[66] "Nexo systems," Jan 2022. [Online]. Available: https://www.nexo-sa.com/systems/legacy-systems/

[67] "Nexo nxamp4x4 powered controller," Dec 2021. [Online]. Available: https://www.nexo-sa.com/products/nxamp4x4/

[68] "Mm 1." [Online]. Available: https://north-america.beyerdynamic.com/mm-1.html

[69] A. Farina, "Simultaneous measurement of impulse response and distortion with a swept-sine technique," in *Audio Engineering Society Convention 108*. Audio Engineering Society, 2000.

[70] ——, "Advancements in impulse response measurements by sine sweeps," *Journal of The Audio Engineering Society*, 2007.

[71] E. British Standard, "Iso 3382-2: 2008," *Acoustics-Measurement of room acoustic*, 2008.

[72] N. M. Papadakis and G. E. Stavroulakis, "Low cost omnidirectional sound source utilizing a common directional loudspeaker for impulse response measurements," *Applied Sciences*, vol. 8, no. 9, p. 1703, 2018.

[73] M. R. Schroeder, "New method of measuring reverberation time," *The Journal of the Acoustical Society of America*, vol. 37, no. 6, pp. 1187–1188, 1965.

[74] A. Ramamurthy, H. Unnikrishnan, and K. D. Donohue, "Experimental performance analysis of sound source detection with srp phat-β," in *IEEE Southeastcon 2009*. IEEE, 2009, pp. 422–427.

[75] H. Ji, X. Cui, Y. Gao, and X. Ge, "3-d ultrasonic localization of transformer patrol robot based on emd and phat-β algorithms," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–10, 2021.

[76] K. Mahdinejad and M. Z. Seghaleh, "Implementation of time delay estimation using different weighted generalized cross correlation in room acoustic environments," *Life Science Journal*, vol. 10, pp. 846–851, 2013.

[77] "Slepian or dpss window." [Online]. Available: https://ccrma.stanford.edu/~jos/sasp/Slepian_DPSS_Window.html

[78] "Timing variations of percussive instrument performance," Dec 1969. [Online]. Available: https://music.stackexchange.com/questions/122068/timing-variations-of-percussive-instrument-performance?noredirect=1#comment221957_122068

[79] R. B. Ivry and R. E. Hazeltine, "Perception and production of temporal intervals across a range of durations: evidence for a common timing mechanism." *Journal of Experimental Psychology: Human Perception and Performance*, vol. 21, no. 1, p. 3, 1995.

[80] [Online]. Available: https://freesound.org/

[81] M. E. Terzakis, "Spatial analysis using the directional impulse response measurements in room acoustics," Ph.D. dissertation, 2018.

[82] J. Patterson and A. Gibson, *Deep learning: A practitioner's approach*. " O'Reilly Media, Inc.", 2017.

[83] C. J. Willmott and K. Matsuura, "On the use of dimensioned measures of error to evaluate the performance of spatial interpolators," *International Journal of Geographical Information Science*, vol. 20, no. 1, pp. 89–102, 2006.

[84] M. Gasior and J. Gonzalez, "Improving fft frequency measurement resolution by parabolic and gaussian interpolation," CERN-AB-Note-2004-021, Tech. Rep., 2004.