Acoustic Solutions for Door Stations

Johan Nisula
ada09jni@student.lu.se

Sebastian Krill
atp09skr@student.lu.se

Department of Electrical and Information Technology
Lund University

Advisor: Mikael Swartling

June 9, 2015

Printed in Sweden
E-huset, Lund, 2015

Abstract

This thesis investigates how the audio quality in a door station can be improved by using multiple microphones and implementing beamforming. The concept of beamforming is explained, and two beamforming algorithms are implemented. These are tested with different microphone configurations in both simulated and real environments. Three already implemented solutions for single microphones are also tested.

The performance of different microphone configurations is analysed, and the beamforming algorithms are compared to the single microphone solutions. Finally a solution for the application is proposed.

Acknowledgements

We would like to thank our advisor Dr. Mikael Swartling for his help, support and for allowing us to use his software during the work on this thesis. We would also like to thank our examiner Dr. Nedelko Grbić for his guidance throughout the work process. Finally we want to thank Johan Adolfsson at Axis Communications for his help and support.

Johan Nisula and Sebastian Krill
June 9, 2015

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter the background and goal of this thesis are presented, with a description of the intended application. A short review of some existing research, with examples of other applications, is made. Finally, the basic concepts, in the context of the application, are thoroughly explained.

1.1 Background

Communication is one of the most important and time consuming activities in today's society, and the simplest and most natural way to communicate is through speech. Therefore the importance of speech quality and intelligibility in communication devices become evident. In order to achieve high quality communication, the speech signal that is intended to be transmitted should be the signal that is actually received. The main problem is that speech is a complex signal with a wideband frequency range, and as devices are built to record speech they will inevitably record other sounds within that range as well. These undesirable sounds are referred to as noise, and this is the problem scientists in the field of speech enhancement try to solve.

Due to the rapid development of computers and devices such as mobile phones there has been a large amount of research done on speech enhancement. The result of this is a collection of algorithms, designed to reduce the noise in a signal from a single microphone, while attempting to preserve the speech. These can be divided into three categories [1], which are described in section 2.4. There is a problem that impairs the performance of these algorithms, causing them to distort the speech as they reduce the noise. The problem is to be able to distinguish the speech from the noise so that only the noise can be removed, without distorting the speech. In order to achieve this, one must have access to information about the environment where the device will be used [1].

The need for knowledge about the intended environment stems from the fact that there are different sources of noise in different environments. For example, if the intended environment is a car, there might be noise from the engine, tires and wind. If, on the other hand, the environment is the inside of an airport, the noise might be multiple unwanted surrounding speech sources. In the car environment a voice detection solution could work because of the difference between the noise and the speech, but the same solution might fail at the airport because of the similarity between the speech and noise.

Another important piece of information, required to design a well functioning algorithm, is the signal-to-noise ratio (SNR) in an environment [1]. SNR is the ratio between the desired signal power and the noise signal power. Information about the SNR is important because it shows which signal range the application will operate in, and thus what range of noise the algorithm should be able to suppress.

As the available computational power have become greater, a new approach to solving the problem of noise have been adopted. By using several microphones, placed at different positions, one can gain access to more than one signal. Because of the fact that it takes time for a sound to travel, the signals will contain spatial information about the same sound. One may exploit this information through different algorithms to, for example, gain knowledge about from which direction the sound was emitted. This technique have been used for a long time in different narrowband applications, such as radar [2].

The concept is known as beamforming, and consists of using the time it takes a wave to propagate between sensors at different locations, together with knowledge of the wavelength, to gain spatial information about the wave. This information can be used to combine the sensor signals, so that the resulting signal may amplify or dampen a wave through constructive and destructive interference respectively. This means that the technique may be used with any type of wave that can be registered by a sensor, and thus it can be used with sound waves. Beamforming will be thoroughly explained in section 1.2.

When beamforming is applied to a narrowband signal, all the waves being manipulated have approximately the same properties. Therefore, an algorithm only have to be applied to one type of wave. With a wideband signal, a new layer of complexity is added. Since the signal contains a wide band of frequencies, there are many waves, with different properties, to manipulate. This results in complexity rising as the width of the frequency band increases. With the increase of computational power, the complexity becomes easier to handle. Because of this, the field of speech enhancement have seen new solutions, tackling the problem of noise with the help of beamforming.

1.1.1 Axis Communications

Axis is one of the world's leading manufacturers of network camera solutions. They provide a variety of security products, ranging from network cameras to physical access control systems. As they are looking to expand their market into the area of door stations, the importance of high quality sound increases. This is due to the fact that a door station is a communication device, and as Axis are looking to produce a high quality product they require high quality communication.

A door station is situated by an entry, where the people entering need to be cleared by another person, situated at, for example, a reception. The station has a camera, a speaker and a microphone. Examples of locations where it might be situated are loading bays and airports, and the environment could potentially have a lot of noise. For the door station to be able to transmit a high quality communication signal, it should be able to reduce the noise with as little distortion of the speech as possible.

The current version of the Axis door station is the A8004-VE, and an image of it can be seen in figure 1.1. Axis is currently using software from an external vendor to achieve high quality sound in the door station but, as they are always looking to improve the quality of their products, additional solutions are being investigated.

This thesis attempts to investigate how the sound quality in the Axis door station may be improved by the use of multiple microphones. Limitations on microphone configurations and sampling rate are based the A8004-VE. The unit is 14 cm wide, with a height of 26 cm, and has a maximum sampling rate of 16 kHz.

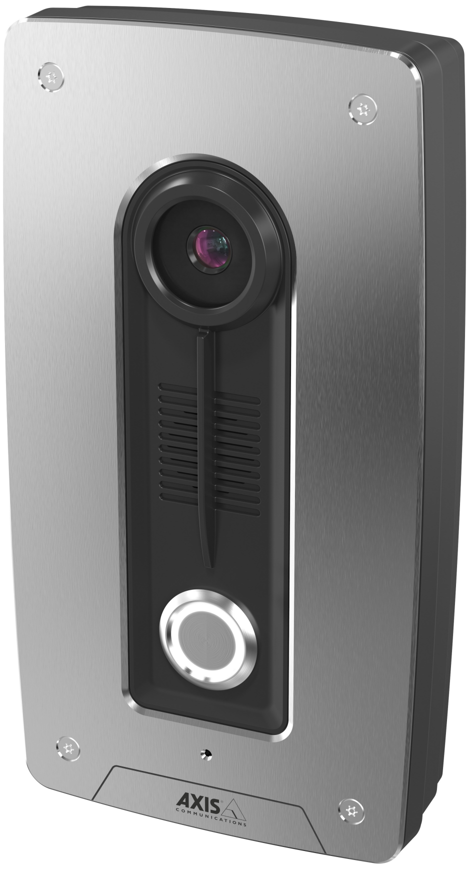


Figure 1.1: Axis A8004-VE door station

1.1.2 Thesis Goal

The purpose of this thesis is to investigate how sound quality can be improved through the implementation of spatial filtering, considering the specific application of a door station. This is done through the implementation of two beamforming algorithms, with the final goal of presenting a suitable solution for the application.

1.1.3 History and Existing Research

Because the purpose of this thesis is to examine beamforming with respect to speech enhancement, this section will only address the history and research in the field of acoustic beamforming. The term refers to the use of beamforming to do spatial filtering on sounds, either to localize a sound source, or to focus on a source and suppress sounds from other locations.

The first real application in this field was an acoustic telescope, used to locate sound sources on jet engines, proposed by John Billingsley in 1974 [3]. The purpose of the telescope was to find sources of noise on the engine used in the Concorde airplane. Finding noise sources has been a major part of the research field in acoustic beamforming. One application that has been studied is that of finding noise sources on moving objects. There has been research done using beamforming to analyze the sound of trains and airplanes to determine from what part of the vehicle noise is emitted [3].

An application that indirectly relates to the subject of this thesis is that of noise measurements in wind tunnels. Because of the high background noise in a wind tunnel it is hard to measure the noise created by the object being tested. There has been research done on how to reduce the background noise so that the desired noise may be measured [3].

Another research subject that is not directly related to this thesis, but to the physical design of a door station, is that of the combination of beamforming and a camera. In 1999 a so called acoustic camera was presented by GFaI [3]. The device was a camera with a number of microphones, which implemented beamforming. This was done in order to create an overlay onto the image of the camera, displaying a heatmap of the sounds in the image.

Research that directly relates to this thesis has been performed by Nedelko Grbić in [4], where the two beamforming algorithms that are implemented in this thesis are tested and evaluated in a hands-free application. Grbić also implements and evaluates an adaptive beamforming algorithm, which is compared to the other algorithms, together with other solutions for speech enhancement.

1.2 Beamforming

The word 'beamforming' refers to using multiple receiving or transmitting elements, e.g. microphones or antennas, to achieve a directionally dependent gain of the transmitted or received signal. In the receiver case, which is the one considered in this thesis, this is achieved by weighting and summing all of the received signals from the different receiving elements to create a single output signal.

The basic principle of beamforming is that of destructive and constructive interference of waves. Two sinusoids with the same frequency will, when added together, interfere so that the sum is a sinusoid with the same frequency. Depending on the phase difference between the sinusoids, the amplitude of the sum will be dampened or amplified. In the receiver case, a signal will be received with different time delays at the elements of the array depending on the location it originated from, which allows for directional selectivity.

Using multiple microphones, the sound travelling from a talking person to the microphones can be preserved by having unit gain for this direction. Other directions where potential noise may be coming from can be attenuated, leading to an improvement in speech intelligibility.

This section will explain the basics of beamforming, and what is required to implement it considering ideal microphones. The coordinate system that will be used is shown in figure 1.2.

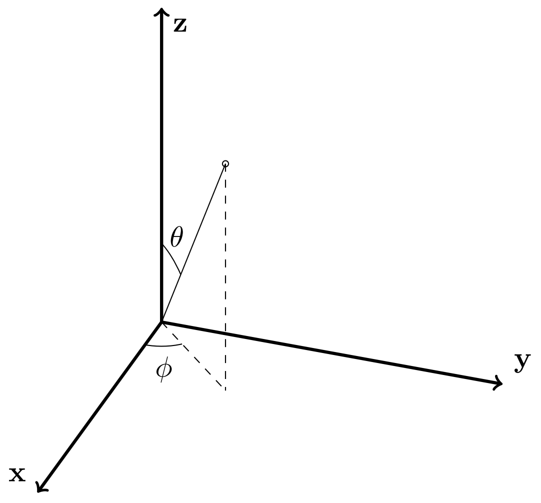


Figure 1.2: Angles in the spherical coordinate system

1.2.1 Microphone Arrays

To perform beamforming, more than one microphone is needed. The collection of microphones used is referred to as a microphone array. With more than one microphone comes the possibility of different spatial arrangements. The simplest array is created by placing the microphones on a straight line, which will be referred to as a linear array. With more than two microphones other configurations are certainly possible, such as planar arrays where the microphones are confined to lie in the same plane, or three-dimensional arrays where the microphones could be placed freely. In this thesis only linear and planar microphone arrays are considered.

1.2.2 Narrowband Beamforming

As mentioned before, beamforming relies on wave interference to produce a directionally dependent gain. Since two sinusoids with different frequencies cannot amplify or dampen each other consistently it is natural to study what will happen to a narrowband signal, containing only a single frequency, when it is processed by the microphone array.

Sound propagation

Sound is the result of a periodical perturbation of the pressure in a gas, i.e. waves travelling through the gas medium. Under the assumptions of a homogeneous and non-viscous medium its propagation is governed by the wave equation

$$\nabla^2 x(t,\mathbf{r}) - \frac{1}{c^2}\frac{\partial^2 x(t,\mathbf{r})}{\partial t^2} = 0 \tag{1.1}$$

where $x(t,\mathbf{r})$ is the sound pressure at location $\mathbf{r} = \begin{bmatrix} x & y & z \end{bmatrix}^T$ and time t. The equation allows for both plane wave and spherical wave solutions.[5]

The solution to the wave equation for a plane wave is known as

$$x(t,\mathbf{r}) = Ae^{j(\omega t - \mathbf{k}\cdot\mathbf{r})} \tag{1.2}$$

where A is the amplitude of the wave, $\omega = 2\pi f$ with the wave frequency f and $\mathbf{k}$ is the so called wavenumber vector, which describes the speed and direction of the wave, given by

$$\mathbf{k}(\phi,\theta) = \frac{2\pi}{\lambda}\begin{bmatrix} \sin(\theta)\cos(\phi) & \sin(\theta)\sin(\phi) & \cos(\theta) \end{bmatrix}^T \tag{1.3}$$

where λ is the wavelength, related to the frequency and propagation speed c by $\lambda f = c$ [5].

In this thesis only plane waves with propagation speed 330 m/s will be considered, and therefore the equation for spherical waves is omitted. Figure 1.3 shows a comparison between spherical and plane waves as they propagate towards a microphone array with two microphones. The full lines are the wavefronts, the dotted lines are the subsequent waves, separated by the wavelength λ, and the points on the dashed lines are the microphones.

The reason for considering only plane waves is that as a wave source, emitting spherical waves, moves further away from a sensor the incident waves begin to appear as plane waves. Why this occurs can be seen in the magnified parts of figure 1.3. An expression that governs when spherical waves can be considered plane for an evenly spaced linear array is [5]

$$|r| > \frac{2(Nd)^2}{\lambda} \tag{1.4}$$

where $|r|$ is the distance between the array and the sound source, N is the number of microphones and d is the distance between the microphones.

As an example, the distance between an array with a length of 13 cm and a source emitting sound with a frequency of 2000 Hz should be less than 26 cm for the waves to begin to appear as spherical. In the application of a door station this can be viewed as a short distance, and therefore only plane waves are considered in this thesis.

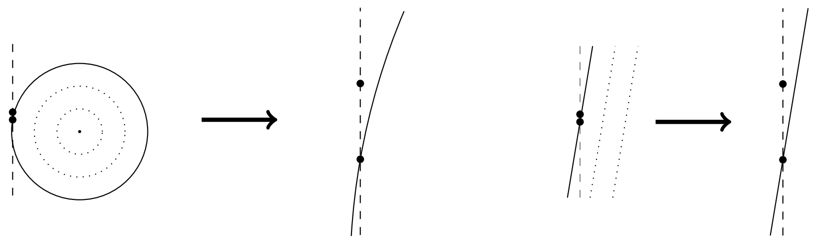


Figure 1.3: Comparison of far-field spherical wave and plane wave

Time delay

In Figure 1.4, a wavefront of a plane wave arriving at a linear array with two microphones is shown. The microphones are spaced by the distance d and the wave has propagation angle ϕ with the angle θ fixed at 90°. At the time when the wave arrives at the lower microphone it will still have to travel an additional distance D before it arrives at the upper microphone, resulting in a delay of $\frac{D}{c}$ seconds.

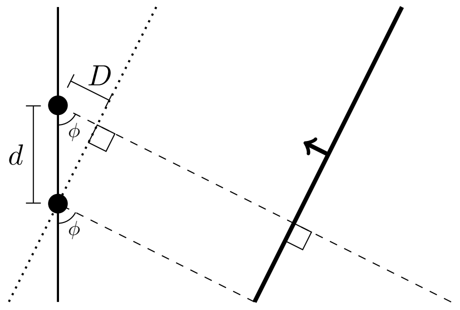


Figure 1.4: Plane wave arriving with angle ϕ to a linear array with two microphones

In the simple case, shown in figure 1.4, the additional distance the wave has to travel is found as $D = d\cos(\phi)$. In the more general case where θ is not fixed, the distance D can be calculated using scalar projection. Defining a unit vector

$$\hat{\mathbf{k}}(\phi,\theta) = \begin{bmatrix}\sin(\theta)\cos(\phi) & \sin(\theta)\sin(\phi) & \cos(\theta)\end{bmatrix}^T \tag{1.5}$$

that points in the direction which the wave is travelling, and a position vector $\mathbf{m}$ for each microphone. The distance D can be calculated as $D = \mathbf{m} \cdot \hat{\mathbf{k}}(\phi,\theta)$ with $\mathbf{m} = \mathbf{0}$ being the reference point for which $D = 0$.

It can be noted that, for a linear array, the delays between the microphones are not affected by rotating the propagation direction of the wave around the array axis. This has the consequence that only the delays in the plane containing the array can be controlled, and all rotations of this plane around the array axis will have the same delays between the microphones. For a planar array this is not the case, and the delays for incident waves from the entire space, located on one side of the plane containing the array, can be controlled. The delays for incident waves with the same relative angle from the opposite space will be the same.

Directional Gain

The directional gain of a beamforming microphone array can be analyzed by observing its output when a complex sinusoid is received as a plane wave. Considering an array of N microphones, and a continuous time sinusoid signal $s(t) = e^{i\omega t}$ with frequency $f = \frac{\omega}{2\pi}$ and propagation direction (ϕ, θ). The vector of received microphone signals $\mathbf{x}$ can be expressed as

$$\mathbf{x} = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_N(t) \end{bmatrix} = \begin{bmatrix} s(t-\tau_1) \\ s(t-\tau_2) \\ \vdots \\ s(t-\tau_N) \end{bmatrix} = \begin{bmatrix} e^{i\omega(t-\tau_1)} \\ e^{i\omega(t-\tau_2)} \\ \vdots \\ e^{i\omega(t-\tau_N)} \end{bmatrix} = \underbrace{e^{i\omega t}}_{s(t)} \overbrace{\begin{bmatrix} e^{-i\omega\tau_1} \\ e^{-i\omega\tau_2} \\ \vdots \\ e^{-i\omega\tau_N} \end{bmatrix}}^{\mathbf{d}(\omega,\phi,\theta)} \tag{1.6}$$

where τ_i denotes the time delay to microphone number i relative some reference point, which can be calculated by

$$\tau_i(\phi, \theta) = \frac{D_i(\phi, \theta)}{c} = \frac{\mathbf{m}_i \cdot \hat{\mathbf{k}}(\phi, \theta)}{c} \tag{1.7}$$

where $\mathbf{m}_i$ is the position vector of i'th microphone. The vector $\mathbf{d}(\omega, \phi, \theta)$, often called the steering vector, contains information about the frequency dependent delay for a given array [5].

To find the directional gain of a microphone array the microphone signals are weighted with their respective weights w_i and summed. The magnitude of the output signal $y(t)$ is calculated as [5]

$$|y(t)| = \left| \sum_{i=1}^{N} w_i^* x_i \right| = \left| \mathbf{w}^H \mathbf{x} \right| = \overbrace{\left| \mathbf{w}^H \mathbf{d}(\omega, \phi, \theta) \right|}^{P(\omega,\phi,\theta)} |s(t)| \tag{1.8}$$

where the function $P(\omega, \phi, \theta)$ is the directional dependent gain of the signal. A plot of this function is known as the beam pattern for the microphone array.

Beampatterns and Array Geometry

The beam pattern of a linear array with four microphones, evenly placed along the x-axis, with spacing $d = 0.165$ m and weights $w_i = \frac{1}{4}$ is shown in figure 1.5. The angle θ is fixed at 90° and the beam pattern is symmetric around the angle $\phi = 180°$, because of symmetry around the x-axis, so only the part between 0° and 180° needs to be considered. The function graph shows that there will be four angles where the gain is exactly zero, meaning that a signal with this frequency will be completely cancelled if it arrives from these directions. The gain at $\phi = 90°$ is one, meaning that the signal will not be attenuated if it arrives from this direction.

The part of the graph that is located between 60° and 120° is called the main lobe and the other parts, located between the zeros, are called sidelobes.

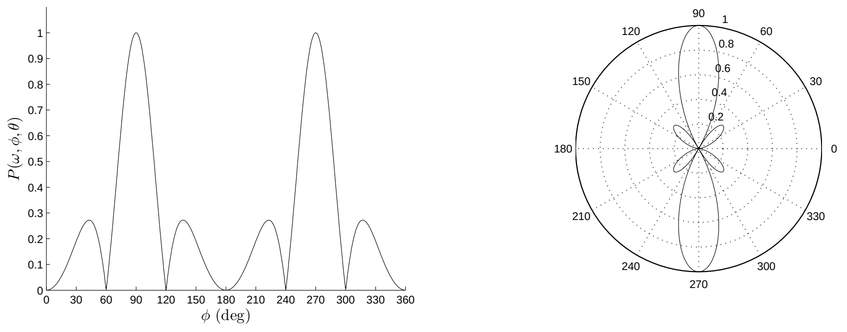


Figure 1.5: Beam pattern and its polar plot for the frequency $f = 1$ kHz

Figure 1.6 shows the effects of varying the microphone spacing d and the number of microphones for an evenly spaced array placed along the x-axis. In the left function graph d is varied, with an array of five microphones, showing that larger d results in a narrower main lobe. In the right function graph the number of microphones is varied, with the total array length fixed to 0.66 m, showing that increasing the number of microphones reduces the sidelobe level.

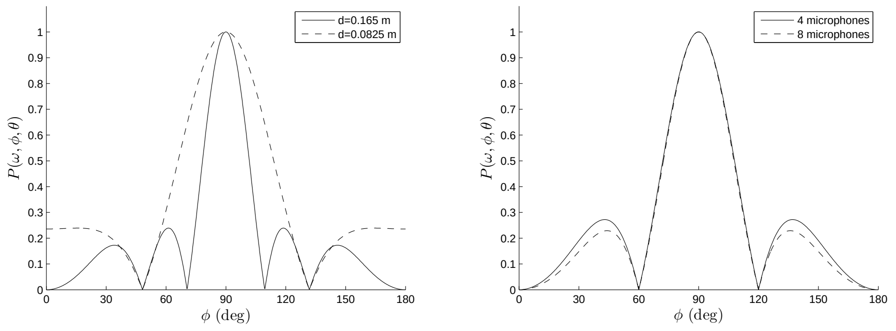


Figure 1.6: Effects of varying spacing d (left) or number of microphones (right) for the frequency $f = 1$ kHz

A three-dimensional image of the beam pattern can be created by varying both ϕ and θ. The result of this can be seen in figure 1.7, displaying the difference in beam pattern between a linear array and a square-shaped planar array. In this figure ϕ is varied from 0° to 360° and θ is varied from 0° to 180°, with the linear array placed along the x-axis and the planar array placed in the xz-plane. What can be seen is the result of how the linear array is only able to control the delays for a plane that is rotated around the x-axis, while the planar array is able to control the delay from all angles of incidence.

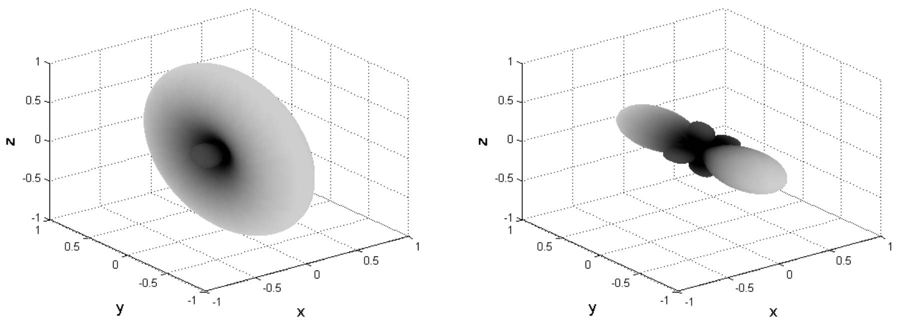


Figure 1.7: 3D spherical beam patterns for linear (left) and planar (right) arrays for the frequency $f = 1$ kHz

By using weights other than $w_i = \frac{1}{N}$ the beam pattern can be steered. Since

$$\mathbf{d}(\omega, \phi, \theta)^H \mathbf{d}(\omega, \phi, \theta) = N \tag{1.9}$$

choosing the weight vector as

$$\mathbf{w} = \frac{1}{N} \mathbf{d}(\omega, \phi_0, \theta_0) \tag{1.10}$$

will result in a gain of one in the direction (ϕ_0, θ_0) as is shown in figure 1.8. This corresponds to phase shifting the signals arriving from (ϕ_0, θ_0) so that they are exactly in phase when the signals are summed, resulting in constructive interference. Since the phase shift is equivalent to a time delay, this is called delay-and-sum beamforming.

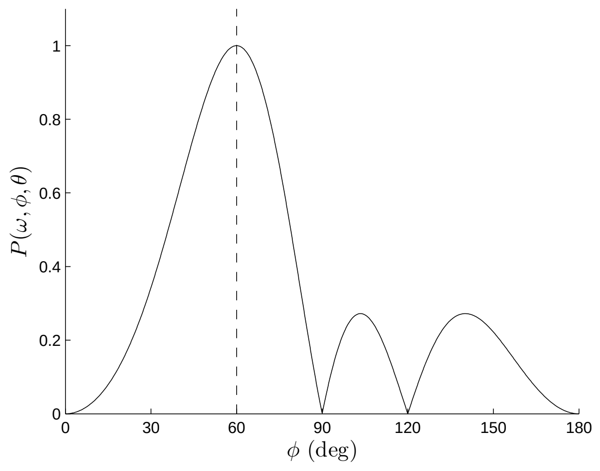


Figure 1.8: Linear array steered to $\phi_0 = 60°$

1.2.3 Wideband Beamforming

When considering signals that do not have such a narrow frequency range that they can be considered as a single frequency, the delay-and-sum beamforming method is not suitable. This is because the steering delays are determined for one frequency, and would not yield the desired result for frequencies other than the one that they were calculated for. The wideband signal can, however, be considered as a composition of many single frequency sinusoids.

A class of beamformers, called filter-and-sum beamformers, can be used to perform beamforming on wideband signals. This type of beamformer applies more extensive filter operations on each microphone signal, before they are summed. The design of such beamformers is displayed in figure 1.9. By using filters h_i, a frequency dependent gain and phase shift can be achieved, allowing a more appropriate time delay for the different frequency components of the signal.

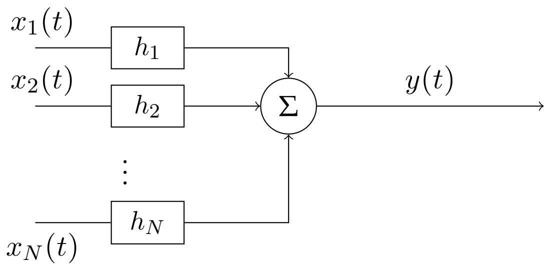


Figure 1.9: Filter-and-sum beamforming structure

Subband Beamforming

One approach to wideband beamforming is called subband beamforming. The idea is to first decompose each of the wideband microphone signals into K narrowband signals, which can then be treated as if they contained only a narrow range of frequencies.

As shown in figure 1.10, where $x_i^{(k)}(t)$ is subband signal number k of the i'th microphone signal, the narrowband signal components are fed to K narrowband beamformers which combine the N microphone subband signals to produce one output per frequency band. The K narrowband outputs are then recombined to create the fullband output signal $y(t)$.

Beampatterns

Since the purpose of wideband beamforming is to manipulate several frequencies, a different way of displaying beam patterns is often used. A three-dimensional plot, where one dimension shows the frequency, reveals how the beam pattern changes with frequency. An example of this can be seen in figure 1.11, where ϕ is varied from 0° to 180° and the frequency range from 0 to 8000 Hz. This type of plot can only be done when either ϕ or θ is kept constant.

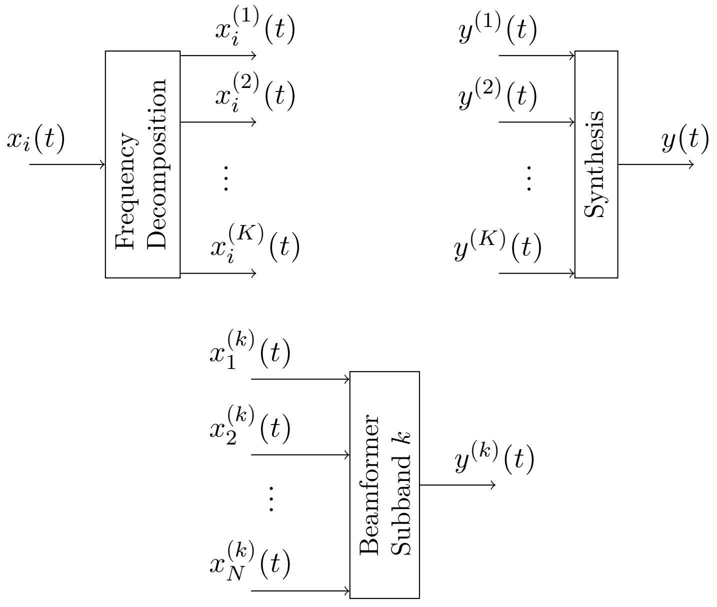


Figure 1.10: Subband beamforming structure

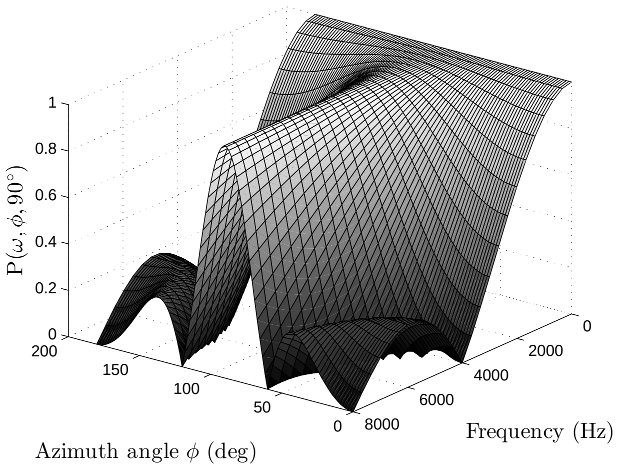


Figure 1.11: Frequency-dependent beam pattern for a linear array with 4 microphones

1.2.4 Limitations

From previous discussion it is made apparent that more microphones decreases sidelobe levels, resulting in more attenuation of the undesired directions. However, using more microphones results in more signals to process, which leads to a higher computational demand.

Another way to improve the performance is to increase the microphone spacing, resulting in a narrower main beam. There are, however, limitations to increasing the microphone spacing. The first and most obvious is the limitations on available space in the application where the beamformer will be used. The second limitation is due to so called spatial aliasing.

Spatial aliasing is a phenomenon similar to the aliasing which occurs when a signal is not sampled with a high enough sampling rate. When using multiple microphones, sampling is also done in space. The sampling rate is then determined by the distance between the microphones, i.e. how densely do the microphones occur in space. If two microphones in the array are separated by a distance so large that the relative delay between them is greater than one half period of the highest frequency in the signal, the direction of arrival cannot be uniquely determined. This essentially means that the beam pattern will repeat itself, and a copy of the main lobe will appear at a different set of angles than the desired one.

To avoid spatial aliasing for a frequency f, the distance d between a microphone and the closest neighbouring microphone must satisfy [5]

$$d \leq \frac{c}{2f}. \tag{1.11}$$

For a wideband signal, the highest frequency in the signal determines if there will be spatial aliasing, according to equation (1.11). A beam pattern showing the effect of spatial aliasing for a frequency of 1 kHz is shown in figure 1.12.

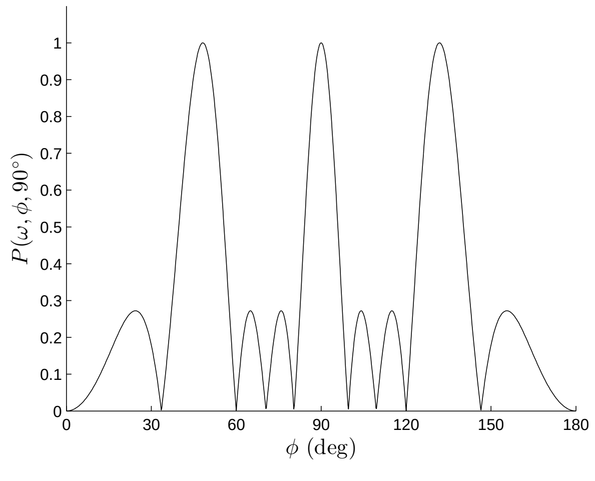


Figure 1.12: Beam pattern with spatial aliasing for a linear array of four microphones, evenly spaced, with $d = 0.495$ m

Chapter 2

Method

This chapter presents the method of this thesis. Two beamforming algorithms, and how they were implemented and tested in both simulated and real environments, are described. A short description of three single microphone algorithms, and their test process, is provided. Tests made with the current Axis product are described, and finally, the measurements used to evaluate the performance are presented.

2.1 Implemented Beamforming Algorithms

It was decided to implement two calibrated beamforming algorithms in a subband structure. The advantage of a calibrated approach is that calculation of the steering delays, and design of filters, is done by using recorded signals, emitted from the desired directions. This means that the differences between the microphones does not have to be considered, but are included in the calculations automatically, because the calibration signals are recorded with the microphones. Differences between microphones might cause poor performance, if they are not included when calculating the steering delays.

A subband structure can improve the computational complexity if the frequency decomposition is implemented efficiently. This can be achieved by using Fast Fourier Transform(FFT). However, using a frequency decomposition will lead to an added delay, which might be undesirable if it is too large.

Discrete time processing

In earlier discussions, for ease of presentation, the signals were considered to be continuous. For the implementation, the signals are sampled in time and digital filters are used. Instead of the time variable t, the time index n is used. The signals are assumed to be uniformly sampled, and properly bandlimited to half of the sampling frequency F_s. The time index n relates to continuous time t through the relation $t = \frac{n}{F_s}$.

Signal Model

The observed signal x_i from the i'th microphone is modelled as

$$x_i[n] = s_i[n] + v_i[n] \tag{2.1}$$

where $s_i[n]$ is the desired signal, from the desired direction, and $v_i[n]$ is the undesired signal, containing both directional and non-directional noise. When considering the frequency decomposition of the signals, which is used for subband beamforming, the signal model for the subband with index k becomes [4]

$$x_i^{(k)}[n] = s_i^{(k)}[n] + v_i^{(k)}[n]. \tag{2.2}$$

To calibrate the beamformers implemented in this thesis, signals representing $s_i[n]$ and $v_i[n]$ are required. These signals must be recorded separately when only the desired and undesired signal is active respectively.

The main goal of using the calibration signals is to capture the information of where they originated from, without relying on a model to explicitly calculate steering delays. However, since the beamforming is performed over a wide range of frequencies, another aspect is how the beamformers behave for different frequencies. The type of calibrated beamformers implemented in this thesis makes it possible to affect this aspect in order to get different gain for different frequencies. This is done by choosing the desired and undesired signal as signals containing desired and undesired frequencies respectively.

Beamformer output signal

The output of a filter-and-sum beamformer, with filters of length L, can be formulated as a scalar product. This is done by stacking the filter weights w_i for each microphone i as a column vector

$$\mathbf{w} = \begin{bmatrix} w_1[0] & \dots & w_1[L-1] & \dots & w_N[0] & \dots & w_N[L-1] \end{bmatrix}^T \tag{2.3}$$

and stacking the L most recent samples of the microphone signals correspondingly as

$$\mathbf{x} = \begin{bmatrix} x_1[n] & \dots & x_1[n-L+1] & \dots & x_N[n] & \dots & x_N[n-L+1] \end{bmatrix}^T \tag{2.4}$$

resulting in the beamformer output signal $y[n]$ being formulated as [4]

$$y[n] = \sum_{i=1}^{N} \sum_{j=0}^{L-1} w_i^*[j] x_i[n-j] = \mathbf{w}^H \mathbf{x}. \tag{2.5}$$

In the case of a subband beamformer, the output at frequency bin k is found in the same way with

$$y^{(k)}[n] = \sum_{i=1}^{N} \sum_{j=0}^{L-1} {w_i^{(k)}}^*[j] x_i^{(k)}[n-j] = {\mathbf{w}^{(k)}}^H \mathbf{x}^{(k)}. \tag{2.6}$$

2.1.1 Least Squares Beamformer

The least squares subband beamformer is derived from the minimum mean square error subband beamformer, which uses the optimality criterion [4]

$$\mathbf{w}_{opt}^{(k)} = \arg\min_{\mathbf{w}^{(k)}} \mathbb{E}\left[\left| y^{(k)}[n] - s_r^{(k)}[n] \right|^2 \right] \tag{2.7}$$

where $y^{(k)}[n]$ is the beamformer output, $s_r^{(k)}[n]$ is the signal of microphone r when only the desired signal is active, k is the subband index and $\mathbb{E}[\cdot]$ denotes the expectation operator. Using equation (2.2) to model the signals, the optimal weights are [4]

$$\mathbf{w}_{opt}^{(k)} = \left[\mathbf{R}_{ss}^{(k)} + \mathbf{R}_{nn}^{(k)}\right]^{-1} \mathbf{r}_s^{(k)} \tag{2.8}$$

where $\mathbf{R}_{ss}^{(k)}$ and $\mathbf{R}_{nn}^{(k)}$ are the autocovariance matrices for when only the desired and undesired signals are active respectively, defined as

$$\mathbf{R}_{ss}^{(k)} = \mathbb{E}\left[\mathbf{s}^{(k)}\mathbf{s}^{(k)^H}\right] \tag{2.9}$$

$$\mathbf{R}_{nn}^{(k)} = \mathbb{E}\left[\mathbf{v}^{(k)}\mathbf{v}^{(k)^H}\right] \tag{2.10}$$

where $\mathbf{s}^{(k)}[n]$ and $\mathbf{v}^{(k)}[n]$ are defined as

$$\mathbf{s}^{(k)}[n] = \begin{bmatrix}\mathbf{s}_1^{(k)}[n] & \mathbf{s}_2^{(k)}[n] & \dots & \mathbf{s}_N^{(k)}[n]\end{bmatrix}^T \tag{2.11}$$

$$\mathbf{v}^{(k)}[n] = \begin{bmatrix}\mathbf{v}_1^{(k)}[n] & \mathbf{v}_2^{(k)}[n] & \dots & \mathbf{v}_N^{(k)}[n]\end{bmatrix}^T . \tag{2.12}$$

The vectors

$$\mathbf{s}_i^{(k)}[n] = \left[s_i^{(k)}[n] \dots s_i^{(k)}[n-L+1]\right] \tag{2.13}$$

and

$$\mathbf{v}_i^{(k)}[n] = \left[v_i^{(k)}[n] \dots v_i^{(k)}[n-L+1]\right] \tag{2.14}$$

contain the L most recent samples of the desired and undesired signal from microphone i, in subband k, respectively. L is the chosen number of time lags to use in the subband filter for each microphone. $\mathbf{r}_s^{(k)}$ is the cross-correlation vector of the signal from one microphone, chosen as reference, and the other microphone signals, when only the desired signal is active

$$\mathbf{r}_s^{(k)} = \begin{bmatrix}r_1^{(k)}[0] & \dots & r_1^{(k)}[L-1] & \dots & r_N^{(k)}[0] & \dots r_N^{(k)}[L-1]\end{bmatrix}^T \tag{2.15}$$

where

$$r_i^{(k)}[j] = \mathbb{E}\left[s_i^{(k)}[n]s_r^{(k)^*}[n-j]\right] . \tag{2.16}$$

Since the true covariance matrices and cross-correlation vector are not known, they need to be estimated. If the expectation is replaced by time averaging to estimate the covariance, the new equations become [4]

$$\hat{\mathbf{R}}_{nn}^{(k)} = \frac{1}{M}\sum_{m=0}^{M-1} \mathbf{v}^{(k)}[m]\mathbf{v}^{(k)}[m]^H \tag{2.17}$$

$$\hat{\mathbf{R}}_{ss}^{(k)} = \frac{1}{M}\sum_{m=0}^{M-1} \mathbf{s}^{(k)}[m]\mathbf{s}^{(k)}[m]^H \tag{2.18}$$

$$\hat{\mathbf{r}}_s^{(k)} = \frac{1}{M}\sum_{m=0}^{M-1} \mathbf{s}^{(k)}[m]s_r^{(k)^*}[m] \tag{2.19}$$

these correspond to the Least Squares(LS) optimality criterion [4]

$$\mathbf{w}_{LS,opt}^{(k)} = \arg\min_{\mathbf{w}_{LS}^{(k)}} \sum_{m=1}^{M-1} \left[\left| y^{(k)}[m] - s_r^{(k)}[m] \right|^2 \right] \tag{2.20}$$

and can be calculated using the corresponding equation [4]

$$\mathbf{w}_{LS,opt}^{(k)} = \left[\hat{\mathbf{R}}_{ss}^{(k)} + \hat{\mathbf{R}}_{nn}^{(k)} \right]^{-1} \hat{\mathbf{r}}_s^{(k)}. \tag{2.21}$$

2.1.2 Signal-to-Noise plus Interference Beamformer

The subband Signal-to-Noise plus Interference Beamformer(SNIB) is similar to the LS beamformer in that it uses calibration signals to find the beamformer weights, however, it has a different optimality criterion. It seeks to maximize the quotient of the mean signal power and the mean noise plus interference power, for each subband, at the output. Using the model presented in this thesis, this means the SNIB tries to maximize the quotient of the mean power of the desired and undesired signal. This leads to the following optimality criterion [4]

$$\mathbf{w}_{opt}^{(k)} = \arg\max_{\mathbf{w}^{(k)}} \frac{\mathbb{E}\left[y_s^{(k)}[n] y_s^{(k)^*}[n] \right]}{\mathbb{E}\left[y_n^{(k)}[n] y_n^{(k)^*}[n] \right]} = \arg\max_{\mathbf{w}^{(k)}} \frac{\mathbf{w}^{(k)^H} \mathbf{R}_{ss}^{(k)} \mathbf{w}^{(k)}}{\mathbf{w}^{(k)^H} \mathbf{R}_{nn}^{(k)} \mathbf{w}^{(k)}} \tag{2.22}$$

where $y_s^{(k)}[n]$ is the beamformer output in subband k when the desired signal $s^{(k)}[n]$ is active alone, and $y_n^{(k)}[n]$ is the beamformer output when the undesired signal $v^{(k)}[n]$ is active alone. The optimal weights for each subband are calculated as the eigenvector corresponding to the largest magnitude eigenvalue found when solving the generalized eigenvalue problem [4]

$$\mathbf{R}_{ss}^{(k)} \mathbf{w}^{(k)} = \lambda^{(k)} \mathbf{R}_{nn}^{(k)} \mathbf{w}^{(k)}. \tag{2.23}$$

In the same way as for the LS beamformer, the covariance matrices are replaced by the time averaged estimates. Other than for small covariance matrices, the eigenvalue problem cannot be solved explicitly and numerical methods must be used. The Lanczos algorithm, which is available in MATLAB through the `eigs` command, was used to achieve this. The eigenvectors $\mathbf{w}_{opt}^{(k)}$ found by this algorithm are normalized so that

$$\mathbf{w}_{opt}^{(k)^H} \mathbf{R}_{ss}^{(k)} \mathbf{w}_{opt}^{(k)} = \lambda^{(k)} \tag{2.24}$$

and

$$\mathbf{w}_{opt}^{(k)^H} \mathbf{R}_{nn}^{(k)} \mathbf{w}_{opt}^{(k)} = 1 \tag{2.25}$$

where $\lambda^{(k)}$ is the eigenvalue corresponding to the eigenvectors.

In contrast to the LS beamformer, the optimal SNIB weights are not uniquely determined from the optimality criterion. The weights can be scaled by any complex constant $\alpha^{(k)}$ and still be optimal as

$$\frac{(\alpha^{(k)}\mathbf{w}^{(k)})^H\mathbf{R}_{ss}^{(k)}(\alpha^{(k)}\mathbf{w}^{(k)})}{(\alpha^{(k)}\mathbf{w}^{(k)})^H\mathbf{R}_{nn}^{(k)}(\alpha^{(k)}\mathbf{w}^{(k)})} = \frac{\left|\alpha^{(k)}\right|^2}{\left|\alpha^{(k)}\right|^2}\frac{\mathbf{w}^{(k)H}\mathbf{R}_{ss}^{(k)}\mathbf{w}^{(k)}}{\mathbf{w}^{(k)H}\mathbf{R}_{nn}^{(k)}\mathbf{w}^{(k)}} = \frac{\mathbf{w}^{(k)H}\mathbf{R}_{ss}^{(k)}\mathbf{w}^{(k)}}{\mathbf{w}^{(k)H}\mathbf{R}_{nn}^{(k)}\mathbf{w}^{(k)}}.$$

This is somewhat problematic when the criterion is used independently in each subband. It is unclear to the authors exactly how the frequencies in the desired signal and the achieved gain in the corresponding subbands are related. Since a solution using normalization could not be found, the weights need to be scaled in some way.

It was decided to calculate the eigenvalues of $\mathbf{R}_{ss}^{(k)}$, which are related to the power in subband k for the desired signal, and use them to scale the weights. The scaling factor for subband k was introduced as

$$\alpha^{(k)} = \sqrt{\frac{\Lambda_s^{(k)}}{\lambda_{s,\max}}} \tag{2.26}$$

where $\Lambda_s^{(k)}$ is the sum of the eigenvalues for $\mathbf{R}_{ss}^{(k)}$ and $\lambda_{s,\max}$ is the largest eigenvalue found over all subbands. The idea was to increase the output signal power in subbands where the desired signal power was large, and decrease it in subbands where the power was small. This scaling turned out to work reasonably well, and due to time constraints it was chosen as the final solution.

2.1.3 Filter Bank

In order to perform subband beamforming, a frequency decomposition producing several narrowband components, from one wideband signal, is required. The basic concept is to filter the wideband signal with a number of bandpass filters, each producing a narrowband component referred to as a subband. Beamforming can then be performed in each subband, after which the components are reconstructed to a single signal. The decomposition and reconstruction is known as the analysis and synthesis filter bank respectively.

It was decided to use a short-time Fourier transform filter bank from [1]. This type of filter bank relies on a prototype low-pass filter, modulated by a Discrete Fourier Transform (DFT), to produce a bank of bandpass filters with equal bandwidth. An efficient implementation can be made by using FFT in the analysis filter bank, and Inverse Fast Fourier Transform(IFFT) in the synthesis filter bank.

Analysis Filter Bank

To obtain K subband signals with a DFT modulated prototype filter, a DFT of length K must be used [1]. In this thesis it was chosen to use blocks of K samples windowed with a Hamming window, also of length K, as the prototype filter. The Hamming window was created by using the MATLAB function `hamming`, implementing the equation

$$w_H[n] = 0.54 - 0.46\cos(2\pi\frac{n}{K-1}) \quad 0 \leq n \leq K-1. \tag{2.27}$$

K subband samples are produced by taking the DFT of the windowed signal block, with each sample calculated as

$$x^{(k)}[n] = \sum_{j=0}^{K-1} x[j]w_H[j]e^{-i2\pi\frac{kj}{K}}. \tag{2.28}$$

This was implemented with the MATLAB function `fft`, which uses FFT. The resulting subband samples are stored in a vector

$$\mathbf{X}[n] = \left[x^{(0)}[n] \dots x^{(K-1)}[n]\right]^T. \tag{2.29}$$

The subband samples could be calculated for every signal sample, increasing the time index n only one sample before calculating the next DFT. However, due to the reduced bandwidth of the subband signals, caused by the bandpass filtering, decimation can be performed without creating aliasing in the subband signals [1]. This can be used to increase the efficiency of the filter bank. A Hamming window of length K has a bandwidth of $\frac{2}{K}$ in normalized frequency, implying that new samples of the subband signals must be calculated for every $\frac{K}{4}$ samples of the input signal to avoid aliasing [1]. This means that in order to avoid aliasing in the subband signals, there must be a 75% overlap between subsequent signal blocks.

The decimated time index m is introduced, related to time index n through $n = \frac{K}{4}m$. In summary the analysis filter bank operates as follows:

1. Obtain K samples of the input signal $\mathbf{x}[n] = [x[n] \dots x[n-K+1]]^T$
2. Window the K samples with the Hamming window
3. Transform the windowed signal block with FFT to produce the subband samples $\mathbf{X}[m]$
4. Increase time index n by $\frac{K}{4}$
5. Repeat from step 1

Synthesis Filter Bank

The synthesis filter bank was implemented using a technique known as overlap-add [1]. Output signal $\mathbf{y}[m]$ is created by inverse transforming the subband sample vector $\mathbf{X}[m]$, to recover the windowed signal block. This signal block is then added to the previous signal block, with a 75% overlap, resulting in $\frac{K}{4}$ finished output samples, starting at time m.

The overlap-add method relies on the fact that the sum of the overlapping windows is close to a constant value, as is illustrated in figure 2.1. As can be seen in the lower plots, summing the overlapping windows results in an almost constant gain, denoted as C.

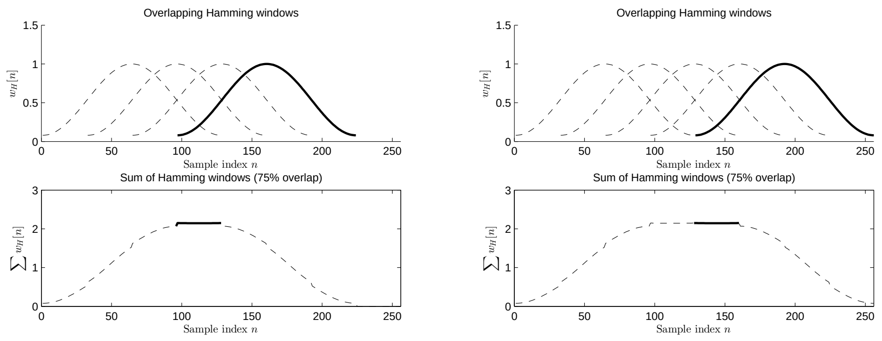


Figure 2.1: Overlapping Hamming windows and their sum

Performing overlap-add continuously during processing introduces a delay of $\frac{3K}{4}$ samples in the output signal. The tapering effect seen in figure 2.1 will only occur at the very beginning and end of processing, and this effect is so small that it is not noticeable in the resulting output.

Since the input signal to the analysis filter bank is real-valued, the DFT has the conjugate symmetry property

$$x^{(K-1-k)}[n] = x^{(k)^*}[n] \tag{2.30}$$

meaning that the values for subbands $k = \frac{K}{2} \mathinner{..} K-1$ can be determined from the first $\frac{K}{2}$ values. This property can be exploited to save some computations when performing subband filtering. Only the first $\frac{K}{2}$ subband samples needs to be filtered and the rest can be found using (2.30).

The synthesis filter bank was implemented with the MATLAB function `ifft`, which uses IFFT, and the conjugate symmetry property was utilized. Its operating procedure can be summarized as:

1. Transform filtered subband samples $\mathbf{X}[m]$ with IFFT to obtain signal block $\mathbf{y}[m] = [y[m] \ldots y[m-K+1]]^T$
2. Overlap and add the first $\frac{3K}{4}$ samples of $\mathbf{y}[m-1]$, to the last $\frac{3K}{4}$ samples of $\mathbf{y}[m]$
3. Divide the last $\frac{K}{4}$ samples of $\mathbf{y}[m]$ by the window gain C and output them
4. Increase the decimated time index m by 1
5. Repeat from step 1

2.2 Simulation

In order to evaluate the performance of the two beamforming algorithms, with a large variety of microphone configurations, simulated environments were implemented in MATLAB. The intention of creating a simulation was to be able to investigate the effects of varying the shape, size and number of microphones in an array, without having to reconfigure a physical array for every test case. Through this approach, a much larger variety of microphone configurations could be tested and evaluated.

Four environments were designed, two for the calibration process and two for the test process. The reasons for using the different environments, and how they differ, are described in sections 2.2.2 and 2.2.3. The results from the simulations are presented in section 3.1.

2.2.1 Microphone Configurations

Five different microphone array shapes were tested, with a varied number of microphones. The shapes were chosen as samples of geometrical figures, with varied size, in order to evaluate how the spatial relation between the microphones affects performance of the beamforming algorithms. Each shape, with an example of how they were sampled, can be viewed in figure 2.2. The figure also shows the variables used to vary the size of the arrays. Table 2.1 shows the variations on size, and number of microphones, tested for each shape.

Each microphone signal was simulated for each sound source. This was done by delaying the signal from a source, according to the time it would take a plane sound wave to travel from a reference microphone to the simulated microphone. Calculating the travel distance was done with the angles of incidence of the source and the distance between the microphones. The delayed signals from all sound sources were then added, in order to create the simulated microphone output.

Array	d (cm)	h (cm)	Microphones
Linear(U)	1-15	-	2-8
Linear(N)	4-15	-	4,6,8
Rectangular	1-5	1-5	4,6,8
Trapezoidal	1-5	1-5	5,7
Circular	1-7.5	-	5-8
Triangular	2-15	-	3,6

Table 2.1: Simulated microphone array variations

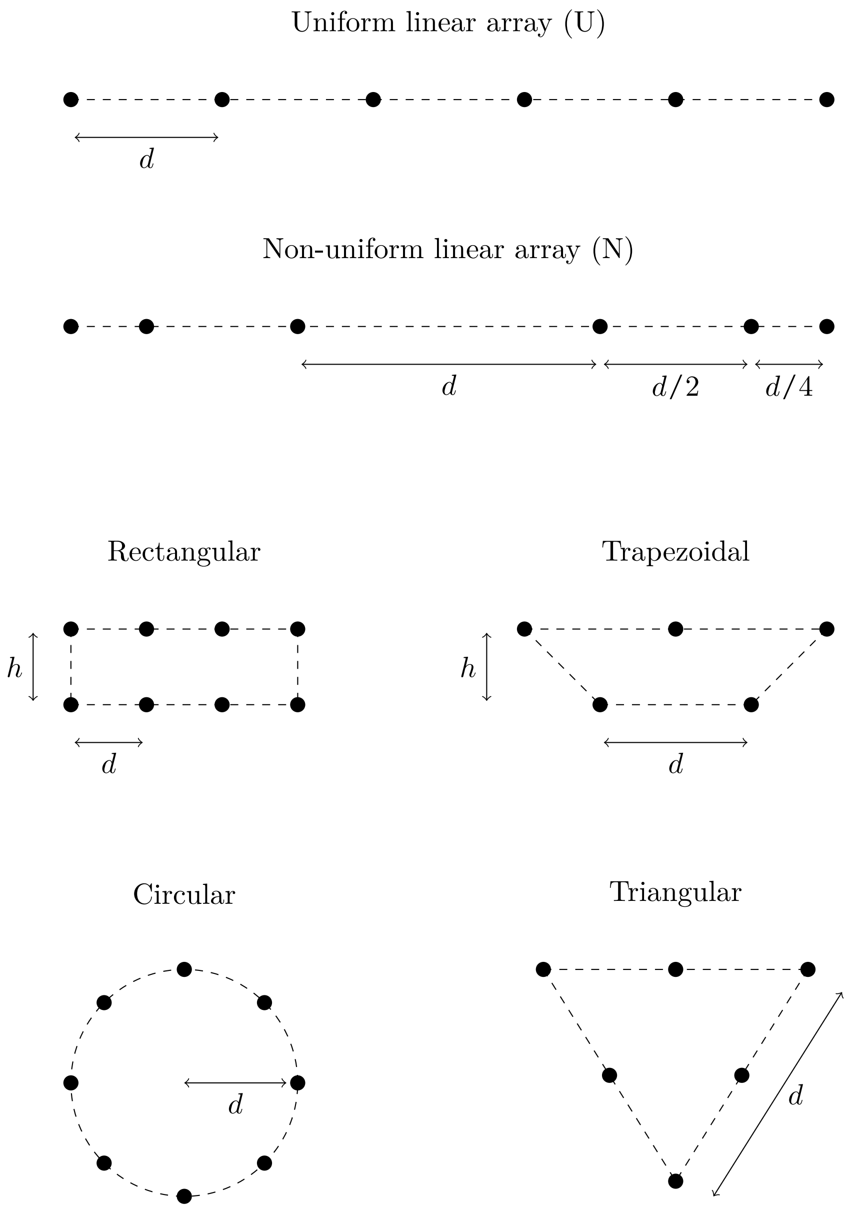


Figure 2.2: Simulated microphone array shapes

2.2.2 Sound Configuration

Since all sound in the simulation was treated as plane waves, sound sources were defined by two angles of incidence. For the planar arrays, both elevation angle θ and azimuth angle ϕ were varied between 0° and 180°. As for the linear arrays, only ϕ was varied while θ was kept constant. The reasoning behind this choice comes from the different beampatterns, seen in figure 1.7, produced by the arrays. They show that the effects of varying a linear array can be evaluated by studying its performance in one plane. This property was exploited in order to reduce calculations, and sound sources were placed only in one plane for the linear arrays. It should be noted that this creates different test environments, which implies that the simulation results for linear and planar arrays should not be compared.

The purpose of the beamforming in this application is to suppress noise, independent of its angles of incidence. To achieve this, the ideal calibration environment would be to have noise of the same level incident from all directions. This is also known as a diffuse noise field. In order to simulate such a field, multiple sound sources were spread out evenly between 0° and 180°, creating the calibration noise field. This was, however, not ideal for the test environments, due to the fact that the main beam actually have a different width, depending on the microphone configuration. Because of this, the noise fields for the test environments were chosen to have a 25° gap on each side of the speech source, as to minimize the effect on measurements from a change in beam width.

Placement of the calibration and test sources for the linear arrays are shown in figure 2.3, with an example of a microphone array, in order to illustrate the relative placement. Microphones are represented by circles, noise sources by crosses and speech sources by triangles. For the planar arrays, calibration and test sources are simply spread out over the elevation angle as well, forming two hemispheres.

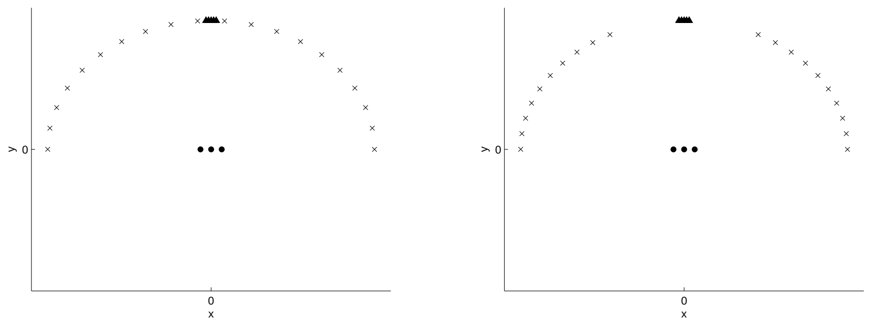


Figure 2.3: Placement of calibration (left) and test (right) sources for the linear array simulations

2.2.3 Calibration and Test Sequence

Signals with a sampling rate of 16 kHz, and a SNR of −5 dB, were used to calibrate and test the beamforming algorithms. The calibration sequence used two different signals of the same length. One for desired and one for undesired signal, emitted from the speech and noise sources respectively. In order to achieve frequency gain control, a signal representing human speech was used as desired signal. White noise was used as undesired signal, because of its flat spectrum. An example of using the signal representing human speech, compared to using white noise, as desired signal is shown in the beampatterns in figure 2.4.

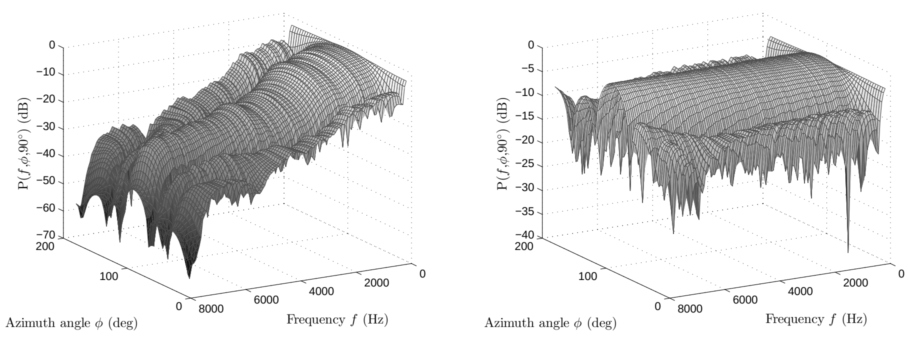


Figure 2.4: Beampatterns for the LS beamformer, calibrated with a representation of human speech (left) and white noise (right)

White noise was generated for each noise source, in order to create uncorrelated noise. Two calibration environment simulations were done for each array, with only one of the two signals active at a time. This was done to be able to calculate the autocovariance matrices used in the beamforming algorithms.

For the test sequence, the signal representing human speech was changed to actual human speech. This was done in order to be able to hear how the algorithms affected a speech signal. The noise signal was kept as uncorrelated white noise, with the same length as the speech signal. Simulation of the test environment was done three times for each array: one where both speech and noise were active, one where only the speech was active, and one where only the noise was active. The simulations with either speech or noise active was done to be able to calculate the measurements described in sections 2.6.1 and 2.6.2.

2.2.4 Implementation Problems

Two main problems arose when implementing the simulation. The first was to realistically simulate the delay between the microphones, and the second was related to the fact that the simulated sound sources behaved as point sources.

Simulating the delays became a problem because the actual delays did not translate to an exact number of samples in the signal. The delays were calculated as the time t it takes sound from a source to travel to the microphones, and the result seldom translates to an exact number of samples. This means that the signal must be delayed according to the fractional part of the sample delay. In order to achieve this, and create a realistic simulation, a fractional delay filter was implemented. The selected filter is based on the ideal interpolation function

$$\mathrm{sinc}(t) = \frac{\sin(\pi t)}{\pi t} \tag{2.31}$$

which interpolates a properly bandlimited signal to continuous time. The interpolation function is sampled at the desired fractional delay δ_{frac}, and truncated to a length of 100 samples [6]

$$I_{frac}[n] = \mathrm{sinc}(n - \delta_{frac}) \quad -49 \leq n \leq 50. \tag{2.32}$$

The delay in samples is calculated as

$$\delta = \delta_{int} + \delta_{frac} = t_{delay} \cdot F_s \tag{2.33}$$

where F_s is the sampling rate and δ_{int} is the integer part of the sample delay. δ_{int} was realised by appending δ_{int} zeroes to the beginning of the filter vector

$$\boldsymbol{\delta} = \begin{bmatrix} 0 & \dots & 0 & I_{frac}[-49] & \dots & I_{frac}[50] \end{bmatrix}. \tag{2.34}$$

The delayed signal is then created through convolution with $\boldsymbol{\delta}$.

The second problem was a result of defining sound sources as point sources, i.e. exact mathematical points in MATLAB. Having a point source as noise creates a situation where the beamforming algorithms have exactly one precise mathematical point to suppress. This allows the algorithms to perform exact cancellation, which produces unrealistic results. If there are more than one source, the number of sources that can result in exact cancellation is related to the microphone configuration. The largest number of sources that can result in this behaviour is the same as the number of microphones in the array. This problem was addressed by using a large amount of simulated noise sources.

There were two other properties of the simulation, which created an unrealistic environment. But due to the nature of the properties, they were seen as advantages instead of problems. One was that the simulated microphones were ideal microphones. In reality this would not be the case, and the performance of the microphones would differ, due to variables such as manufacturing process and material variations. The other property was the absence of reverberations in the simulated environment. Every real environment contains objects that causes some level of reverberation. Because these properties are never the same in any environment, it was seen as advantageous to keep them, since the results might have reflected some specific property of a certain environment otherwise.

2.3 Real-time Implementation and Testing

A real-time implementation was done in MATLAB, with the help of a framework created by Mikael Swartling [7]. The framework provides communication between MATLAB and a sound card, as well as MATLAB functions for real-time execution. This allows the beamforming algorithms to be tested with real equipment, in order to evaluate the performance in a real environment.

Two array shapes were selected to be tested, based on their performance in the simulated environment. Limits on the array dimensions were based on the dimensions of the Axis A8004-VE. The arrays were designed so that a number of sub-arrays could also be tested. An image of the rack that was built to hold the microphones, and a description of the different configurations that were tested, can be found in section 2.3.3.

Signals were sampled with a sampling rate of 48 kHz, and decimated to 16 kHz before any operations were performed. One array was selected to be used to test different variations on the algorithms, in order to determine their effect on the performance. The variations that were tested will be described in section 2.3.6.

2.3.1 Equipment

The microphones used for testing the real-time implementation were AKG condenser microphones. An image of one of the microphones can be seen in figure 2.5. A preamplifier from Line Audio Design was used to power the microphones, and an audio interface from Terratec was used as sound card. An image of these is shown in figure 2.6. The audio interface was connected to the computer through a Firewire connection.

Two different speaker types were used as sound sources. For the speech, and one source of noise, speakers from Fostex were used. One of these can be seen, together with the spectrum it produces for white noise, in figure 2.7. Another type of speaker was used to produce additional noise, in order to provide the environment with a more evenly distributed noise spectrum. This was a dodecahedron speaker, manufactured by Norsonic, and it can be seen in figure 2.8 with the spectrum it produces for white noise.

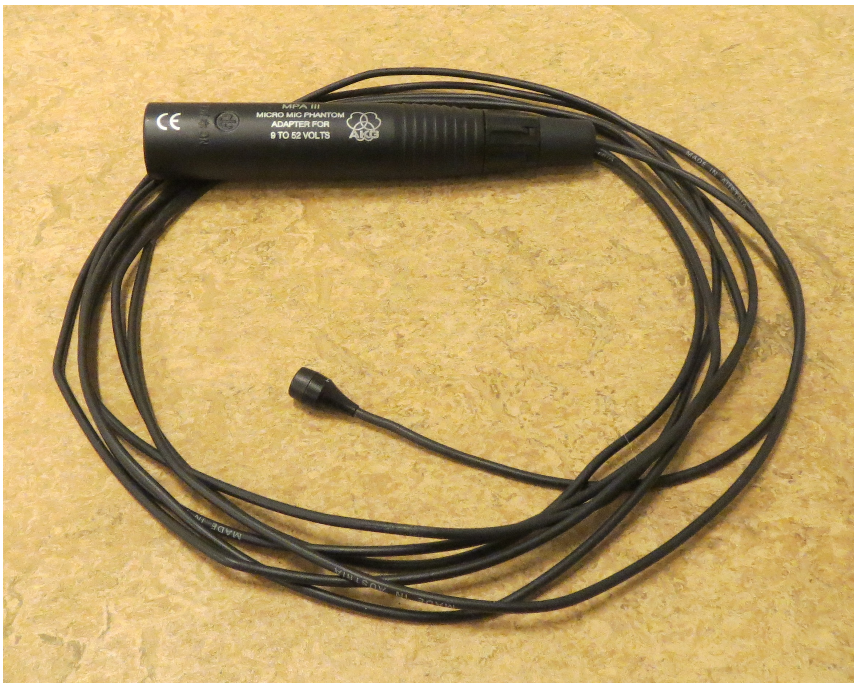


Figure 2.5: AKG C417 condenser microphone with AKG MPA III phantom adapter

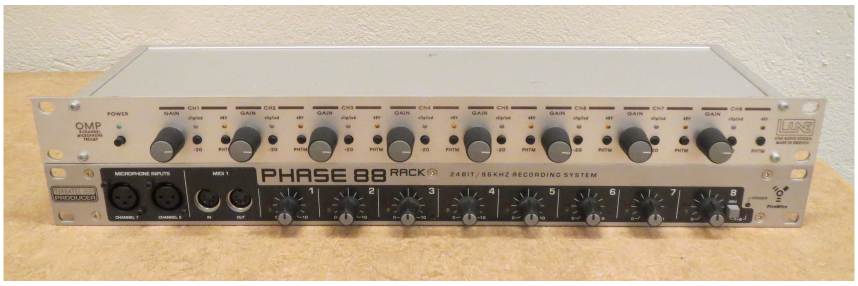


Figure 2.6: 8 channel microphone preamplifier from Line Audio Design and Terratec PHASE 88 Rack FW audio interface

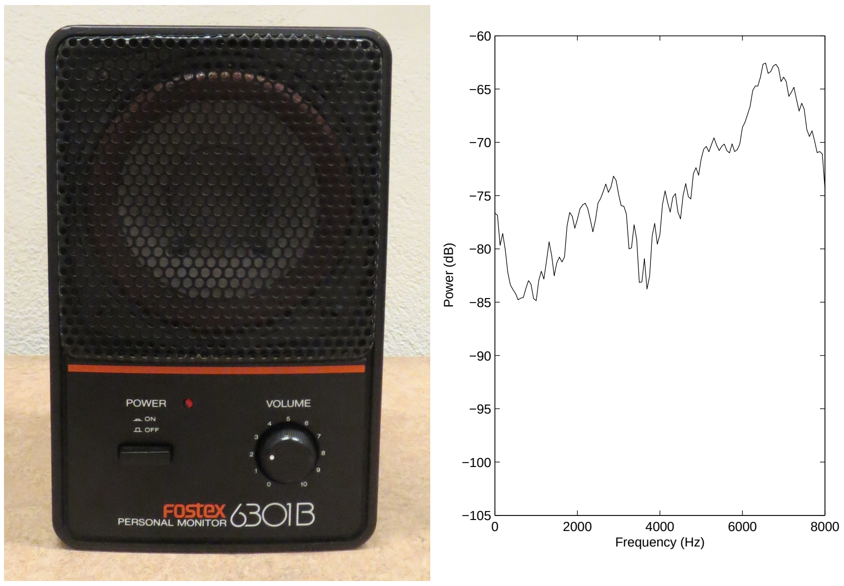


Figure 2.7: Fostex 6301B speaker with spectrum of white noise, recorded with an AKG C417 microphone

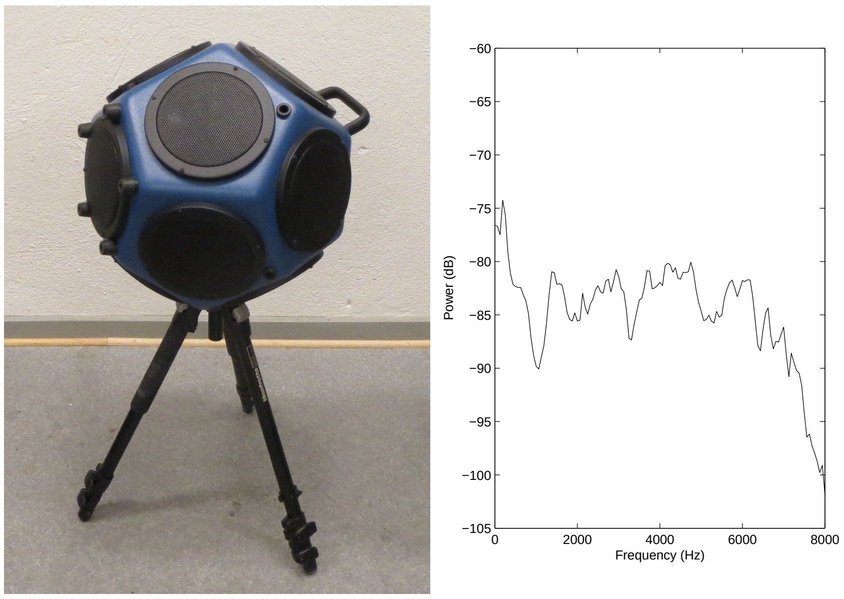


Figure 2.8: Norsonic Nor270 dodecahedron speaker with spectrum of white noise, recorded with an AKG C417 microphone

2.3.2 Environment

Testing of the beamforming algorithms were performed in two real environments. The majority of the tests were done in an anechoic chamber, at Lund University. An anechoic chamber has close to zero reverberations, and the reason for using this type of environment was to make the results independent of reverberations in a specific environment. A part of a wall in the anechoic chamber is shown in figure 2.9, the rest of the chamber, including floor and ceiling, is covered in the same way.

The other environment was a large room, with concrete walls, which was used to test three of the arrays in a reverberant environment. If all of the tests had been done in the room with concrete walls, the results would be dependent on the specific reverberations in that room.

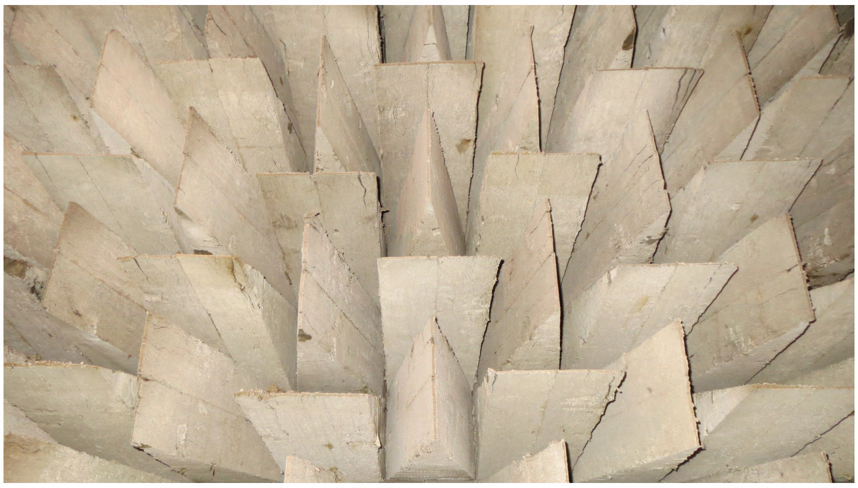


Figure 2.9: A wall in the anechoic chamber in the Perlos antenna laboratory at Lund University

2.3.3 Microphone Configurations

In order to place the microphones at the correct positions, corresponding to the selected arrays, a rack was designed by the authors and built by Axis. This rack can be viewed in figure 2.10. Adhesive compound, used to attach paper to walls, was used to secure the microphones in their positions. An image of how they were secured can be seen in figure 2.11.

Several microphone configurations were tested in the anechoic chamber. These can be viewed in table 2.2, and the results are displayed in section 3.2. For a reference of how the dimension variable is used, see figure 2.2. Three of the arrays were also tested in the reverberant environment, and the result of these test are displayed in section 3.2.1.

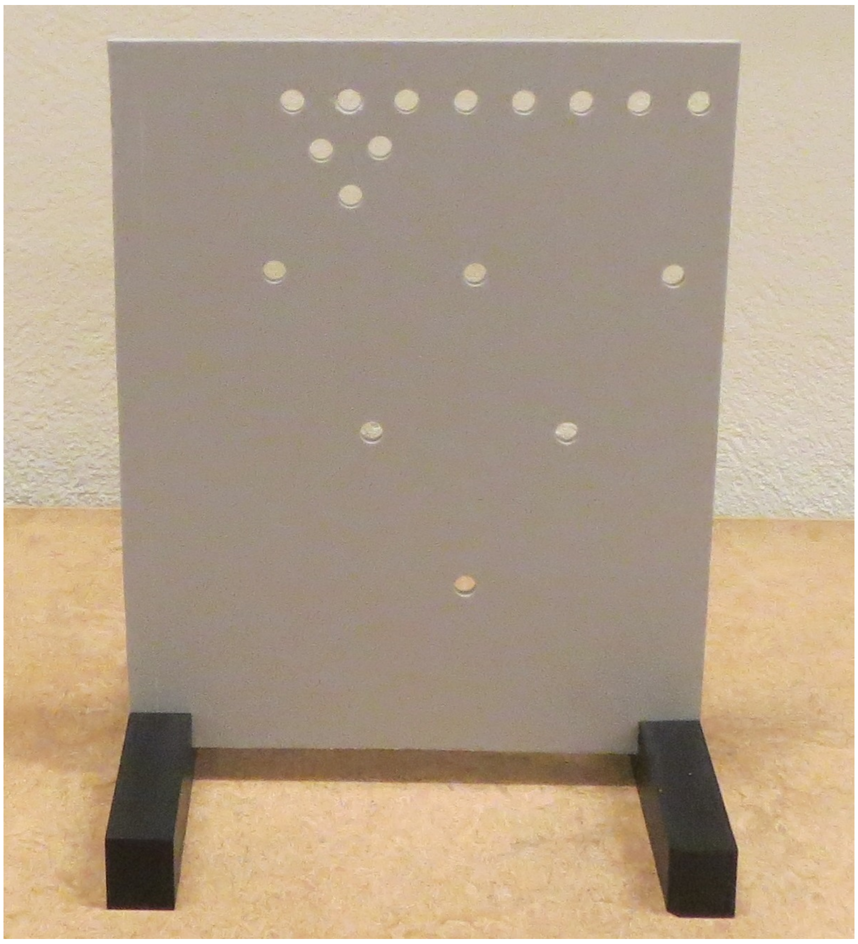


Figure 2.10: Microphone rack used to position the microphones

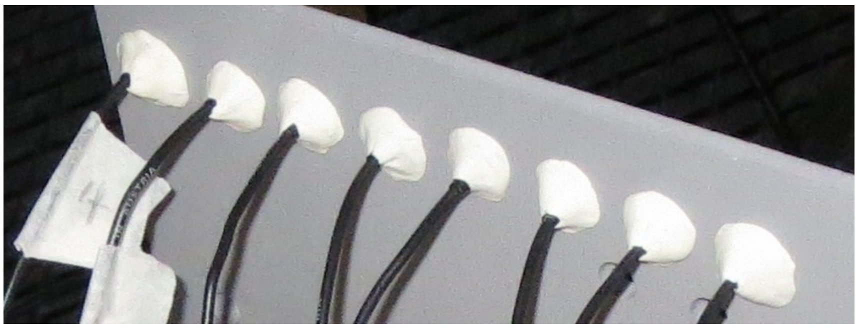


Figure 2.11: Attachment of microphones on the backside of the microphone rack

Array	Micro-phones	d (cm)						
Linear(U)	2	1.85	3.70	5.55	7.40	9.25	11.10	12.95
	3	1.85	3.70	5.55				
	4	1.85	3.70					
	5	1.85						
	6	1.85						
	7	1.85						
	8	1.85						
Linear(N)	4			5.55				
Triangular	3	1.85	3.70	6.50				13.00
	6		3.70					13.00

Table 2.2: Microphone array configurations tested in the anechoic chamber

2.3.4 Sound Configuration

Placement of the speakers, relative to the microphone array, inside the anechoic chamber can be seen in figure 2.12. The speaker directly facing the array was used as speech source, and the two other speakers were used as noise sources. In order to try and mimic the situation with a door station, the distance between the speech source and the array was set to approximately 50 cm.

One noise source was placed on each side of the speech source, to have noise incident from both sides of the speech. The sound sources were placed in approximately the same positions, relative to the array, in the reverberant environment.

2.3.5 Calibration and Test Sequences

The same signals used in the simulation were used to calibrate and test the real-time implementation. These signals, and the reason for using them, are described in section 2.2.3. As stated earlier, signals were recorded at 48 kHz sampling rate and decimated to 16 kHz, before any operations were performed. This was done in order to have the same sampling rate as the Axis A8004-VE. All signals had a SNR of -5 dB, unless stated otherwise.

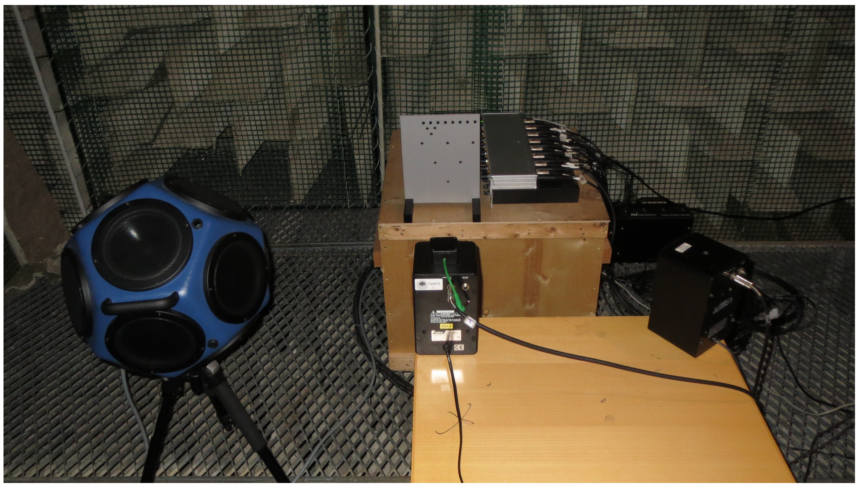


Figure 2.12: Configuration of the equipment inside the anechoic chamber

2.3.6 Algorithm Variations

In order to evaluate the effect on performance, tests were done with variations on three different properties of the algorithms. These tests were performed using a uniform linear array with three microphones, and a microphone spacing of 5.55 cm. The varied properties and their values were:

- Calibration SNR of -10, -5, 0, 5 and 10 dB.
- Filter bank with 32, 64, 128 and 256 subbands.
- Subband filters with 1 to 10 taps.

2.3.7 Implementation Problems

The only problem that arose with the real-time implementation was the execution time of the filtering operations. Since for-loops are very slow in MATLAB, it was not possible to loop through the microphone signals, performing filtering operations, in real-time. Because MATLAB is very effective in matrix computations, the solution to this problem was to create index masks and place the signals in one large matrix. The filter weights could then be stacked in another large matrix, making it possible to perform the filtering operations with matrix computations. This made execution 10 times faster, and allowed the implementation to be executed in real-time.

2.4 Single Microphone Algorithms

As stated in the introduction, algorithms designed for a single microphone can be divided into three classes: spectral subtractive algorithms, subspace algorithms and statistical-model-based algorithms [1]. In order to compare their performance to the beamforming algorithms, one single microphone algorithm from each class was evaluated. MATLAB code used to test them was taken from [1], and detailed descriptions of the algorithms can be found in this book.

The algorithms estimate the noise properties, when there is no speech present in the signal, using Voice Activity Detection (VAD). This can be problematic, if the noise conditions are very poor. When the VAD is not able to properly distinguish between if there is speech or not, parts of the speech might be considered to be noise. The result of this will be distortion of the speech, due to the fact that the algorithms will try to remove the parts of the signal they determine to be noise.

Another problem that can arise is musical noise [1]. It occurs when the properties of the noise are inaccurately estimated, and results in random tones appearing in the processed signal. These tones create a different noise, which can sometimes be considered to be worse than the original noise.

The code for each algorithm was modified to perform the filtering operations on two additional signals. This was done to be able to calculate the measurements in sections 2.6.1 and 2.6.2. Because the algorithms are adaptive, they do not modify a speech-only or noise-only signal in the same way as the signal with both speech and noise.

Tests were done with signals recorded in the anechoic chamber, using the equipment described in section 2.3.1. Recording was done with a sampling rate of 48 kHz, and the signals were decimated to 16 kHz. Three different signals were tested for each algorithm, in order to evaluate how the SNR affects performance. The SNR was set to -5, 0 and 5 dB, and the results of the tests are presented in section 3.3.

2.4.1 Spectral Subtraction

The spectral subtraction algorithm operates in the frequency domain, and uses the same signal model as the subband beamformers, given as

$$x^{(k)}[n] = s^{(k)}[n] + v^{(k)}[n]. \tag{2.35}$$

The algorithm tries to estimate the noise spectrum, and subtract it from the input signal spectrum.

With the use of short-time fourier transform, frequency decomposition is performed. Using VAD, an estimated noise power spectrum is updated at times when speech is absent. Subtraction of the estimated noise power spectrum is performed on the input signal power spectrum, to form the enhanced power spectrum. By taking the square root of the enhanced power spectrum, the magnitude spectrum is obtained. This is then rectified to be non-negative. The enhanced magnitude spectrum is combined with the phase spectrum of the original signal, and an inverse DFT is performed. Overlap-add is used to obtain the enhanced output signal. The MATLAB function used to evaluate this algorithm, taken from [1], is called `specsub`.

2.4.2 Subspace Tracking

The subspace tracking algorithm operates in the time domain, using the signal model

$$x[n] = s[n] + v[n]. \tag{2.36}$$

This algorithm attempts to separate the received signal x, into one speech and one noise component, using a vector space approach. Considering the samples of s and v as linear combinations of past and present samples of x. The idea is to find vectors spanning the subspace corresponding to s, and project the samples of x onto this subspace.

Autocovariance of the noise is estimated using VAD, and autocovariance matrices, for noise and input signal, are computed. The signal matrix is pre-multiplied with the inverse of the noise matrix, in order to whiten the noise. Eigenvalue decomposition is performed on the pre-whitened autocovariance matrix, and eigenvectors corresponding to positive eigenvalues are considered to span the speech signal subspace. The positive eigenvalues are used, together with an estimate of the *a priori* SNR, to form a gain function for the eigenvectors of the speech subspace. Eigenvectors with negative eigenvalues are discarded, by giving them a gain of 0. Gain values, calculated from the gain function, are arranged in a diagonal matrix, which is then transformed back to the original basis. The resulting matrix is pre-multiplied with the noise matrix, in order to undo the whitening. To obtain the enhanced output signal, the resulting matrix is used to project the samples of x onto the speech subspace. The MATLAB function `klt`, taken from [1], was used to evaluate this algorithm.

2.4.3 Wiener Filter using estimated *a priori* SNR

This algorithm operates in the frequency domain, and is based on the Wiener gain function for additive uncorrelated noise

$$H^{(k)} = \sqrt{\frac{\xi^{(k)}}{\xi^{(k)} + 1}}. \tag{2.37}$$

$\xi^{(k)}$ is the *a priori* SNR at frequency bin k, i.e. the ratio of the speech power spectral density to the noise power spectral density.

Frequency decomposition is performed on the input signal, using short-time fourier transform. VAD is used to estimate the power spectral density of the noise, and the *a posteriori* SNR is estimated from this. $\xi^{(k)}$ is then estimated recursively, using the *a posteriori* SNR, and the previously estimated *a priori* SNR. The input signal magnitude spectrum is multiplied with the gain $H^{(k)}$ at each frequency bin k, to create the enhanced magnitude spectrum. An inverse DFT is performed on the combination of the enhanced magnitude spectrum and the phase spectrum of the original signal. Finally, overlap-add is used to obtain the enhanced output signal. Evaluation of this algorithm was done using the MATLAB function `wiener_as`, taken from [1].

2.5 Axis A8004-VE

The A8004-VE was tested in a reverberant environment at Axis Communications in Lund. The signals used for testing were the same as for the other tests, but they were recorded with the A8004-VE. A sampling rate of 16 kHz was used, and signals with a SNR of -5, 0 and 5 dB were tested.

Because of the fact that these tests were not performed with the exact same equipment as the others, e.g. the microphones are different, the results should only be considered as guidelines. The performance of the A8004-VE cannot be directly compared to the performance of the other evaluated solutions in this thesis. As the algorithm used for the A8004-VE is a black box solution, the measurements described in sections 2.6.1 and 2.6.2 cannot be calculated.

2.6 Measurements

To quantify the performance of the beamforming algorithms, and to be able to compare them to the other solutions, four different measurements were used. These will be explained here, together with the motives for using them.

2.6.1 Speech Distortion

In order to measure distortion of the speech signal, caused by the algorithms, the normalized distortion quantity D was used. This measurement was taken from [4], and measures how a processed signal deviates from the source signal.

$$D = \frac{1}{2\pi} \int_{-\pi}^{\pi} |C_d \hat{P}_{y_s}(\omega) - \hat{P}_{x_s}(\omega)| dw \tag{2.38}$$

where $\omega = 2\pi f$, with f as normalized frequency, and the constant C_d is defined as

$$C_d = \frac{\int_{-\pi}^{\pi} \hat{P}_{x_s}(\omega) dw}{\int_{-\pi}^{\pi} \hat{P}_{y_s}(\omega) dw} \tag{2.39}$$

where $\hat{P}_{x_s}(\omega)$ and $\hat{P}_{y_s}(\omega)$ are estimates of the spectral power, when only the speech signal is active, for a single microphone recording and the processed output respectively. The purpose of C_d is to normalize the mean spectral power of the processed output, to the spectral power of the single microphone recording. D is the mean deviation of the spectral power of the processed output, from that of a single microphone recording.

The integrals in the formula were estimated with the sum of the estimated spectral power, calculated using the `pwelch` function in MATLAB. A 256 sample wide Hamming window, and a 50% overlap, was used. Values calculated with (2.38) will be shown in decibel and referred to as speech distortion, with the ideal value being zero.

2.6.2 Noise Suppression

To measure the noise suppression of the algorithms, the normalized noise suppression quantity S_N was used. Like the speech distortion measurement, this was also taken from [4].

$$S_N = C_d \frac{\int_{-\pi}^{\pi} \hat{P}_{y_N}(\omega) dw}{\int_{-\pi}^{\pi} \hat{P}_{x_N}(\omega) dw} \tag{2.40}$$

where C_d is the constant described in (2.39). $\hat{P}_{x_N}(\omega)$ and $\hat{P}_{y_N}(\omega)$ are estimates of the spectral power, when only the noise signal is active, for a single microphone signal and the processed signal respectively.

The integrals are estimated in the same way as for the speech distortion, and the measurement describes the ratio of processed noise to unprocessed noise. A lower value is better, but the negated value will be presented in the results. This is done because suppression, used in this context, implies that a higher value means better performance. Values calculated with (2.40) will be shown in decibel and referred to as noise suppression.

2.6.3 Signal-to-noise ratio improvement

To be able to compare the performance of the algorithms to black box solutions, another measurement that describe suppression of noise was used. This measurement was taken from [1], and calculated with the `comp_snr` MATLAB function from this book.

`comp_snr` takes two signals, one with only speech and one with speech and noise. The function calculates the SNR of the signal with both speech and noise. This is done by simply subtracting the speech from the combined signal, and calculating the SNR of the clean speech and the result. If this is done both for an unprocessed signal and a processed signal, the difference between the two SNR values describe the noise suppression.

Before the values were calculated, the processed signal power was normalized to the clean speech signal power. This was done in order to reduce the effect of general gain differences, using

$$C = \sqrt{\frac{\sigma_s^2}{\sigma_y^2}} \tag{2.41}$$

where σ_s^2 is the variance of the clean speech signal, and σ_y^2 is the variance of the processed signal. The constant C is not optimal, as the processed signal contains noise, but it was used because a processed speech-only signal is not available for adaptive black box solutions.

There are two problems with this measurement, both related to the fact that the speech is simply subtracted from the combined signal. The first is that the signals must be perfectly aligned. This can be achieved by calculating the cross covariance, finding the lag with the largest value, and delaying the unprocessed signal with that amount of samples. A problem that cannot be solved, is that a processed signal with too much speech distortion will result in values that do not represent the noise suppression. This is due to the subtraction removing something that no longer exist in the processed signal, thus adding to the signal.

The difference between the values, calculated with `comp_snr`, of a processed and unprocessed signal will be presented in this thesis. Values will be shown in decibel, with higher values being better, and the measurement will be referred to as SNR improvement.

2.6.4 PESQ improvement

PESQ, which stands for Perceptual Evaluation of Speech Quality, is a telecom industry standard. It is used to evaluate the quality of speech in communication systems. PESQ has been designed to try and describe the actual quality, as perceived by a user. This measurement was chosen in order to present results with a value that follows a widely used standard, which can then be easily compared to other systems. The MATLAB function `pesq`, taken from [1], was used to run the tests needed, and calculate the PESQ value. This function uses the International Telecommunication Union (ITU) standard P.862.

A PESQ value range from -0.5 to 4.5, where higher scores means better quality. In this thesis, the difference between PESQ values for a processed and unprocessed signal will be presented. A larger value is better, and the measurement will be referred to as PESQ improvement.

Chapter 3

Results

The results obtained from the tests, described in the previous chapter, are presented here. Graphs are used when appropriate, otherwise the results are shown in tables.

3.1 Simulation Results

Results for the simulated environments are presented in this section. It was decided to present only speech distortion and noise suppression for these results, because of the simple fact that the performance is reflected well enough by these measurements.

The changes in speech distortion were fairly small for varied array sizes, and the behaviour was not consistent. Therefore, the speech distortion results are shown as the mean value of the minimum and maximum value, together with the maximum deviation. Speech distortion results are shown in table 3.1.

Noise suppression results are displayed as graphs. Each figure shows one graph for the LS beamformer, and one graph for the SNIB. The results can be seen in figures 3.1-3.6. If the width of an array exceeds 13 cm, a point corresponding to an array width of 13 cm is marked on the graph. This is done in order to illustrate the performance range to be expected, for the size of the Axis A8004-VE.

Array	Speech Distortion (dB)	
	LS	**SNIB**
Linear(U)	-57.2 (± 2.8)	-53.7 (± 0.9)
Linear(N)	-57.0 (± 0.9)	-53.6 (± 0.5)
Rectangular	-55.1 (± 0.6)	-53.3 (± 0.5)
Trapezoidal	-55.5 (± 0.2)	-53.4 (± 0.4)
Circular	-55.6 (± 0.8)	-53.4 (± 0.4)
Triangular	-56.1 (± 1.7)	-54.5 (± 1.7)

Table 3.1: Speech distortion for the simulated microphone arrays

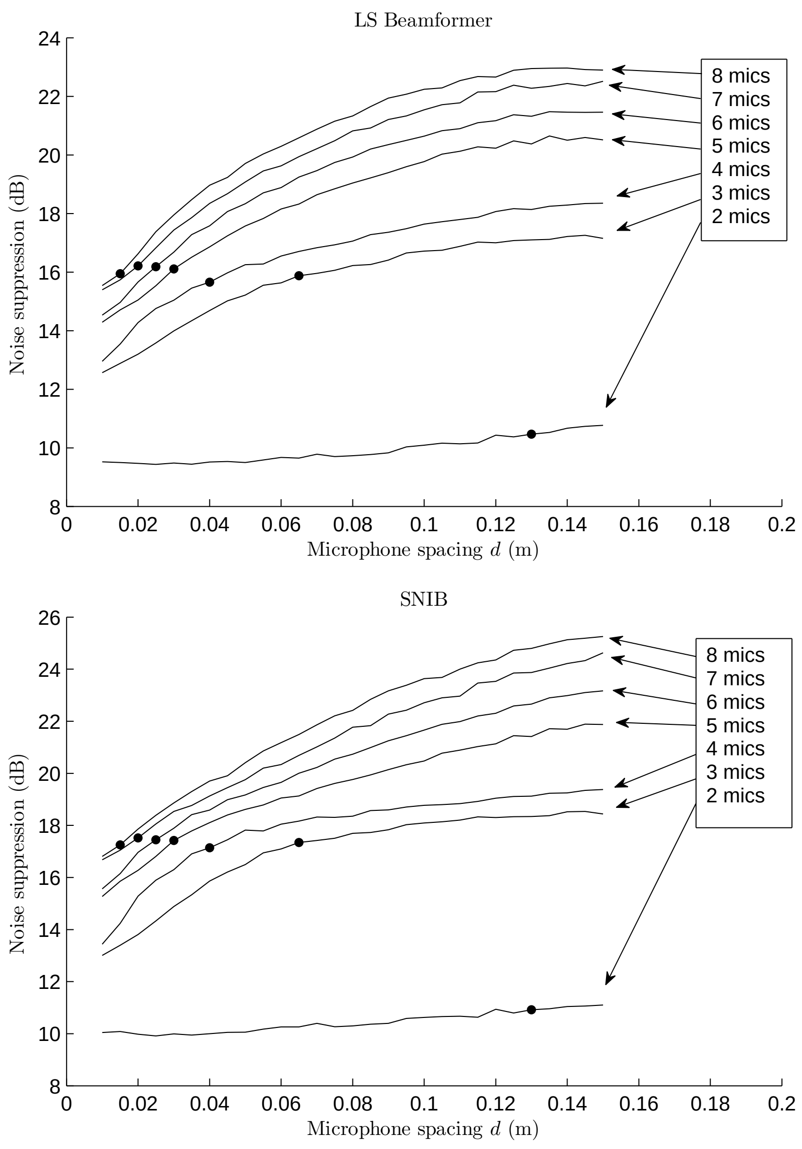
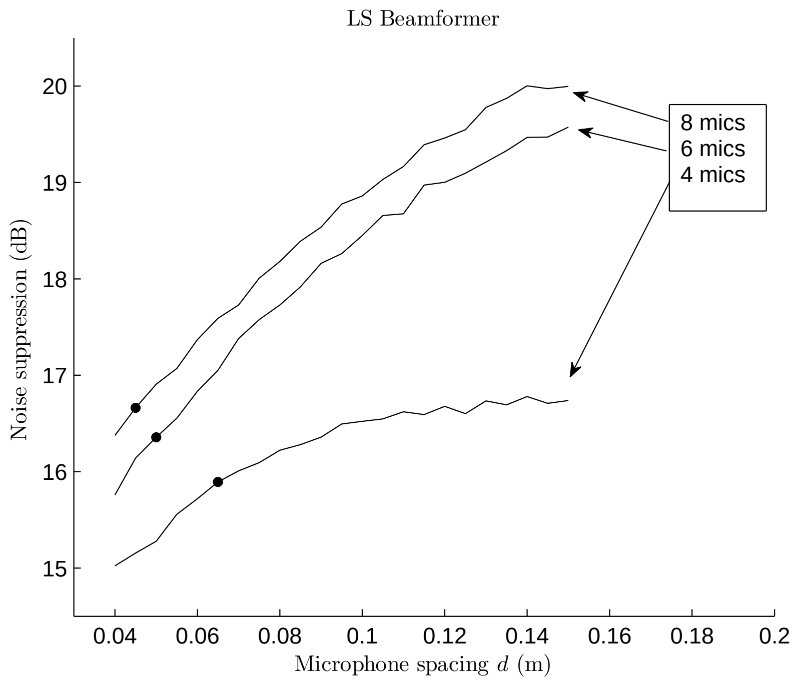


Figure 3.1: Noise suppression for simulated **uniform linear** arrays, with different number of microphones and varied microphone spacing



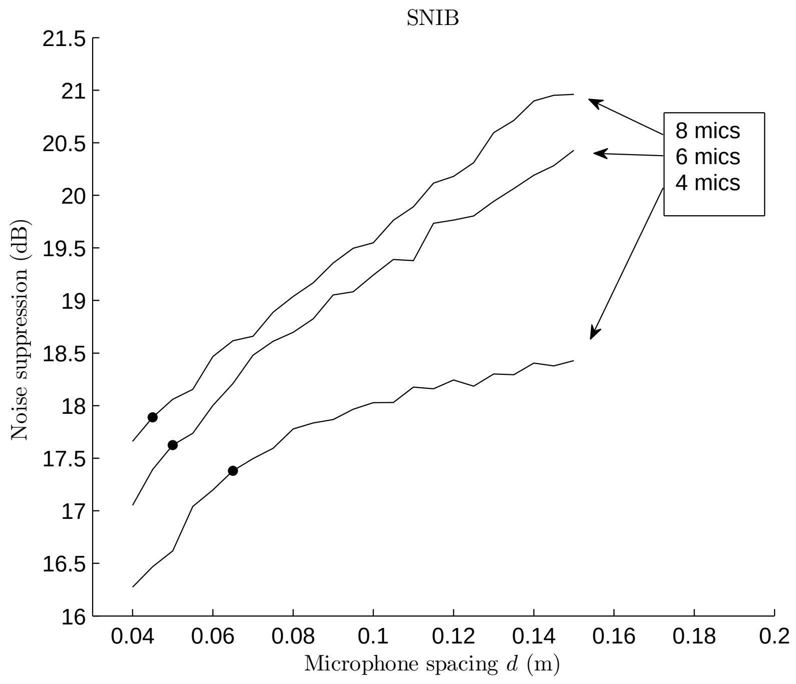


Figure 3.2: Noise suppression for simulated **non-uniform linear** arrays, with different number of microphones and varied microphone spacing

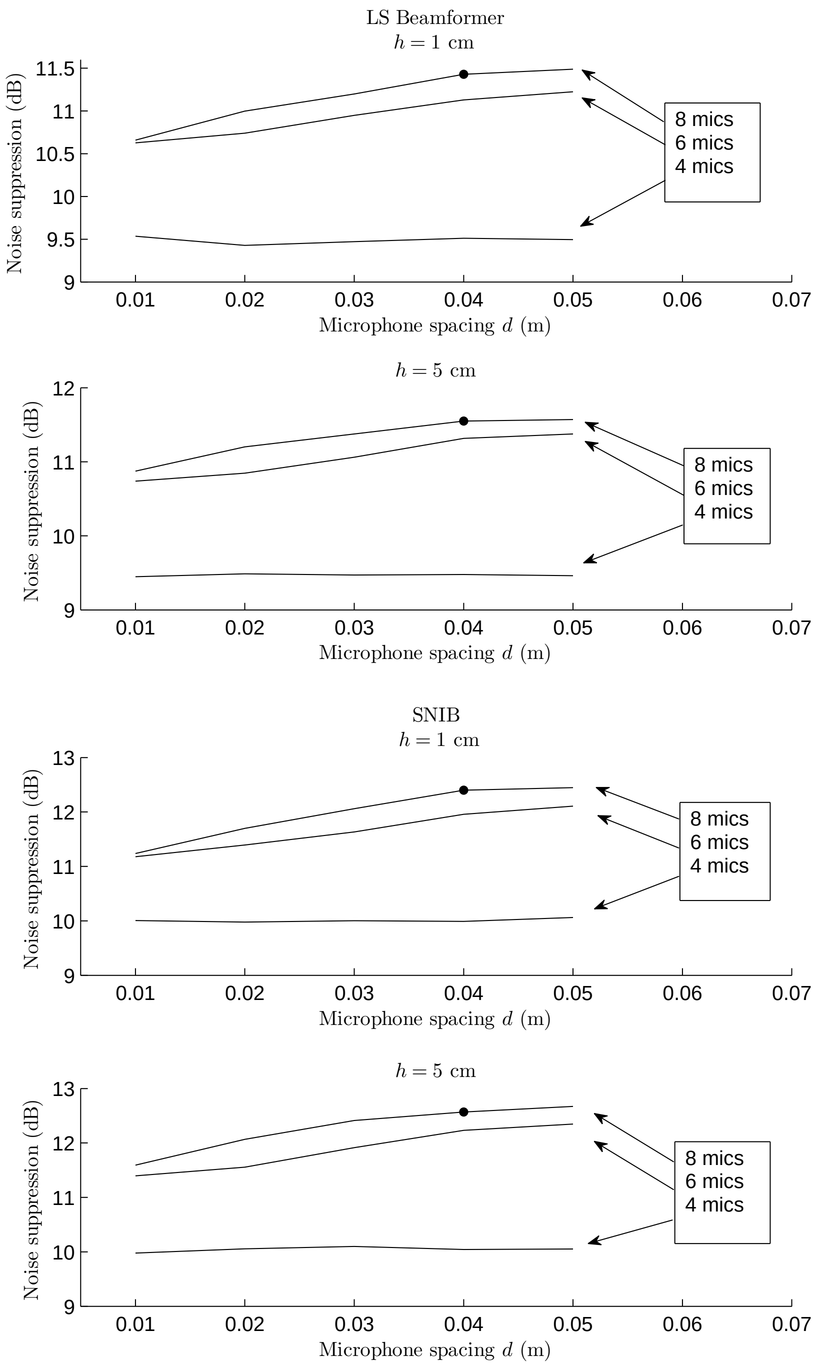


Figure 3.3: Noise suppression for simulated **rectangular** arrays, with different number of microphones and varied microphone spacing

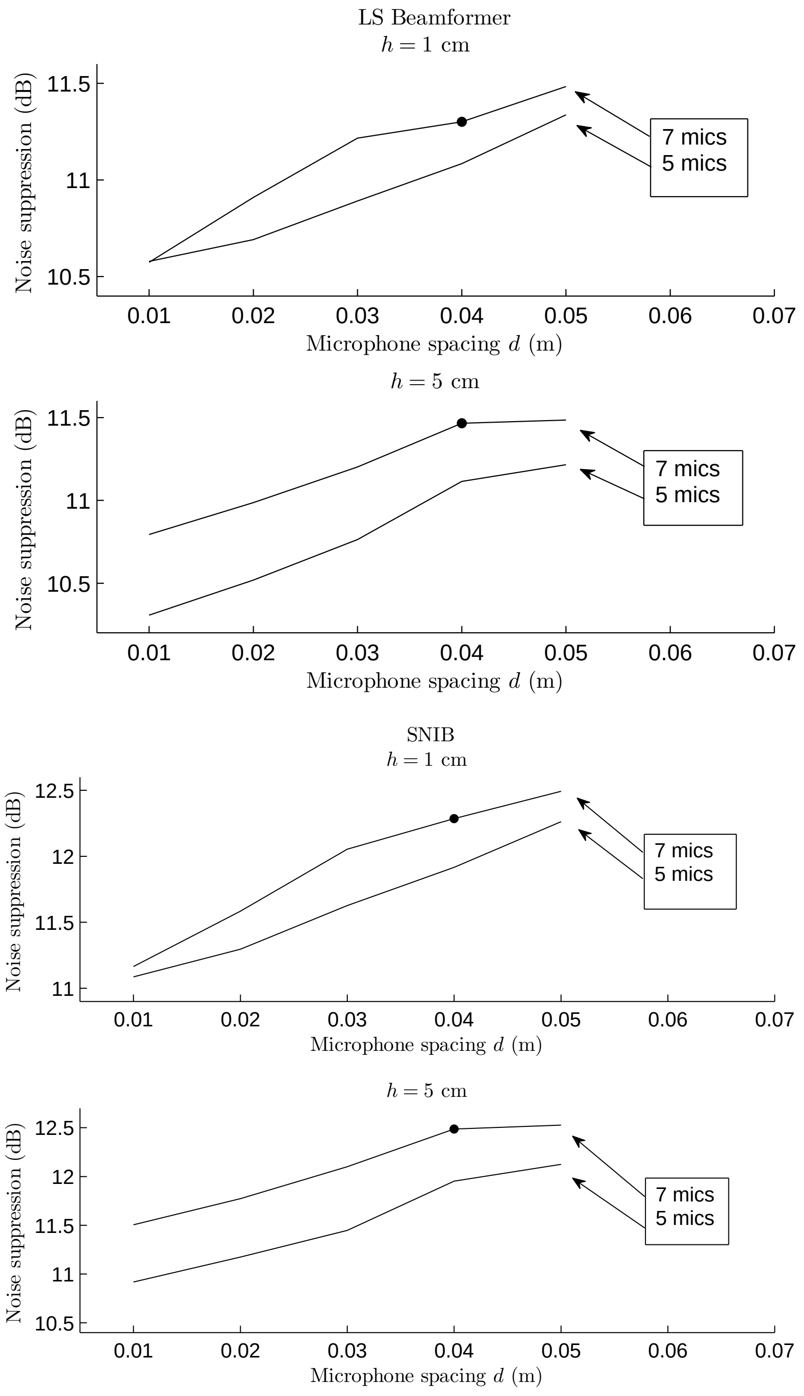
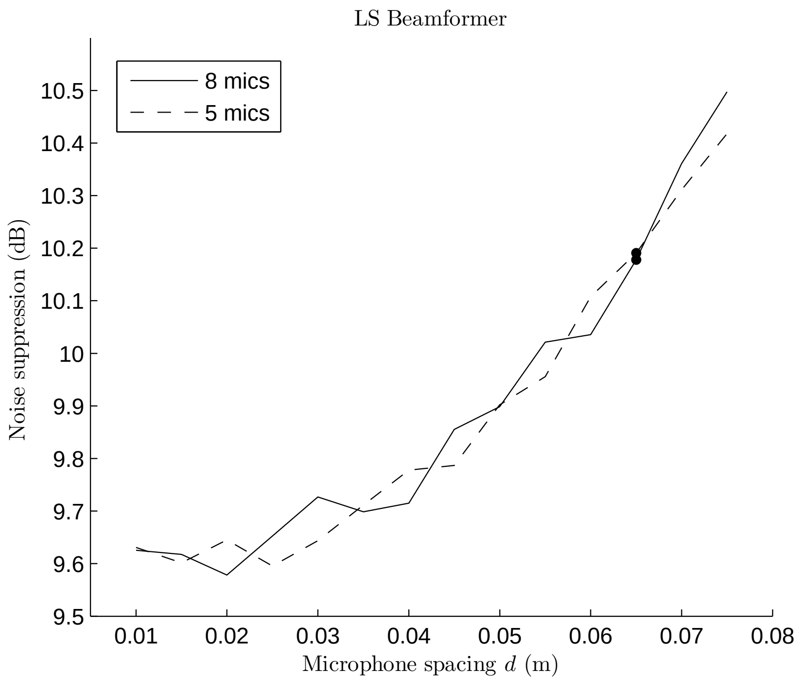


Figure 3.4: Noise suppression for simulated **trapezoidal** arrays, with different number of microphones and varied microphone spacing



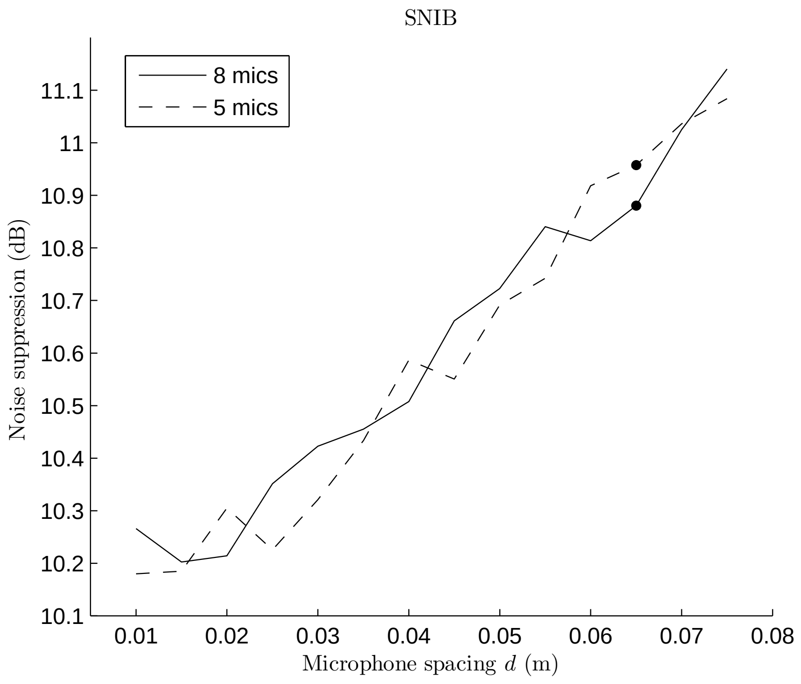


Figure 3.5: Noise suppression for simulated **circular** arrays, with different number of microphones and varied microphone spacing

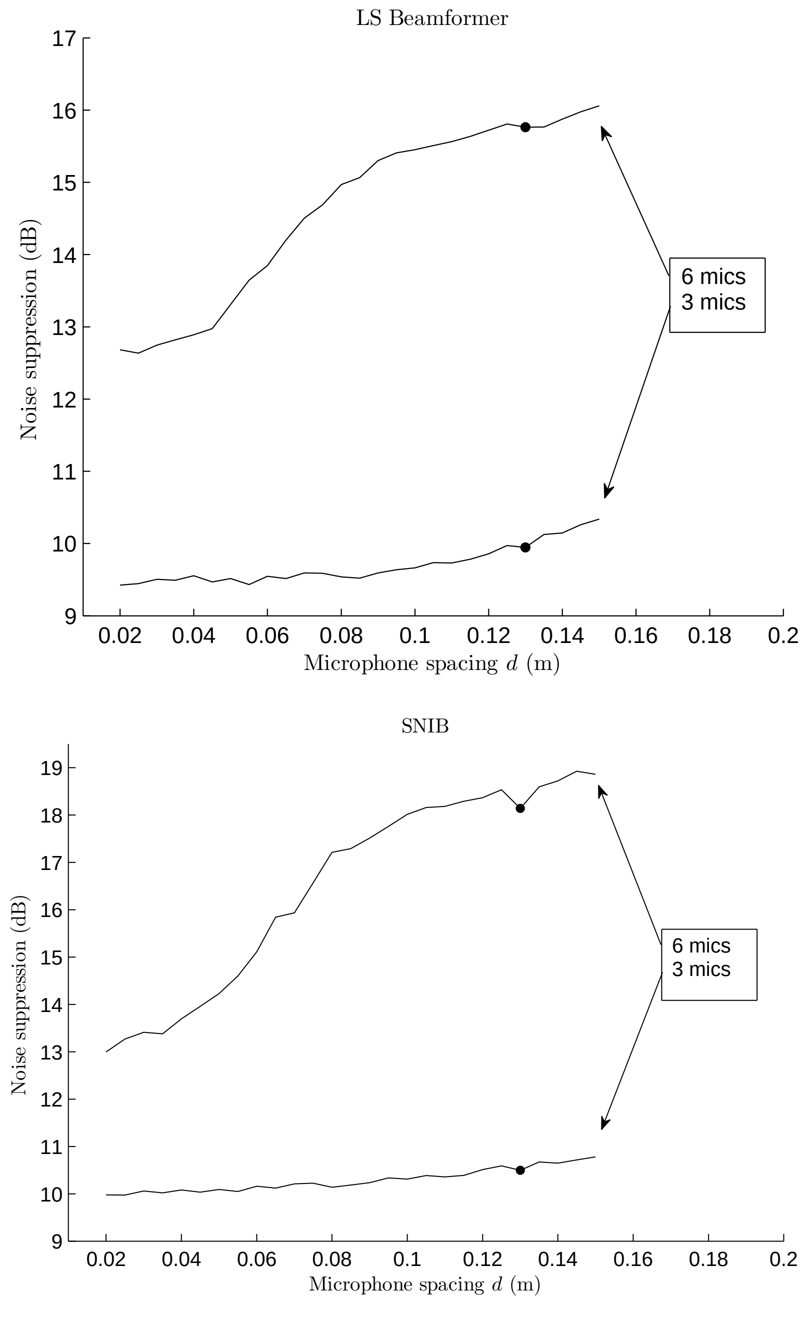


Figure 3.6: Noise suppression for simulated **triangular** arrays, with different number of microphones and varied microphone spacing

3.2 Real-time results

Results obtained with the real-time beamformers, in the real environments, are presented in this section. All four measurements are presented for each array, and results for the variations of the algorithms are shown in section 3.2.1.

Results from the anechoic chamber are displayed in figures 3.7, 3.8, and tables 3.2-3.7. Results obtained in the reverberant environment are shown in tables 3.8-3.11.

Micro-phones	d (cm)	Noise suppression (dB)		Speech distortion (dB)	
		LS	**SNIB**	**LS**	**SNIB**
3	1.85	15.6	8.2	-68.6	-59.2
	3.70	16.9	12.0	-69.7	-60.1
	5.55	17.8	14.8	-70.7	-59.8
4	1.85	16.8	8.7	-70.1	-59.0
	3.70	18.0	14.6	-71.2	-60.0
5	1.85	17.6	10.5	-70.8	-59.1
6	1.85	18.0	9.8	-71.4	-59.7
7	1.85	18.4	9.4	-71.2	-60.4
8	1.85	18.7	10.6	-71.3	-61.7

Table 3.2: Noise suppression and speech distortion for **uniform linear** arrays in the anechoic chamber

Micro-phones	d (cm)	PESQ improvement		SNR improvement (dB)	
		LS	**SNIB**	**LS**	**SNIB**
3	1.85	0.58	-0.04	14.6	1.2
	3.70	0.72	0.46	15.6	0.3
	5.55	0.87	0.57	16.5	0.0
4	1.85	0.71	0.11	9.9	1.6
	3.70	0.90	0.38	16.9	0.0
5	1.85	0.80	0.12	16.5	1.1
6	1.85	0.86	0.04	16.9	1.2
7	1.85	0.91	0.10	17.1	0.8
8	1.85	0.98	0.15	17.4	0.4

Table 3.3: PESQ and SNR improvement for **uniform linear** arrays in the anechoic chamber

Micro-phones	d (cm)	Noise suppression (dB)		Speech distortion (dB)	
		LS	**SNIB**	**LS**	**SNIB**
3	1.85	14.5	7.1	-64.7	-59.5
	3.70	15.0	8.6	-66.5	-60.5
	6.50	15.8	11.5	-70.1	-59.6
	13.00	16.2	14.8	-70.8	-61.4
6	3.70	16.5	6.2	-71.5	-59.2
	13.00	17.5	11.3	-71.3	-60.6

Table 3.4: Noise suppression and speech distortion for **triangular** arrays in the anechoic chamber

Micro-phones	d (cm)	PESQ improvement		SNR improvement (dB)	
		LS	**SNIB**	**LS**	**SNIB**
3	1.85	0.44	0.17	11.7	1.2
	3.70	0.54	0.35	13.3	0.3
	6.50	0.76	0.63	14.8	-0.1
	13.00	0.85	0.47	15.3	-0.5
6	3.70	0.73	-0.06	15.8	1.2
	13.00	1.00	0.36	16.5	0.1

Table 3.5: PESQ and SNR improvement for **triangular** arrays in the anechoic chamber

Micro-phones	d (cm)	Noise suppression (dB)		Speech distortion (dB)	
		LS	**SNIB**	**LS**	**SNIB**
4	5.55	18.0	15.4	-70.8	-59.9

Table 3.6: Noise suppression and speech distortion for a **non-uniform linear** array in the anechoic chamber

Micro-phones	d (cm)	PESQ improvement		SNR improvement (dB)	
		LS	**SNIB**	**LS**	**SNIB**
4	5.55	0.89	0.27	16.7	-0.1

Table 3.7: PESQ and SNR improvement for a **non-uniform linear** array in the anechoic chamber

Micro-phones	d (cm)	Noise suppression (dB)		Speech distortion (dB)	
		LS	**SNIB**	**LS**	**SNIB**
3	5.55	17.4	14.8	-65.9	-57.5

Table 3.8: Noise suppression and speech distortion for a **uniform linear** array in the reverberant environment

Micro-phones	d (cm)	PESQ improvement		SNR improvement (dB)	
		LS	**SNIB**	**LS**	**SNIB**
3	5.55	0.67	0.27	15.4	0.6

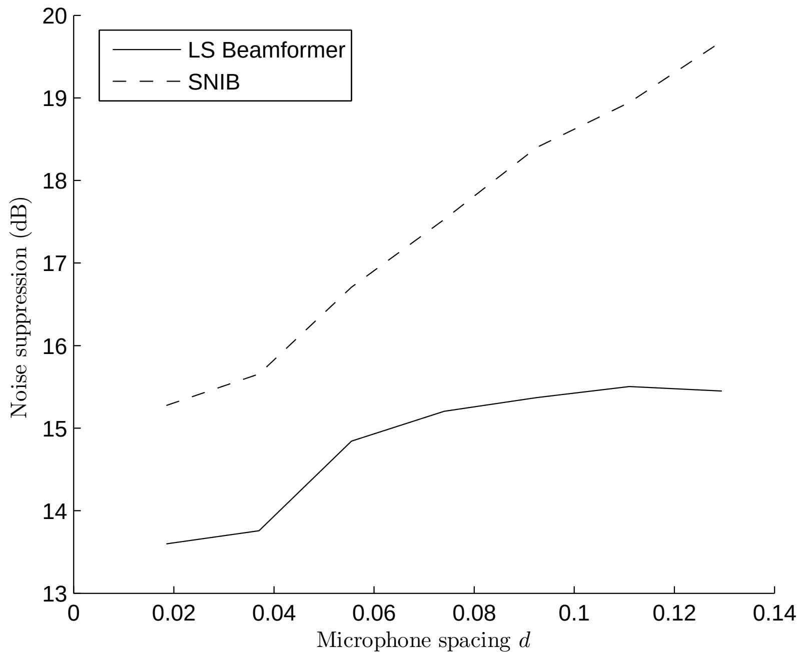
Table 3.9: PESQ and SNR improvement for a **uniform linear** array in the reverberant environment

Micro-phones	d (cm)	Noise suppression (dB)		Speech distortion (dB)	
		LS	**SNIB**	**LS**	**SNIB**
3	13.00	17.4	19.4	-64.7	-59.0
6	13.00	19.1	17.5	-68.5	-58.8

Table 3.10: Noise suppression and speech distortion for **triangular** arrays in the reverberant environment

Micro-phones	d (cm)	PESQ improvement		SNR improvement (dB)	
		LS	**SNIB**	**LS**	**SNIB**
3	13.00	0.73	0.51	14.8	0.5
6		1.09	0.68	17.2	0.9

Table 3.11: PESQ and SNR improvement for **triangular** arrays in the reverberant environment



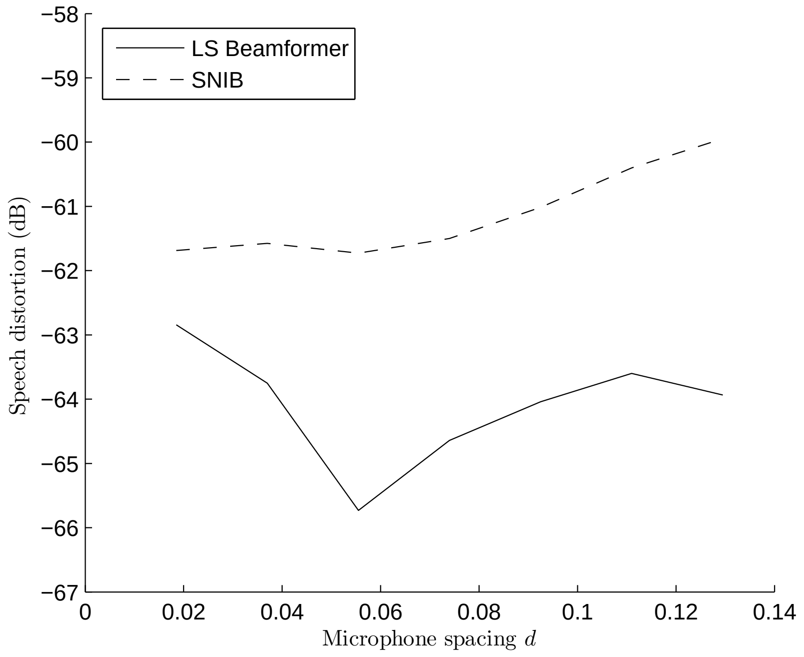


Figure 3.7: Noise suppression and speech distortion for **linear** arrays, with 2 microphones and varied microphone spacing, in the anechoic chamber

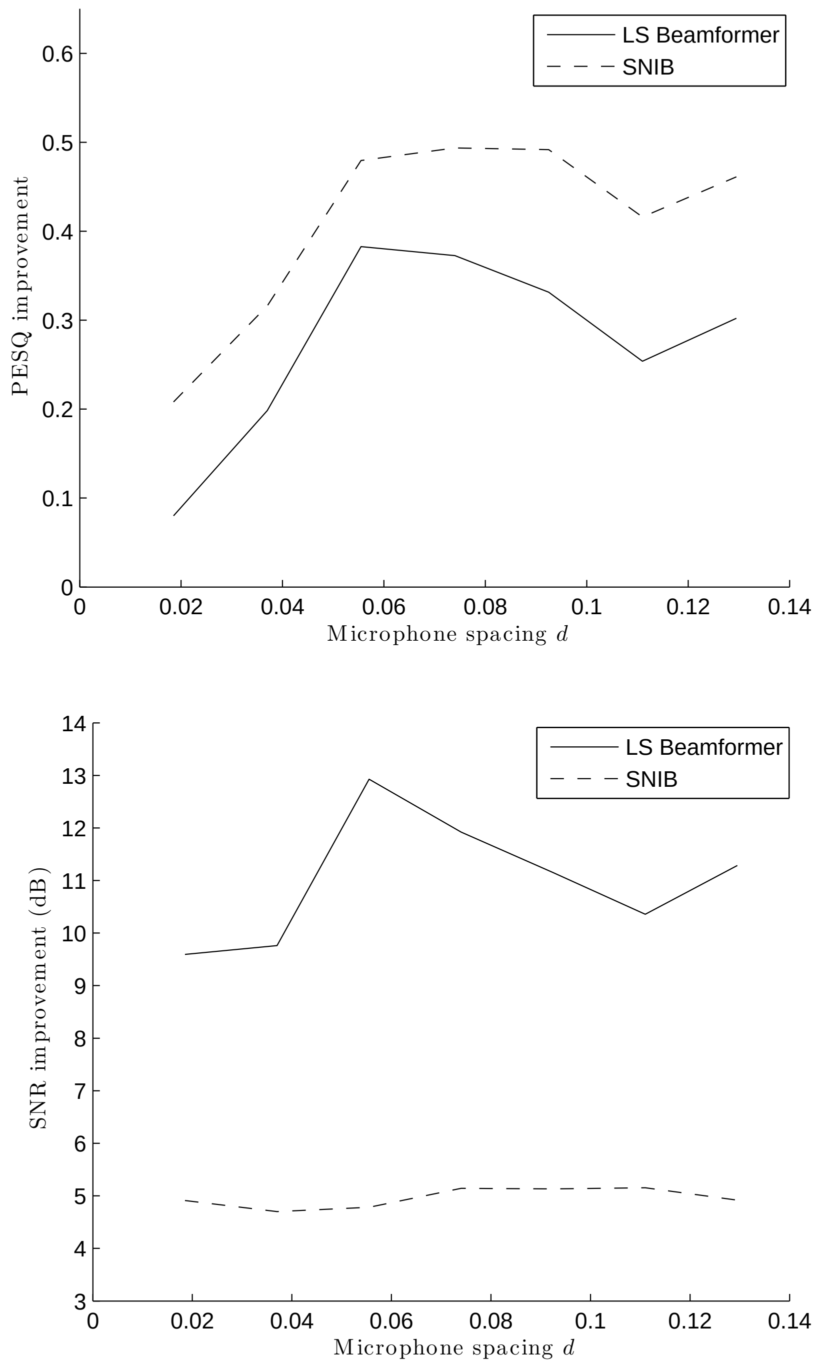
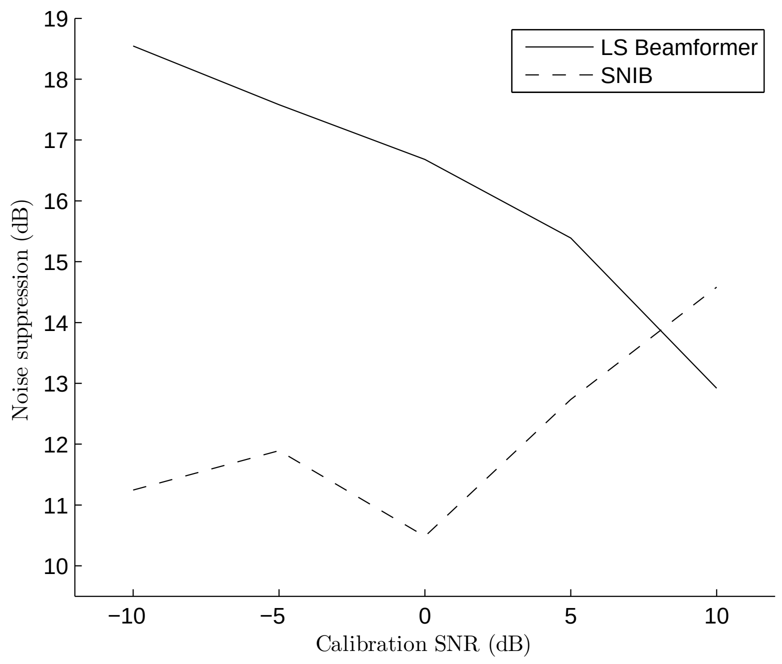


Figure 3.8: PESQ and SNR improvement for **linear** arrays, with 2 microphones and varied microphone spacing, in the anechoic chamber

3.2.1 Algorithm Variations

The results of varying the properties described in section 2.3.6 are presented here. Graphs were created, with all four measurement presented, for each property. Figures 3.9 and 3.10 shows the result of varying the calibration SNR. Results for a varied number of subbands in the filter bank are displayed in figures 3.11 and 3.12, and the results from varying the number of filter taps can be seen in figures 3.13 and 3.14.



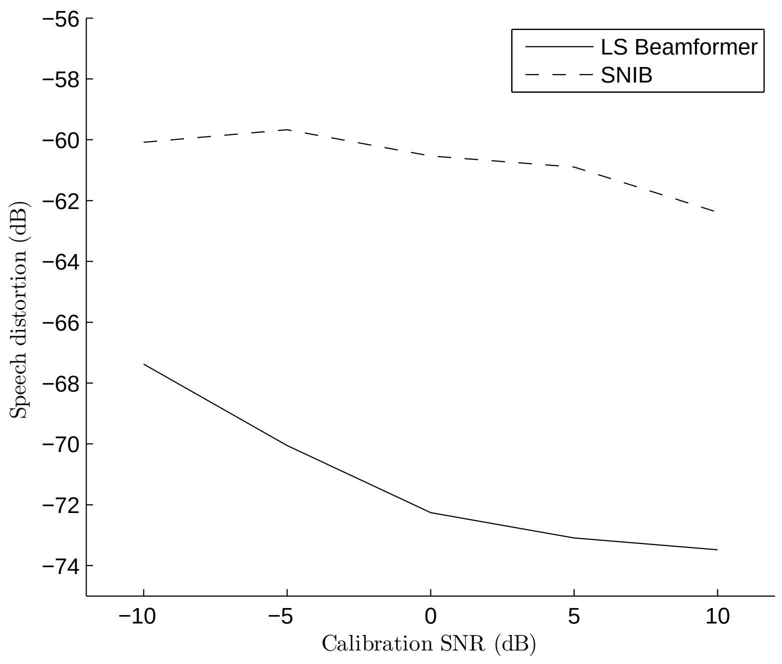
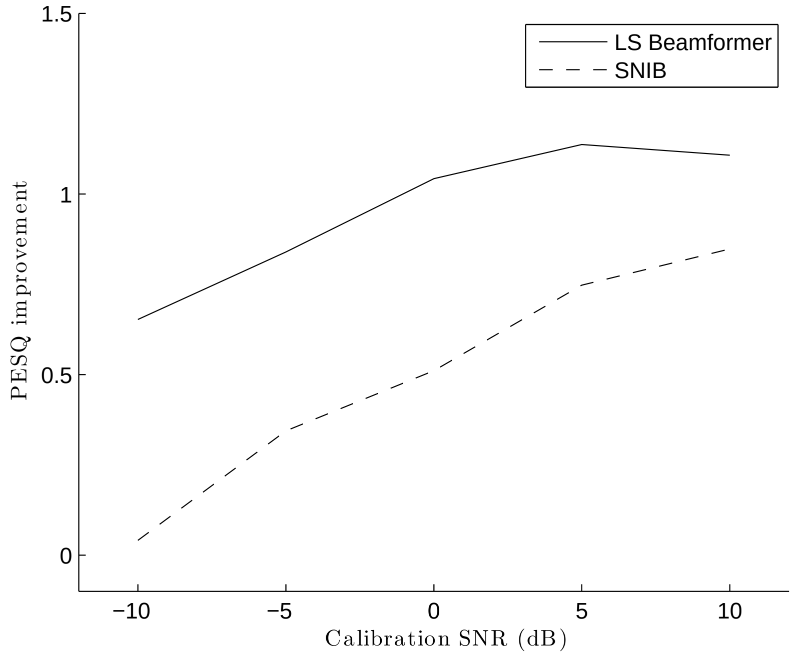


Figure 3.9: Noise suppression and speech distortion for varied calibration SNR in the anechoic chamber



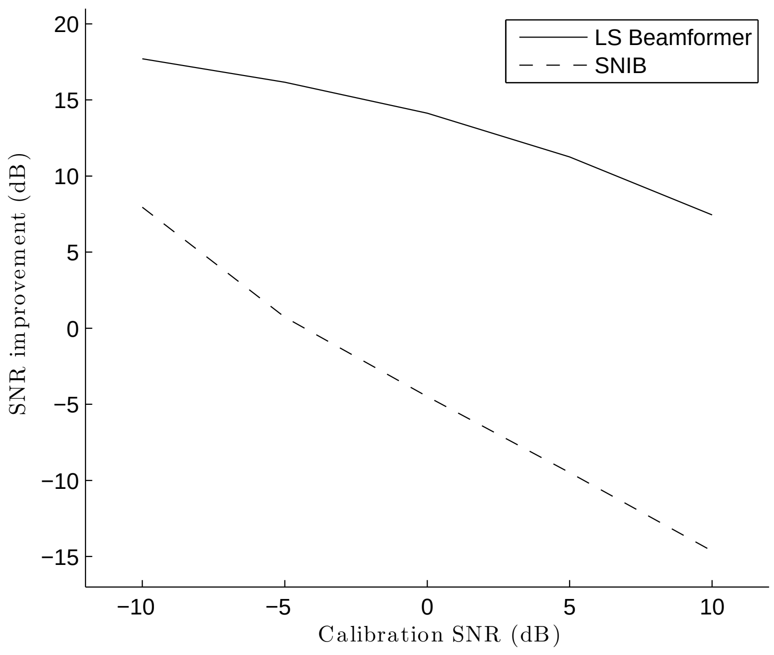
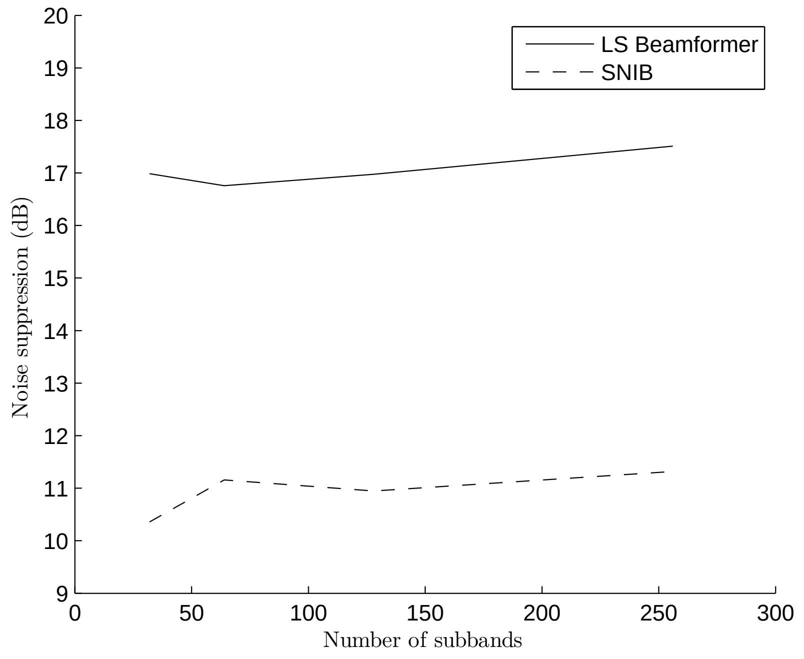


Figure 3.10: PESQ and SNR improvement for varied calibration SNR in the anechoic chamber



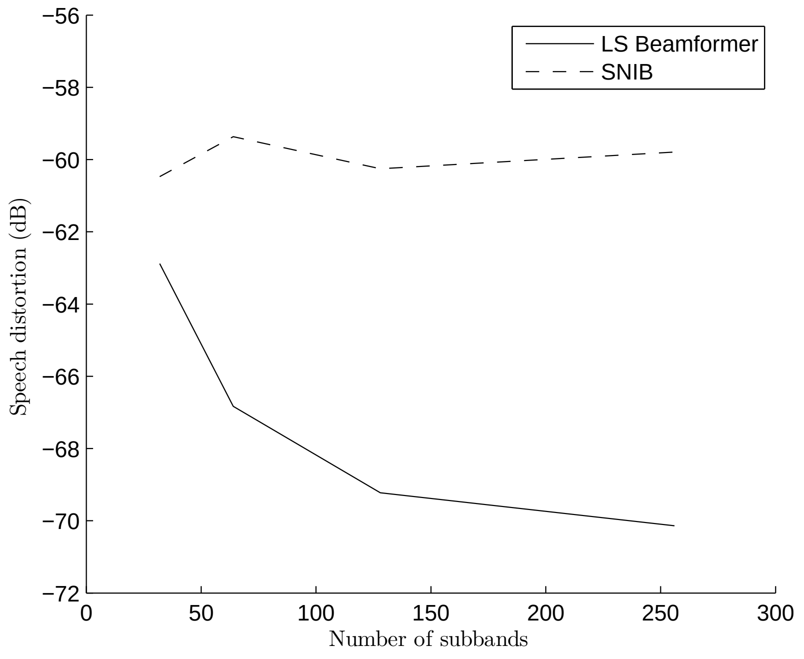
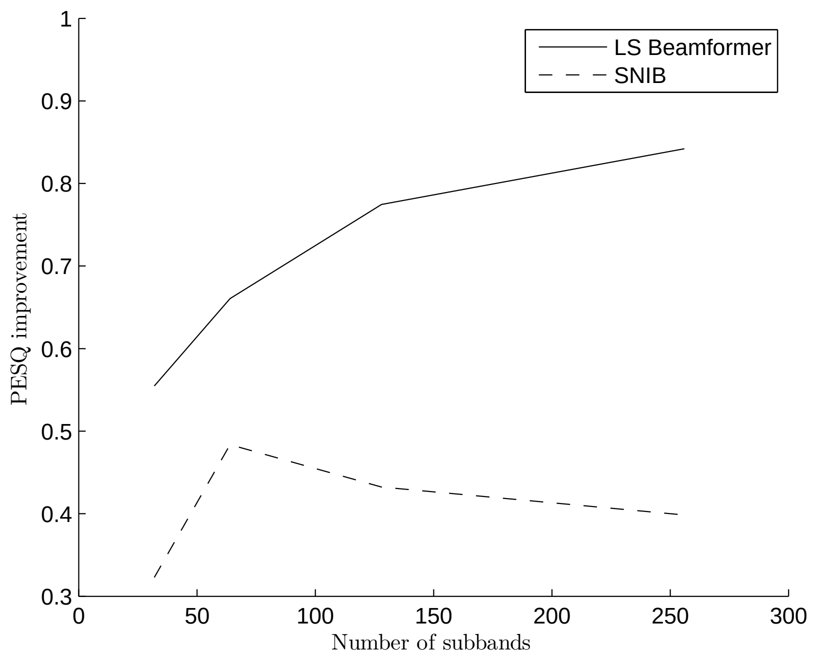


Figure 3.11: Noise suppression and speech distortion for varied number of filter bank subbands in the anechoic chamber



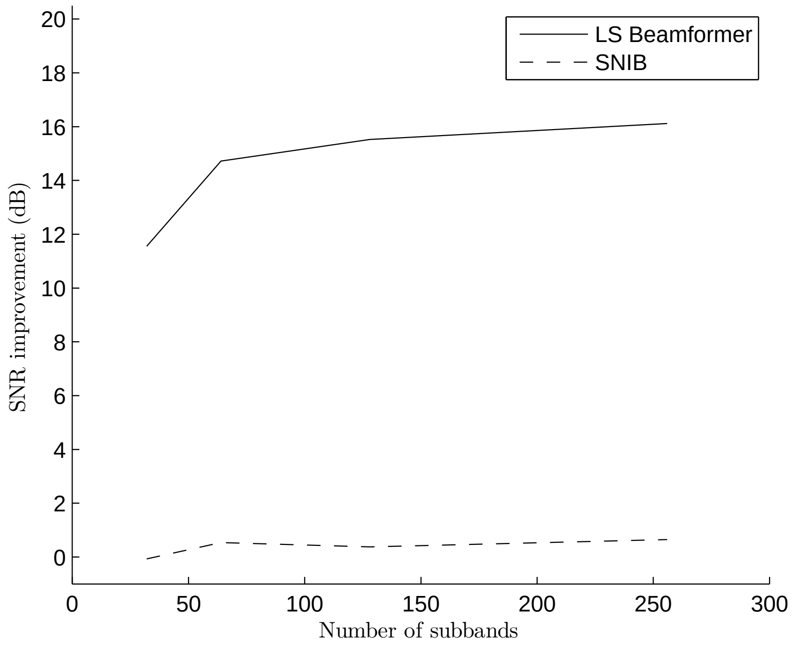
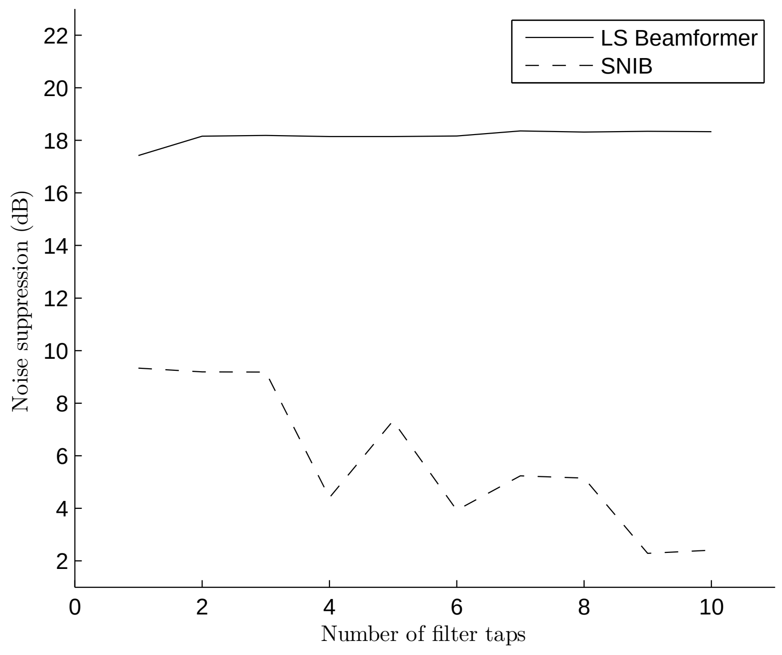


Figure 3.12: PESQ and SNR improvement for varied number of filter bank subbands in the anechoic chamber



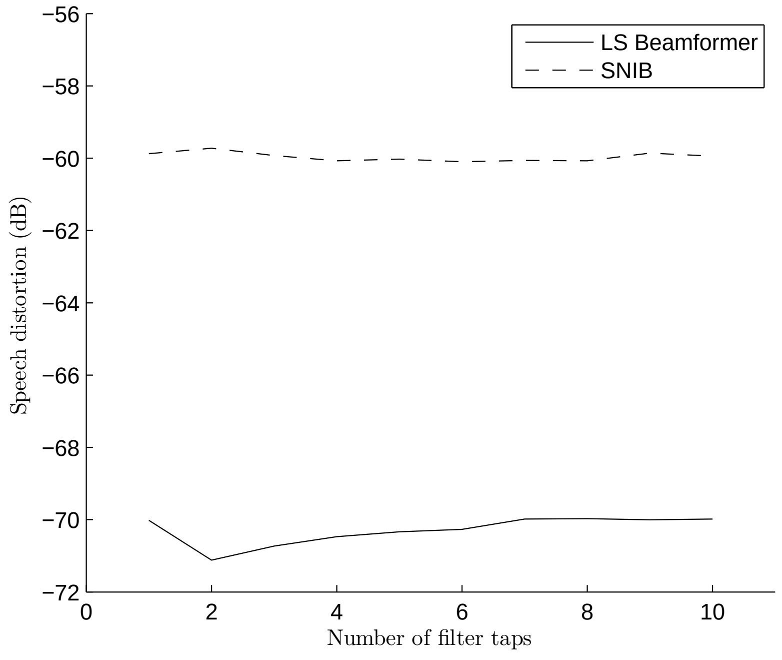
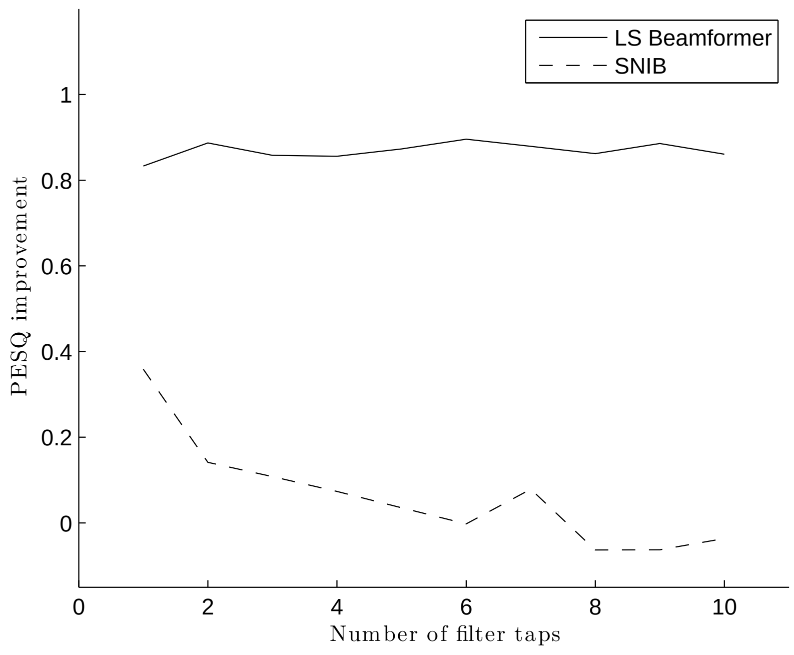


Figure 3.13: Noise suppression and speech distortion for varied number of subband filter taps in the anechoic chamber



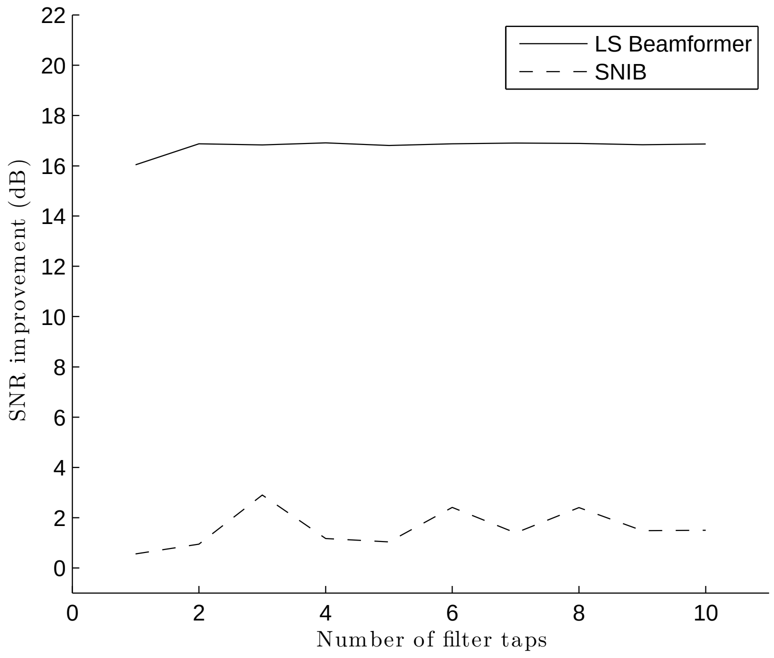


Figure 3.14: PESQ and SNR improvement for varied number of subband filter taps in the anechoic chamber

3.3 Single Microphone Algorithms

The results for the single microphone algorithms are presented in this section. They are displayed in two tables, with noise suppression and speech distortion shown in table 3.12, and PESQ and SNR improvement displayed in table 3.13.

Algorithm	Test SNR (dB)	Noise suppression (dB)	Speech distortion (dB)
Spectral Subtraction	-5	10.7	-59.5
	0	14.7	-62.9
	5	15.2	-66.5
Subspace tracking	-5	17.1	-61.7
	0	15.2	-63.6
	5	12.5	-66.3
Wiener filter	-5	9.2	-63.4
	0	9.1	-65.2
	5	8.4	-67.2

Table 3.12: Noise suppression and speech distortion for the single microphone algorithms

Algorithm	Test SNR (dB)	PESQ improvement	SNR improvement (dB)
Spectral Subtraction	-5	-0.32	8.5
	0	0.05	10.5
	5	0.42	14.3
Subspace tracking	-5	-0.01	14.0
	0	0.35	14.5
	5	0.63	16.5
Wiener filter	-5	0.03	11.5
	0	0.29	12.5
	5	0.52	14.9

Table 3.13: PESQ and SNR improvement for the single microphone algorithms

3.4 Axis A8004-VE

This section contains the results obtained with the Axis A8004-VE. Because the device uses a black box solution, only the PESQ and SNR improvement could be measured. Results are show in table 3.14.

Test SNR (dB)	PESQ improvement	SNR improvement (dB)
-5	0.10	5.7
0	0.14	5.4
5	0.20	6.4

Table 3.14: PESQ and SNR improvement for Axis A8004-VE

Chapter 4

Analysis

In this chapter the authors present their analysis of the results, starting with how well the measurements reflect performance. Simulation and real-time performance of the beamforming algorithms are analysed separately, with focus on the microphone configurations for the simulation, and the algorithms for the real-time results. The single microphone algorithms are anlysed and compared to the beamforming algorithms. A short analysis of the A8004-VE is done, and finally a solution for the application is proposed.

4.1 Measurements

When possible to calculate, the noise suppression and speech distortion measurement reflect the performance of an algorithm quite well. It should be noted that the speech distortion should only be used to compare values obtained with the same speech signal. If the power in the speech signal is altered, the baseline of the measurement will be altered. This means that if the power of the speech signal used for testing is increased or decreased, the speech distortion measurement will also increase or decrease.

The SNR improvement measurement manages to reflect noise suppression decently if the speech is not distorted, but quickly begin to show very misleading values as the speech distortion increases. Because of this, the measurement might not achieve the intended goal of being able to compare black box algorithms to the beamformers, and it is important to keep this in mind when reviewing the results. On the other hand, PESQ improvement reflects how signals were enhanced quite well, making it useful for comparing quality in different solutions.

The authors conclusions about the measurements are based on listening to the signals with various speakers and headphones. It should be noted that it is very hard to convey the perceived quality of sound through numbers.

4.2 Beamforming algorithms

The analysis of the beamforming algorithms is divided into two parts, one for the simulation and one for the real-time implementation. This is done because simulation tests were done in order to evaluate performance with different microphone configurations, and real-time tests were done in order to evaluate actual performance in real environments.

4.2.1 Simulation

The simulation results suggest that the behaviour of the LS beamformer and the SNIB is very similar, with the SNIB having overall better noise suppression and worse speech distortion. While the speech distortion for the LS beamformer showed some variation, although not consistent, the speech distortion for the SNIB seemed fairly stable. As the test environment for linear and planar arrays differed, their behaviours will be discussed separately.

Linear arrays

The most noticeable difference between linear arrays, is that a two-microphone array performs much worse than the other arrays. Interestingly, the performance also seems to improve much slower when the microphone spacing is increased, compared to the arrays with more microphones. After comparing beampatterns, the authors conclude that the behaviour of the two-microphone array is due to the algorithms producing a very wide main beam for all frequencies.

For linear arrays with more than two microphones, the results suggest that only a small performance increase can be achieved by using more than three microphones, considering the size of the A8004-VE. This implies that if the size of the array is limited to 13 cm, it will be unnecessary to use more than three microphones.

An interesting thing to note about the linear arrays is that there is no apparent sign of spatial aliasing in the results, even though several of the arrays violate the condition for this to occur. All of the arrays seem to perform well, even at the largest dimensions. The authors link this behaviour to the calibration signal being a representation of human speech. Higher frequencies, which may cause spatial aliasing, are not prevalent in human speech. Because of this higher frequencies are being suppressed. Non-uniform linear arrays showed no distinct difference in performance, compared to the uniform linear arrays.

Planar arrays

The rectangular, trapezoidal and circular arrays behaved very similar, with a fairly small increase in performance with larger array dimensions. Triangular arrays with six microphones seemed to perform quite well, with a noticeable increase in performance with larger arrays. However, the triangular arrays with three microphones showed the same behaviour as the other arrays.

After studying the results and becoming more familiar with the behaviour of different arrays, the authors realized that the optimal planar array would be a disc covered in microphones. It was concluded that the triangular array with six microphones represent the best sampling of such a disc. This can be seen as sampling two microphones each, on three linear arrays with different rotation and midpoints intersecting. The authors believe this to be the reason for its superior performance.

4.2.2 Real-time implementation

Unlike the simulation results, the real-time results show that the performance of the LS beamformer and the SNIB is very different. In the anechoic chamber, the SNIB noise suppression is consistently much worse than for the LS beamformer, except for a linear array with two microphones. The speech distortion behaves as for the simulation, with the SNIB being worse than the LS beamformer, but the difference is more noticeable. Interestingly, the SNIB seems to perform much better in the reverberant environment, producing results similar to the simulation.

LS Beamformer

The behaviour of the algorithm, with different microphone configurations, is consistent with the simulation results. Larger arrays produce higher noise suppression, and the results confirm that not much is gained by using more than three microphones for a linear array with the limited dimensions. An interesting thing to note is that the linear two-microphone arrays and the triangular three-microphone arrays perform better than expected. This might be due to the limited size of the noise sources, resulting in smaller noise fields than for the simulation.

Results from the anechoic chamber suggest that the performance of triangular arrays is worse than for linear arrays. The authors believe that this behaviour is linked to the positions of the noise sources, and the results from the reverberant environment support this conclusion. These results show that the performance increases slightly when using a triangular array. This points to the fact that a linear array can only attenuate noise in a plane rotated around its axis, creating a situation where some of the reverberations from the floor and ceiling cannot be attenuated at all. Despite this, the performance of the linear array is quite good in the reverberant environment.

Based on the results, the authors consider the LS beamformer to be a very robust solution, capable of good noise reduction in any environment. The PESQ and SNR improvement values are consistent with the noise suppression and speech distortion, implying that the algorithm does not distort the speech signal very much. This was confirmed by listening to the signals, and the authors conclude that the speech distortion created by this algorithm is negligible.

SNIB

In comparison to the LS beamformer, the performance of the SNIB is very different from the simulation. The behaviour when varying array size is still somewhat consistent, with larger arrays resulting in better noise suppression. Surprisingly, the SNIB performs worse when an increased number of microphones is used, and the array with the best performance is actually the linear array with two microphones. The authors link this behaviour to the calculation of the SNIB weights being numerically sensitive. Increasing the number of microphones creates larger autocovariance matrices, making ill-conditioned calculations worse.

Speech distortion for the SNIB is relatively stable, but the SNR improvement values does not reflect noise suppression. The PESQ improvement values implies that the distortion of the speech is very high, with some values even being negative. This was confirmed by listening to the signals, and the authors consider the speech distortion of this algorithm to be quite high.

Interestingly, the performance of the SNIB improved in the reverberant environment, even outperforming the LS beamformer in one case, considering noise suppression. Speech distortion was still quite bad, which is reflected by the PESQ and SNR improvement. According to the authors, this might be related to the reverberations acting as additional noise sources, creating a noise autocovariance matrix of higher rank, resulting in better conditioned calculations.

Considering the results, the authors conclude that the SNIB, as implemented here, is not a very good solution. The algorithms ability to reduce noise is very sensitive to different situations, and the speech distortion is always quite high.

Algorithm variations

Varying the calibration SNR for the algorithms seemed to affect them in opposite ways, considering noise suppression. While the noise suppression of the LS beamformer decreased with increased SNR, the SNIB showed some improvement. The behaviour of the SNIB was somewhat inconsistent, pointing to sensitivity, and the speech distortion was not improved much. On the other hand, the LS beamformer behaved consistently, and the results show that a trade-off between noise suppression and speech distortion exists. A lower calibration SNR results in higher noise suppression, but also more speech distortion.

SNR improvement for a varied calibration SNR shows some interesting results for the SNIB. It suggests that there is an increased speech distortion with higher SNR. However, the PESQ improvement does not reflect this, and by listening to the signals the authors determined that the speech distortion was actually decreased with increased SNR. These results could reflect a slight phase distortion in the subbands, which might not be apparent to a listener. Such phase distortion will have a large impact on the SNR improvement, because of the way the measurement is calculated. According to the authors, this is another evidence for the sensitivity of the SNIB.

Varying the subbands in the filter bank seemed to have little effect on noise suppression for both algorithms. A performance increase for the speech distortion could be seen with an increased number of subbands for the LS beamformer, while the SNIB showed no such behaviour. The results for the LS beamformer indicates that a trade-off between computational complexity and speech quality exists. More subbands require more calculations, and a longer delay in the output signal. According to the authors, this can be viewed as resolution in the filter bank. With less subbands, each subband contain a wider band of frequencies, resulting in inexact manipulation of some frequencies. PESQ and SNR improvement showed no irregular values for these results.

An increased number of filter taps showed no noticeable improvement in performance for the LS beamformer, and the SNIB actually displayed a decrease in noise suppression. The authors consider this as another proof of the sensitivity of the SNIB. As with more microphones, this behaviour can be linked to larger covariance matrices. Increasing the number of filter taps creates bigger matrices, resulting in ill-conditioned calculations.

4.3 Single microphone algorithms

The results for the three single microphone algorithms indicate that their performance is highly dependent on the environment. Varying the SNR produce very different results, with both suppression of noise and distortion of speech being affected. This is the expected behaviour, because the algorithms are adaptive and use VAD. In order to produce good results for an environment, these algorithms need to be tuned to that specific environment.

Listening to the signals made it evident that musical noise was present for all three algorithms, and that the distortion of the speech was quite bad for the -5 dB SNR. The authors conclude that these algorithms might perform quite well in situations with a fairly static SNR, and that their performance would be reduced drastically for situations where the SNR can change a lot.

4.3.1 Comparing performance to the beamforming algorithms

Comparing the single microphone and the beamforming algorithms should be done with the application in mind. As the beamforming algorithms act as static filters, they will always suppress noise and distort speech in the same way, depending on the angle of incidence. The single microphone algorithms, on the other hand, will suppress noise and distort speech depending on the SNR, and the type of noise. This means that if the location of the person producing the speech signal is known, the beamforming algorithms will be superior. If the environment is also likely to create a wide range of SNRs, the performance could even be far superior. However, if the position of the person is not known, the single microphone algorithms will perform better.

It should be noted that there is no reason the beamforming and single microphone algorithms could not be combined. Knowing the position of the person producing speech is still required, but the authors believe this type of solution might be able to produce an even better result. Since a beamforming algorithm will make the noise level more controlled, a single microphone algorithm can be tuned to a situation with less variation.

4.4 Axis A8004-VE

Since the A8004-VE uses a black box solution, the authors have no knowledge of how the noise reduction algorithm works. Based on the obtained results, the authors conclude that the algorithm is tuned to try and achieve low speech distortion, with the result of a reduced noise suppression. When using a single microphone algorithm, this is probably the best solution for high speech quality.

Because the algorithm in the A8004-VE was tested with different equipment, no real comparison can be made with the beamforming algorithms. But after listening to the signals, and considering the application of a door station, the authors believe that the sound quality can be greatly improved by using beamforming.

4.5 Proposed solution

Based on the analysis of the results, and the application of a door station with dimensions of the A8004-VE, the authors propose a solution with three microphones, positioned as a 13 cm uniform linear array, implementing the LS beamforming algorithm. A filter bank with 128 subbands should be used to achieve a very high speech quality, but the number of subbands could be reduced to 64 if some speech distortion is acceptable.

The reason for selecting a linear and not a planar array is the application. When a door station with a camera is used, most of the space in front of the station will be occupied by a person, and thus not producing much noise. As this thesis have shown, using a planar array will require more microphones in order to achieve the same noise suppression as a linear array can achieve in a plane. Considering that a person and the ground will occupy most of the main beam, the authors believe that for this application it is more appropriate to focus the noise suppression to the sides of the person.

References

[1] P. Loizou, *Speech Enhancement: Theory and Practice.* Taylor & Francis, 2007.

[2] W. Liu and S. Weiss, *Wideband beamforming: concepts and techniques.* John Wiley & Sons, 2010.

[3] U. Michel, "History of acoustic beamforming," in *Berlin Beamforming Conference*, 2006.

[4] N. Grbic, *Optimal and adaptive subband beamforming.* Ph. D. dissertation, Blekinge Institute of Technology, 2001.

[5] I. McCowan, "Microphone arrays: A tutorial," *Queensland University, Australia*, 2001.

[6] T. I. Laakso, V. Valimaki, M. Karjalainen, and U. K. Laine, "Splitting the unit delay [fir/all pass filters design]," *Signal Processing Magazine, IEEE*, vol. 13, no. 1, pp. 30–60, 1996.

[7] M. Swartling, *Direction of Arrival Estimation and Localization of Multiple Speech Sources in Enclosed Environments.* Ph. D. dissertation, Blekinge Institute of Technology, 2012.