AALBORG UNIVERSITY
MASTER THESIS

A Parameter Study on the Ray Space Transform Method and its Performance Compared to the Spatial Decomposition Method

AAT1060

Master Thesis

AALBORG UNIVERSITY
MASTER THESIS

Institute for Electronic Systems
Acoustics and Audio Technology
Fredrik Bajers Vej 7B
9220 Aalborg Øst
Tlf.: (+45)99408600
http://es.aau.dk

Title:

A Parameter Study on the Ray Space Transform Method and its Performance Compared to the Spatial Decomposition Method

Project:

AAT10 Master Thesis
Acoustics & Audio Technology

Project period:

Spring 2017

Project group:

1060

Group members:

Simon Boelt Jensen
Gustavo Esteban Chávez Morales

Supervisors:

Christian Sejer Pedersen
Martin Bo Møller

Number of pages: 102
End of project: 08/06/2017

Abstract:

Room acoustics analysis and synthesis using the Spatial Decomposition(SDM) method has become widely known and is used in applications such as room impulse visualization and auralization. The Ray-Space Transform Method(RSTM) is a new framework to perform analysis and synthesis of a room, and, due to the novelty of the method many characteristics have not been studied in depth. The purpose of this thesis is to study the RSTM framework and its paramters, and then determine its advantages and disadvantages and compare it with the SDM method for acoustical room analysis, by comparing the performance on sound source estimation. The thesis defines the significant parameters of the RSTM and their impact on estimation of sound sources. The comparison of the methods is done on parameters which are common for each method. The comparison finds that in many aspects the methods perform similarly, with some exceptions. Following the comparison study, some tests on real data is performed on the RSTM, to verify the simulations presented in earlier chapters. Tests are done for a non-anechoic environment using a low amount of microphones, an anechoic chamber using the same settings as for the simulations, and for a standard listening room. The tests on real measurements show that the RSTM performs as the simulations predicted.

Table of Contents

1

Introduction

1.1 Motivation

The interest for sound-field analysis and synthesis has been growing for a long time, and still is. This is true due to its importance in areas in acoustics such as: music recordings, multimedia communications, auditorium acoustics and noise control[2][13]. The importance in these areas stems from the fact that spatial acoustical information is very important for a human to experience a sound field, and get a sensation of immersion of the sound[14]. Some of the main challenges in capturing sound fields concern the application of methods of sound field analysis in non-laboratory conditions, where robustness to noise and interference is very important[17]. This creates a desire to explore new methods in the field to try and find something which performs well as a general solution to sound field tasks, including analysis and synthesis. Using microphone arrays to capture sound fields is currently the norm, and multiple methods already exist for this purpose. A novel microphone array method for sound analysis has recently been discovered and implemented, called the "Ray Space Transform Method"(RSTM), which while still early in its development shows promise both in sound field analysis and synthesis. The motivation of the thesis is to explore the parameters and components of the method and then compare its performance in sound field analysis to a contemporary widely used method, the Spatial Decomposition Method(SDM).

1.2 Wave Field Processing

One of the classic problems in processing of acoustical signals is the estimation of acoustical field representations. Microphones arrays are a common factor between many techniques used for acoustical field analysis and synthesis. The process to reach this representation is called spatial encoding or wave field processing and depending on the way that the field is represented receives various names. Different methods of the acoustic field processing are often divided into categories, these being parametric, nonparametric and geometric representations. The main difference seen in these is that the parametric representation works on some level of a priori knowledge of the acoustic field, such as environment geometry. Often here, the analysis is carried out by beamforming or analysis of (Time Difference of Arrival)TDoAs. The nonparametric representation assumes no a priori knowledge, utilizing either plane-wave representations, spherical-wave representations or cylindrical representations. The geometric representation uses acoustic rays, which are found when looking at simple acoustic propagation, and carry the acoustic information along straight lines in space. This birthed the concept of the ray space, a domain where points represent the plane-wave components of the sound field. [14]

1.3 Contribution of the Thesis

RSTM and SDM are both methods for sound-field analysis, synthesis and derived acoustic tasks as source localization and others. While the implementation that we are analyzing for SDM was published in 2013 [1] and has been used in several real life scenarios as can be seen for example in articles [20] [21] [22], the actual implementation of the RSTM was published in late 2016 [2] and have not been used in non laboratory conditions as far as we researched it. Both techniques differ in formulation, methodology and results as can be seen in Chapter2 and to compare them a systematic approach is necessary. This thesis aims to accomplish two main contributions. The first one is to explore the RSTM performance regarding the multiple parameters that defines it, and the second one is to compare RSTM and SDM for room analysis. The metric used to accomplish these tasks is the distance and angle error present in the estimation of acoustical events in different scenarios This exploration tries to be systematic, first studying the methodology and parameters in each of the methods, then performing simulations in anechoic conditions, simulations considering different possible sources of estimation errors, simulations in reverberant conditions and finally laboratory measurements. The purpose of having these results is to know what is expected from both methods in different scenarios and to define advantages and disadvantages between them.

1.4 Structure of this report

The report is structured in five chapters.

- Chapter 1 introduces the motivation to carry out this work and the main contribution of it.
- Chapter 2 introduces the methods and presents the state of the art and theory behind them. Additionally a study on the line detection methods used in RSTM.
- Chapter 3 introduces and carries out an analysis on the influence of the parameters which are specific to each method.
- Chapter 4 compares the methods using parameters which are common for both methods in order to determine the advantages and disadvantages of the RSTM compared to SDM.
- Chapter 5 carries out a study on real measured data to verify the simulated data.
- Chapter 6 concludes and discusses the results and reflects on possibly of future works.

2

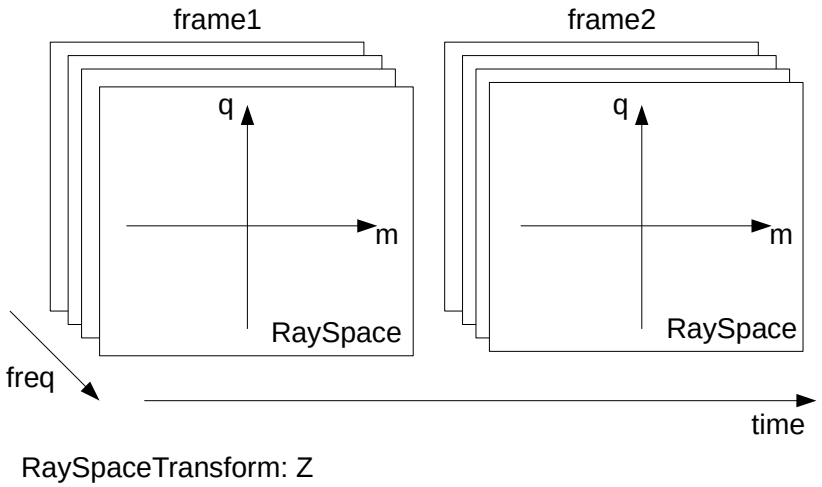
Theory

This chapter presents the necessary theoretical framework to understand the RSTM and SDM methods and the simulations performed. It is composed by the next sections:

- Section 2.1 explains the RSTM formulation, the path that a signals follows with this technique until a ray space representation is obtained and additionally two important definitions for the RSTM. The plenacoustic function and the Gabor frame transform.
- Section 2.2 presents the SDM algorithm and conceptual formulation.
- Section 2.3 summarizes the objectives and products and both methods and the theoretical similarities and differences between them
- As the product of RSTM is a visual representation, Section 2.5 presents three different strategies for image line detection.
- Section 2.7 presents the theory necessary, and the algorithm used to simulate the RIR in anechoic and reverberant conditions

2.1 Introduction to the Ray Space Transform

In RSTM the estimated location is not a single number, but a matrix for a predefined number and range in the ray-space (m slope, q intercept) and the similarity (energy) between the captured acoustic field and shifted and modulated copies, for each frequency and for each time frame.



RaySpaceTransform: Z

An initial framework for simulating the ray space is made to achieve a general understanding of how the algorithm works. We delve into how a ray space is formed, which algorithms and

methods are applied, and later on then attempt to explain these. The ray space algorithm, as described in [2], uses a linear array of microphones with equal spacing, distributed along the vertical cartesian z-plane, the locations in this axis are defined by:

$$z(l) = l * d - d * (l + 1)/2 \tag{2.1}$$

Where z is the array holding the microphone locations in the z-plane, d is the spacing between microphones, and l is the current microphone, $l = 1, 2, 3...L$. L is the number of microphones in the array. See figure 2.1 (a) for a visual representation of the array.
To simulate a signal received on each of these microphones, a white noise signal is produced, $wgn(n_{total}, 1, 0)$, where n_{total} is the length of the signal. To represent what the receivers 'see'[1] when a speaker emits the signal from a chosen position $\mathbf{r}'$, a transfer function dependent on frequency and microphone position is applied to the signal for each microphone position.

$$p(z, \omega) = s(\omega) \cdot h(z|\mathbf{r}', \omega) \tag{2.2}$$

Where z denotes the 1-D coordinates of the individual microphones, and ω denotes the angular frequency. $s(\omega)$ is the FFT of the generated white noise.

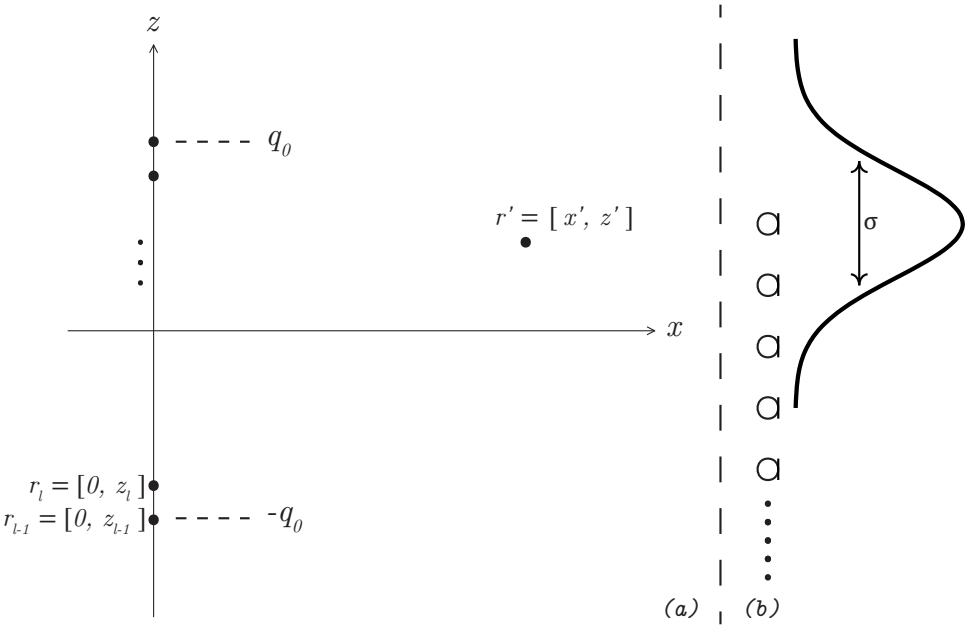


Figure 2.1: *(a) Microphone array and source topology. (b) Gaussian windowing over microphones.*

The transfer function $h(z|\mathbf{r}', \omega)$ as is defined in [2]:

$$h(z|\mathbf{r}', \omega) = \frac{e^{-j\omega||\mathbf{r}-\mathbf{r}'||/c}}{4\pi||\mathbf{r} - \mathbf{r}'||} \tag{2.3}$$

Where c denotes the speed of sound, $||\cdot||$ denotes the $\mathcal{L}_2$ norm, $\mathbf{r}'$ is the coordinates of the sound source, and $\mathbf{r}$ hold the current microphone coordinates.
Having produced an approximation of sound propagation across the array from a source, we can

[1] Ideal conditions, no reflections.

start transforming the data into the ray space. A current method, as presented in [2] is shown in the following equation:

$$[\mathbf{Z}]_{i,w}(\omega) = d\sum_{l=0}^{L-1} p(z_l,\omega)e^{-\frac{jkz_l m_w}{\sqrt{1+m_w^2}}} e^{-\frac{\pi(z_l-q_i)^2}{\sigma^2}} \tag{2.4}$$

Here we recognize the first part of the equation, $p(z,\omega)$, but the two exponentials have so far not been discussed.

The first exponential in the equation, $e^{-\frac{jkz_l m_w}{\sqrt{1+m_w^2}}}$, beamforming across a chosen range of angles, defined in $m_w = tan(\Theta_w)$, $w = 1,...,W$, $W = \frac{\bar{m}}{2m_{max}}$. The wavenumber, $k = \frac{\omega}{c}$, represents the frequency in the spatial domain.

The second exponential, $e^{-\frac{\pi(z_l-q_i)^2}{\sigma^2}}$, is a Gaussian weighting function, which while we iterate over the microphone array, attenuates the signal on microphones more the further away from the current microphone they are. A rough depiction of this is seen in 2.1(b). σ controls the width, or standard variance of the Gaussian curve. Iterations are made across the microphones as well as points between these on the z axis, contained in q_i, $i = 1,...,I$. This splits the array of microphones into I sub-arrays of microphones.

The $\mathbf{Z}$ matrix of equation 2.4 then contains the ray space for the current frame[2] and chosen frequency, ω. Using the following values for each variable, figure 2.2 shows the magnitude heatmap of the $\mathbf{Z}$ matrix, visualizing the ray space.

Variable	Symbol	Value
Frequency	ω	$2\pi 1000[Hz]$
Source Location	r'	$[3,0][m]$
Amount of microphones	L	16
Microphone distance	d	0.1[m]
Beamforming angle	θ	-70 to 70
Sub-array distance	$\bar{q}_i$	7.5[mm]
Slope axis resolution	$\bar{m}_w$	0.03
Gaussian window width	σ	0.2[m]

Table 2.1: *Parameters used in figure 2.2*

See section 3 for more insight into the variables.

[2] A frame is a set number of samples, i.e. it is the time component.

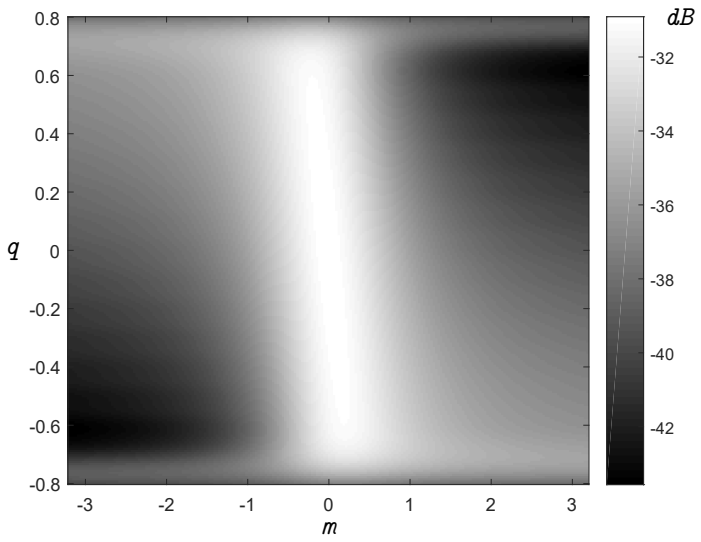


Figure 2.2: *The ray space for* ***r'*** *= [3, 0].*

From the ray space, it is now possible to derive the location of the sound source. Assuming that the analyzed frequency is present in the source and that the pattern is not distorted by other acoustical phenomena, the sound source will show itself as a linear pattern in the ray space defined by[2]

$$y = mx + q \tag{2.5}$$

Where x and y are coordinates in the real world, m and q are coordinates of the ray space. Detecting a line in the ray space is challenging in itself, especially if the data representation is noisy or contains error. Section 2.5 examines line detection in the ray space.

2.1.1 Signal Path Overview

Figure 2.3 gives a simple visual representation of the path of the sound signals received on the array from microphone to ray space. First the signal is sampled on the microphone, whereafter it is converted to the frequency domain by fast fourier transform. The gaussian window is applied across the microphone array, and thus a much larger array of signals is made, containing weighted information of multiple microphones in each element. Beamforming over a range of angles defined by $m_w = tan(\Theta_w)$[2] is then performed. Summing over each microphone then gives us the Ray Space, as seen in equation 2.4.

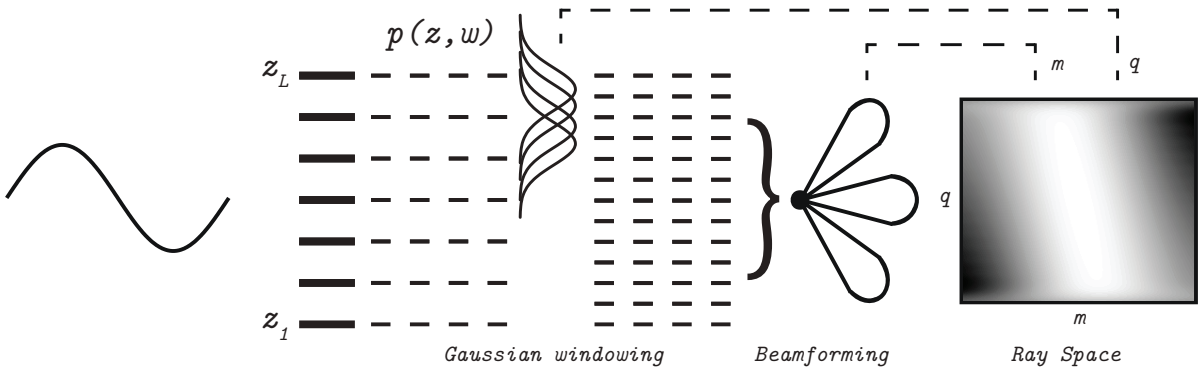


Figure 2.3: *From left to right, the signal path is first captured on the microphone array, in simulations this is carried out by applying a transfer function to represent each microphone position. Converting this to frequency we have the level at position and frequency in $p(z,\omega)$. The gaussian windowing splits the signal in to smaller steps in position in the sub-arrays, and these positions are then beamformed upon to achieve the ray space.*

2.1.2 Short Time Fourier Transform

To make use of the short frame averaging concept in RSTM, we use short time fourier transform(STFT) to get the frequency content of each frame. We implement this by doing FFTs on the data contained in each frame. The effect of this is that in a very short time window, we gather a lot of data from the sound field, which can be averaged to produce smoother heat maps. See section 3.2.4 for more about frame averaging. Following piece of MATLAB code shows the concept.

```
for j = 1:frames
  sw(j,:) = fft(st(1-N+N*j:N*j), Fs);
end
```

Where: **frames** is the number of frames used, **N** is the length of the frame in samples, **sw** is $s(\omega)$, the frequency content of the signal, **st** is $s(t)$, the input time signal, **j** is the variable iterating over the current frame, **Fs** is the sampling frequency.

No window or overlap is introduced on the STFT, since using white Gaussian noise the signal is time invariant. If the STFT is to be used with signals that are time variant, such as music or speech, a Hanning window with overlap is introduced, and the STFT then looks like the following MATLAB snip.

```
for j = 1:frames
  sw(j,:) = fft(hann(length(N*j/2 - N/2 + 1 : N*j/2 + N/2))
            .* st(N*j/2 - N/2 + 1 : N*j/2 + N/2), Fs);
end
```

Where hann is a built-in MATLAB function to generate a Hanning window with the length of the window as the argument.

2.1.3 The Plenacoustic Function and the Ray Space Domain

The plenacoustic function is an acoustic adoption of the more well-known plenoptic function, which is a function used in optics to describe an image in any position looking in any angle,

at any point in time [26]. The plenacoustic function(PAF), introduced by Ajdler et.al. in [6] as the sound pressure recorded at location (x,y,z) at time t, is non directional information that can be transformed into a directional PAF, as the PAF contains phase information, but requires multiple sampling of the sound-field in space. The directional plenacoustic function describes the pressure in every direction through every point in space, it can be written as a function of position, direction, frequency and time $f(x, y, \theta, \omega, t)$
As stated in [17], in an homogeneous medium an acoustic ray is a line co-linear with the wave vector which acoustic radiance remain constant along it (as long as propagation losses are not considered), being suitable to use them to parametrize the directional plenacoustic function.
The ray space transform objective, explained in 2.1 represent all the elements of the plenacoustic function, as represents acoustics rays by time and frequency (while the position is observed from the images), and can be considered a representation of the directional PAF.

2.1.4 Gabor Frames in the Ray Space

As explained in [27] the Gabor transform and Gabor expansion are tools to analyze and synthesize a time signal respectively. The objective of the Gabor transform is to represent a signal in the joint time-frequency domain and usually remove undesired features from it. In this sense it can be stated that the Gabor transformation is similar to the Short Time Fourier Transform as it represents the signal in the time-frequency domain. The coefficients show the similarity between an analysis window which is shifted in time and modulated in frequency.
The Gabor transformation is represented by the formula:

$$C_{m,n} = \sum_{i=0}^{L-1} s(i) \cdot \gamma^*_{m,n}(i) \tag{2.6}$$

$$\gamma_{m,n}(i) = \gamma(i - m \cdot \triangle M) \exp(2\pi \cdot n \cdot \triangle N \cdot i/L) \tag{2.7}$$

Where:

Symbol	Definition
$C_{m,n}$	Gabor coefficients
L	Time sampling points
$s(i)$	Time signal
m	Time element
n	Frequency element
$\triangle M$	Time sampling interval
$\triangle N$	Frequency sampling interval
M	Time sampling points
N	Frequency sampling points
$\gamma_{m,n}(i)$	Window function

The Ray Space uses an adaptation of the Gabor transform. It maps a time signal into a new domain. The main differences between the Gabor transform and the adaptation in the Ray Space transform are:

1. The signal is not sampled in time but in space through spatially distributed microphones.
2. The input signal is not represented in time, but in the frequency domain.

3. The signal is not mapped into time-frequency domain, but into the ray-space domain which is $(m,q)\epsilon\mathbb{R}x\mathbb{R}$. Where m y is the slope of the incident ray source-receiver and q is the intercept with the z axis. As is shown in figure 2.4
4. The analysis window function is a normal Gaussian distribution.

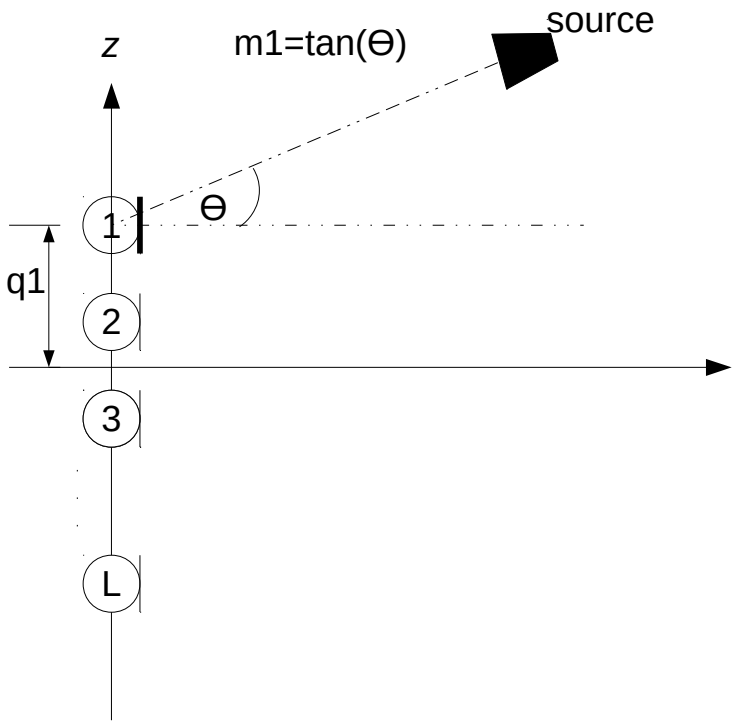


Figure 2.4: *Source-receiver ray representation for the ray space domain.*

The discrete Ray-Space transform is represented by the formula[2]

$$[Z]_{i,w(\omega)} = d\cdot \sum_{l=0}^{L-1} p(z_l,\omega)\exp(\frac{-j\cdot k\cdot z_l\cdot m_w}{\sqrt{1+m_w^2}})\exp(-\pi(z_l-q_i)^2/\sigma^2) \tag{2.8}$$

Where:

Symbol	Definition
$[Z]_{i,w(\omega)}$	Ray space coefficients
d	Distance between microphones
$p(z_l,\omega)$	Pressure signal for a given microphone and frequency
m_w	Slope element
q_i	Space element
k	Wavelength number of the current input signal
z_l	Microphone position
L	Space sampling points
N	Frequency sampling points
$\exp(-\Pi(z_l-q_i)^2/\sigma^2)$	Gaussian window function

2.2 The Spatial Decomposition Method

This section summarizes the method described article [1], as is the implementation used through the thesis.

2.2.1 Objective

The main objective of SDM is to analyze a spatial room impulse response. The algorithm will result in a set of image-sources, one for every analyzed time step, described by the discrete pressure values and their corresponding locations as shown in figure 2.5 extracted from [1] To achieve the location component,a set of room impulse responses captured with a microphone array, is analyzed through a least squares solution for TDOA. The pressure component that represents the room impulse response at the center of the array, can be either selected from the microphone array, if there is an omni-directional microphone at that position or can be formed with the other microphones, if that is not the case.

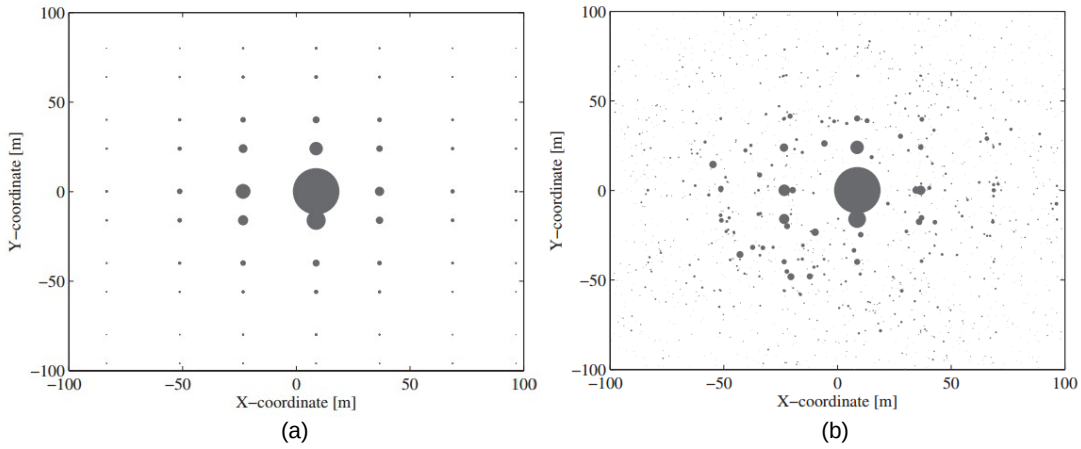


Figure 2.5: *An example of the locations and amplitudes of (a) simulated image-sources and (b) decomposed image-sources with SDM from a spatial impulse response. The area of each filled circle illustrates the energy of that image-source. The image-sources with the highest energy are correctly analyzed (figure and text extracted from [1])*

2.2.2 Basic assumption and general method

As explained in [1] SDM is derived under two basic assumptions. The sound propagation direction is the average of all the waves arriving to a microphone array at the same time and this propagation direction is associated with the impulse response sound pressure in the geometric center of the array. The method uses the information captured by the microphone array and compose a spatial room impulse response noted as $H(t) = \{h_n(t)\}_{n=1}^{N}$ where N is the the total number of microphones in the array and $h_n(t)$ is the signal on each microphone. Then the method analyzes the time difference of arrival, TDOAs, of the spatial room impulse response, sample by sample, to get a location referred to the center of the array. The set of locations is named DOA, after Direction Of Arrival, and is represented in Cartesian coordinates. The pressure component at the center of the array that represent the magnitude component of the sound, represented as O can be predicted from the spatial room impulse response, if there it is not possible to have one

microphone at the center of the array or simply use the captured pressure values if a microphone at the center is available. Figure 2.6 shows a block diagram of the SDM method.

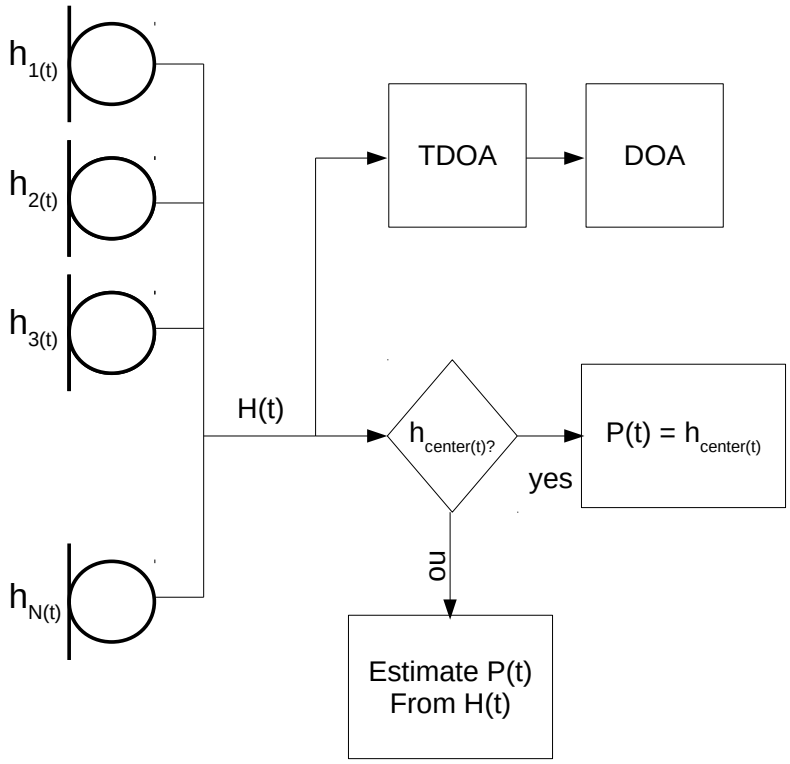


Figure 2.6: *Block diagram of the SDM method*

More on Location estimation

First the TDOAs are be obtained through generalized correlation method with direct weighting described in [24] and then each TDOA estimate is interpolated as explained in [25]. Each TDOA for every sample, is analyzed in a short time window, of length L. The TDOAs are represented by $\hat{\tau_k}$ being k the total number of possible pairs of microphones, the corresponding microphone positioning difference vectors are noted with V. This information is used in the least squares solution for TDOA leading to the location estimation.

Estimation of the Sound Source

As explained in 2.3 the SDM product will be the direction of arrival, in cartesian coordinates, and the energy value for each time sample of the RIR at the center of the array. In practical terms, in the implementation that we are using two matrices are the result of the SDM algorithm. The first matrix contains all the estimated directions of arrival, DOA, and the second one contains all the energy values at the center of the array. From this information we can estimate the position of the sound by simply looking into the value with the highest energy, retrieving the index for it and getting the DOA for that index as explained in figure 2.7.

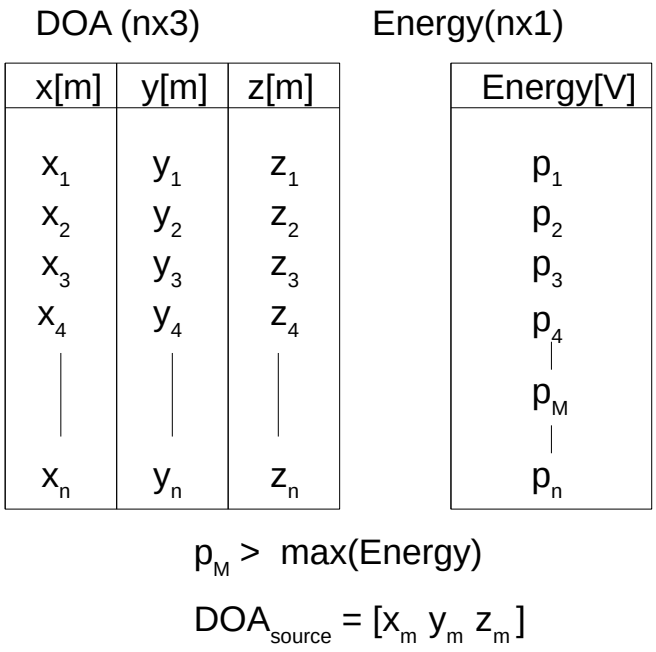


Figure 2.7: *SDM source estimation*

2.3 Similarities and Differences Between RSTM and SDM

Similarities

1. The analysis final "product" is in both cases showing the location and energy over time of the room impulse response. For SDM is represented as Cartesian coordinates plus the magnitude of pressure at every sampling period. While for RSTM the location is not a single number, but a linear pattern in the ray space.
2. Synthesis of the spatial impulse response can be achieved with both methods.
3. Sound Source Localization is feasible with both methods.

Differences

1. The input signals in RSTM need to be represented in the frequency domain first in order to apply the algorithm, while in SDM the method the signal needs to be represented in time domain.
2. The pressure values in the the RSTM are not included directly, instead is a measure of similarity, in RSTM
3. SDM has been used in analysis and synthesis of spatial impulse responses as can be seen in [20][20][20] and [23], RSTM has been used principally as the framework for processing tasks like sound source filtering and acoustic localization as can be seen in [3] and [17]
4. While SDM is a method specific to analysis and synthesis of spatial room impulse responses, RSTM has been used widely in different others fields like communications and optics.

2.4 Identifying the Local Maxima

To obtain data which line detection can process, we identify the local maxima in the ray space, which will contain the 'ridges' of the rays in the ray space. In noisy scenarios some maxima will also appear outside the ray, and these need to be disregarded by line detection. The local maxima are found in the ray space by a simple comparison. A function iterates over each value in $Z(q,m)$, and compares it to the previous and next values in the m-axis, $Z(q,m-1)$ and $Z(q,m+1)$. If the value is greater than that of the value on either side of it, the index is saved in an array. When complete, the array containing the indices of local maxima in Z along with the matrix Z itself is sent to be processed in the RANSAC function. The following MATLAB code snip performs this function.

```
for qiscan = 1:length(q_i)
    for mwscan = 2:length(m_w)-1
        if (Z(qiscan, mwscan+1) < Z(qiscan, mwscan)) && ...
            (Z(qiscan, mwscan-1) < Z(qiscan, mwscan))
            Z_ransac(ncount, 2) = q_i(qiscan);
            Z_ransac(ncount, 1) = m_w(mwscan);
            ncount = ncount + 1;
        end
    end
end
```

Where **Z** is the matrix holding the ray space, **qiscan** and **mwscan** are the variables iterating over their respective axes, **Z_ransac** holds the local maxima indices, and **q_i** and **m_w** are arrays containing the indices of the axes.

2.5 Line Detection Algorithms

To obtain the source locations in the ray space, a method of finding linear patterns is necessary.[2][3] The source location in the ray space shows itself as a linear pattern in a heatmap. Here the slope of the linear pattern corresponds to the real x coordinate, and the value of q at the intersect between the linear pattern and $m = 0$ in the ray space corresponds to the real y coordinate. Some known algorithms used in line detection are briefly explained and then evaluated in regard to its usability in the project.

The main criterion for the method is to be able to accurately recognize linear patterns when noise is present in the point cloud generated from the local ray space maxima, i.e. a certain level of robustness is required so that it doesn't misidentify linear patterns in the noise.

The algorithms we examine are: Least-Squares Linear Regression[8], Hough Transform[9][10], Random Sample Consensus(RANSAC)[5]. Although the least-squares regression is not known to be particularly robust, it is a nice reference since it is one of the more used standard methods of line detection. The latter two, Hough Transform and RANSAC, are known as more robust solutions to line detection.[2][3] The Hough transform is proposed as a method for line detection in the ray space in [3]. RANSAC is proposed as an alternative in [11].

2.5.1 Least Squares Regression

Least squares linear regression[8] is one of the most widely used methods of detecting linear patterns. It produces a single solution as long as the system, $\mathbf{A}\bar{x} = \bar{b}$, provided is consistent. The solution is found by:

$$\mathbf{A}^T\mathbf{A}\bar{x} = \mathbf{A}^T\bar{b} \tag{2.9}$$

$$\bar{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\bar{b} \tag{2.10}$$

The weakness of the least squares regression arrives when the maxima provided contain outliers which are uncorrelated with the relevant linear pattern. The least squares regression will consider all the points provided and assume that they are part of the pattern. Without any pre-processing of the data, if outliers are present, the least squares regression will misinterpret the data and not estimate source location accurately.

2.5.2 Hough Transform

The Hough transform is a method of linear pattern detection in image analysis. It was developed specifically to detect collinear points by transforming points in a figure into straight lines in a parameter space. This is usable in the ray space, since the ray space heatmap data representation is the same as that of a picture[11][9][10]. In [11] the authors apply the Hough transform to get the first approximation of multiple sources in a ray space image, where the peaks are assigned to their corresponding sources. To achieve accurate estimation, the Hough transform requires a very high grid density on the map of points that it creates in its parameter space. To avoid having to produce this, the authors in [11] use the Hough transform to find a first approximation of the source locations, where the peaks of the corresponding sources are assigned to them. After this, linear regression is used over measurements of these peaks assigned to sources to obtain better results than the Hough transform manages by itself.

2.5.3 RANSAC

RANSAC[5] is an iterative method to discover patterns in point cloud datasets. While it is non-deterministic, it is a very robust method, in which the accuracy of estimation is very much determined by the user-controlled parameters. RANSAC is meant for use in datasets where outliers, which have no correlation with the desired pattern, are present. Through iteration over randomly chosen datapoints, RANSAC seeks to determine which points in the cloud can be classified as 'inliers', which will shape the proposed pattern, and 'outliers', which will be discarded and not have any effect on the outcome.

2.5.4 Evaluation

Linear regression can be suitable for ideal condition, where it does not have to consider points which are not part of the relevant linear pattern(s). We seek something more robust that can work in scenarios outside of ideal simulated conditions, and such we disregard the use of pure linear regression. The Hough transform is closer to what we need in our simulations and for the real data, but in the [11] the authors argue that linear regression is preferred to the Hough transform in localization of single sources in relatively noiseless conditions. Here we figure it

would be preferable to have a single method of finding source locations in all purposes, which leaves us with RANSAC. In addition to the previous arguments, authors in [3] refer to using Random Sample Consensus(RANSAC) as an improved concept of line detection in the ray space compared to the Hough transform used in [11]. Based on this distinction and evaluation, this thesis will heed the advice and use RANSAC as the only method of detecting lines and obtaining source location coordinates from the Ray Space.

2.6 Random Sample Consensus

In this section we further explore the choice of line detection method, Random Sample Consensus. RANSAC[5] is well suited for discovering only linear patterns, and can be configured to only look for these. The user-controlled parameters are the number of iterations, the inlier threshold, and the inlier ratio. The number of iterations defines how many times the algorithm will look for inlier pairs, and include them if they uphold the other criteria. It is highly correlated with the accuracy of the estimation, although there will be some limit where adding more iterations has little to no effect. The inlier threshold decides how close the proposed point in the cloud has to be to the line pattern for it to be considered an inlier. The inlier ratio decides how many points in total are needed to construct a line pattern, to make sure it only recognizes something as a line if it has atleast a number of points within it, relative to the total number of points in the cloud[3][5].

While 'outliers' may not exist in a perfect, simulated scenario with a single source, RANSAC should still perform adequately given a set of data with clear patterns. The advantage, then, of adapting RANSAC to the algorithm lies in it's robustness. In the case of exploring the performance of the RSTM beyond noiseless simulations, the robustness of detecting linear patterns in the ray space will be crucial. RANSAC is also well suited for identifying more than one linear pattern, by applying the algorithm to the resulting set of outliers which contains all points in the cloud except for those making up the previous linear pattern. This can in theory be done indefinitely for each sampling of the sound field, providing sufficient resolution in the image, and enough iterations during RANSAC to identify these.

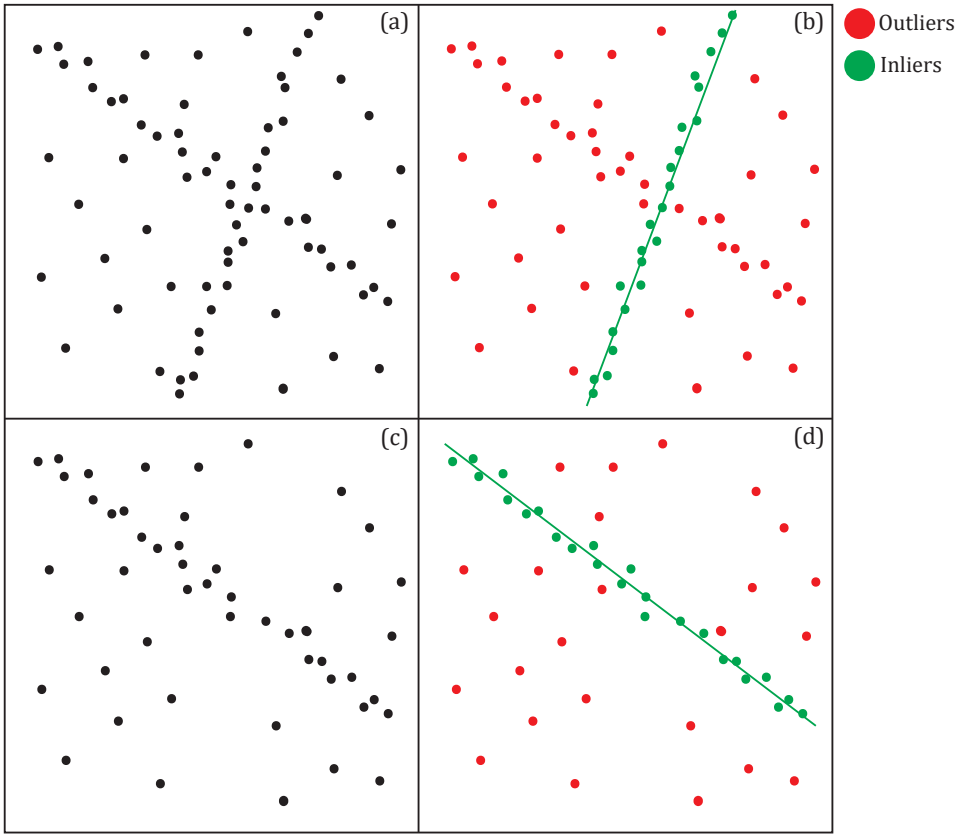


Figure 2.8: *RANSAC line pattern detection. (a) and (b) shows datasets before RANSAC processing. (c) and (d) show what the RANSAC sees in the pattern. Here the red dots are classified as 'outliers' and are not considered belonging to the linear pattern, while the green dots are 'inliers', which constitute the linear pattern.*

Figure 2.8 shows a case of how the RANSAC method finds linear patterns in a point cloud. Figure 2.8 (a) shows the initial point cloud introduced to the algorithm. In (b) the algorithm has identified a linear pattern, and the points the best fit the line, the inliers, in green. These are then logged, and removed from the set. The remaining red points, then represent the outliers, which the algorithm deemed uninfluential on the proposed linear pattern. These can be used as an input, see (c), to the next iteration of the algorithm, to see whether more linear patterns exist in the dataset, beyond the first which was eliminated. The algorithm then repeats process over the 'new' dataset and finds another linear pattern (d). This can be continued until no clear linear pattern can be found in the set. A stop criterion is necessary if the algorithm is to identify linear patterns without supervision. The inlier ratio is used for this.

2.7 Room Impulse Response Model

The simulations presented in this report consider two scenarios regarding the Room Transfer Function (RTF) or Green's function in which they take place. The first RTF considered is the anechoic or free space model, where the generated soundfield is unbounded by any reflective element. The second RTF considers a rectangular room with defined reflection coefficients for each of the walls that bound the soundfield. This section presents an explanation on how each

of the RTFs is approximated and how the implementation is done in [16]. The equations and theorems in the section use [16] as reference.

Wave Equation

The propagation in an homogeneous medium of the sound waves that compose a sound field is explained by the wave equation 2.11

$$\nabla^2 p(\mathbf{r},t) - \frac{1}{c^2}\frac{\partial^2 p(\mathbf{r},t)}{\partial t^2} = 0 \tag{2.11}$$

To consider the sound field generated by a source in a specific room a source function and boundary conditions are needed. Being $s(\mathbf{r},t)$ the source function, then the wave equation results in 2.12

$$\nabla^2 p(\mathbf{r},t) - \frac{1}{c^2}\frac{\partial^2 p(\mathbf{r},t)}{\partial t^2} = -s(\mathbf{r},t) \tag{2.12}$$

Helmholtz Equation

Lets define the Fourier transform of the sound pressure

$P(\mathbf{r};\omega) = \int p(\mathbf{r},t) exp(i\omega t) dt = \mathcal{F}\{p(\mathbf{r},t)\}(\omega)$

Applying Fourier transform to 2.12 lead to the time independent Helmholtz equation 2.13, that can be understand as the representation of the wave equation in frequency domain, where k is the wave number.

$$\nabla^2 P(\mathbf{r};\omega) + k^2 P(\mathbf{r};\omega) = S(\mathbf{r};\omega) \tag{2.13}$$

Room Transfer Function, Green's Function

Considering a unit amplitude harmonic point source at position $\mathbf{r}_s = [x_s, y_s, z_s]$, and being the Kronecker delta function, $\delta(\cdot)$ the source function becomes $S(\mathbf{r};\omega) = \delta(\mathbf{r}-\mathbf{r}_s)\delta(x-x_s)\delta(x-y_s)\delta(x-z_s)$.

In this conditions Equation 2.13 becomes equation 2.14 in which $H(\mathbf{r},\mathbf{r}_s;\omega)$ is the Room Transfer Function or Green's function

$$\nabla^2 H(\mathbf{r},\mathbf{r}_s;\omega) + k^2 H(\mathbf{r},\mathbf{r}_s;\omega) = \delta(\mathbf{r}-\mathbf{r}_s) \tag{2.14}$$

The solution of 2.14 depends on the boundary conditions imposed by the enclosed space and the position of the sound source. A function $\Psi_m(\mathbf{r};\omega)$ that satisfies the equation and the boundary conditions is called an eigenfunction while each of the coefficients that depends on the position

of the sound source are represented by $C_m(\mathbf{r}_s;\omega)$ Subsequently a general expression for the RTF using the eigenfunctions is 2.15

$$H(\mathbf{r},\mathbf{r}_s;\omega) = \sum_{m=0}^{\infty} C_m(\mathbf{r}_s;\omega)\Psi_m(\mathbf{r};\omega) \tag{2.15}$$

The solution of equation 2.14 have a special practical importance. For any source function, the sound pressure can be calculated if the RTF is known through equation 2.16.

$$P(\mathbf{r};\omega) = \int\int\int_{v_s} H(\mathbf{r},\mathbf{r}_s;\omega)S(\mathbf{r}_s;\omega)d\mathbf{r}_s \tag{2.16}$$

Anechoic RTF

For a point source in a space without reflective surfaces, equation 2.14 is solved into equation 2.17

$$H(\mathbf{r},\mathbf{r}_s;\omega) = \frac{exp(-i\omega\|\mathbf{r}-\mathbf{r_s}\|/c)}{4\pi\|\mathbf{r}-\mathbf{r_s}\|} \tag{2.17}$$

Rectangular Room

Considering a rectangular room of dimensions (L_x, L_y, L_z) and perfectly reflecting walls the eigenfunctions, normally referred as modes, where $\mathbf{m} = (m_x, m_y, m_z)$ are positive integers are:

$$\Psi_m(\mathbf{r}) = cos\left(\frac{m_x\pi}{L_x}x\right) cos\left(\frac{m_y\pi}{Ly_x}y\right) cos\left(\frac{mz_x\pi}{Lz_x}z\right) \tag{2.18}$$

The solution for 2.14 for a rectangular room is 2.19

$$H(\mathbf{r},\mathbf{r}_s;\omega) = \sum_{\mathbf{m}\epsilon\mathcal{M}} \frac{\Psi_{\mathbf{m}}(\mathbf{r})\Psi^*_{\mathbf{m}}(\mathbf{r}_s)}{\Lambda_{\mathbf{m}}(k^2-k^2_{\mathbf{m}})} \tag{2.19}$$

where $\mathcal{M}$ represent all the desired triplets of m and $\Lambda_{\mathbf{m}}$ is a normalization constant for the associated eigenvector.

Implementation of the Rectangular Room Response

Allan and Berkley image method [15] allows to efficiently compute a Finite Impulse response (FIR) that models the acoustic channel between a source and a receiver in rectangular rooms. The implementation of this method done in [16], page eleven. Here, a list of input and optional parameters is also found.

2.8 Summary

The theoretical framework for the RSTM and SDM algorithms has been presented, and will be used to construct a simulation framework in the next chapters. A common ground for the comparison has been introduced in section 2.3, which will be used as basis to define the performance metrics in the following chapters. Section 2.5 presents an analysis of three different line detection techniques and RANSAC has been selected as the one to be used for the RSTM product processing. And finally in section 2.7 the anechoic and rectangular room RTF's has been introduced and will be used in the parameter study and comparison studies.

3

Parameter Study

A study on the parameters of the RSTM and SDM and their effect on the performance of the algorithms is made to gain a general sense of how the methods work. Although, before this can be studied, the influential parameters has to be identified. The following list contains the influential parameters we found for each method.

- RSTM parameters
  - Beamforming range
  - m_w resolution
  - q_i resolution
  - Gaussian window width
  - Number of frequencies analyzed
  - Number of frames averaged
- RANSAC parameters
  - Inlier ratio
  - Number of iterations
  - Distance threshold
- SDM parameters
  - Maximum spacing between a pair of microphones
  - Temporal Window Size
  - Number of microphones
  - Microphone array geometry
- Common parameters
  - Number of microphones
  - Microphone spacing
  - Sound source distance
  - Sound wave incident angle to center of array
  - Microphone position error
  - Number of sound sources
  - Signal to noise ratio
  - Microphone phase mismatch
  - Sound source signal type
  - Estimation in reverberant conditions

Where the RSTM and RANSAC parameters are specific to the RSTM, the SDM parameters specific to the SDM, the common parameters are common for both methods. The common parameters will be used as a base to create a performance comparison study between RSTM and SDM.

3.1 Default Parameter Settings

To be able to look into one parameter at a time, a default state[1] of each of the following variables is chosen beforehand[2]:

RSTM
Beamforming angle range, $\theta = -78°...78°$
m_w interval, $\bar{m_w} = 0.03$
q_i interval, $\bar{q_i} = 7.5 * 10^{-3}$ m
Gaussian window width, $\sigma = 0.2$
Microphone spacing, d = 0.1 m
Number of frequencies analyzed, G = 1
Frequency, f = 1000Hz
Number of frames averaged, 1
Samples per frame, 192

RANSAC
Inlier ratio = 0.25
Number of iterations = 5000
Distance threshold = 0.03

SDM
Maximum spacing between a pair of microphones, $d_{max} = 0.25[m]$
Temporal Window Size, $L = 70$ samples at 48000[Hz] Fs
Microphone array geometry, *Random*

Common
Source signal type for RSTM, white Gaussian noise $\mathcal{N}(0, 1)$.
Source signal type for SDM, ideal dirac delta function
Source location, r = [3, 0] - source distance 3 m.
Number of microphones, L = 16
Number of sound sources, 1

Unless stated otherwise, the parameters will be as defined above in the remaining sections of this chapter.

The speed of sound, c, is set to 343 m/s throughout the report

3.2 Ray Space Parameters

This section contains a study on the RSTM specific parameters. A block diagram in figure 3.1 gives an overview of the ray space simulations from start to finish. First the input signal is generated, then it is transformed to frequency domain by STFT. Here after the acoustic transfer

[1] This state is based on settings from [2].
[2] Errors, noise and phase are zero in default state.

function which simulates the signal coming from a simulated speaker position for each microphone in the array. This yields $p(z, \omega)$, the pressure values at microphones for each frequency. This is the input for the ray space transform, which produces the ray space matrix, $Z(m, q)$. Local maxima are found in the generated ray space, $Z(m, q)$, and a new matrix only containing the coordinates of the local maxima is made, $Z_{ransac}(m, q)$, which is sent to the RANSAC block. RANSAC then finds linear patterns in the ray space and estimates the source coordinates, x and y in meters.

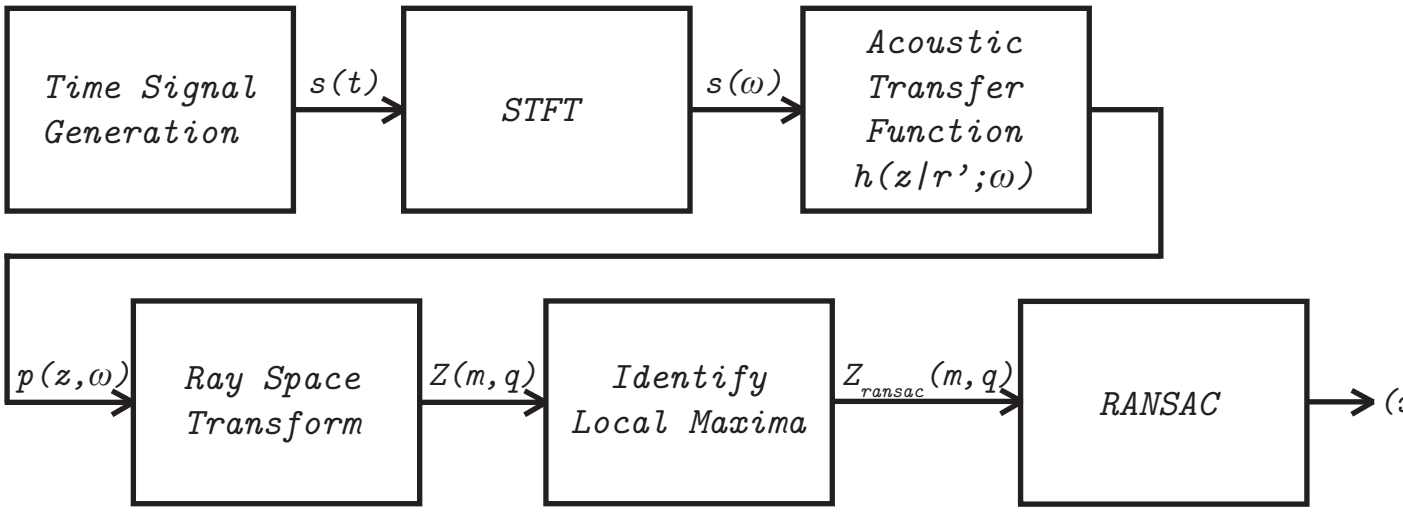


Figure 3.1: *Block diagram showing the process of simulating the RSTM.*

3.2.1 Resolution in the m-plane and Angular Beamforming Range

m_w, $w = 1, 2, 3...W$, controls the beamforming, in that it is defined by the beam width angle, θ. $m_w = tan(\theta)$ [2]. The default setting is looking from -78° to 78°, using an sample interval value in m_w of $\bar{m_w} = 0.03$. We want to explore how changing m_w and $\bar{m_w}$, changes the source position estimation. The parameters are tested for a source located in [2, 2], which in default settings result in an estimation at [2.049, 2.043]. Using $\theta = -78°...78°$ results in an accurate estimation, as is seen above. Reducing it to $\theta = -40°...40°$, the estimation finds [2.463, 2.335]. Figure 3.2 shows the Ray Space generated using the two different values of θ,(a) for $\theta = \pm 40°$,(b) for $\theta = \pm 78°$, the maxima found as the blue stars, and the linear patterns these produce. The coordinate [2, 2] arrives at an incidence angle of 45°, and thus reducing the visible region to ±40° from center of array, the source is only visible to the outmost edge of the array, and reducing θ further would render the source invisible to the array.

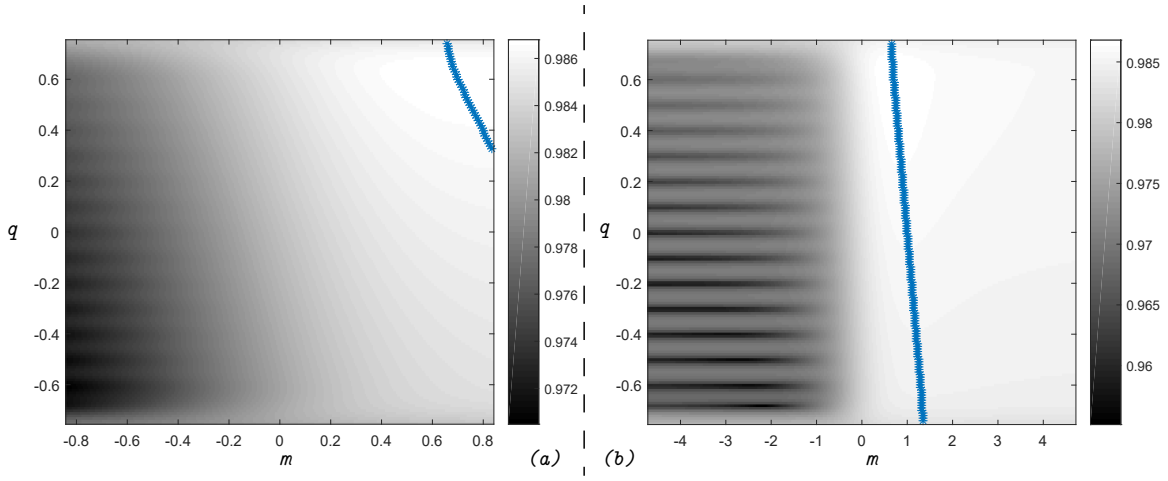


Figure 3.2: *(a) Ray Space, $\theta = \pm 40°$, true source location r' = [2, 2] estimation finds [2.463, 2.335]. (b) Ray Space, $\theta = \pm 78°$, , true source location r' = [2, 2] estimation finds [2.049, 2.043].*

Increasing θ increases W, which increases the complexity of the algorithm. Having θ run at a field incidence as opposed to the full range from -90° to 90° is due to $tan(90) = \text{inf}$, meaning W approaches infinity as the beamforming range approaches 90°. And as such, going beyond the chosen 78° exponentially increases W, which in turn exponentially increases complexity of the algorithm. E.g. with a spacing of $\bar{m_w} = 0.03$, $\theta = \pm 78°$, W = 315, with $\bar{m_w} = 0.03$, $\theta = \pm 85°$, W = 765.

Since m_w directly maps the Ray Space, it is assumed that by increasing the resolution, $\bar{m_w}$, decreases performance while decreasing it results in an increase in performance. Setting W to 63, five times larger interval than default, and setting the source further away to [5, 5], the estimation yields [5.307, 5.250]. With W = 1575, five times lower interval than default, estimation finds [4.892, 4.897]. With default settings, W = 315, we find [5.123, 5.123]. In this case it is not proven that the performance increases with smaller interval, likely due to the default resolution being plenty for the task given. Increasing the distance of the source to [20, 20], we estimate [6.899, 7.012] for W = 63, [20.493, 20.493] for W = 315, and [20.545, 20.588] for W = 1575.

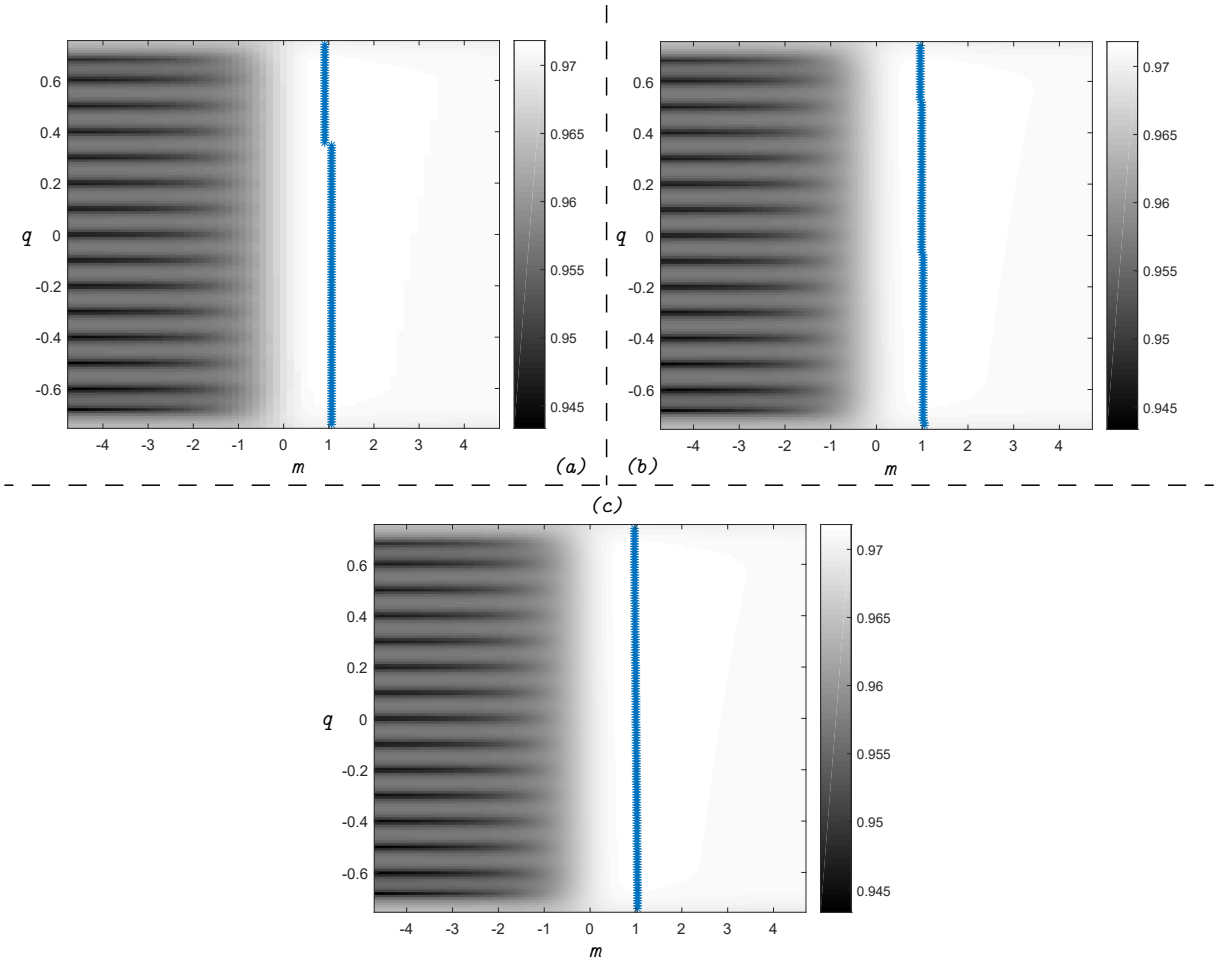


Figure 3.3: *(a) Ray Space, W = 63 ∴ $\bar{m_w} = 0.15$, true source location r' = [5, 5] estimation finds [5.307, 5.250]. (b) Ray Space, W = 315 ∴ $\bar{m_w} = 0.03$, true source location r' = [5, 5] estimation finds [5.123, 5.123]. (c) Ray Space, W = 1575 ∴ $\bar{m_w} = 0.006$, true source location r' = [5, 5] estimation finds [4.892, 4.897].*

Figure 3.3 shows the Ray Space generated using the three different values of $\bar{m_w}$, (a) for $\bar{m_w} = 0.15$, (b) for $\bar{m_w} = 0.03$, and (c) for $\bar{m_w} = 0.006$, source location in [20, 20]. Here it shows that some level of resolution is necessary to estimate when sources a far from the array. Figure 3.3(a) makes it clear that having a too low resolution makes the linear pattern in the ray space imprecise, as the resulting estimation also shows. Using the default resolution still achieves good estimations even at far distances, so no reason is found to increase this for simulation purposes.

3.2.2 Resolution in the q-plane

q_i, $i = 1, 2, 3...I$, represents the range and resolution of the Gaussian window which is applied to each sub-array[2]. The resolution of q_i, $\bar{q_i}$, can also be viewed as the spacing between sub-arrays. The default state of q_i is a range of $\pm q_0$, where q_0 is the maximum z value of the microphone array, z_L, and a resolution, $\bar{q_i}$, of $10 * 10^{-3}m$, I = 151. As with the m_w parameter we want to explore the effects on source estimation when changing resolution and range. It can be argued that changing the range from $\pm q_0$ does not make sense since it ensures that we get all possible

combinations within the microphone array.

Changing the sub-array spacing, on the contrary, can be used to either reduce complexity or improve resolution, as is the case with the resolution in m_w. Furthermore, the space between sub-arrays define when the observed sound has origin in the far field, i.e the smaller spacing, the shorter the distance to the far field. Normally, considering microphone arrays, the far field is defined to be bounded by $r = \frac{2*\Delta^2}{\lambda}$ [2][4][7], where Δ is the length of the array, and λ is the wavelength. Implementing the sub-array structure the requirement is lowered by using far field components for each sub-array, replacing the length of the entire array, Δ, in the equation, by the length of the sub-arrays, Δ_l.

Testing for a source in [20, 20], we find [20.493, 20.493] using default settings. Setting $\bar{q_i} = 1 * 10^{-3}m$, $I = 755$, a fifth of the default interval value yields [20.912, 20.632]. Decreasing the resolution fivefold, $\bar{q_i} = 50 * 10^{-3}m$, $I \sim 30$, yields [-Inf, -Inf]. Figure 3.4 shows the Ray Space generated using the three different values of $\bar{q_i}$, (a) for $\bar{q_i} = 50 * 10^{-3}m$, (b) for $\bar{q_i} = 10 * 10^{-3}m$, and (c) for $\bar{q_i} = 2 * 10^{-3}m$, source location in [20, 20]. It is obvious that the resolution on the q axis is improved in the Ray Space heatmap going from figure (a) to (c), source localization for (b) and (c) are good, while for a low resolution in (a) it completely fails. This happens due to the few points available for line detection is not enough data points to make a guess that is not just a vertical line.

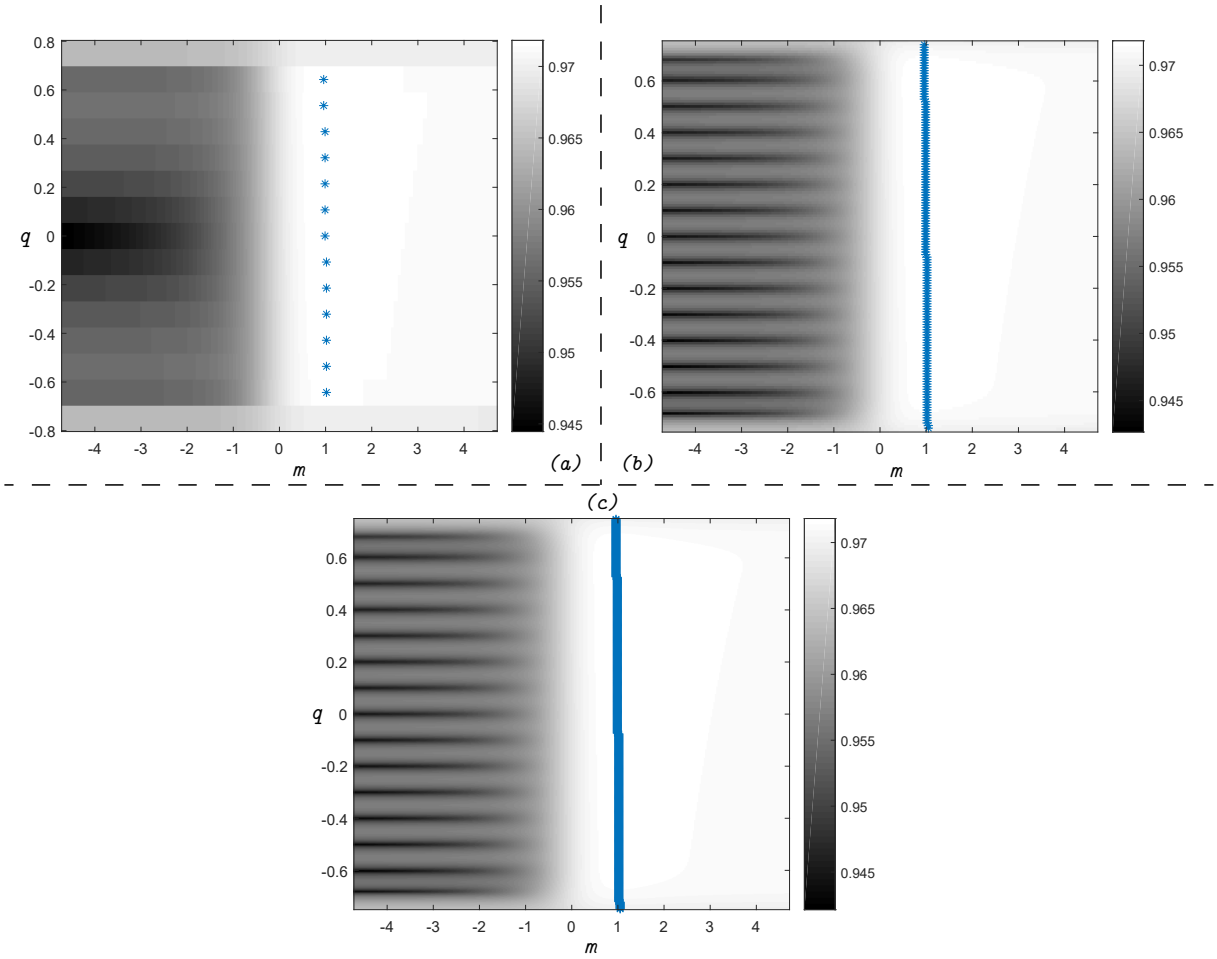


Figure 3.4: *(a) Ray Space, $I = 755$, $\bar{q}_i = 50 * 10^{-3}m$, true source location r' = [20, 20] estimation finds [20.912, 20.632]. (b) Ray Space, $I = 151$, $\bar{q}_i = 10 * 10^{-3}m$, true source location r' = [20, 20] estimation finds [20.493, 20.493]. (c) Ray Space, $I \sim 30$, $\bar{q}_i = 2 * 10^{-3}m$, true source location r' = [20, 20] estimation finds [-Inf, -Inf].*

3.2.3 Frequency

ω is the frequency analyzed in the Ray Space transform. While only one frequency is used per generation of Ray Space(Z matrix), the product sum of resulting matrices can be used to include the content of more frequencies. The product sum taken and used, as this is how the it is applied in [3]. Whether or not averaging over frequencies also work, it is not delved into in this thesis. Three Ray Spaces are produced, one with $\omega = 200 * 2\pi$, another with $\omega = 1000 * 2\pi$, and one with $\omega = 4000 * 2\pi$, see figure 3.5.

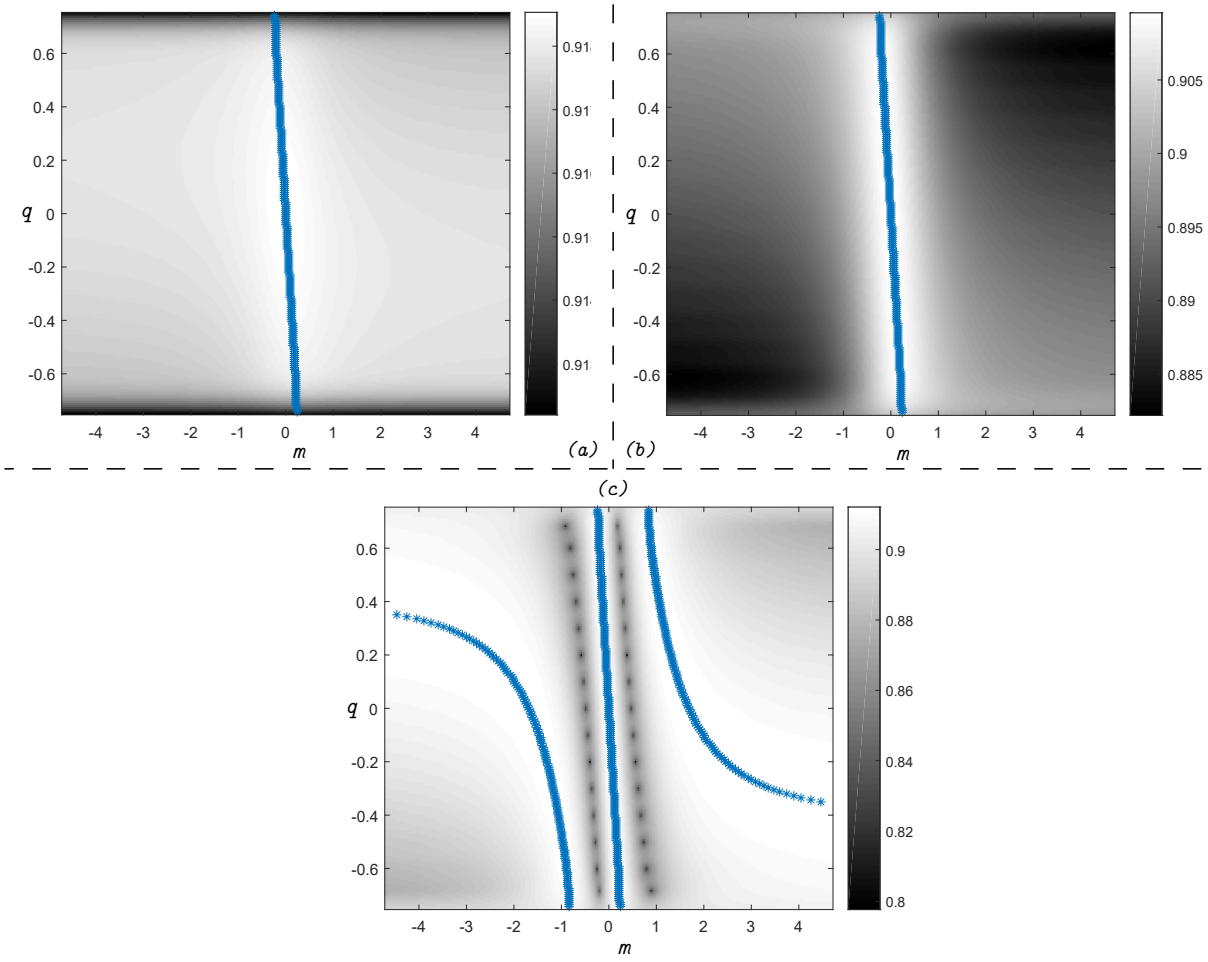


Figure 3.5: *(a) Ray Space, $\omega = 200 \cdot 2\pi$. (b) Ray Space, $\omega = 1000 \cdot 2\pi$. (c) Ray Space, $\omega = 4000 \cdot 2\pi$*

In each of the three ray spaces the source estimation is accurate, although the heatmaps look very different. It is hard to visually identify the 'ridge' of maxima in the 200 Hz ray space, the heatmap is overall more flat than those of higher frequency. This is caused by the wavelength of the signal, since the wavelength is significantly longer on the 200 Hz map, the signal will vary less on each spatial sampling in the microphone array than the others. The 4 kHz map shows three patterns, but only one linear pattern. The two polynomial patterns surrounding the linear pattern is a product of spatial aliasing, which happens when: $f < \frac{c}{2*d}$, where c is the speed of sound, d is the spacing between microphones[19]. The ray space represents this aliasing as non-linear patterns and as such the spatial Nyquist criterion can be safely ignored. The reason for this is a property of the uniform linear array layout, and a further explanation of can be read in article [11] section *C*. Avoiding the spatial aliasing criterion, the upper frequency limit is defined by the Nyquist frequency criterion, $f < Fs/2$. Using data from more than one frequency to generate a ray space ensures that the data is weighted for different frequency bands. This is helpful if the signal is very frequency specific.

3.2.4 Frame Averaging

Averaging over multiple frames in ideal conditions makes no difference in source estimation, since each frame would contain the same data. To gauge how averaging frames will affect the result in the real world, we add noise to a single frame of the input signal of the ray space transform. Averaging of frames is carried out by equation 3.1[2], which is the mean of each ray space frame.

$$\frac{1}{frames} \sum_{k=1}^{frames} Z(m,q,k) \tag{3.1}$$

Where **frames** is the total number of frames averaged, **Z** is the ray space matrix.
Averaging over five frames this leaves four frames with noiseless ray space estimations and one with added noise. White Gaussian noise with a level 6 dB lower than the signal is used for the noisy frame. The ray space with added noise is seen in figure 3.6(a), the ray spaces of the noiseless frames looks as figure 3.6(b).

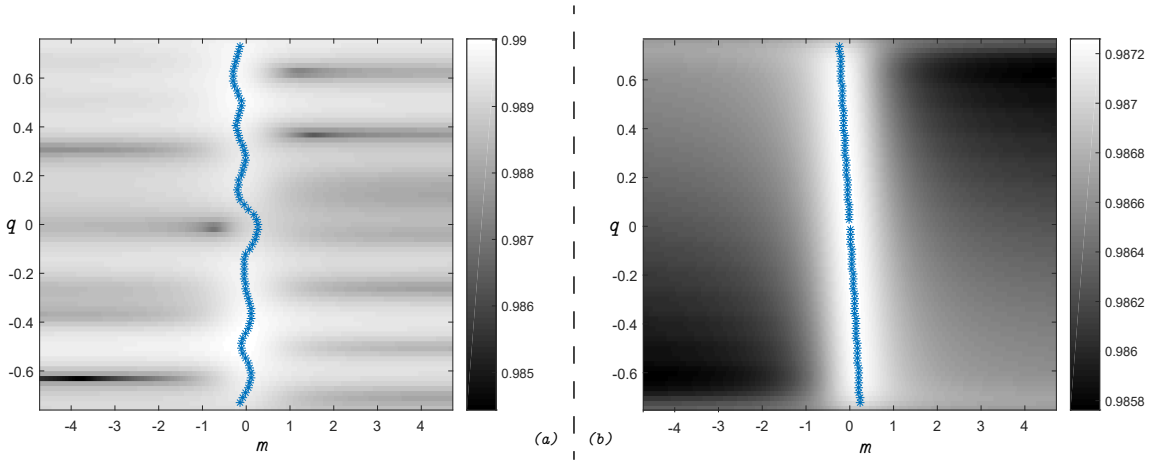


Figure 3.6: *(a) Ray Space, -6 dB white Gaussian noise added. (b) Ray Space, no noise.*

Averaging over the five frames, we obtain a ray space seen in figure 3.7. While the noise can still be seen, it now only accounts for a fifth of the data, and is suppressed by the noiseless frames.

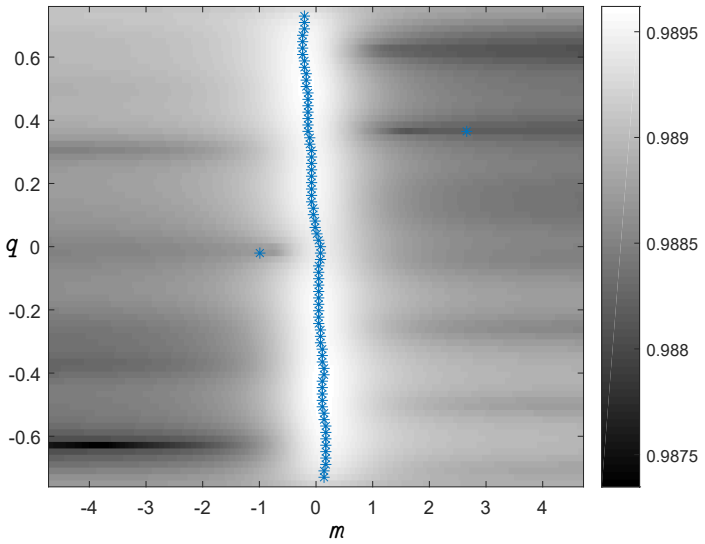


Figure 3.7: *Ray space averaging five frames.*

3.2.5 Width of the Spatial Window

As explained in section 2.1, the microphone array data is divided into subgroups and weighted spatially through the use of a moving spatial Gaussian window[2]. As can be seen in figure 3.8, the width of the Gaussian window, σ, can effect the formation of the sub-arrays and is mainly related with the distance between microphones.
Following observations can be stated:

1. If the Gaussian window is too wide compared to the distance between microphones, the formed sub-arrays will almost contain the same data each time, as can be seen in figure 3.8 (a).
2. If the Gaussian window is too narrow, compared to the distance between microphones, a beamforming operation will not happen at all, since the weighting never will take data from more than a single sensor, as can be seen in figure 3.8(b).

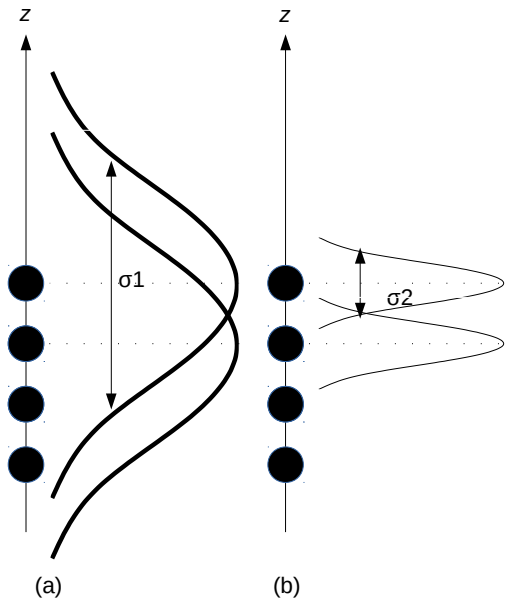


Figure 3.8: *Gaussian window width and distance between microphones*

Test Cases

To verify some the effects of the width of the Gaussian window, a simulation with two extreme cases and a five reasonable values are evaluated. Considering the default configuration scenario, with the sound source at position [3, 0] and a σ of 0.01[m] (one tenth of d) and 1[m] (ten times d), the ray spaces are in figure 3.9. Having a too low σ shows in that much of the ray space is without any actual acoustic data, since no maxima is found for many of the horizontal lines(or rows of the Z matrix), resulting in only few points of interest and a very poor resolution. Having a too high σ makes it consider almost every microphone for each iteration, generating an overflow of data, as well as artifacts which show themselves as nonlinear patterns of local maxima around the actual line. We try again with $\sigma = 0.05$ and 0.2, half of d and twice d. Note that authors in [2] use 0.2 for a microphone spacing of 0.1 m. Figure 3.10 shows these. At $\sigma = 0.05$, all lines now have data within them, but never from more than a single microphone, and often not much data from that microphone either. This makes itself clear in the striped pattern on the ray space, which shows that less data is present if the microphone is only at the edge of the spatial window. $\sigma = 0.2$, the default state, shows a much smoother ray space, with no visible problems in the figure.

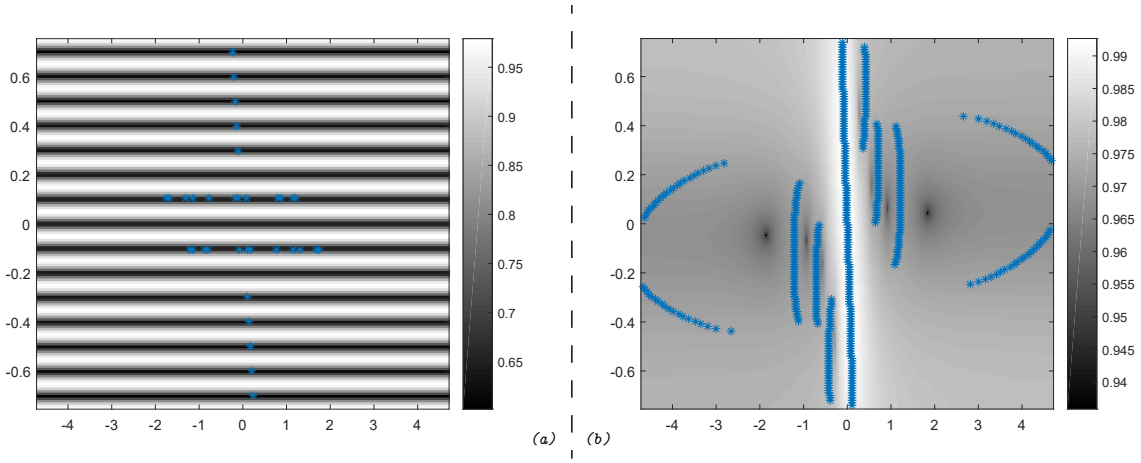


Figure 3.9: *(a) ray space with a spatial window width of 0.01[m]. (b) ray space with a spatial window width of 1[m]*

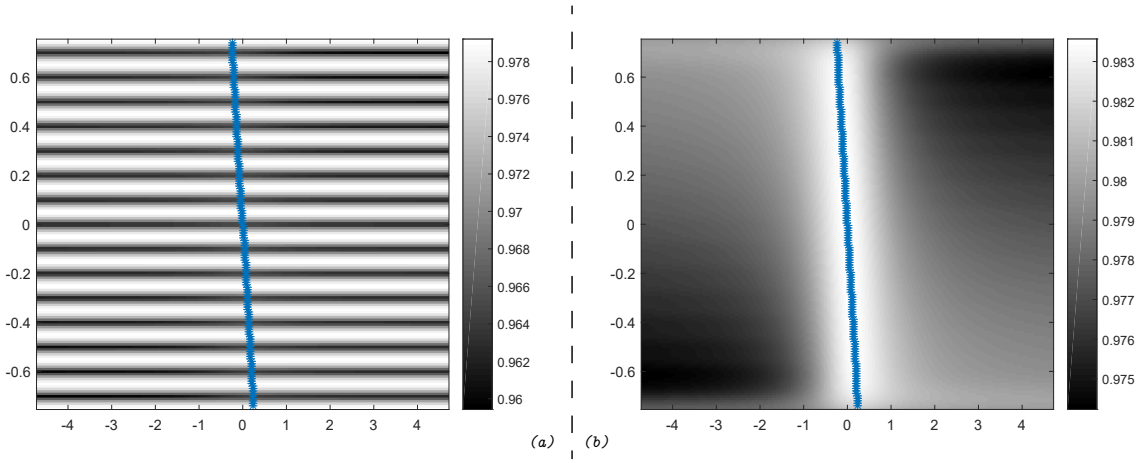


Figure 3.10: *(a) ray space with a spatial window width of 0.05[m]. (b) ray space with a spatial window width of 0.2[m]*

To verify that the spatial window is related to the distance between microphones, we lower the distance between microphones to $d = 0.04$, and reproduce the ray space in figure 3.10 with a σ of 0.02 and 0.08, half and double the spacing between microphones. See figure 3.11 for these.

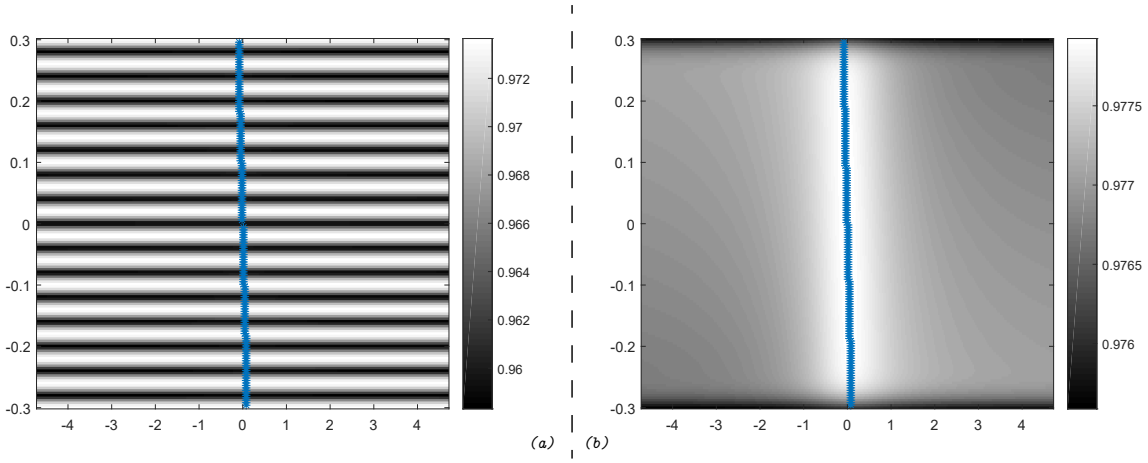


Figure 3.11: *(a) ray space with a spatial window width of 0.02[m]. (b) ray space with a spatial window width of 0.08[m]*

Observations

Having a spatial window width of around two times the distance between microphones seems to provide the better looking ray spaces with clear linear patterns. The estimation of source location is equivalent for the values tested, but this is due to the ideal simulated conditions. Increasing σ to use information from more microphones per movement of the spatial window might prove useful in scenarios not considered here.

3.2.6 Microphone Spacing

Along with the number of microphones in the linear array, L, the microphone spacing d defines the length of the array[2][3]. In itself it also determine the resolution of microphones along the array. Lowering d will lower the length and increase the resolution of the array and opposites for increasing d. We here study how changing this affects estimation of source location.
With static L = 16, we explore the following levels of d. Articles [2][3] use d = 0.1 and 0.06 respectively, so the values chosen are based around these.

- d = 0.02
- d = 0.04
- d = 0.06
- d = 0.08
- d = 0.1
- d = 0.12
- d = 0.14
- d = 0.16
- d = 0.18

Since σ defines how many microphones the Gaussian window averages over, we decide to make it scale with d in this case. Article [2] uses σ = 0.2 with d = 0.1, so here we use σ = d*2 to make sure it covers the same amount of microphones for each trial. Table 3.1 shows the values of d and the estimated source location. True source location r'_{true} = [3, 0].

d [m]	$r'_{estimated}$
0.02	[2.641, 0.155]
0.04	[2.834, 0.026]
0.06	[2.995, -0.036]
0.08	[2.963, -0.052]
0.10	[3.112, -0.004]
0.12	[3.092, -0.039]
0.14	[3.100, -0.039]
0.16	[3.092, -0.004]
0.18	[3.137, -0.030]

Table 3.1: *Level of d versus estimated values.*

At very low d, we see error in estimation. This is most likely due to lowering d roughly translates into the source being moved further away from array, from the perspective of the array. This is true since the lower d is, the earlier in space we enter the far field, see the second paragraph of subsection 3.2.2. Not much changes in estimation beside those, but the ray spaces show some differences due to the changing size of the array. Figure 3.12 presents four levels of d in the ray space, d = 0.02, 0.06, 0.10, 0.18.

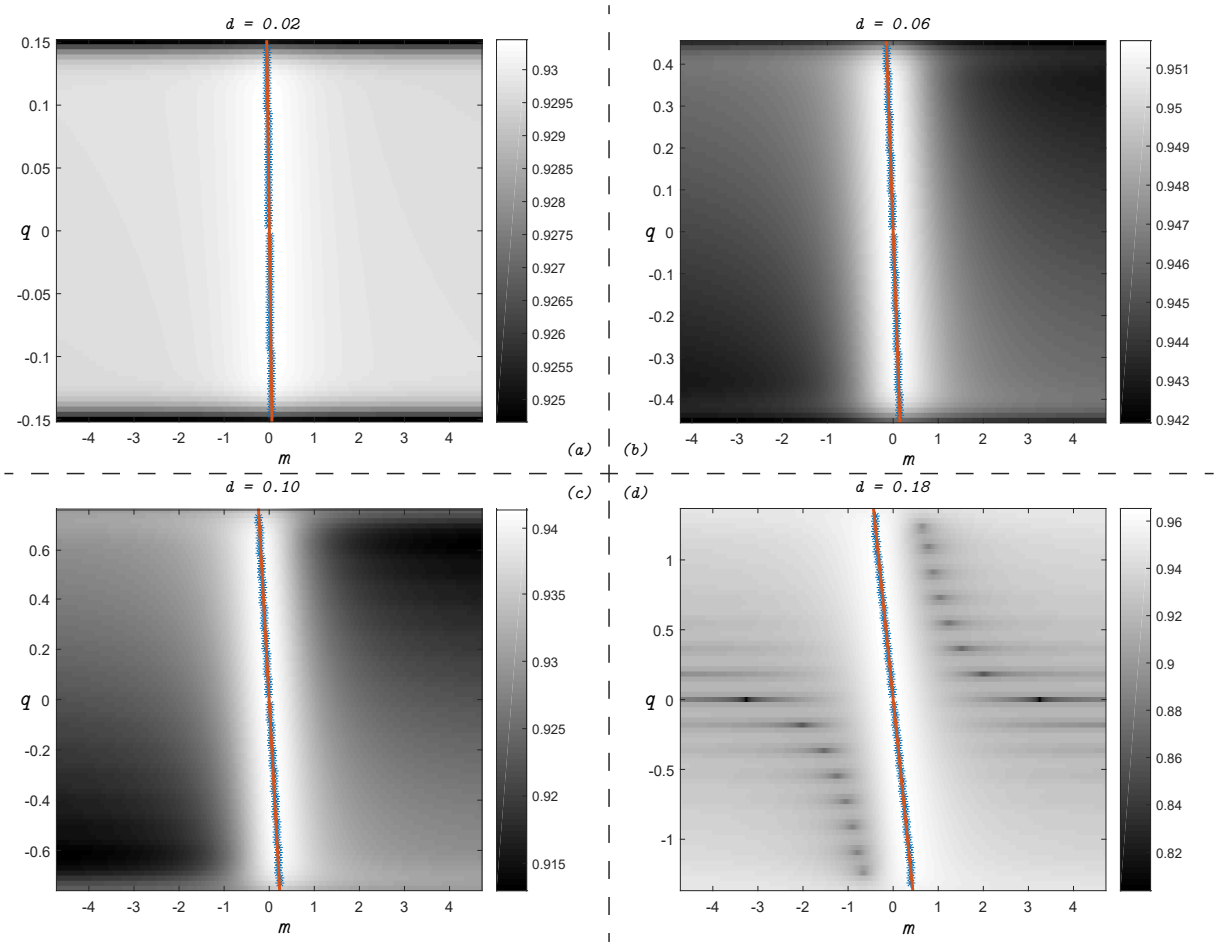


Figure 3.12: *(a) Ray space, d = 0.02. (b) Ray space, d = 0.06. (c) Ray space, d = 0.10. (d) Ray space, d = 0.18.*

3.3 Detection of Multiple Sources in the Ray Space

The Ray Space allows for the estimation of multiple sources in every iteration [3]. This section looks into the performance detecting two simultaneous sources in simulated ideal conditions. Using the same signal model as in the framework for a single source, we simply add more sources by creating more instances of $p(z, w)$ and adding them together. These are made by multiplying the same noise signal, $s(w)$, to transfer functions, $h(z|\mathbf{r}', w)$, with other sets of source coordinates, $\mathbf{r}'$. Setting two sources to $\mathbf{r'_1} = [2, 2]$ and $\mathbf{r'_2} = [2, -2]$. See figure 3.13 for a diagram showing the source positions relative to the microphone array. We use contributions of a combined 5 points in frequency, these being $f = [500, 801, 1105, 1409, 1713]$. In addition, the algorithm is averaged over 10 frames of 192 samples. $\sigma = 0.5$ is used in place of the default setting 0.2, since it is found during simulations that the method finds the sources using this setting and not the default, however this is not found to be true when estimating single sources. A deeper study into this was not carried out as a result of deadlines. We obtain source locations at [2.119, 2.085] and [2.091, -2.051].

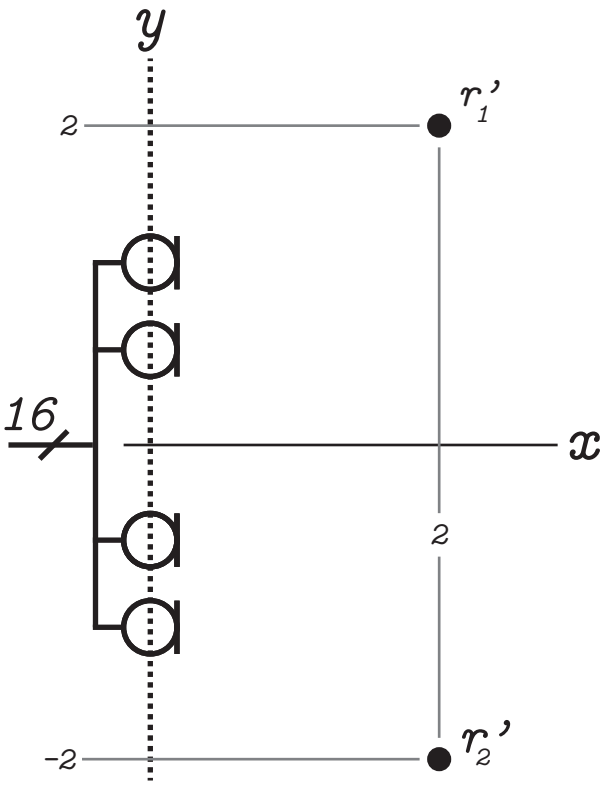


Figure 3.13: *Diagram showing the true source distances to microphone array. The filled black circles denote the speaker positions.*

The same test is performed for sources in [4, 4] and [4, -4]. Here we estimate the locations to lie in [4.059, 4.014] and [4.059, -4.014]. See the corresponding ray spaces in figure 3.14 (a) and (b).

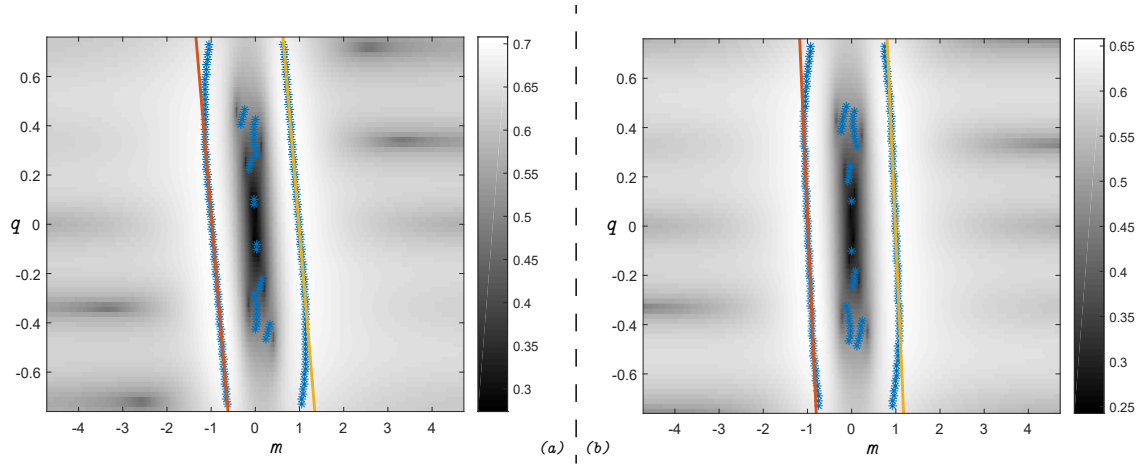


Figure 3.14: *(a) Ray Space,* $\mathbf{r'_1}$ *= [2, 2] and* $\mathbf{r'_2}$ *= [2, -2], estimations find* $\mathbf{r'_{1}}_{est}$ *= [2.119, 2.085] and* $\mathbf{r'_{2}}_{est}$ *= [2.091, -2.051]. (b) Ray Space,* $\mathbf{r'_1}$ *= [4, 4] and* $\mathbf{r'_2}$ *= [4, -4], estimations find* $\mathbf{r'_{1}}_{est}$ *= [4.059, 4.014] and* $\mathbf{r'_{2}}_{est}$ *= [4.059, -4.014].*

We then attempt placing the sources closer to each other, with sources in [2, 1] and [2, -1]. Here we find [2.353, 1.142], [2.300, -1.137], see figure 3.15 (a). A 10% offset is seen in the first dimension here, to explore further we see how it handles [2, 0.5] and [2, -0.5]. Now the method does not recognize any linear patterns in the ray space, and as can be seen in figure 3.15 (b), the rays interfere with each other and are obscured.

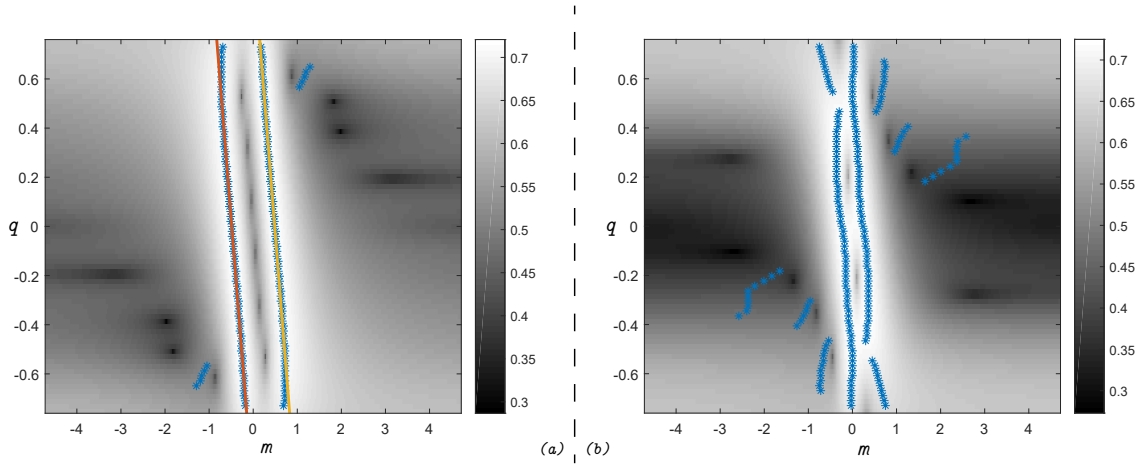


Figure 3.15: *(a) Ray Space,* $\mathbf{r'_1}$ *= [2, 1] and* $\mathbf{r'_2}$ *= [2, -1], estimations find* $\mathbf{r'_{1}}_{est}$ *= [2.353, 1.142] and* $\mathbf{r'_{2}}_{est}$ *= [2.300, -1.137]. (b) Ray Space,* $\mathbf{r'_1}$ *= [2, 0.5] and* $\mathbf{r'_2}$ *= [2, -0.5], estimations find* $\mathbf{r'_{1}}_{est}$ *= [N/A] and* $\mathbf{r'_{2}}_{est}$ *= [N/A].*

We then try [2, 0] and [4, 0] to see how the performance is with a source behind another. Here, the algorithm detects a single source in [2.773, 0], roughly between the two sources. Thus it seems if two sources are located in the same direction they are seen as a single source in between them. The ray space in figure 3.16 (a) shows this. The reason for this is that in the ray space the data of the two sources seem to blend together to make one source in between them. Changing some parameter might assist in this, but the time needed to go through them all is not available

during this project period.
Setting three simultaneous sources in $\mathbf{r'_1}$ = [2, 2], $\mathbf{r'_2}$ = [2, -2], and $\mathbf{r'_3}$ = [2, 0] the algorithm estimates locations in [2.337, 2.306], [2.183, -2.169], and [1.926, 0]. See figure 3.16 (b) for the ray space image.

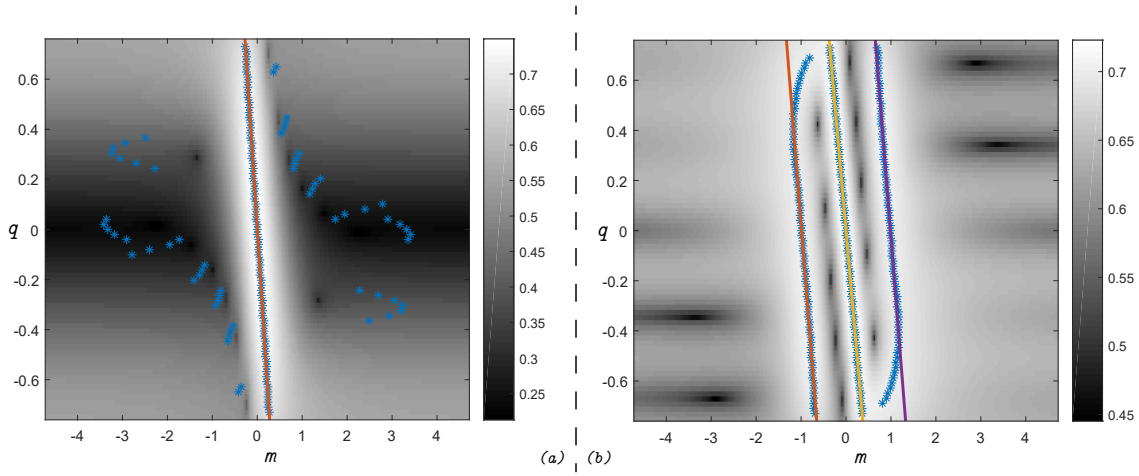


Figure 3.16: *(a) Ray Space, $\mathbf{r'_1}$ = [2, 0] and $\mathbf{r'_2}$ = [4, 0], estimations find single source in $\mathbf{r'}_{est}$ = [2.773, 0]. (b) Ray Space, $\mathbf{r'_1}$ = [2, 2], $\mathbf{r'_2}$ = [2, -2], and $\mathbf{r'_3}$ = [2, 0], estimations find $\mathbf{r'_{1}}_{est}$ = [2.255, 2.230], $\mathbf{r'_{2}}_{est}$ = [2.183, -2.169], and $\mathbf{r'_{3}}_{est}$ = [1.926, 0].*

Observations

Although the Ray Space RANSAC finds the multiple sources when they are at a certain distance from each other in the y axis, it is found that when one or more sources are significantly closer than others, the farther sources are invisible to the sound camera. It also misinterprets the sound field if two equal sources are very close to each other as they influence each others linear patterns. A more in depth study in to this is required to determine whether this is fixable by changing parameters.

3.4 RANSAC Parameters

This section covers the programmable parameters of the RANSAC line detection.

While different adaptations of the RANSAC algorithm has been developed to try and help with problems such as finding a correct inlier ratio for a dataset[12], we here only use the standard version of the RANSAC algorithm, since it is out of scope of this thesis to go very in-depth regarding line detection.

3.4.1 Inlier Ratio

The inlier ratio determines how high a percentage of the points in the provided dataset has to be an inlier on the linear pattern for the algorithm to accept the line[3][5]. If the inlier ratio criterion is not upheld, the RANSAC will not accept the proposed pattern and thus not return a proposed line. I.e. if this criterion is not upheld, there is not an acceptable linear pattern present in the dataset. It is important to keep this criterion strict, while not too harsh. This is true since

if the criterion is too lenient, it may include points in the linear pattern which to not belong, such as noise or points belonging to a separate linear pattern in the set. If the criterion is too harsh, the algorithm might struggle to ever find a linear pattern, since some points belonging to the pattern might not lie precisely on the linear pattern and thus be discarded. Figure 3.18 shows three values of inlier ratios: 0.05, 0.25 and 0.5 for a dataset.

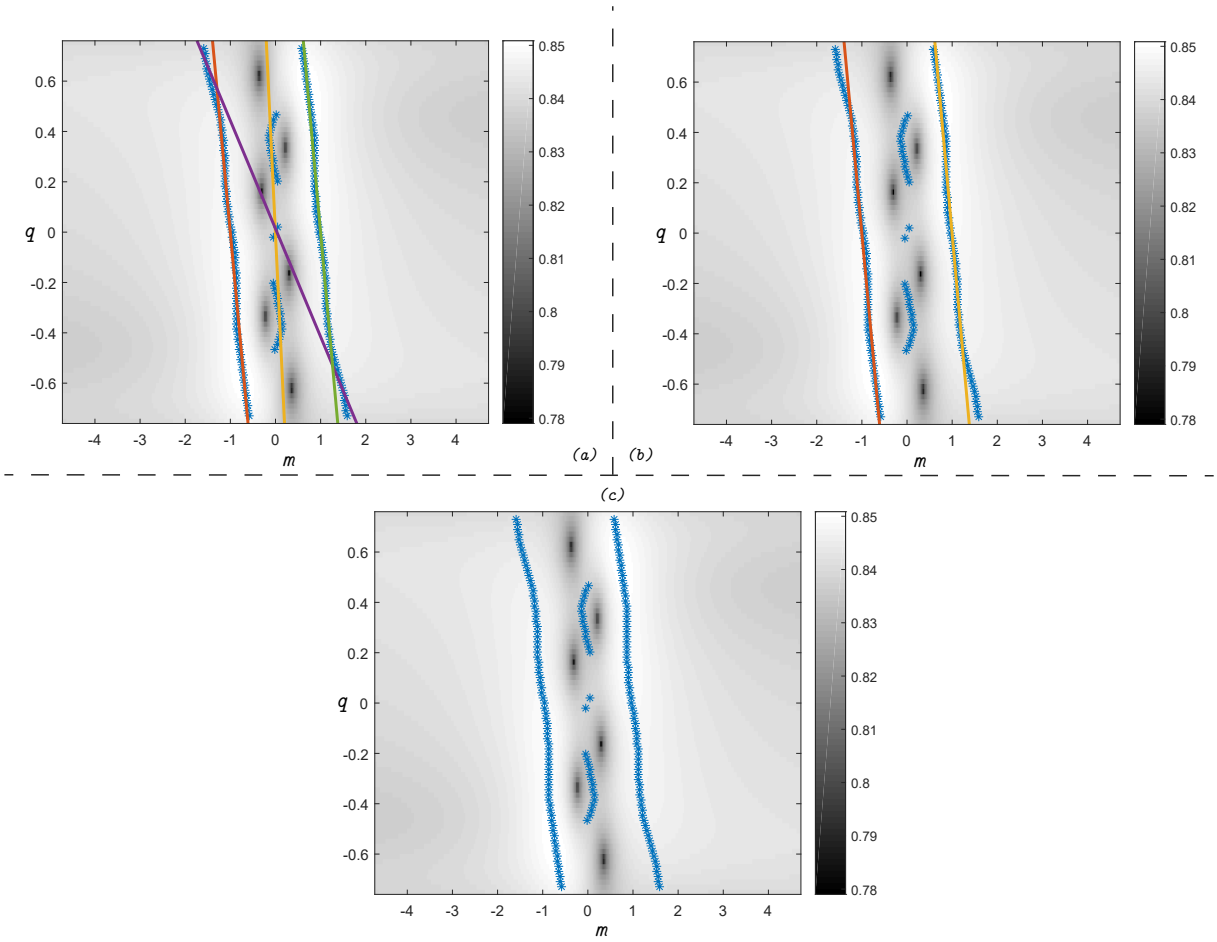


Figure 3.17: *(a) Ray Space, inlier ratio = 0.05. (b) Ray Space, inlier ratio = 0.25. (c) Ray Space, inlier ratio = 0.5.*

It is seen that a too low inlier ratio will find undesirable patterns while one that is too high will not find any patterns at all.
While .25 suits this dataset, the fact remains that different levels of inlier ratios will fit different types of datasets. For example if a dataset is very contaminated and contains very high amounts of outliers, the inlier ratio has to be quite low for the algorithm to accept any linear patterns within the noise.

3.4.2 Number of Iterations

The number of iterations determines how many times the algorithm will try to find inlier pairs and include them in the proposed linear pattern[3][5]. The most important factor here is to make sure enough iterations are performed to ensure that the correct linear pattern is found. The number required will scale with the number of local maxima input to the algorithm, which

scales with the resolution of the q and m axes, and how much noise is present in the data. 5000 iterations are used as a default state, and since it does not significantly change the time to run the algorithm, and no cases are found where increasing the number improves the outcome of the algorithm, this number is held constant for the time being. In the case that the RANSAC shows signs of bad performance, the number can be increased without concern.

3.4.3 Distance Threshold

The distance threshold sets the distance limit each point has to be within a line to be considered an inlier of it[3][5]. Simply put, increasing the distance threshold makes the algorithm consider the linear pattern to be wider, and decreasing makes it consider it as thinner. As is true with the inlier ratio, finding an optimal distance threshold varies for types of datasets. If a dataset is very contaminated, having a high distance threshold will likely make the algorithm consider large amounts of the noise present as inliers. While if the distance threshold is too low, small variations on the linear pattern might be enough to make the line invisible to the algorithm. Figure ref (a), (b), and (c) show ray spaces with distance thresholds of 0.01, 0.03, 0.3 respectively.

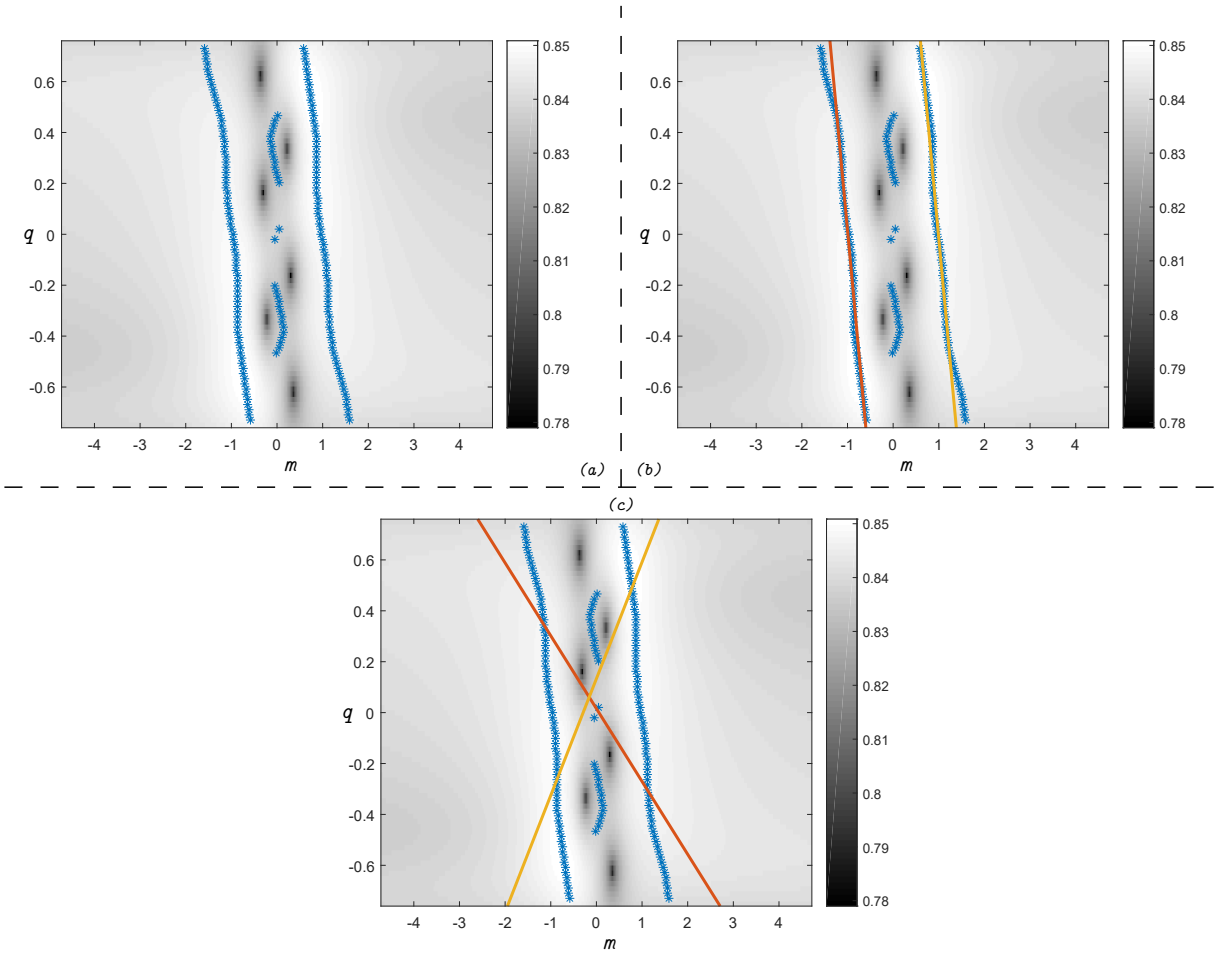


Figure 3.18: *(a) Ray Space, distance threshold = 0.01. (b) Ray Space, distance threshold = 0.03. (c) Ray Space, distance threshold = 0.3.*

With a low threshold it is seen that the algorithm does not recognize any linear patterns, and with very high distance threshold it finds arbitrary patterns which simply hold as many points

as possible while upholding the criterion.

3.5 SDM Parameters

This section contains a study on the SDM specific parameters. A block diagram in figure 3.19 gives an overview of the SDM simulations from start to finish. First the input signal is generated, then it is transformed to frequency domain by FFT. As in the RSTM, we apply the acoustic transfer function which simulates the signal coming from a simulated speaker position for each microphone in the array. While this can be done never leaving the time domain, implementing the exact same function on both methods ensure that the signal model is equivalent for the methods. After this we have $p(z, \omega)$, the pressure values at microphones for each frequency. By IFFT we obtain $p(z, t)$, the time signal at each microphone. This is used as input for the SDM estimation, which produces an array containing all possible estimated locations of the source, DOA. This is then ordered by which estimated locations have more energy in the reference microphone, p_{ref}.

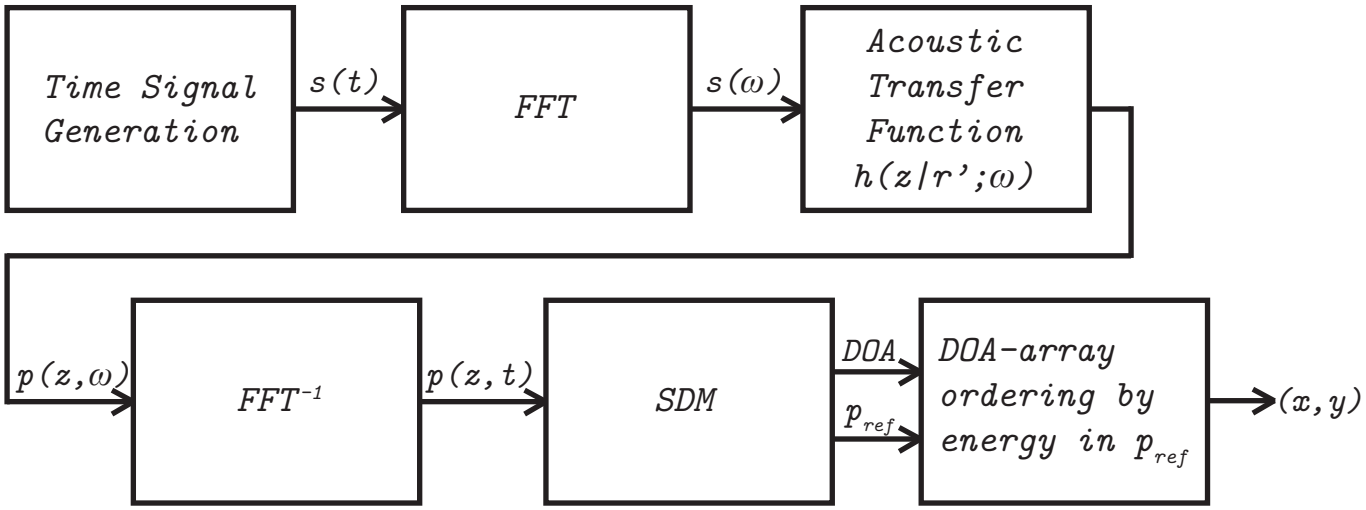


Figure 3.19: *Block diagram showing the process of simulating the SDM.*

An implementation of the method from one of the authors of article [1] is available for Matlab, and will be used to study performance aspects of this method regarding the parameters that defines it. In the same article some guidelines are stated about the parameters and are shown in table 3.2

Parameter	Guideline
Window Length - L	Larger than $2d_{max}c$
Maximum microphone spacing- d_{max}	The smaller the better
Number of microphones - N	The larger the better
Geometry of the array	-
Sampling Frequency -F_s	The higher the better

Table 3.2: *SDM parameters*

From this parameters, the number of microphones will be studied in comparison with the RSTM

as a physical parameters and the window length, L and maximum distance between two receivers, dmax, are studied in the same test case, as one defines the other.

3.5.1 Geometry

The microphone array geometry used in SDM is not specified in the method as in the implementation of RSTM[2], which we are analyzing. The following statements are taken from [1].

1. For 3-D spatial sound encoding, the minimum number of microphones is four, which are not in the same plane. It can be inferred that for 2-D sound analysis the minimum number is 3 complying with the condition of not being in the same line and forming a plane in the region of interest.
2. One microphone should be ommidirectional, or is possible to approximate a virtual one.
3. The dimensions of the array should be less or equal to the dimensions of a human head.

Looking at these requirements the following four configurations will be used within a simulation scenario. These four variations try to maintain the parameters discussed in table 3.2 in order to maintain a fair comparison. The analysis window length, L, is approximately 70 samples for all the configurations, and the number of microphones is sixteen for all of them, with exemption of the rectangular array in which is seventeen. Figure 3.20 shows a representation of the geometries used.

1. Half Wheel.
2. Random. The microphones will be distributed using a random positioning between them. This type of configuration is chosen because theoretically it presents advantages regarding the aliasing problem introduced by repeated sampling spacing which leads to ghost images of high energy due to regular grid configurations.[18]
3. Circular.
4. Rectangular.

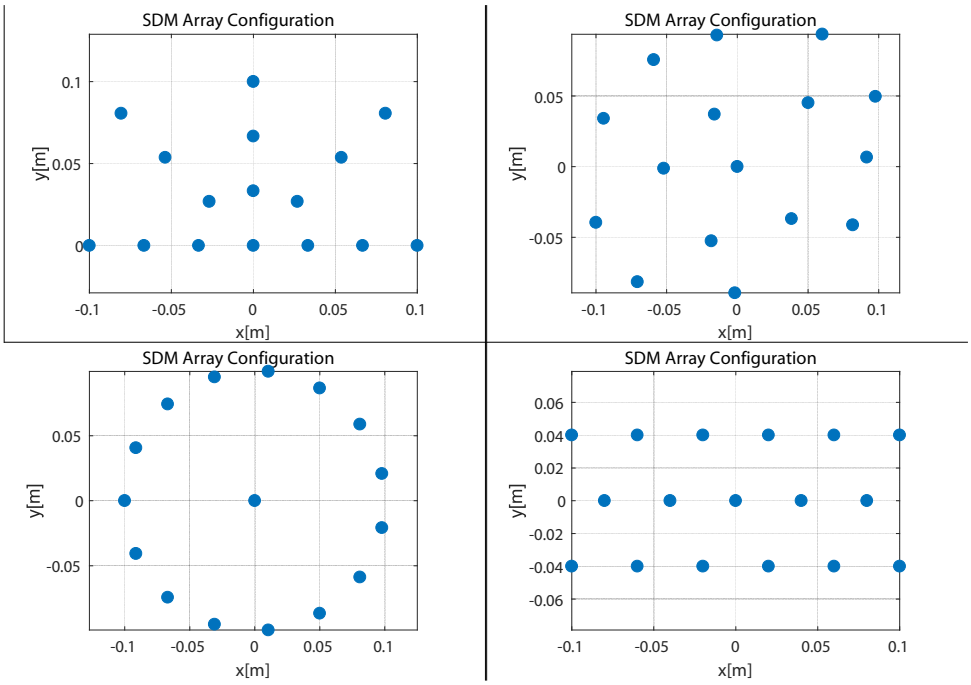


Figure 3.20: *Configurations evaluated for SDM*

Scenario

An anechoic scenario, with a ideal impulse emitted by single source is considered. Ten different positions, lying in a half circle around the origin coordinates is used for each array configuration. Figure 3.21

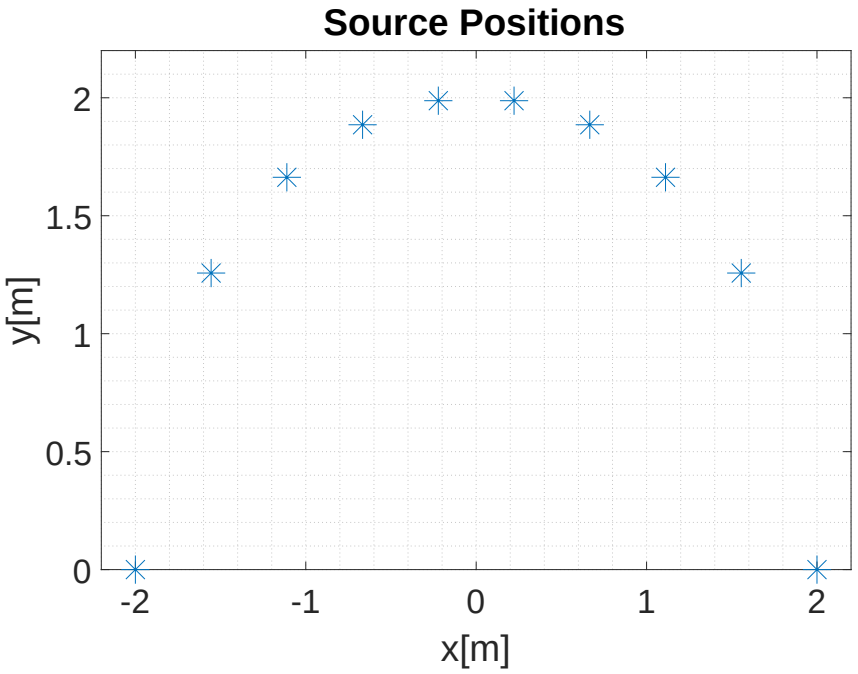


Figure 3.21: *Source positions used, one at a time*

Results

Figures 3.22 and 3.23 show the distance and angle error among the 10 considered positions.

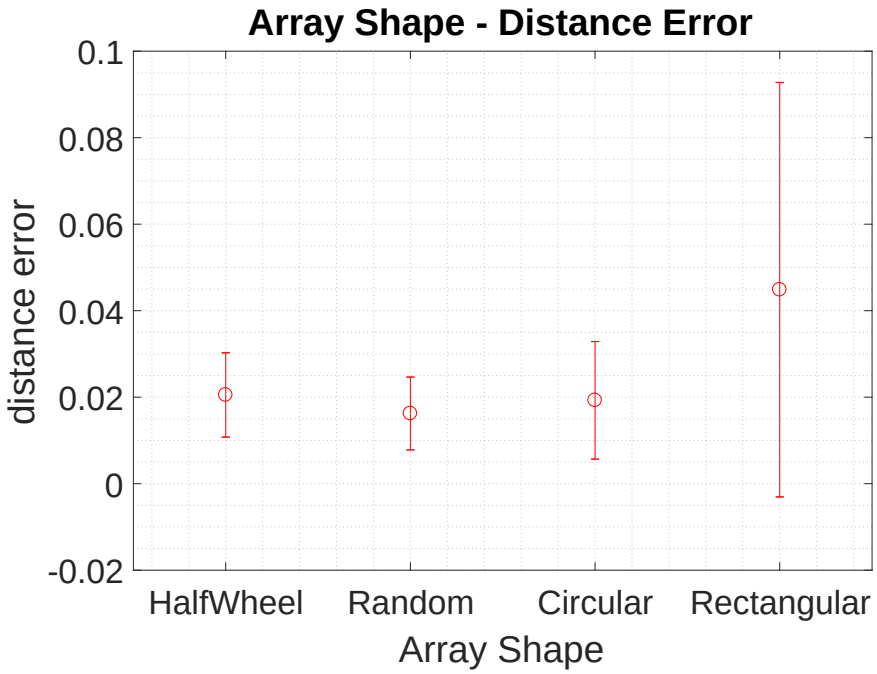


Figure 3.22: *Mean and standard deviation for the distance estimation error for the used array configurations*

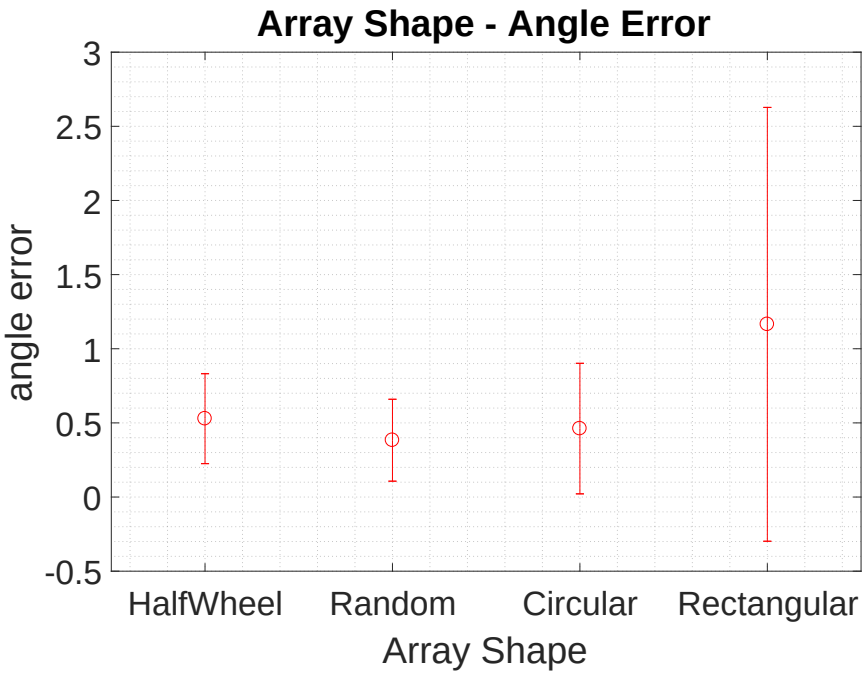


Figure 3.23: *Mean and standard deviation for the angle estimation error for the used array configurations*

Observations

The results show that the estimation for distance and angle presents the smaller mean and deviation error values for the random configuration. The configuration that performs worst is the rectangular, in which the distance error can be up to 9 [cm] in distance and 2.6 [deg] in angle. The other two configurations present performances close to the random array.

3.5.2 dmax and Temporal Window Size L

The maximum distance between any pair of microphones defines the recommended temporal window length. The window length should be slightly bigger than $2d_{max}/c$ [1]. To show how these parameters affect the estimation 3 configurations were tested. As the random array configuration is the one with the lower estimation error, it will be used in two enlarged versions. The first one is enlarged by a factor of 4, while the second one is enlarged by a factor of 5. The implementation used calculates the necessary window size value to comply with the described criteria. For the sampled configurations the values of L are 70, 254 and 314 samples, from the smallest to the largest array, being the dmax values 0.25[m], and 0.9[m], 1.1[m] respectively as can be seen in figure 3.24. The scenario explained in section 3.5.1 is used for the source position.

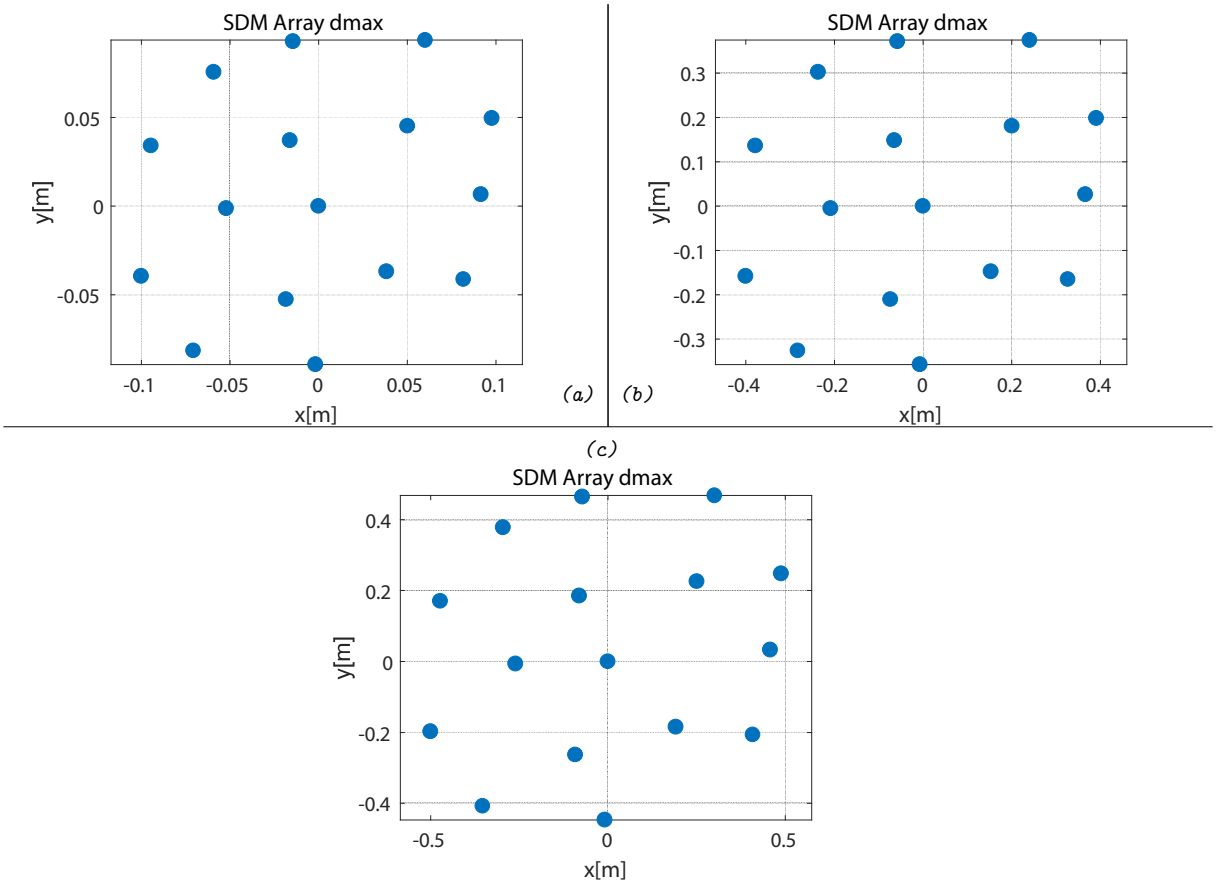


Figure 3.24: *(a) Original, dmax = 0.25 m. (b) 4 times original dmax = 0.9 m. (c) 5 times original dmax = 1.1 m.*

Results

The distance mean error and angle mean error among the proposed source locations are presented in figures 3.25 and 3.26

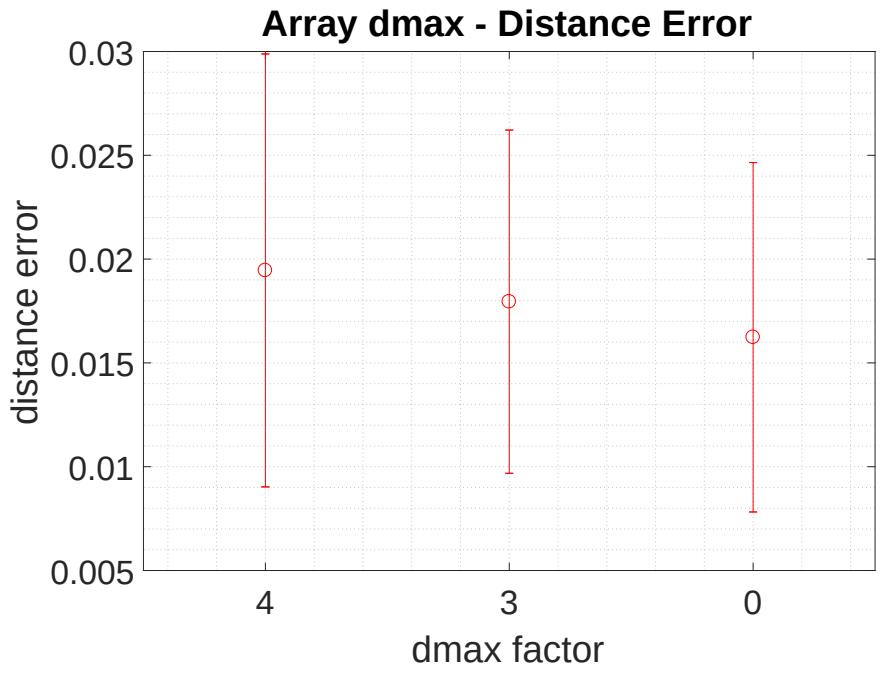


Figure 3.25: *Mean and standard deviation for the distance estimation error*

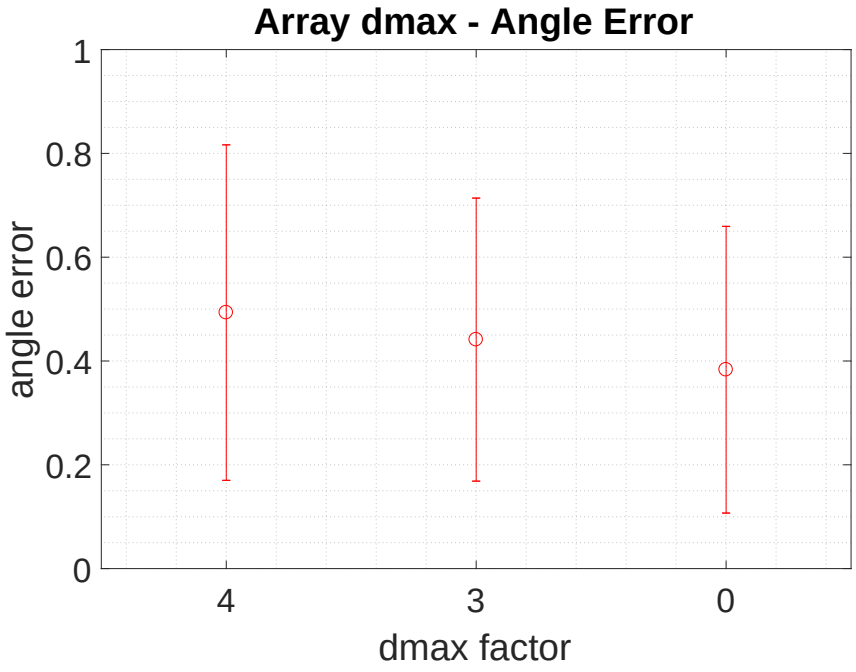


Figure 3.26: *Mean and standard deviation for the angle estimation error*

Observations

The smaller array presents the best mean estimation for the angle and distance estimation but the values are not of great significance, as they go from 2 to 1.5 [cm] for distance and 0.4 to 0.5 degrees in angle estimation. The difference between the standard deviation between the analyzed configurations is not different and don not present any important information. The real observed

difference between the configurations is the time that the algorithm takes to process them, being the larger one the one that takes more time to process.

3.6 Summary

A summary of the observations made on each method specific parameter.

RSTM Specific Parameters

- **[Resolution in the m-plane and Angular Beamforming Range]**: The beamforming angle range, θ, controls the angular range of which the array is able to see sources. If a source lies just outside the range of the center microphone, it might still be visible and able to be estimated if it lies within the beamforming range of the edges of the array. Increasing the beamforming angle range also increases complexity, and does so more the closer you get to grazing incidence of $\theta = 90°$. Having a very high angle range will result in a very high W. Having too low resolution in the m-plane can impair the estimation especially in sources far away, since the slope of the ray in the ray space becomes increasingly steeper the further away the source is. If the resolution then is too low, the line shows up as a vertical line with no slope, which results in invalid estimation. The default setting is found to be plenty for the test cases.
- **[Resolution in the q-plane]**: The resolution in the q-plane defines the size of the sub-arrays which are analyzed between microphones. As was the case for the m-plane, if the resolution in the q-plane becomes too small, the algorithm will not have sufficient variation in data points on the axis, and will result in invalid estimations if it becomes too small. The default setting is found to be plenty for the test cases.
- **[Frequency]**: The frequency parameter has two aspects to it: number of frequencies analyzed and chosen frequencies of interest. By analyzing multiple frequencies the method will use data from more frequency ranges, which can be of assistance if the signal lies within a certain frequency band. The frequencies used matter in the way that they define the bands of frequency one wishes to include, but spatial aliasing also becomes a point of interest when looking at frequencies above the criterion, $\frac{c}{2*d}$[19]. In the ray space the spatial aliasing shows itself as nonlinear patterns, and these can be ignored provided they do not interfere with any of the rays in the ray space. In the case of multiple sources, spatial aliasing will likely interfere with source rays in the ray space.
- **[Frame Averaging]**: Frame averaging is a tool to use data from sampled datasets at different points in time. This is especially helpful if noise is present in the signal, so that a smoother ray space can be generated from multiple noisy ray spaces, due to the variance being averaged over frames.
- **[Width of the Spatial Window]**: The width of the spatial window, σ, is found to be correlated to the microphone spacing, d. This is likely the case since σ decides how many microphones are considered per windowing, and how heavily the microphones are weighted. For single source estimation it is found that $\sigma = 2*d$ works well. A study was not performed to see how it might affect other scenarios, but it is of interest since the optimal value might change.
- **[Microphone Spacing]**: Keeping the $\sigma = 2 * d$, varying the microphone spacing only shows error in estimation when the spacing is at very low levels, around $d = 0.02m$. This is due to the far field being entered much earlier, thus making the source seem much farther away in the perspective of the microphone array. See the second paragraph of subsection 3.2.2 for a bit more on this.

RANSAC Specific Parameters

- **[Inlier Ratio]**: The inlier ratio defines how high a percentage of the local maxima need to be inliers for the method to accept the linear pattern. The optimal value for this will change with the scenario. For example if lots of the local maxima are the result of noise, then the percentage has to be low so that it actually accepts the linear pattern. The opposite is true for data with few maxima from noise, if it is too low it might start considering finding acceptable linear patterns in the noise after finding the real linear patterns.
- **[Number of Iterations]**: The number of iterations determines how many times the algorithm will try to find inlier pairs and include them in the proposed linear pattern. It is decided that since it does not in particular affect the run time of the algorithm, the number is kept at the default state of 5000. If more are needed, the number can always be increased without having to worry.
- **[Distance Threshold]**: The distance threshold determines the distance limit each point has to be within a line to be considered an inlier. As is the case for the inlier ratio, the optimal value for this will change with the scenario examined. If there is noise causing the ray in the ray space to be slightly nonlinear for parts of the line, while if you look at the ray as a whole, a linear pattern is clear, having a too low distance threshold will cause the algorithm to reject the line. Similarly if noise is present and the distance threshold is too high, it will start considering noise as inliers just because it is in the vicinity of the linear pattern.

SDM Specific Parameters

- **[Maximum spacing between a pair of microphones - d_{max}]**: This parameter is related with the temporal window size required. In the three versions of the random array used, none presented a significant difference in estimation of the source. A clear difference in the processing time is observed though, as the larger arrays require larger windows sizes, so the smaller array is preferred.
- **[Temporal Window Size L]**: As the smaller array is preferred, the smaller temporal window size is also preferred. The concern that can arise from these is the representation of low frequencies.
- **[Microphone array geometry]**: In the test-cases a smaller estimation error was observed for the random array configuration. The definition of the geometry of the array is a subject of investigation itself, but the only purpose of the test-case was to observe if some geometry performs better than the other proposed and use it.

4

Comparison Study

4.1 Introduction

The objective of the chapter is to compare the performance and capabilities of the SDM and RSTM methods applied in source localization. The input signal model will be the same for both methods. The main parameters to be assessed are the localization error, ϵ, defined as he distance between the estimated position $\hat{r'}$ and the real position of the source r', and the angle error, ϕ, defined by the absolute angle difference between the estimated position $\hat{r'}$ and source r'.

$$\epsilon = \left\| r' - \hat{r}' \right\| \tag{4.1}$$

$$\phi = \left\| atan(\frac{r'(2)}{r'(1)}) - atan(\frac{\hat{r}'(2)}{\hat{r}'(1)}) \right\| \tag{4.2}$$

4.1.1 Simulation Set-up

The next tables summarize the parameters used during the simulations. The values chosen are based on the default settings for the parameter study which are based on settings used in [2]. Beyond these, some parameters are changed in accordance with the more optimal values found in the parameter study and the rest remain in their original state since changing them did not improve performance. The number of frames analyzed is set to 10, to ensure some averaging is done when working with noise. The same goes for using five frequencies, to use content from frequencies across multiple ranges. The highest frequencies used here is 1713 Hz, since the nonlinear patterns which show when aliasing occurs might obscure the data when significant noise is induced.

Parameter	Description	Value
N	Number of samples per frame	192
frames	Number of analyzed frames	10
fs	Sampling frequency	$48k$Hz
L	Microphones in the linear array	16
d	Distance between the microphones	0.1 m
I	Samples on the q axis	150
W	Samples on the m axis	300
σ	Width of the spatial window	Single source, 0.2. Multiple sources, 0.5.
G	Number of analyzed frequencies	5
ω	Analyzed frequencies	[500 801 1105 1409 1713] Hz
θ	Range of the analyzed frequencies	[-78° to 78°]
y	Input signal	White Gaussian noise, 0 dBw

Table 4.1: *RSTM parameters*

Parameter	Description	Value
L	Microphones in the random array	16
d	Minimum distance between the microphones	0.04 m
fs	Sampling frequency	$48k$Hz
N	Number of samples	1024
y	Input signal	Ideal impulse

Table 4.2: *SDM parameters*

Parameter	Description	Value
interNum	Number of iterations	5000
thDist	Distance inliers need to be to the proposed line	0.03
thInlrRatio	Percentage of the total points need to be inliers	0.25

Table 4.3: *RANSAC parameters*

4.2 Source Distance

To determine how increasing or decreasing distance from array to source affects the source position and angle estimation, the x coordinate in r' is set to different values, and the position is estimated for each method. The distances examined are:
$r_{dist} = [0.1, 0.25, 0.5, 1, 2, 3, 5, 7, 10, 13, 16, 19, 22][m]$
A diagram of these relative to the microphone array is seen in figure 4.1.

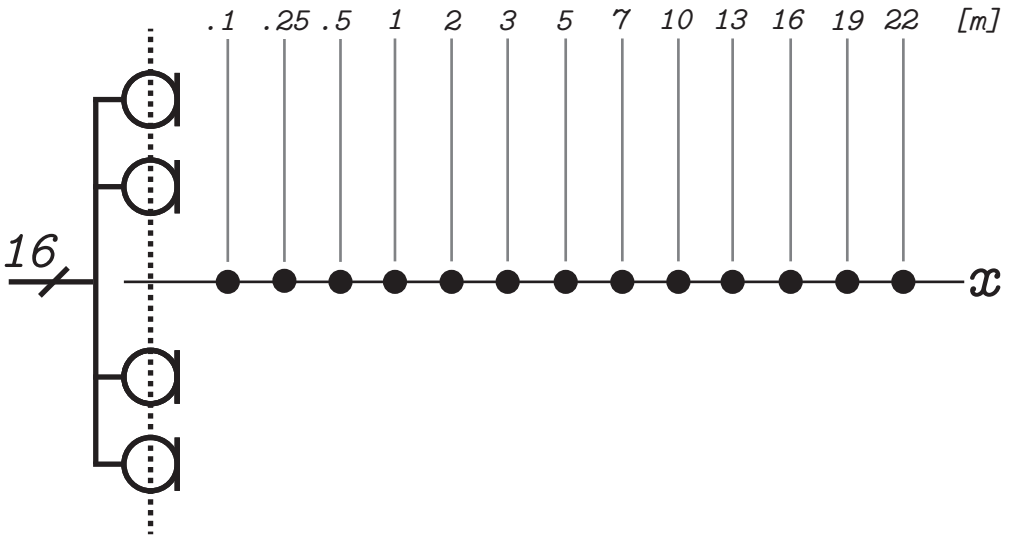


Figure 4.1: *Diagram showing the true source distances to microphone array. The filled black circles denote the speaker positions.*

RSTM

Figure 4.2 shows the results.

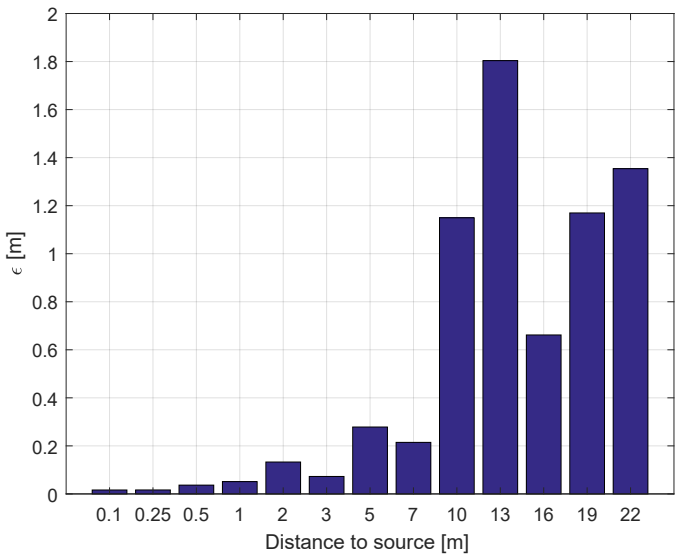


Figure 4.2: *Figure showing results for increasing source distance in RSTM, x axis shows the distance to source, y axis show the distance error, ϵ.*

The estimation is accurate from at short ranges, at least distances down to 0.1 meters, to distances up to 7 meters. At far distances the ability to estimate sources distance diminishes. At 10 meters and beyond the estimation error becomes erratic and fluctuates between 0.5 and 2 meters error. Reason for this is likely the fact that source distance is measured from the slope of the linear pattern in the ray space. When distance increases, the slope gets steeper, and becomes harder to

accurately estimate, since at this point very small changes in slope lead to very large changes in source distance. The ray space for the source at 22 meters distance in shown in figure ref, where the steepness of the slope can be seen. The angle estimates correctly for all distances.

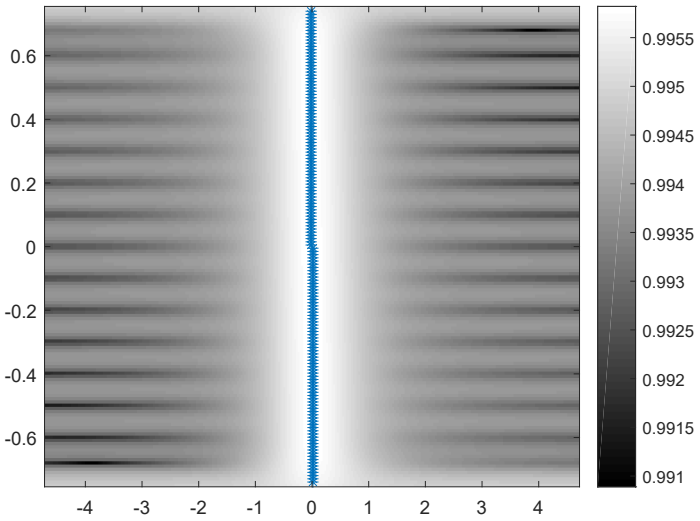


Figure 4.3: *Ray space at distance 22 meters, steepness of the linear pattern makes it hard to accurately estimate source distance.*

SDM

Figure 4.4 shows the results.

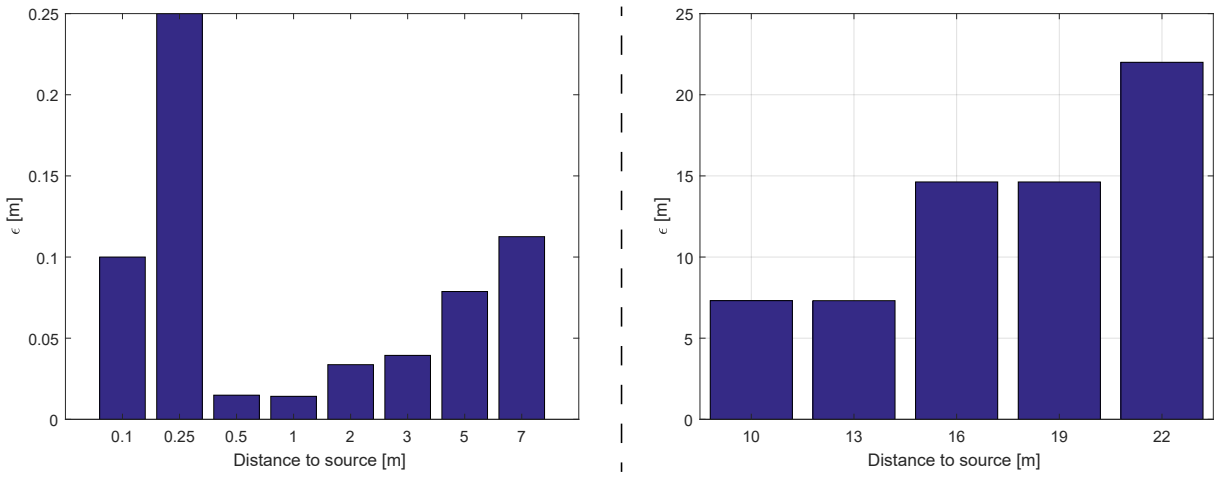


Figure 4.4: *Figures showing results for increasing source distance in SDM, x axis shows the distance to source, y axis show the distance error, ϵ.*

The figure shows that sources within .25 meters are not estimated correctly. The implementation assumes far field conditions, thus expecting a plane wave at the array. This makes it unable to handle sources in the near field. The estimations are accurate from .5 meters up to seven meters, whereafter the distance error increases with increase of distance. The error beyond seven meters

should to the authors best knowledge not present a magnitude so large as the observed, so it is assumed that it is a fault in not considering the delay time of the time signal compared to the samples analyzed. This was only realized at very late stage of the project, so no time was available to look objectively into the cause of it in the program.

4.3 Incidence Angle

By leaving the source distance static, and only altering the angle from the center of the microphone array to source position we find the relation between incident angle and estimation performance. The source distance is kept at 3 m, while we move in steps of 10°. Only one side is necessary to examine, since in ideal conditions we have symmetry. Figure 4.5 shows a diagram of the positions relative to the microphone array.

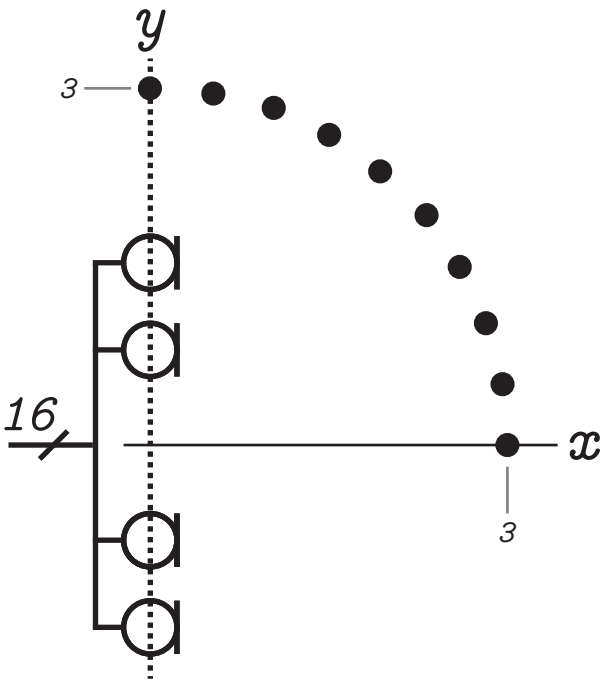


Figure 4.5: *Diagram showing the true source distances to microphone array. The filled black circles denote the speaker positions.*

RSTM

Incidence Angle	r'_{true}	$r'_{estimated}$	Distance error, ϵ [m]	Angle error, ϕ [°]
0°	[3, 0]	[3.112, -0.004]	0.112	0.075 °
10°	[2.954, 0.522]	[2.996, 0.518]	0.042	0.205°
20°	[2.818, 1.029]	[2.931, 1.068]	0.120	0.048°
30°	[2.598, 1.501]	[2.638, 1.530]	0.050	0.101°
40°	[2.297, 1.930]	[2.300, 1.930]	0.003	0.040°
50°	[1.928, 2.299]	[1.968, 2.340]	0.057	0.075°
60°	[1.498, 2.599]	[1.533, 2.655]	0.066	0.045°
70°	[1.025, 2.819]	[1.050, 2.883]	0.069	0.023°
80°	[0.519, 2.955]	N/A	N/A	N/A°
90°	[0, 3]	N/A	N/A	N/A°

Table 4.4: *True source position values versus estimated values for specific angles of incidence.*

It is seen in table 4.4 that the estimation performs well until we arrive at an angle beyond the beamforming range, θ, as would be expected.

SDM

Incidence Angle	r'_{true}	$r'_{estimated}$	Distance error, ϵ [m]	Angle error, ϕ [°]
0°	[3, 0]	[3.008,-0.039]	0.040	0.734°
10°	[2.954, 0.522]	[2.963, 0.520]	0.009	0.074°
20°	[2.818, 1.029]	[2.827, 1.030]	0.009	0.041°
30°	[2.598, 1.501]	[2.601, 1.512]	0.011	0.149°
40°	[2.297, 1.930]	[2.292, 1.949]	0.020	0.343°
50°	[1.928, 2.299]	[1.928, 2.310]	0.011	0.135°
60°	[1.498, 2.599]	[1.505, 2.605]	0.009	0.045°
70°	[1.025, 2.819]	[1.033, 2.826]	0.011	0.100°
80°	[0.519, 2.955]	[0.532, 2.961]	0.014	0.221°
90°	[0, 3]	[0.001,3.008]	0.008	0.013°

Table 4.5: *True source position values versus estimated values for specific angles of incidence.*

Since the SDM does not have any angular requirement, it estimates locations correctly for every defined incidence angle.

Observations

It is found that while within its beamforming range, θ, the RSTM performs similarly to the SDM. Both show low ϵ and ϕ when this is the case. The RSTM can not estimate sources outside its beamforming range, thus the SDM outperforms it here, showing no increase in error when the angle increases toward 90 °.

4.4 Signal to Noise Ratio

This section discusses the estimation of a single source position in an anechoic simulated environment but adding different amounts of white Gaussian noise in each of the receivers. The addition of the noise can be seen as any disturbance in the middle of the whole system.

Model of the added noise

In the frequency domain, the signal on a receiving microphone can be represented as $u(\omega) = h(\omega)s(\omega) + v(\omega)$. Where $h(\omega)$ represents the transfer function between the source and the receiver on an anechoic environment, $s(\omega)$ represents the signal produced by the source and $v(\omega)$ represents a noise signal with zero mean, defined variance equally distributed in all the frequencies. This is all based on the simulation study in section V of article [2].

To define the input signal to noise ratio(iSNR), we use the expression:
$iSNR(\omega) = \frac{|h(\omega)|^2 \phi_s}{\sigma_v}$
where σ_s is the variance of the source and σ_v is the variance of the noise. Additionally we represent iSNR in dB($20log_{10}$) for a better representation of the cases where the expression becomes fractional.

Source Location estimation

We compare the RSTM to the SDM in source localization.

Parameter	Description	Value
iSNR	Input signal to noise ratio	[-20 to 10] dB, 1 dB step size
Monte Carlo simulations	Number of realizations averaged	100 per iSNR level
r'	Source position	[1 0] [m]

Table 4.6: *Simulation Scenario*

Results

Figure 4.6 shows the amount of iSNR against the distance error, ϵ. The y axis has been limited to between 0 and 2 meters, since if it was expanded to contain larger error, it would not be possible to see the differences when the methods perform well.

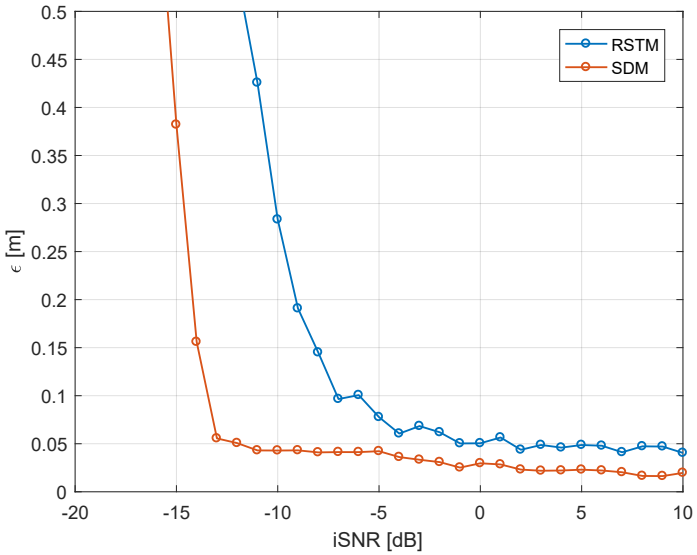


Figure 4.6: *iSNR vs distance error.*

The data in figure 4.6 shows that both methods perform well with noise with iSNR down to -5 dB, at least. The RSTM estimations become significantly worse towards -10 dB iSNR. The SDM performs slightly better, and falls off around -15 dB iSNR.

Observations

- When the iSNR is positive, meaning that the signal power in the receiver is higher than the noise power. The two methods are valid. The best approximation is made by the SDM method, followed by the RANSAC method and the regression method with a higher floor.
- The SDM method presents the best approximation even with a lot of noise added. Only producing invalid approximations when the iSNR is below -13[dB]
- The noise affects the RSTM earlier than SDM, and becomes invalid at -10dB iSNR.
- In general a very similar trend is seen comparing the RSTM found here and that found in [2] section V. The data we find, however, seem to have the RSTM perform better at higher levels of noise, even considering that we use $20log_{10}$ and in [2] $10log_{10}$ is used.

4.5 Multiple Sources

We compare the methods in estimating multiple sources to see how the methods perform in a more difficult task than single source estimation. To compare the two methods in estimating multiple sources, we define a region of space wherein two sources will be randomly generated for each iteration of the Monte Carlo simulations. This space is defined as only being in the positive x axis[1], between 0.5 and 3.5 meters, and between -1.5 and 1.5 meters on the y axis. 100 Monte Carlo simulations are performed on the estimation of the then hundred sets of sources generated. The comparison variable will then be the mean distance and standard deviation between sources. The two sources are generated separately with different seeds, and are thus uncorrelated and will not interfere with each other.

[1] Reason being the Ray Space only works in positive x for the configuration used.

RSTM

See subsection 4.1.1 for settings used in the ray space transform algorithm.

The simulation yields 63 pairs estimations of the source positions, and 37 cases of the algorithm detecting none, or only one source. Across the 63 pairs of estimation, a mean distance of 0.211 m to true position with a standard deviation of 0.358 is found. Thus when the algorithm succeeds in finding the correct amount of sources, it also fairly accurately finds their locations. To ensure that the number of Monte Carlo simulations is sufficient, figure 4.7 (a) shows cumulative mean of the distance error and standard deviation across the successful trials. Figure (b) shows the same figure, but with outlying results removed[2].

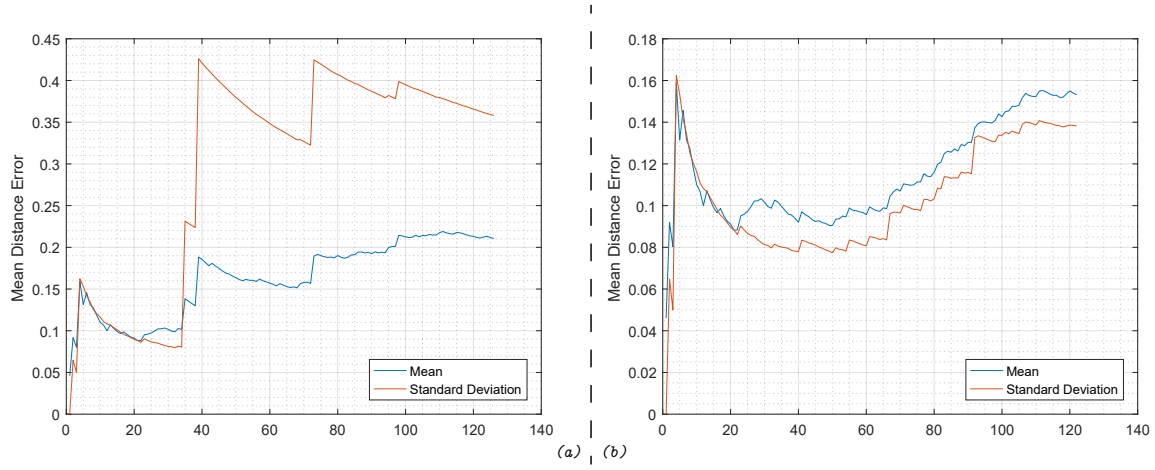


Figure 4.7: RSTM *(a) Cumulative mean and standard deviation distance error, including outlying results. (b) Cumulative mean and standard deviation distance error, excluding outlying results.*

Reasons as to why the algorithm does not always correctly find the sources is caused by the random generation of sources. As was determined in section 3.3, if sources are very close to each other, they influence the generated rays on the Ray Space, and may be represented as a combined ray of the two speakers, not representing any of them. Figure 4.8 shows a ray space of a case where the linear patterns are obscured.

[2]Results with distance error $>$ 2 m.

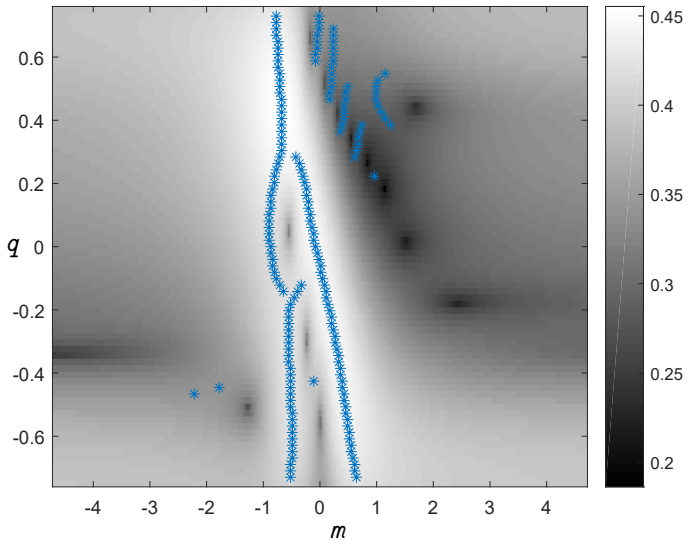


Figure 4.8: *Ray space of two sources interfering with each other, rendering localization impossible.*

SDM

See subsection 4.1.1 for settings used in the SDM algorithm.

The SDM Monte Carlo simulations find 100 pairs of source position estimations, thus none failed. A mean error of 0.141 m is registered with standard deviation of 0.369. The SDM finds the sources without fail and shows very similar mean and standard deviation in estimation to the RSTM. Figure 4.9 shows the cumulative mean and standard deviation for the SDM as figure 4.7 does for RSTM. Using these implementations, per estimation the SDM is much more confident in finding multiple sources.

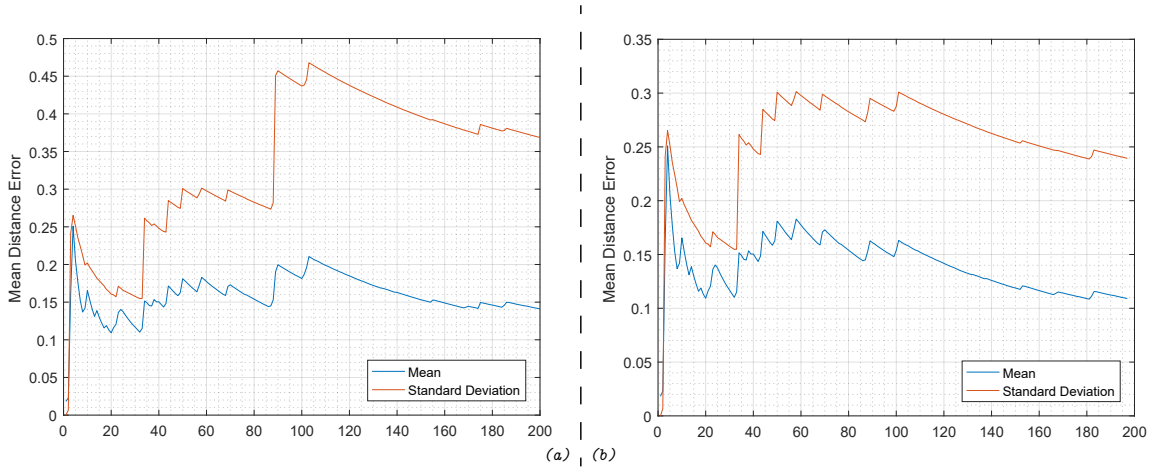


Figure 4.9: ***SDM*** *(a) Cumulative mean and standard deviation distance error, including outlying results. (b) Cumulative mean and standard deviation distance error, excluding outlying results.*

4.6 Microphone Position Error

A comparison is made on estimation with error in the microphone positions to obtain insight into the robustness of the two methods. Seeded random uniform noise is added to every microphone in both methods during estimation of source location. The methods are then compared in how well the estimation is performed while the microphones are shifted slightly out of position.

RSTM

To simulate microphone position error in the Ray Space, we introduce ten levels of uniform noise with a range of ±1 mm to ±10 mm, with 1 mm spacing, to the x- and y-coordinate of each microphone position. 100 Monte Carlo simulations are performed on each level of noise, each estimating a speaker located in [1,0].
Figure 4.10 shows the means and standard deviance of the distance error, ϵ, per noise level. Noise level 0 is estimation without noise, and levels 1 to 10 correspond to the increasing levels of noise. While minor differences are seen in the different noise levels, the standard deviation and mean remains low. It can be concluded that the method is robust and works with significant error in microphone positioning.

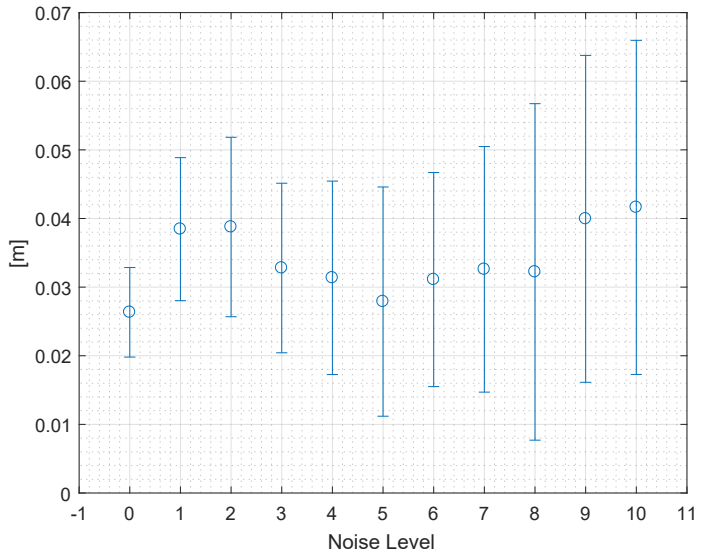


Figure 4.10: *Distance error mean and standard deviation per noise level.*

SDM

The same noise is applied for SDM, here both on x and y coordinates. Figure 4.11 shows the mean and standard deviation of the distance error estimation for each level of noise. Noise level 0 is estimation without noise, and levels 1 to 10 correspond to the increasing levels of noise. The distance error means and standard deviations is slightly lower than that of the RSTM across the noise levels.

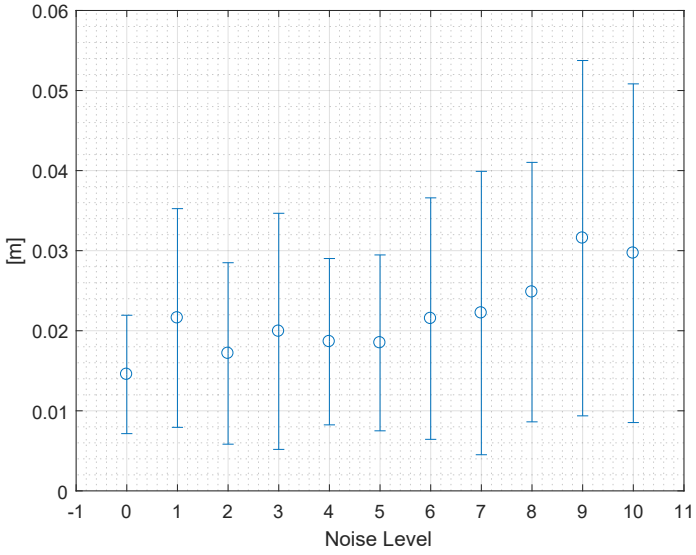


Figure 4.11: *Distance error mean and standard deviation per noise level.*

4.7 Microphone Phase Mismatch

When using non-phase matched microphones in the microphone array, which almost always will be the case, slight mismatches in phase will likely occur in between the microphones. In this section we simulate and attempt to determine the effect that phase differences in microphones has on source estimation. We introduce the phase mismatch in simulations as a delay on the input signal in random microphones. We then look at the distance -and angle error means and standard deviations through Monte Carlo simulations. Different amounts of delay are introduced in different numbers of microphones to get an overview of the effect of the phase delay, these are:

[1]: One random microphone delayed by 15 frames, corresponding to a delay of ∼0.3 ms.
[2]: Two random microphones delayed by 6 and 12 frames, corresponding to a delay of ∼0.12 ms and 0.16 ms, respectively.
[3]: Four random microphones each delayed by 3 frames, corresponding to a delay of ∼0.6 ms.

RSTM

Delay type **[1]**:
100 Monte Carlo simulations yield a mean distance error of 0.135 m with a standard deviation of 0.097 m. A mean angular error of 0.337° with a standard deviation of 0.242°. Figure 4.12 shows a ray space of one of the simulations, and shows how the error presents itself here.

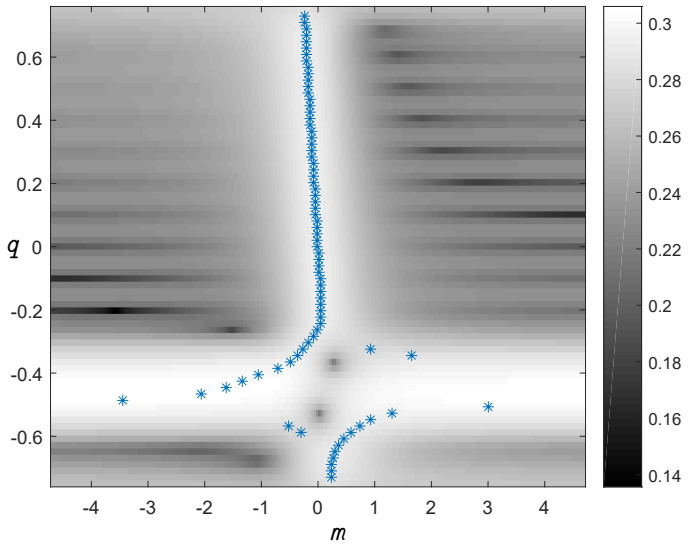


Figure 4.12: *Ray Space, 15 frame delay at microphone number 4(z = -0.45).*

The error is very visible in the ray space, but has no significant impact on the estimation accuracy since the RANSAC line detection still easily recognizes the linear pattern.

Delay type **[2]**:
100 Monte Carlo simulations yield a mean distance error of 0.135 m with a standard deviation of 0.137 m. A mean angular error of 0.362° with a standard deviation of 0.253°. Figure 4.13 shows a ray space of one of the simulations, and shows how the error presents itself here.

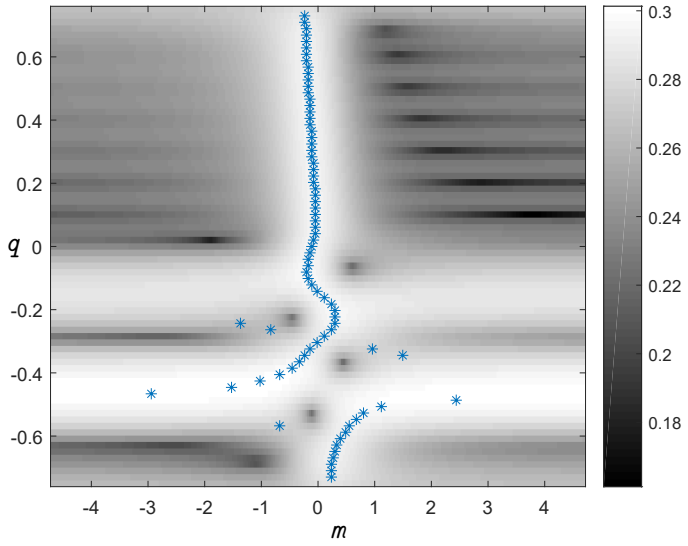


Figure 4.13: *Ray Space, 6 frame delay at microphone number 7(z = -0.15), 12 frame delay at microphone number 4(z = -0.45).*

As with delay type **[1]**, the RANSAC looks past the visible error in the ray space and continuously accurately estimates the source location. No significant difference in estimation is found between the two types of delays.

Delay type **[2]**:
100 Monte Carlo simulations yield a mean distance error of 0.135 m with a standard deviation of 0.137 m. A mean angular error of 0.362° with a standard deviation of 0.253°. Figure 4.13 shows a ray space of one of the simulations, and shows how the error presents itself here.

Delay type **[3]**:
100 Monte Carlo simulations yield a mean distance error of 0.498 m with a standard deviation of 0.652 m. A mean angular error of 0.006° with a standard deviation of 0.031°. Here the distance error increases significantly, which is the product of more of the ray space being distorted by mismatching. As seen in figure 4.12 and 4.13, if the noise only appears on few microphones, the linear pattern stays mostly intact, while if more microphones are affected, the line becomes more obscured.

SDM

Delay type **[1]**:
100 Monte Carlo simulations yield a mean distance error of 0.237 m with a standard deviation of 0.166 m. A mean angular error of 4.471° with a standard deviation of 3.205°.

Delay type **[2]**:
100 Monte Carlo simulations yield a mean distance error of 0.245 m with a standard deviation of 0.162 m. A mean angular error of 4.616° with a standard deviation of 3.123°.

Delay type **[3]**:
100 Monte Carlo simulations yield a mean distance error of 0.174 m with a standard deviation of 0.116 m. A mean angular error of 0.153° with a standard deviation of 0.066°.

SDM produces a slightly larger angular error than the RSTM in the first two types of delay, while the distance error is very close to that of the RSTM. No significant difference in the two types of delay are seen in the SDM either. The SDM outperforms the RSTM on delay type **[3]** on distance estimation.

Observations

Both methods show robustness to different types of phase mismatch. While slight larger angular error is seen in SDM tests, neither method fails to estimate sources with the types of delay introduced. In delay type **[3]**, the RSTM shows increase in distance estimation, due to the ray space linear pattern being obscured by having delay on multiple microphones. It can be argued that delay type **[1]** and **[2]** are very large and would probably result in changing the equipment used in the test. Nevertheless, the test shows that with even large mismatch error in few microphones, both methods are able to estimate correctly.

4.8 Input Signal Type

The methods performance in estimating sources with different types of input signals is done in this section. This gives insight into some of the different types of scenarios the methods are

usable within.

RSTM

In this section three different input signal types are tested to verify the validity of the RSTM for them. The configuration explained in section 3 is used primarily and any changes to it are explained.

1. A Gaussian white random process
2. An ideal dirac delta impulse
3. A Gaussian pulse with a defined frequency bandwidth and length
4. A sine wave

Gaussian Noise

This simulation uses a Gaussian noise signal with a total of 1600 samples, as can be seen in figure 4.14, is analyzed in 10 frames each with a length of 160 samples. The ray space transform of 4 frequencies ranging from 50 Hz to 1500 Hz are analyzed individually. Source location in [3, 0].

1. Length of the signal, length(y) = 1600
2. Samples by frame, N = 160
3. Frames = 10
4. Frequencies analyzed = [50 500 1000 1500]

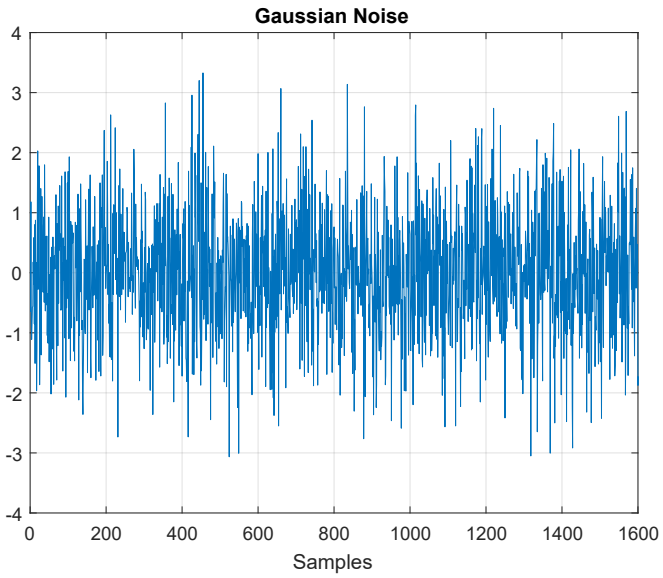


Figure 4.14: *Gaussian noise as input.*

The ray-space is presented for each frequency, alongside the location of the identified peaks in figure 4.15. Additionally the ray space heatmap for the inter-frames and -frequencies is presented in figure 4.16.

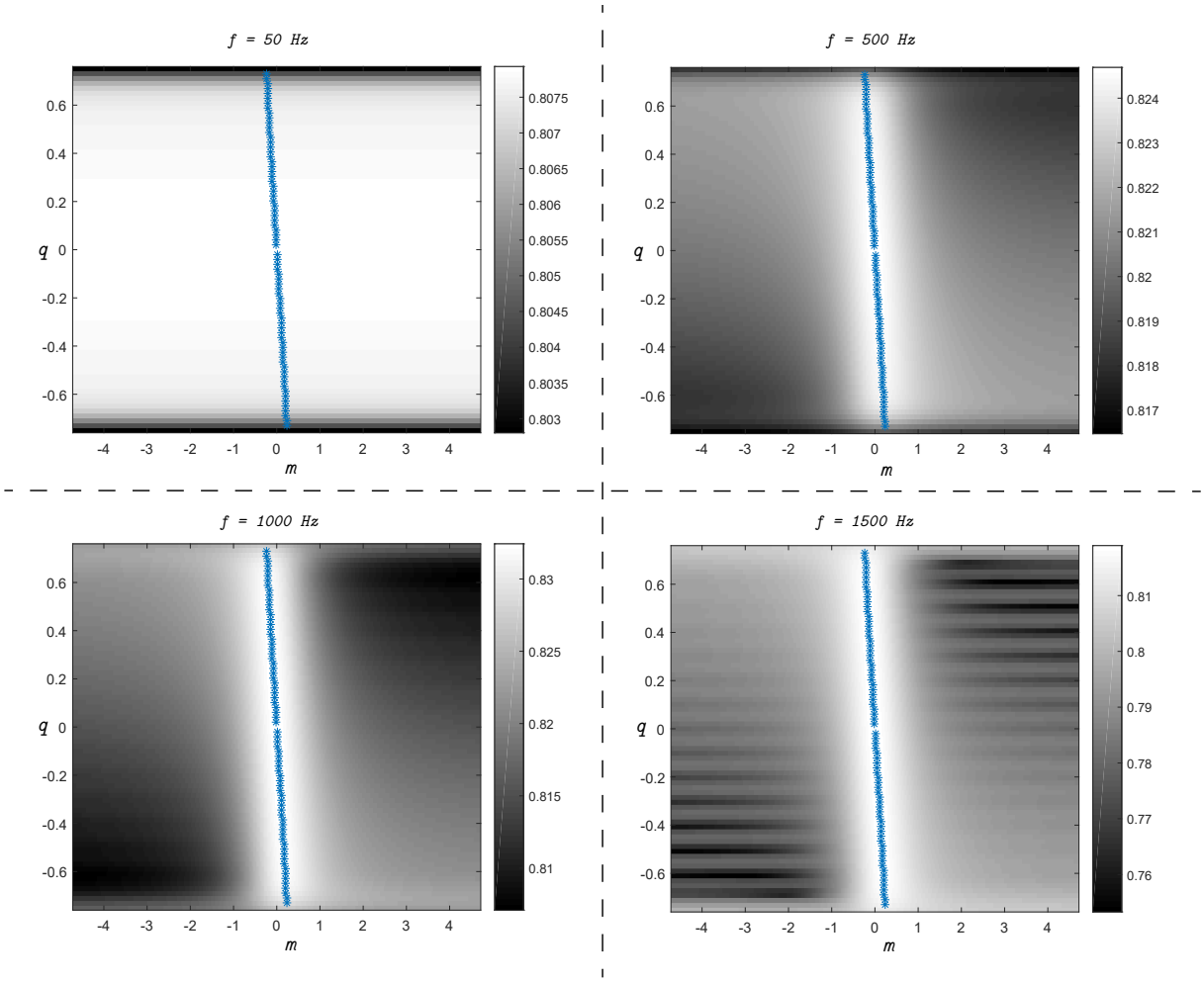


Figure 4.15: *Ray space heatmaps for four frequencies, using Gaussian noise.*

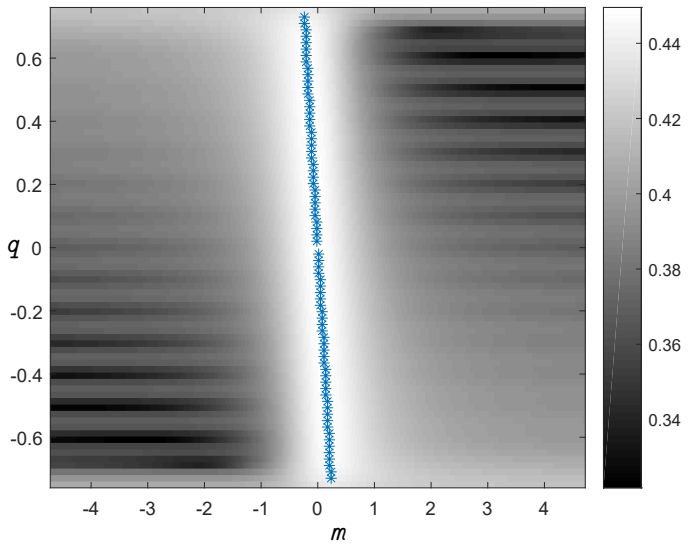


Figure 4.16: *Ray space heatmap, combined contribution of the four frequencies.*

It can be seen that in ideal conditions, combining frequencies does not change the outcome of

the ray space algorithm.

Dirac Delta

This simulation uses a signal of a total of 1600 samples, the dirac delta signal can be seen in figure 4.17.

1. Length of the signal, length(y) = 1600
2. Samples by frame, N = 160
3. Frames = 10
4. Frequencies analyzed = [50 500 1000 1500]

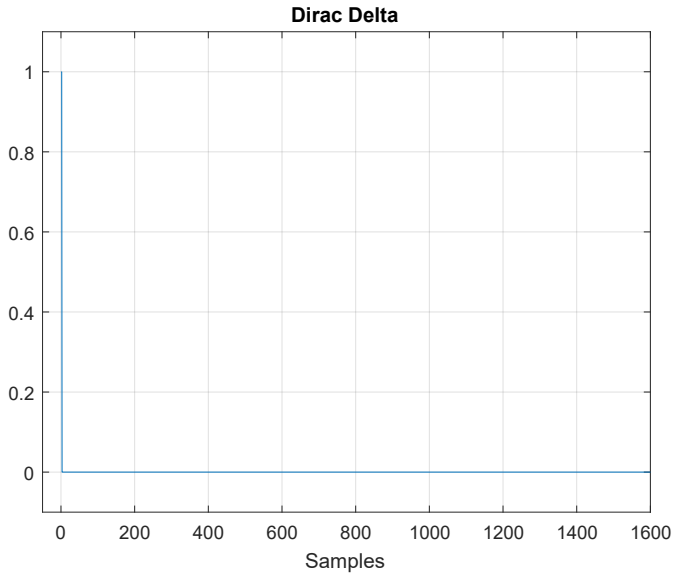


Figure 4.17: *Dirac delta as input.*

The ray spaces for each frequency analyzed for the delta dirac pulse are in figure REF. No combined ray space for these are shown since it is equivalent to that of Gaussian noise.

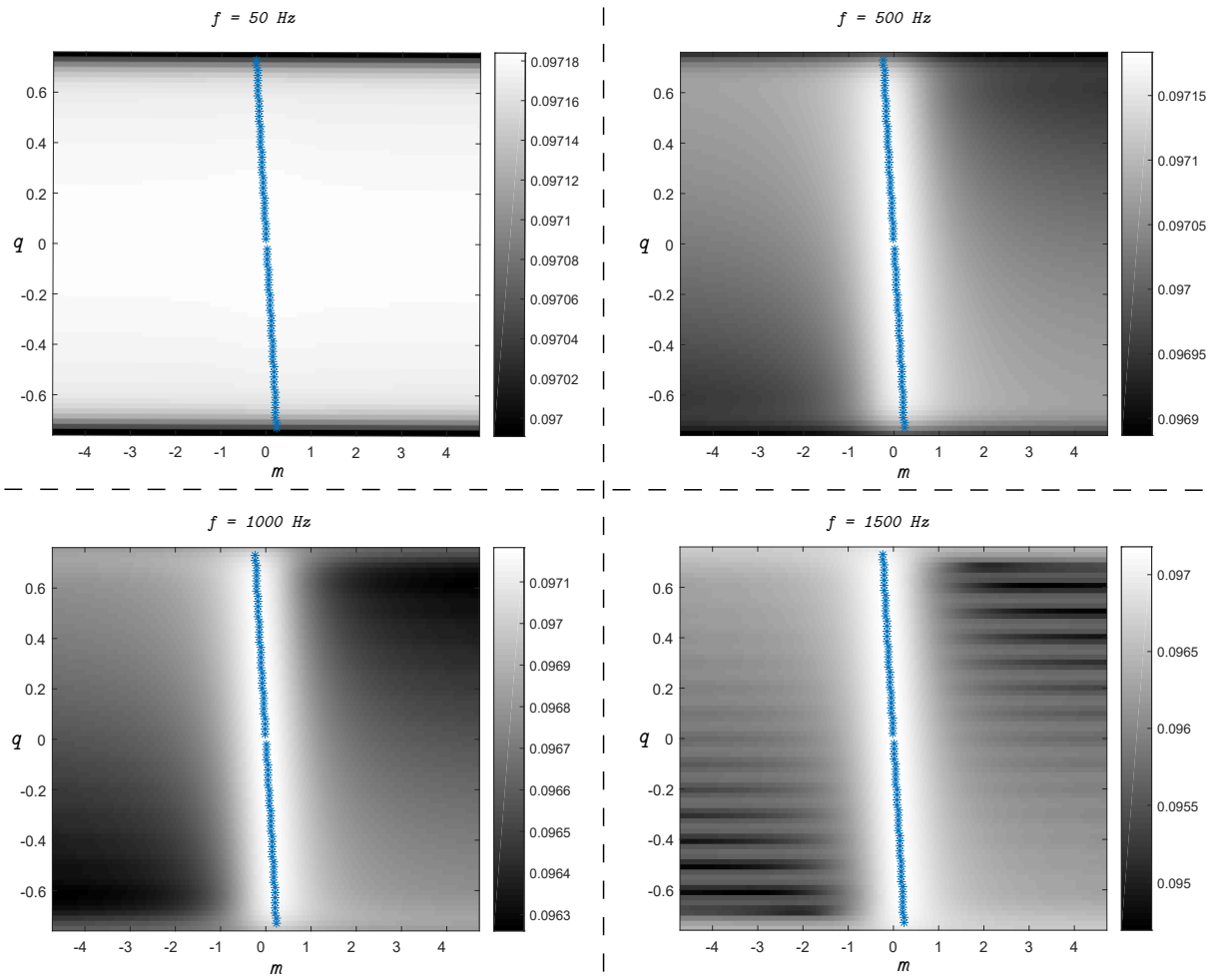


Figure 4.18: *Ray space for four frequencies, using a dirac delta pulse.*

Gaussian Pulse

A Gaussian pulse with a defined frequency bandwidth and length is tested and can be seen in figure 4.19.

1. Length of the signal, length(y) = 1600
2. Samples by frame, N = 160
3. Frames = 10
4. Frequencies analyzed = [50 500 1000 1500]
5. Central frequency = 1000 Hz
6. Bandwidth = 10

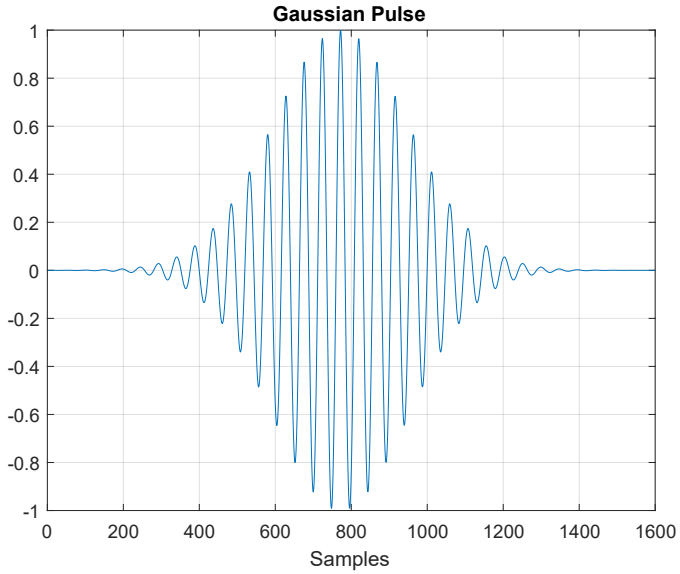


Figure 4.19: *Gaussian pulse as input.*

The ray-space is presented for each frequency in figure 4.20. No combined ray space for these are shown since it is equivalent to that of Gaussian noise.

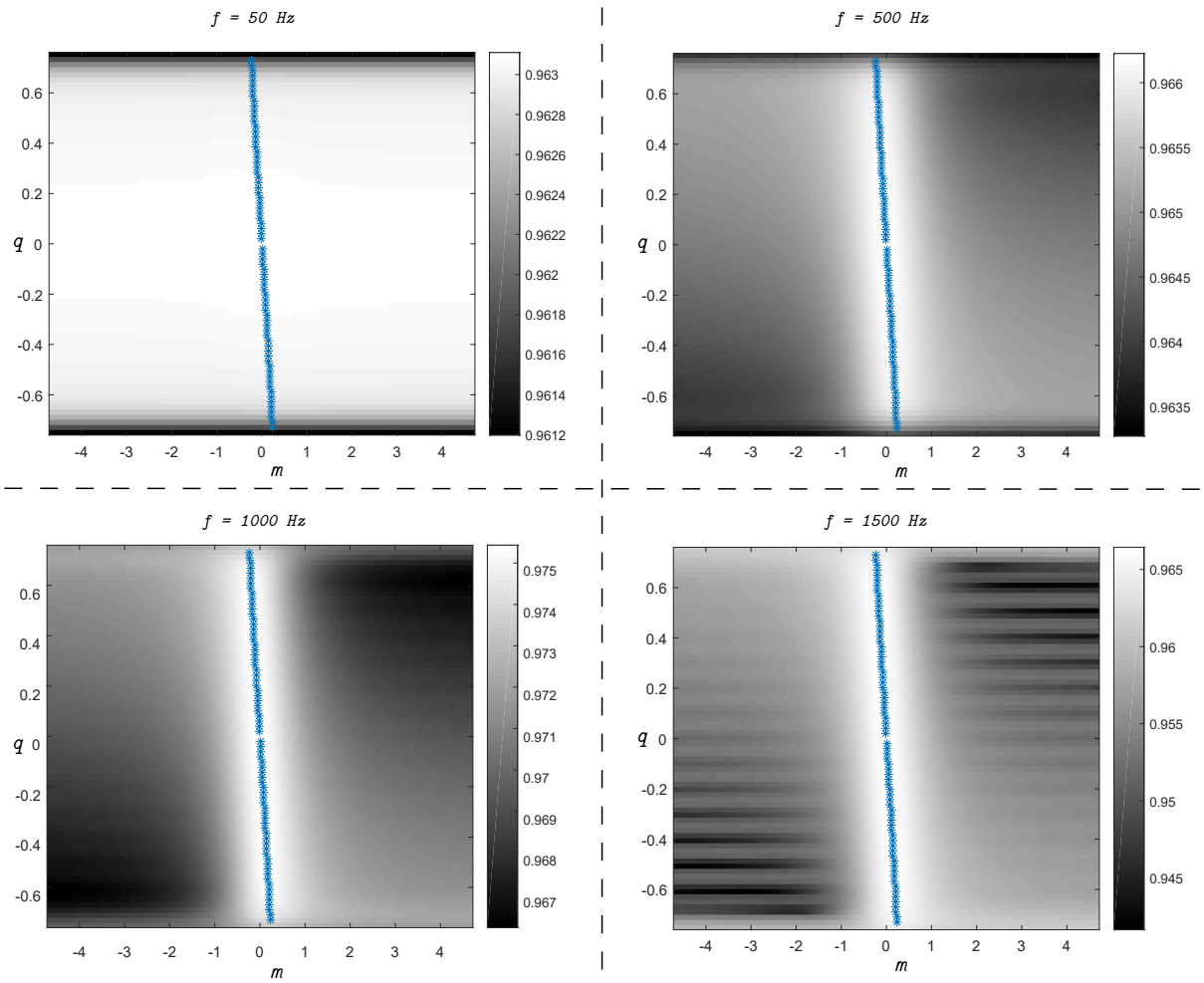


Figure 4.20: *Ray space for four frequencies, using a Gaussian pulse.*

Sine Wave

A sine wave with frequency of 1000 Hz and length is tested and can be seen in figure 4.21. Following settings are used:

1. Length of the signal, length(y) = 1600
2. Samples by frame, N = 160
3. Frames = 10
4. Frequencies analyzed = [50 513.3 1006.7 1500]
5. Sine wave frequency = 1000 Hz

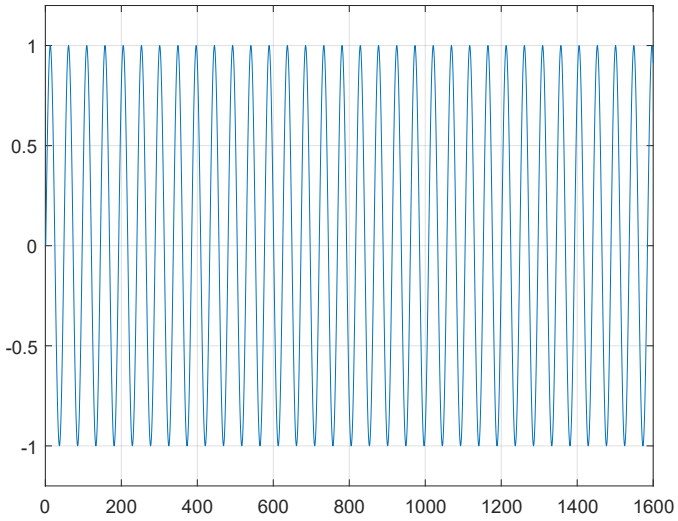


Figure 4.21: *Sine wave as input.*

The ray-space is presented for each frequency in figure 4.15. No combined ray space for these are shown since it is equivalent to that of Gaussian noise.

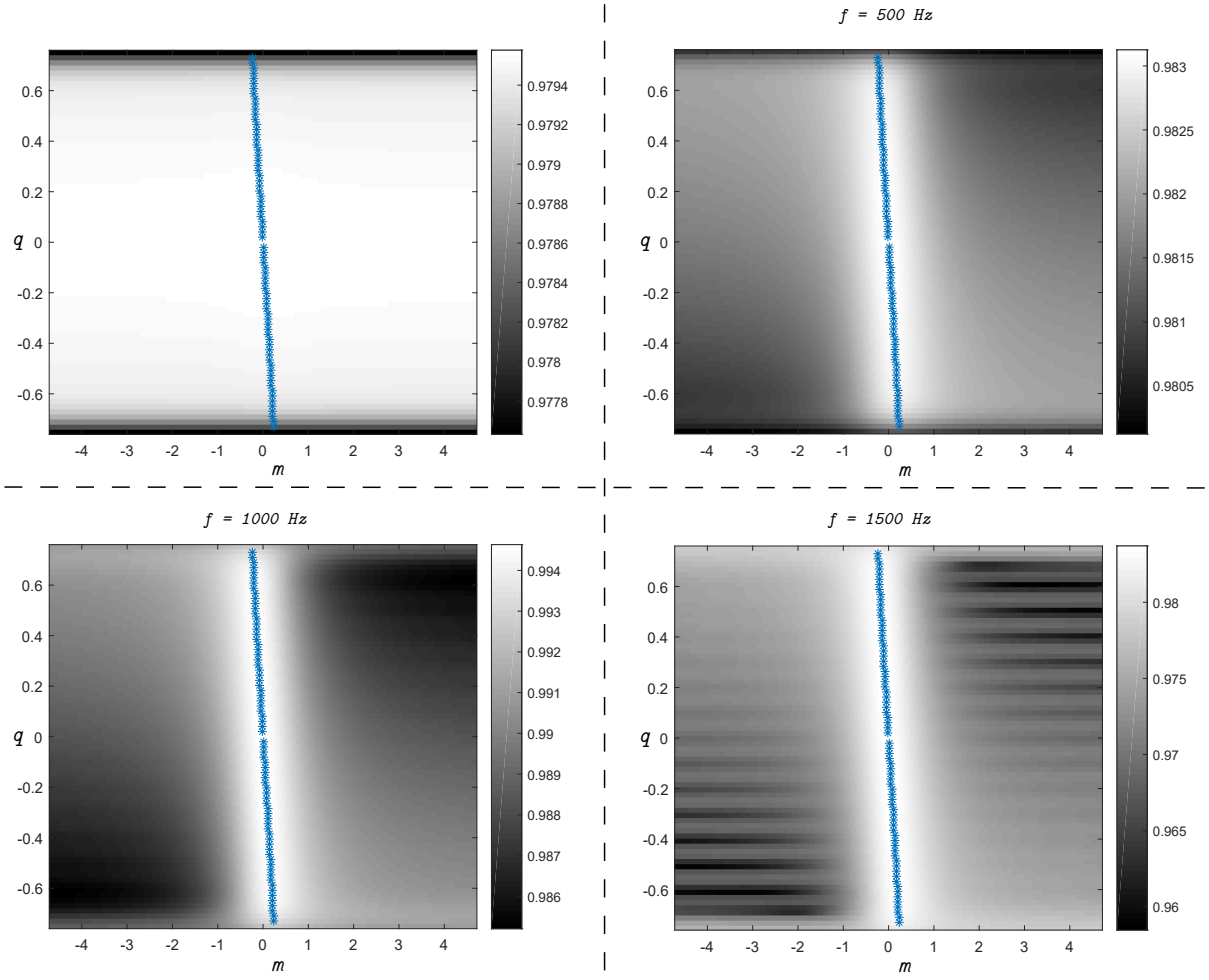


Figure 4.22: *Rays pace for four frequencies, using a 1000 Hz sine wave.*

SDM

A known limitation of SDM is that it needs an impulse as input for source localization[1], in this section we simulate and estimate using first and impulse and then Gaussian white noise to show that it does not work in the method. Source location in [3, 0].

Figure 4.23 shows a source image representation for the dirac impulsive function while figure 4.24 shows the same for the Gaussian white noise. As can be seen on the figures the estimation of the source is correctly observed for the impulse a localization error of ϵ= 0.039 m and an angle error of ϕ= 0.734 °. For the Gaussian noise, the method does not estimate correctly at all. At some points of estimation it does find the direction of the source, but not reliably.

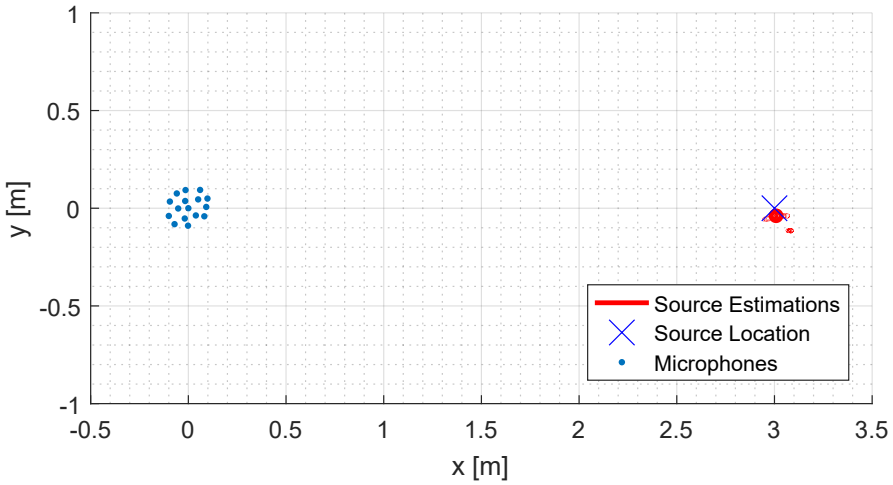


Figure 4.23: *DOA and energy of values up to 1/1000 of the maximum energy, dirac impulse.*

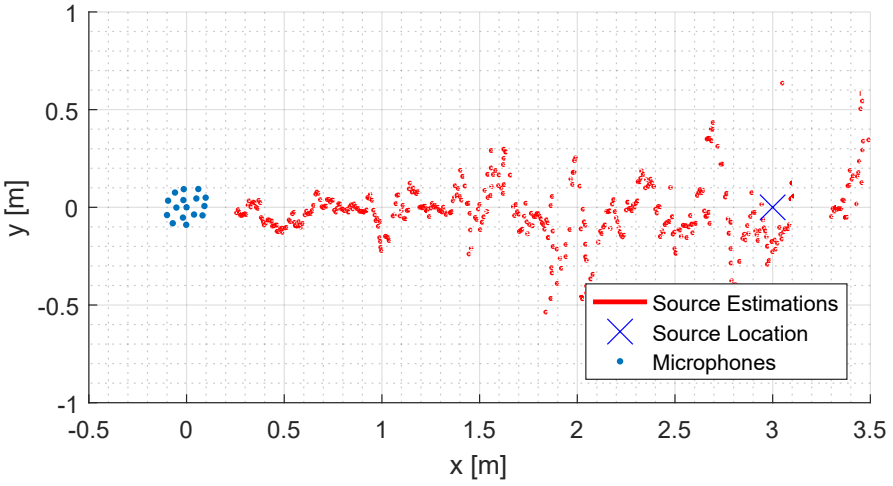


Figure 4.24: *DOA and energy of values up to 1/1000 of the maximum energy, Gaussian noise.*

4.9 Number of Microphones

Simulations using different amounts of microphones are carried out in this section to determine how many receivers are necessary for the methods to perform adequately in the configurations we use in the report. This has significance since compact solutions for sound field analysis are preferred.

RSTM

As was introduced earlier, the geometry of the microphone array layout used in the RSTM is linear and spacing between microphones is equidistant. In this context its length is defined by the number of microphones, L, and the distance between each of them, d. Scenarios with different number of microphones, L, using the same spacing, d, are presented. Authors in [2][3] use configurations L = 16, d = 0.1 and L = 16, d = 0.06, and so we use this as our reference point, and move from there. The following configurations are studied:

- L = 4, d = 0.1
- L = 6, d = 0.1
- L = 8, d = 0.1
- L = 10, d = 0.1
- L = 12, d = 0.1
- L = 14, d = 0.1
- L = 16, d = 0.1
- L = 18, d = 0.1
- L = 20, d = 0.1

Table 4.7 shows the results of the source estimations.

L	$r_{estimated}$	Distance error, ϵ [m]	Angle error, ϕ [°]
4	[5.282, -0.034]	2.282	0.374°
6	[3.757, 0.019]	0.758	0.283°
8	[3.507, -0.008]	0.507	0.129°
10	[3.221, -0.010]	0.221	0.180°
12	[3.070, 0.004]	0.071	0.069°
14	[3.140, 0.019]	0.141	0.341°
16	[3.112, -0.004]	0.112	0.075°
18	[3.066, -0.016]	0.068	0.301°
20	[2.992, -0.004]	0.009	0.082°

Table 4.7: *RSTM. Number of microphones versus estimated values.*

The result improves for a while when increasing the number of microphones, but when we have 12 or more in the array, the estimation finds the location within 15 cm, indicating that at least for this problem 12 is enough to accurately estimate source location. The variation seen across estimation with $L = 12, 14, 16, 18, 20$ can be explained by inherent noise in estimation using the method, as figure 4.7 in section 4.5 shows. In general it can be said that increasing the number of microphones will increase performance, but it is a trade-off since it increases algorithm complexity, number of microphones needed, and array size if d is kept constant. The ray space for L = 4, 12, 16, and 20 are seen in figure 4.25

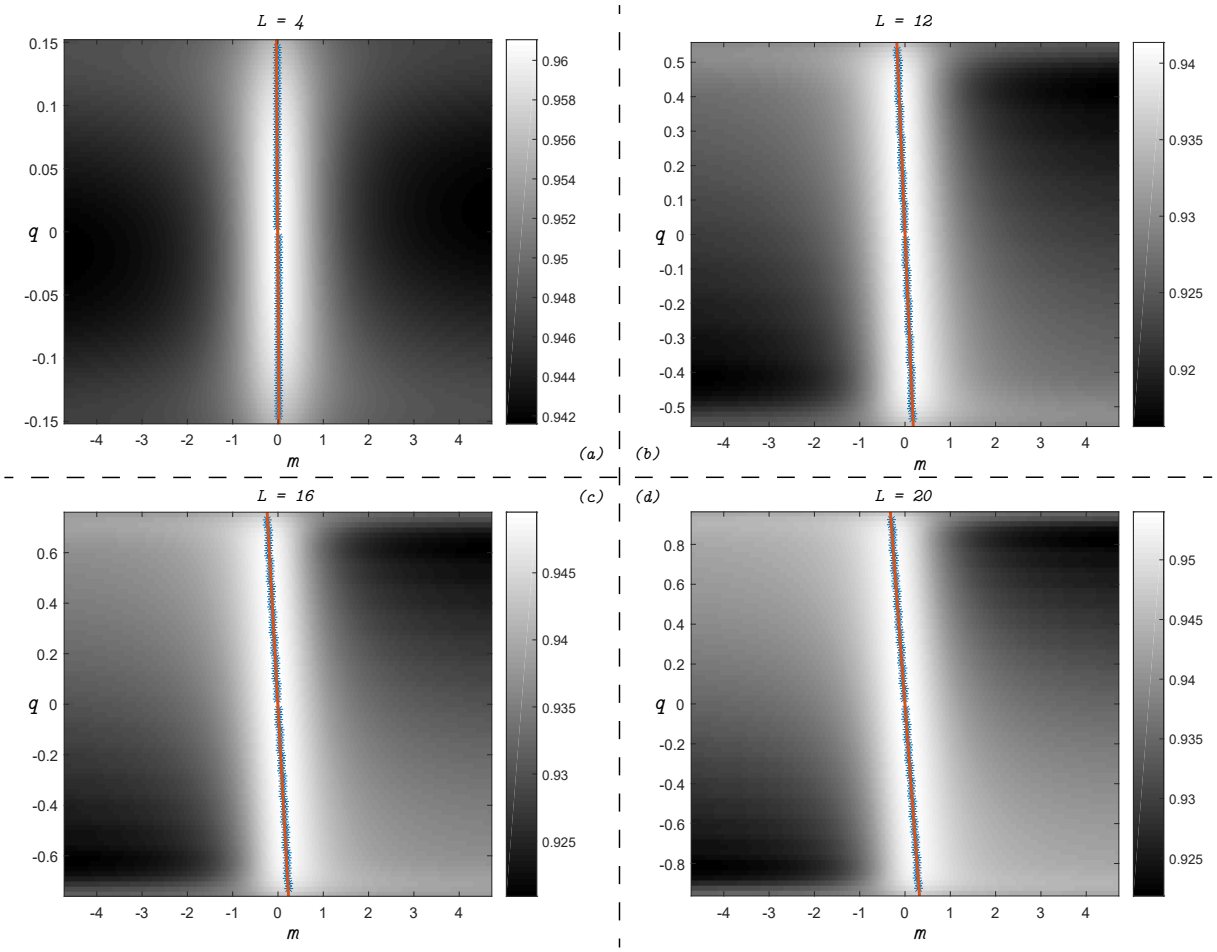


Figure 4.25: *(a) Ray space, L = 4. (b) Ray space, L = 12. (c) Ray space, L = 16. (d) Ray space, L = 20.*

SDM

The configuration of the SDM microphone array is random locations within -1 and 1 on both the x and y axis, with a minimum distance of 0.04 m between each microphone. While fewer microphones are necessary in estimation with SDM[1], we use the same numbers of microphones as for RSTM, and estimate source locations similarly. Table 4.8 shows the results of the source estimations.

L	$r_{estimated}$	Distance error, ϵ [m]	Angle error, ϕ [°]
4	[3.004, 0.155]	0.155	2.945°
6	[3.008, 0.026]	0.028	0.503°
8	[3.008, -0.036]	0.037	0.678°
10	[3.008, -0.052]	0.052	0.982°
12	[3.008, -0.048]	0.048	0.910°
14	[3.008, -0.039]	0.040	0.747°
16	[3.008, -0.039]	0.039	0.734°
18	[3.008, -0.041]	0.041	0.774°
20	[3.008, -0.030]	0.031	0.572°

Table 4.8: *SDM. Number of microphones versus estimated values.*

The SDM perform well even only with four microphones, which is expected since one of the major strengths of SDM is the ability to estimate with few receivers[1]. Figure 4.26 shows the SDM estimations for L = 4, 12, 16, and 20. The red points show where the SDM estimates the highest points of energy.

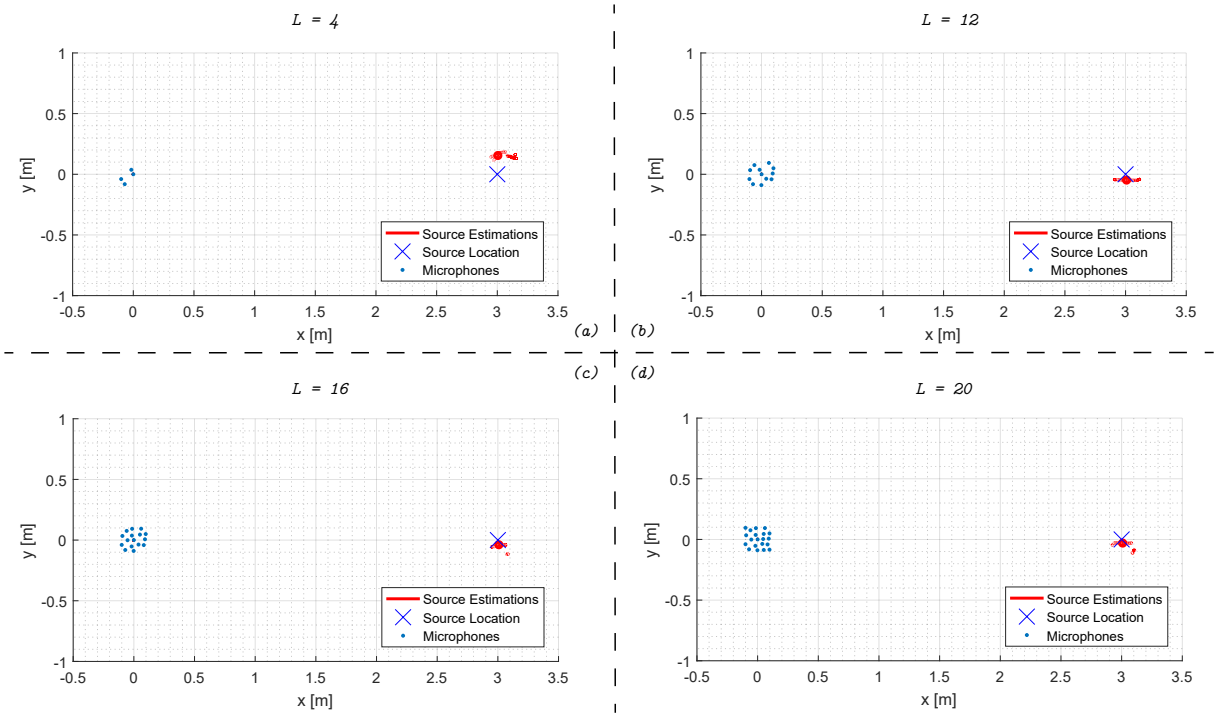


Figure 4.26: *(a) SDM estimation, L = 4. (b) SDM estimation, L = 12. (c) SDM estimation, L = 16. (d) SDM estimation, L = 20.*

4.10 Estimation in Reverberant Conditions

The purpose of this section is to observe the performance of both methods in the estimation of the position of a source in a reverberant rectangular room. The RIR of the room is calculated using the implementation made by Habets in [16] of the image method developed in [15]. For the RSTM once that the RIR time signal for each position of the microphone array are calculated, they are convolved to an anechoic white noise signal as explained in section 2.7. In the SDM case, the RIR signals are used directly as input of the algorithm.

Scenario

The array and source positions can be observed in figure 4.27 The simulated room have the next characteristics:

- Dimensions.- The dimensions are defined the same as the standard listening room in Aalborg University. Which are 8x4x3 LxWxH [m]
- Beta.- The reflection coefficient β needs to be specified for each of the walls. In these simulations the same coefficient is specified for all the walls excepting the wall in the back of the arrays, which β zero. A total number of three reflection coefficients are considered [0 0.65 0.80]. Figure 4.29 shows an example of two time signals generated with different

reflection coefficients. Specifying a β of magnitude zero, generate anechoic transfer functions and is presented a verification testcase.

- Source.- A single source in the room at 3 different positions is considered
- Reflection_Order.- Only reflections up to fourth order are considered in the simulations.

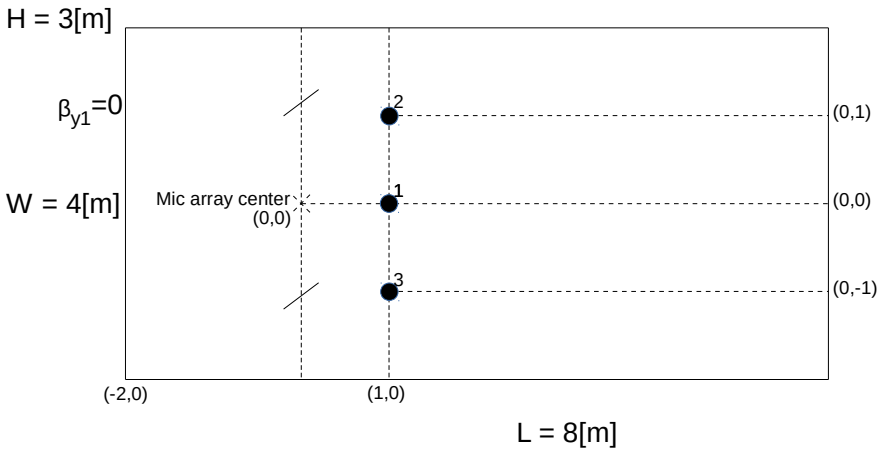


Figure 4.27: *Scenario considered for the reverberant simulations*

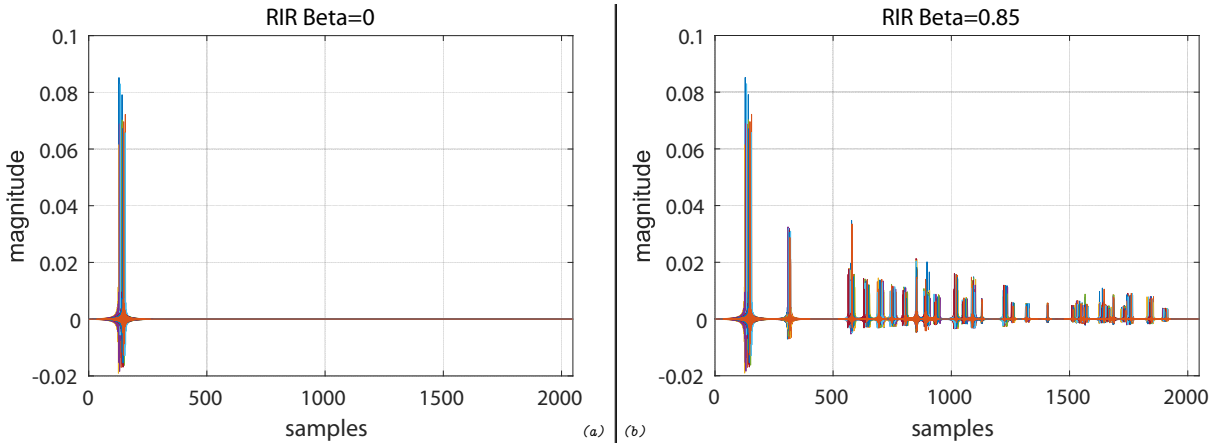


Figure 4.28: *(a): $\beta = 0$, (b): $\beta = 0.85$ for position 1*

Results

Table 4.9 show the results for SDM, while table 4.10 show the results for RSTM

Position	β					
	0		0.65		0.80	
	ϵ[m]	ϕ[°]	ϵ[m]	ϕ[°]	ϵ[m]	ϕ[°]
[1 0]	0.0082	0.1768	0.101	0.2575	0.101	0.2587
[1 1]	0.0082	0.1768	0.101	0.2575	0.101	0.2587
[1 -1]	0.0082	0.1768	0.101	0.2575	0.101	0.2587

Table 4.9: *SDM distance and angle error for the considered positions and reflection coefficients*

Position	β					
	0		0.65		0.80	
	ϵ[m]	ϕ[°]	ϵ[m]	ϕ[°]	ϵ[m]	ϕ[°]
[1 0]	0.0081	0.1064	0.0439	0.4175	0.0439	0.4175
[1 1]	0.0277	0.6134	0.0292	1.0790	0.2183	3.5020
[1 -1]	0.0756	0.3347	0.3063	4.0321	0.2993	4.5518

Table 4.10: *RSTM distance and angle error for the considered positions and reflection coefficients*

Figure 4.29 shows how the ray spaces are distorted due to contributions of the present reflections of the near wall to the source.

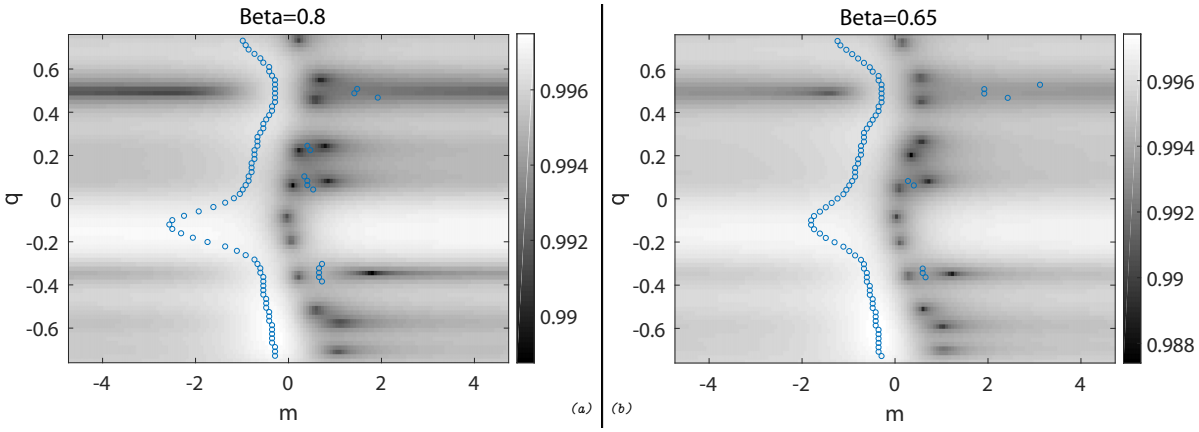


Figure 4.29: *(a):* $\beta = 0.8$ *position [1 -1], (b):* $\beta = 0.65$ *position [1 -1]*

Observations

SDM shows a very low estimation error for all the positions and reflection coefficients specified. In the other hand RSTM shows low estimation error for positions located at angle 0 degrees but present errors up to 30[cm] and 4.5 degrees when reflections are present. Figure 4.2 shows the two worst cases analyzed.

4.11 Summary

Here we summarize the observations made on each comparison parameter.

- **[Source Distance]**: In varying distance on the sound source, the RSTM slightly outperforms the SDM by being able to see sources very close to the array, down to atleast 0.1 m, while the SDM is able to see sources from around 0.5 and further away. Both methods start failing beyond 7 m.

- **[Incidence Angle]**: With locked source distance and varying signal incidence angle, the SDM clearly outperforms RSTM, since it is not bounded in this regard and can estimate sources from any direction. The RSTM here is limited by the fact that it is only able to see sources in positive x coordinates, and only within a chosen beamforming angle range relative to the center of the linear microphone array, θ. This limitation is a product of the linear geometry of the array, adapting the method for other geometric layouts could remove the limitation.

- **[Additive Noise]**: Adding different levels of noise to the microphones gives insight into how well the methods can find linear patterns, when significant noise is present. The SDM method performs slightly better than the RSTM when input signal to noise ratio(iSNR) is -5 dB or higher. When nearing -10 dB iSNR, the RSTM fails and no longer completes valid estimations. The SDM persists a bit longer, and becomes invalid when the iSNR reaches -15 dB.

- **[Multiple Sources]**: Both methods perform well when two sources are located, but the RSTM returns erroneous results when the sources are placed too close to each other in the real world. This causes the ray space to be obscured, since the rays corresponding to each source interfere with each other.

- **[Microphone Position Error]**: Introduction uniform noise in levels from $U \sim [-1mm, 1mm]$ to $U \sim [-10mm, 10mm]$ and averaging Monte Carlo simulations of the levels, neither method shows sensitivity to changes in microphone positions. Very slight increase in standard deviation is seen towards the higher levels of noise, but not enough to be deemed significant.

- **[Microphone Phase Mismatch]**: Both methods perform well when one or two microphone are phase mismatched. The SDM experiences slight error in angle estimation. The delay shows itself very clearly in the ray space, and interferes the linear pattern at the location of the delayed microphone. If many microphones are mismatched, the error in the RSTM increases due to the interference along more of the linear pattern.

- **[Input Signal Type]**: Signal type is one of the major parameters in which the RSTM excels. The SDM is limited to the use of impulse signals in sound field analysis, while RSTM is able to estimate regardless of the type of signal provided. The fact that it is able to use content from many different frequencies also enables it to use very frequency specific content as input signal.

- **[Number of Microphones]**: Increasing the number of microphones increases the performance of either algorithm, but only to a certain point. In estimating a single source in

noiseless conditions, with spacing constant at d = 0.1, the RSTM needs atleast 12 microphones to perform well. The SDM needs only four to perform decently($\epsilon = 0.155$ m). In more complex problems, both methods will perform better with more microphones, and in general SDM performs well with much fewer microphones than the RSTM.

- **[Estimation in Reverberant Conditions]**: SDM estimations for the position and reflection coefficients present very low estimation error. RSTM presents estimation error up to 30[cm] with the default parameter settings for positions close to a wall. It can be observed from the images produced that the linear pattern is more or less distorted depending on the reflection coefficient.

4.12 Conclusions

Generally the methods perform similarly over the compared parameters. Significant observations to make between the two methods are: The SDM performs well even with very few microphones, we find that it performs well down to four microphones using the random geometry. Using a different type of geometry might be better with a small number of microphones and could make the required number even lower. On the contrary, RSTM, with the linear configuration used here, needs atleast 12 microphones to accurately estimate at a distance of 3 meters. Other configurations of the array will likely make it perform differently, but none other than the linear array configurations are studied here, and have not been studied in-depth in previous studies on the method, to the best of the authors' knowledge. Another significant observation is that the ray space can work with any type of acoustic signal from the source. The SDM requires an impulse. The RSTM is also in this configuration limited to only estimating sources in the positive x-plane, also limited by the beamforming range θ. The SDM can estimate sources from any direction. The SDM also performs better on detection of multiple sources, where the RSTM in about a third of the cases can not find the sources. The SDM finds the sources for every Monte Carlo simulation. In the two thirds that the RSTM does estimate, the methods perform similarly in distance -and angle error.

5

RSTM Verification

A series of tests done using real microphones capturing real data is done to test the performance of the RSTM on real data, and to verify the simulations presented in previous chapters.

5.1 Initial Test in Non-Anechoic Conditions

A test is performed using a linear array of eight microphones of the same model, captured on a single eight channel A/D converter, which is sampled on a sound card and saved on a PC. Figure 5.1 shows the set-up and table 5.1 provides the equipment used in the test.

Equipment List

AAU #	Description	Model
08718	Sine/Noise Generator	B&K Type 1049
75525, 75545 - 75551	Eight 1/2-Inch Microphones	G.R.A.S. 40AZ
75557, 75577 - 75583	Eight 1/4-Inch Pre-Amplifiers	G.R.A.S. 26cc
56543	Eight Channel A/D Converter	Behringer Ultragain Pro-8 Digital
56553	Sound Card	RME Audiolink 96 Multiset
77008	PC laptop with PCMCIA port	Fujitsu Simenes Lifebook E Series
02125-08	Speaker	FBT Jolly^3 A

Table 5.1: *Equipment used in the initial test.*

Set-Up

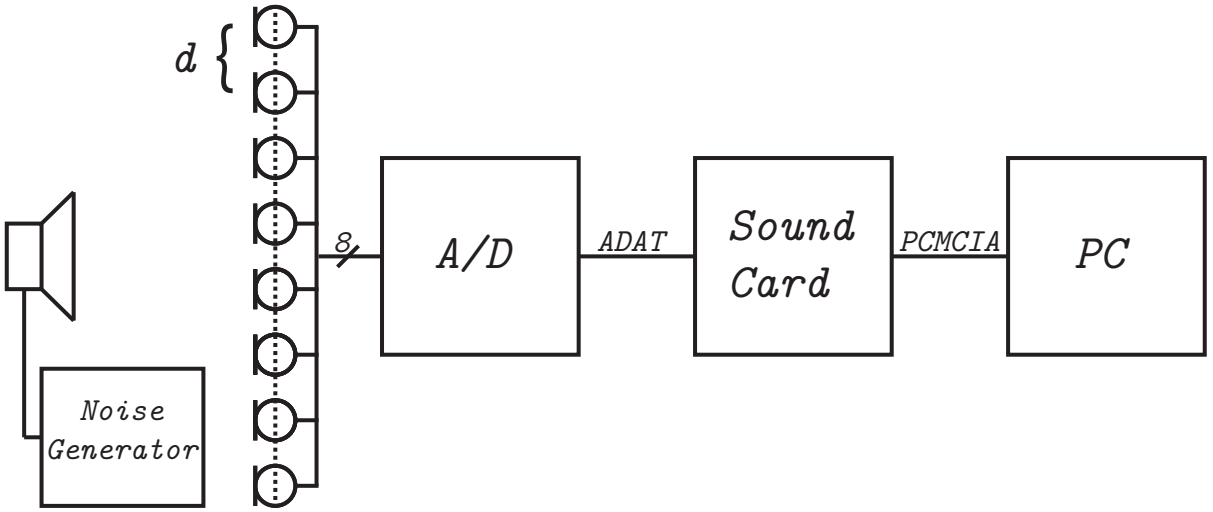


Figure 5.1: *Set-up for the initial test.*

The environment is not anechoic, but has reduced reflective properties with absorptive surfaces on the walls and carpeting on the floor. An ideal test would be completed with a 16 microphone array in anechoic conditions, but the resources to complete this is at the time of this test unavailable. It is reasoned that performing an initial test with fewer microphones in non-ideal conditions can bring some insight into the performance of the algorithm. Pictures in figure 5.2 shows pictures of the environment in which the measurements were captured, with the array in place and the other equipment removed.

Figure 5.2: *Measurement environment.*

The speaker is set in position [3, 0], reference to [0, 0] which is the center of the microphone array. Co-ordinates are represented as meters in the real world. White noise is output on the speaker via the Noise Generator. Comparing the level of the recording to the background noise, we obtain a signal-to-noise ratio of 33.99 dB.
To use direct signals instead of simulated signals through the signal model, we simply apply an FFT to the signals and retrieve the values of the frequencies we are using. With a sampling rate of $fs = 48$ kHz, we look at a five frequencies, f = $[500, 801, 1105, 1409, 1713]$, averaging 50 frames each of 192 samples. The remaining of settings used in the algorithm are as we use in comparison simulations in section 4.1.1, except changing σ to 0.2, and L to 8 microphones.
Using this on the recorded signal we achieve the ray space seen in figure 5.3.

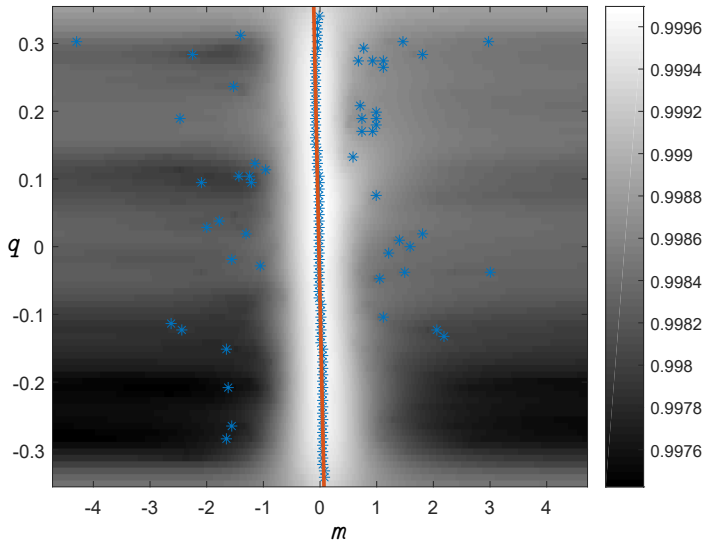


Figure 5.3: *Ray space of measured data, source in [3, 0]. Estimation yields [3.901, -0.080].*

The source is estimated in [3.901, -0.080], which produces a distance error $\epsilon = 0.905$ m, and an angle error of $\phi = 1.175°$. The directional estimation is correct, but some error is seen in distance estimation.

A series of tests is carried out to see whether the angle consistently will be estimated correctly. Same settings as for the previous test is used, but estimating single sources sources placed in [2, 0.8], [2, -0.8], [2, 1.5], and [2, -1.5]. The resulting ray spaces are seen in figure 5.4.

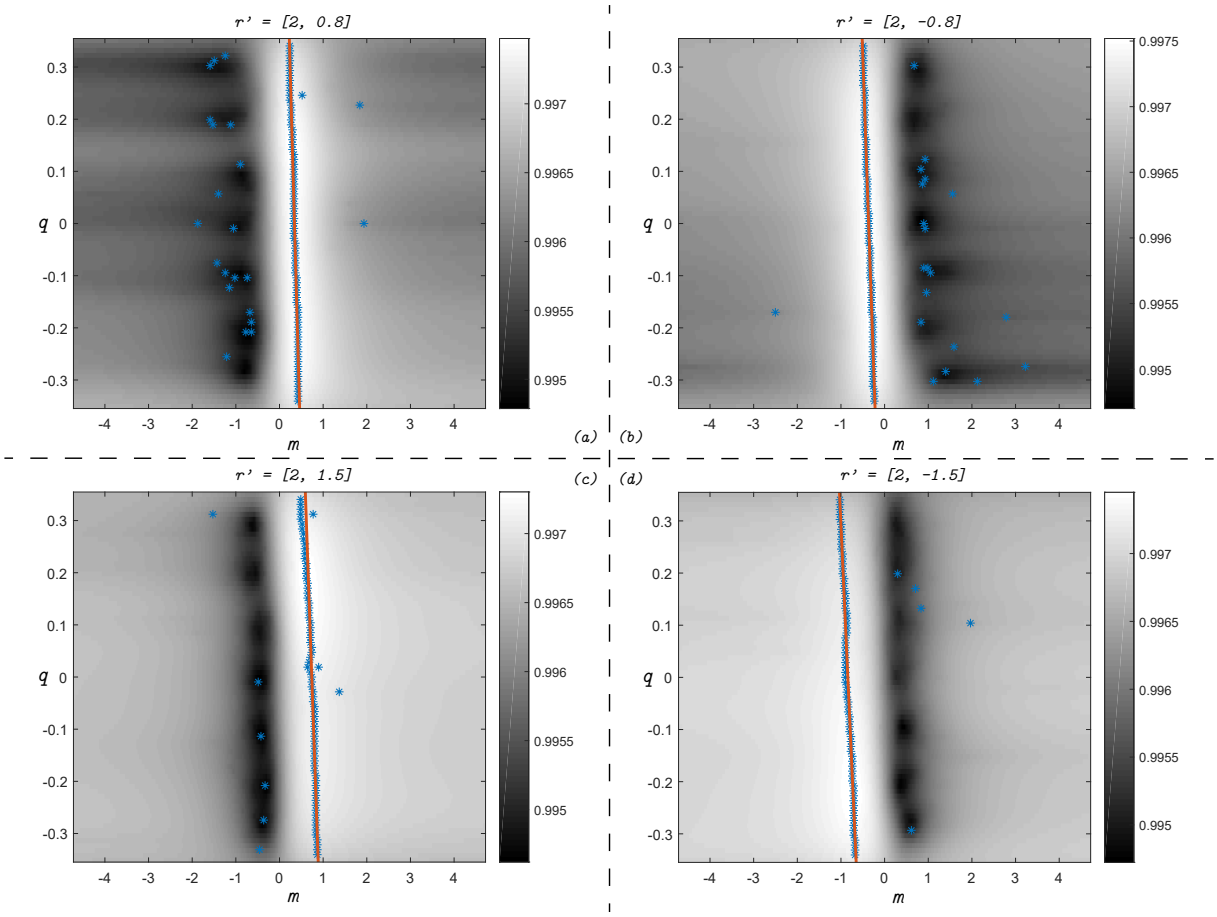


Figure 5.4: *Ray spaces of measured data. (a) source in [2, 0.8], estimation finds [3.006, 1.031]. (b) source in [2, -0.8], estimation finds [2.405, -0.880]. (c) source in [2, 1.5], estimation finds [2.405, 1.778]. (d) source in [2, -1.5], estimation finds [1.854, -1.562].*

The resulting estimations find:

- **(a)**, $r' =$ **[2, 0.8]**: Estimated location in [3.006, 1.031], $\epsilon = 1.032$, $\phi = 2.870°$
- **(b)**, $r' =$ **[2, -0.8]**: Estimated location in [2.405, -0.880], $\epsilon = 0.413$, $\phi = 1.704°$
- **(c)**, $r' =$ **[2, 1.5]**: Estimated location in [2.405, 1.778], $\epsilon = 0.491$, $\phi = 0.395°$
- **(d)**, $r' =$ **[2, -1.5]**: Estimated location in [1.854, -1.562], $\epsilon = 0.159$, $\phi = 3.244°$

Angle is estimated correctly for all cases, with a maximum error of just over 3°. The error in distance is as would be expected for a set-up using only eight microphones. This expectation stems from section 4.9, wherein we find a distance error of $\epsilon = 0.507$ meter using eight microphones, estimating in ideal conditions.

5.1.1 Observations

The test serves well as verification for discoveries made earlier in simulations. Section 4.9 states that at least 12 microphones are needed for correct estimations at a 3 meter distance, so some error on distance is to be expected using an array with only eight. The estimations in the non-anechoic room were expected to perform worse than was the case, since many other factors come into play when outside ideal simulations. Reflections come into play in the estimations, although you would expect this to offset the angle estimation, which is not the case. Background noise is

unlikely to affect the result since we found a 33.99 dB signal to noise ratio in the room. Although this shows that the method performs corresponding to our simulations, a separate test using a 16-microphone array is needed to further verify the simulations, to give a comparison to the simulated data where the parameters are as equal as possible.

5.2 Anechoic Chamber Tests

The microphone array is expanded to the full size, using 16 microphones in a linear array. Tests are performed in an anechoic chamber to get a comparison as close to the simulations as possible. The signal is sampled on 16 G.R.A.S. 40AZ microphones with G.R.A.S. 26cc pre-amps. These are sampled on two eight channel A/D converters, which are sent to a pc through a sound card on USB. See table 5.2.

Equipment List

AAU #	Description	Model
08718	Sine/Noise Generator	B&K Type 1049
75525, 75530, 75533 - 75537, 75540, 75542, 75545 - 75551	Eight 1/2-Inch Microphones	G.R.A.S. 40AZ
75557, 75562, 75565 - 75569, 75572, 75574, 75577 - 75583	Eight 1/4-Inch Pre-Amplifiers	G.R.A.S. 26cc
56543	Eight Channel A/D Converter	Behringer Ultragain Pro-8 Digital
86838	Sound Card	RME Fireface
N/A	PC laptop	Lenovo ThinkPad E530c
02125-08	Speaker	FBT Jolly^3 A

Table 5.2: *Equipment used in the initial test.*

Set-Up

The set-up used is seen in figure 5.13. Each speaker and its position represents a set of coordinates in which a test is performed in the anechoic chamber.

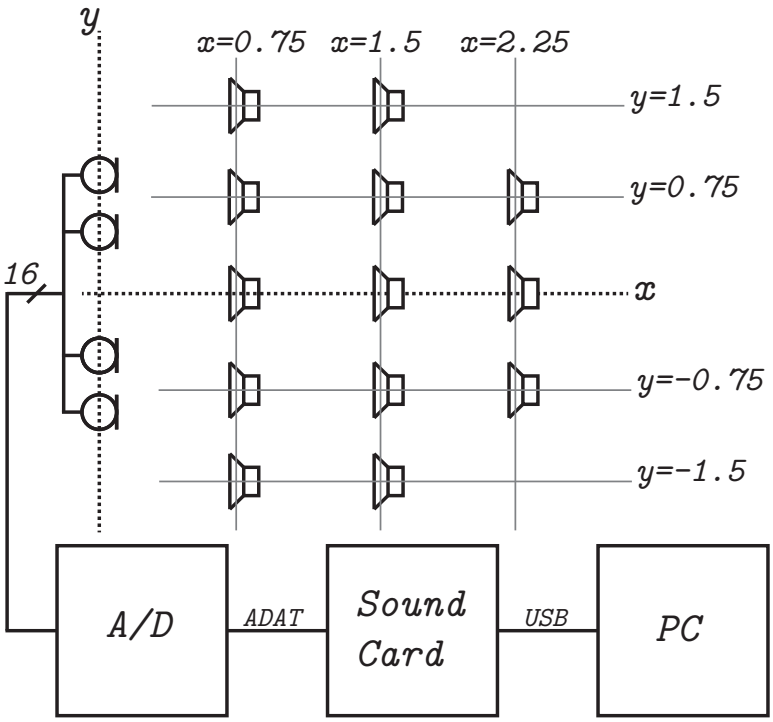


Figure 5.5: *Set-up for the anechoic chamber tests.*

Pictures of the anechoic chamber with the array set in place in figure 5.6.

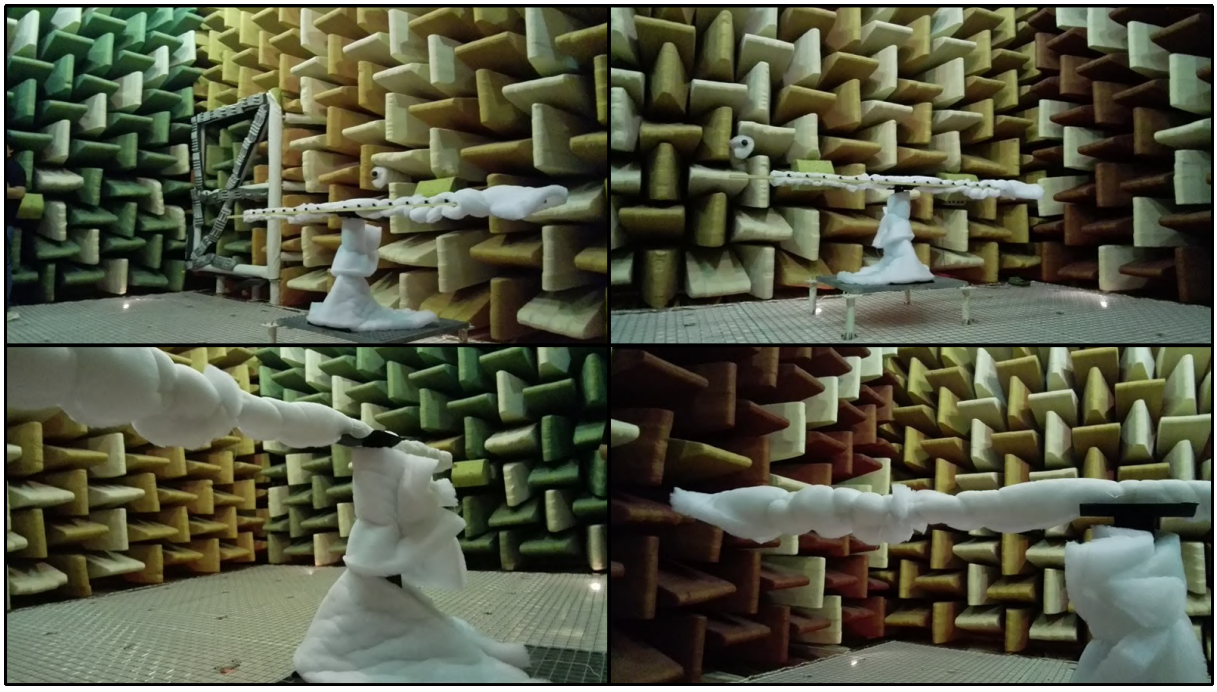


Figure 5.6: *Anechoic chamber environment.*

Individual estimations of source positions are carried out with the speaker at each of the

designated positions seen in figure 5.13. Parameters used are identical to those of section 5.1, with the exception of using L = 16 microphones. Figure 5.14 shows the ray spaces and the estimated positions for the first 'line' of speakers, positioned along the line at $x = 0.75$.

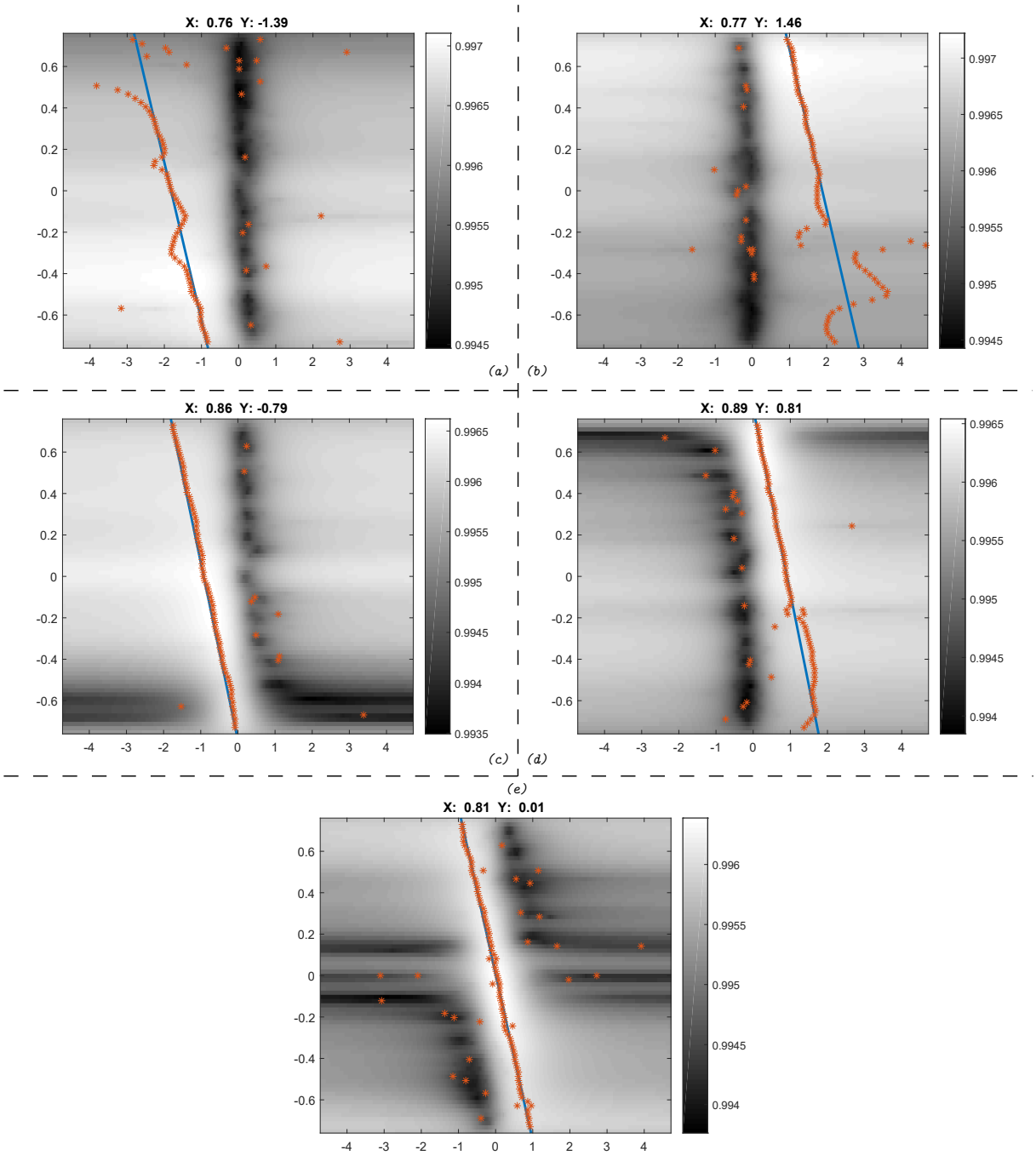


Figure 5.7: *Estimation of speaker positions. (a) r' = [0.75, -1.5], estimation finds [0.76, -1.39]. (b) r' = [0.75, 1.5], estimation finds [0.77, 1.46]. (c) r' = [0.75, -0.75], estimation finds [0.86, -0.79]. (d) r' = [0.75, 0.75], estimation finds [0.89, 0.81]. (e) r' = [0.75, 0], estimation finds [0.81, 0.01].*

The estimations find:

- **(a), $r' =$ [0.75, -1.5]**: Estimated location in [0.76, -1.39], $\epsilon = 0.111$, $\phi = 2.103°$

- **(b)**, $r' =$ **[0.75, 1.5]**: Estimated location in [0.77, 1.46], $\epsilon = 0.045$, $\phi = 1.242°$
- **(c)**, $r' =$ **[0.75, -0.75]**: Estimated location in [0.86, -0.79], $\epsilon = 0.117$, $\phi = 2.429°$
- **(d)**, $r' =$ **[0.75, 0.75]**: Estimated location in [0.89, 0.81], $\epsilon = 0.152$, $\phi = 3.244°$
- **(e)**, $r' =$ **[0.75, 0]**: Estimated location in [0.81, 0.01], $\epsilon = 0.061$, $\phi = 0.707°$

The same procedure is done for speaker positions along the line at $x = 1.5$. Figure 5.8 shows the ray spaces.

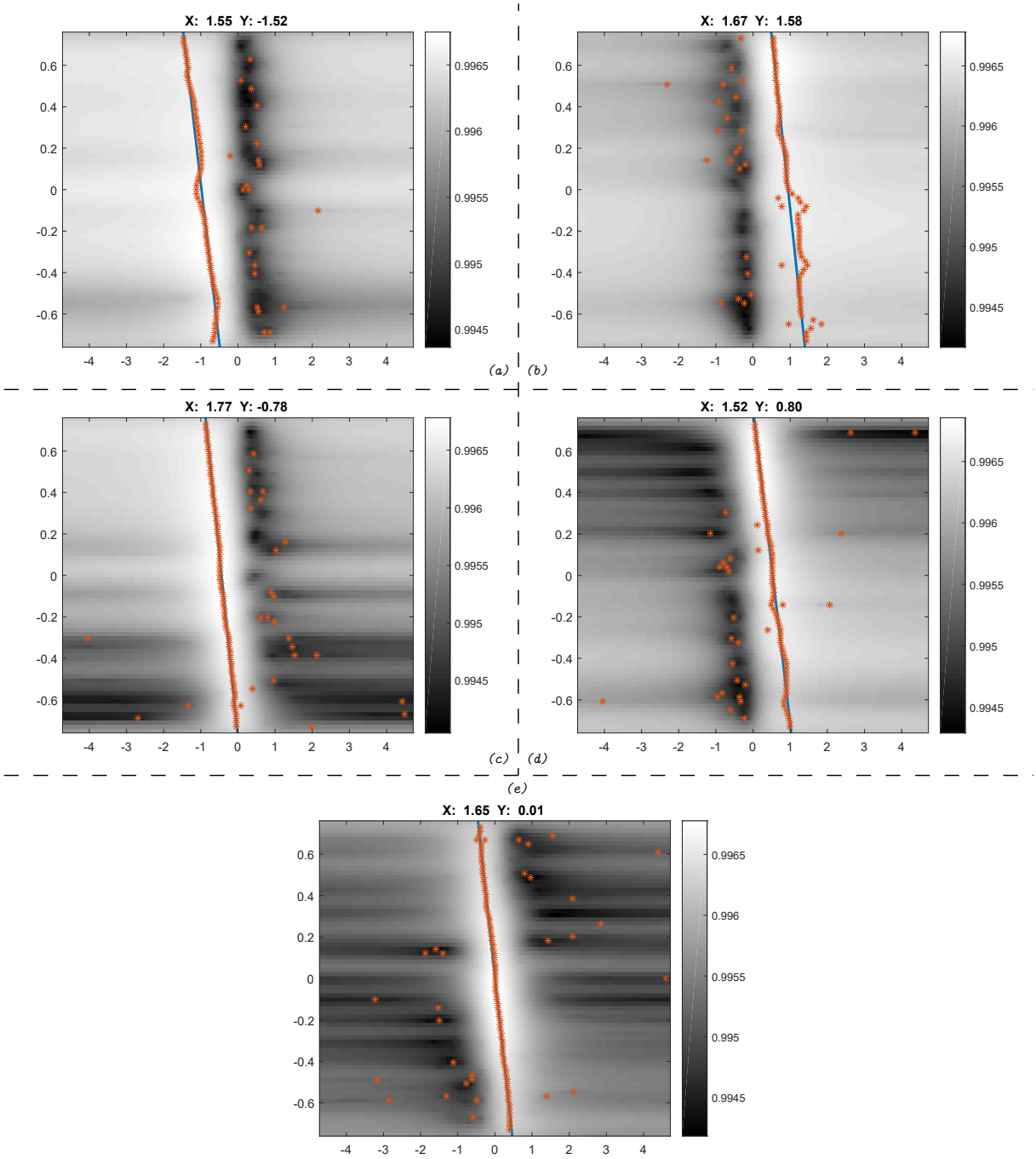


Figure 5.8: *Estimation of speaker positions. (a) r' = [1.5, -1.5], estimation finds [1.55, -1.52]. (b) r' = [1.5, 1.5], estimation finds [1.67, 1.58]. (c) r' = [1.5, -0.75], estimation finds [1.77, -0.78]. (d) r' = [1.5, 0.75], estimation finds [1.52, 0.80]. (e) r' = [1.5, 0], estimation finds [1.65, 0.01].*

The estimations find:

- **(a)**, $r' =$ **[1.5, -1.5]**: Estimated location in [1.55, -1.52], $\epsilon = 0.054$, $\phi = 0.560°$
- **(b)**, $r' =$ **[1.5, 1.5]**: Estimated location in [1.67, 1.58], $\epsilon = 0.188$, $\phi = 1.586°$
- **(c)**, $r' =$ **[1.5, -0.75]**: Estimated location in [1.77, -0.78], $\epsilon = 0.272$, $\phi = 2.783°$
- **(d)**, $r' =$ **[1.5, 0.75]**: Estimated location in [1.52, 0.80], $\epsilon = 0.054$, $\phi = 1.194°$
- **(e)**, $r' =$ **[1.5, 0]**: Estimated location in [1.65, 0.01], $\epsilon = 0.150$, $\phi = 0.347°$

The final line in $x = 2.25$, figure 5.9 shows the ray spaces.

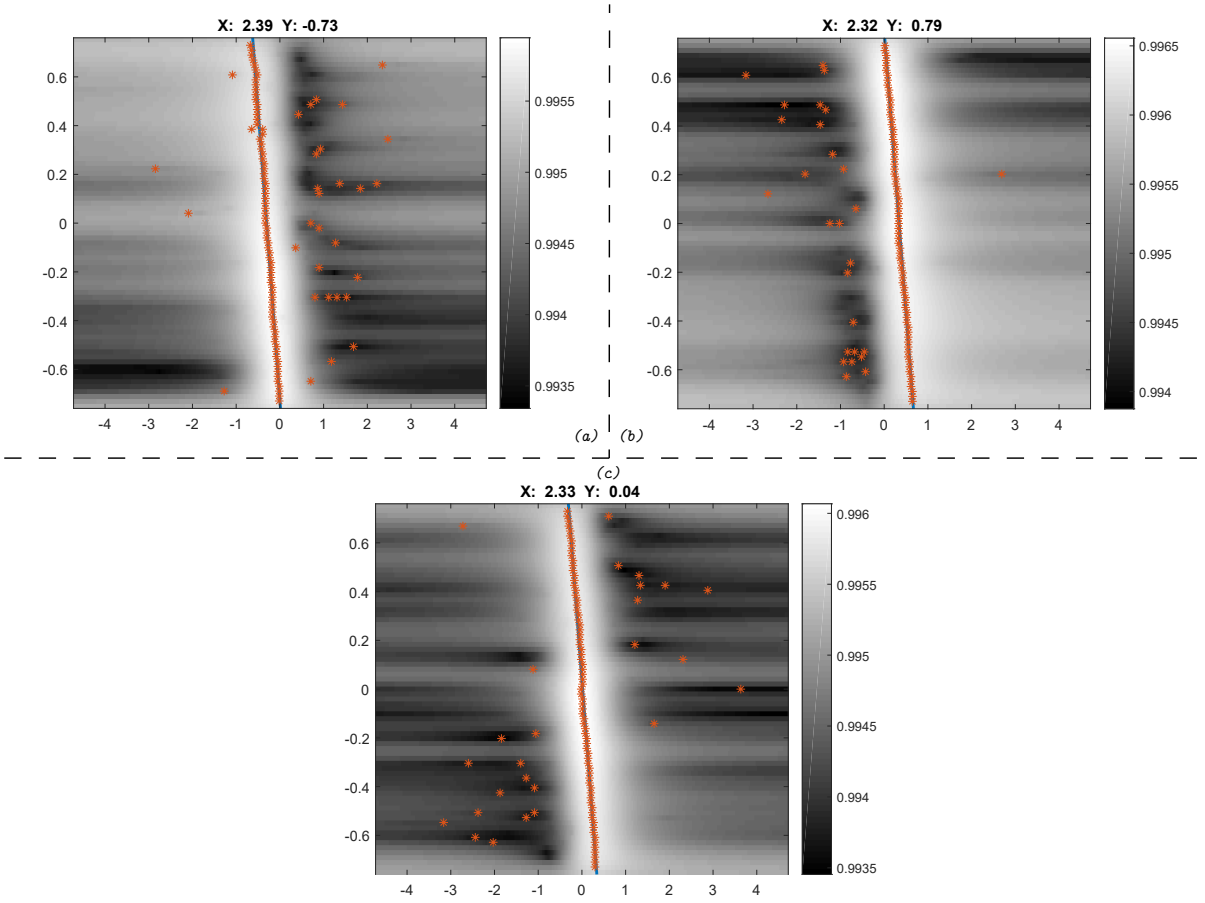


Figure 5.9: *Estimation of speaker positions. (a) r' = [2.25, -.75], estimation finds [2.39, -0.73]. (b) r' = [2.25, .75], estimation finds [2.32, 0.79]. (c) r' = [2.25, 0], estimation finds [2.33, 0.04].*

The estimations find:

- **(a)**, $r' =$ **[2.25, -.75]**: Estimated location in [2.39, -0.73], $\epsilon = 0.141$, $\phi = 1.45°$
- **(b)**, $r' =$ **[2.25, .75]**: Estimated location in [2.32, 0.79], $\epsilon = 0.081$, $\phi = 0.370°$
- **(c)**, $r' =$ **[2.25, 0]**: Estimated location in [2.33, 0.04], $\epsilon = 0.089$, $\phi = 0.984°$

The RSTM estimates every location within 20 centimeters, and within an angle of 3 degrees. The distance error varies among positions, but in general the greater angle from speaker to center of array, the larger the error becomes. This happens due to the beamforming range of 78°, which in some source locations make some of the outer microphones lose line of sight to the source. This can be seen in figure 5.14 (a) and (b), where the ends of the ray spaces are obscured due to loss of line of sight at microphones at the end of the array.

Multiple Sources

To compare with the simulated performance of estimating multiple sources with the RSTM, we place two speakers of the same model[1], in positions [2, 2] and [2, -2], and white noise is output on both of them. Same settings as for the previous test are used, with the exception that σ is increased to 0.5, as it was in simulations for multiple sources. Figure 5.10 shows the ray space produced.

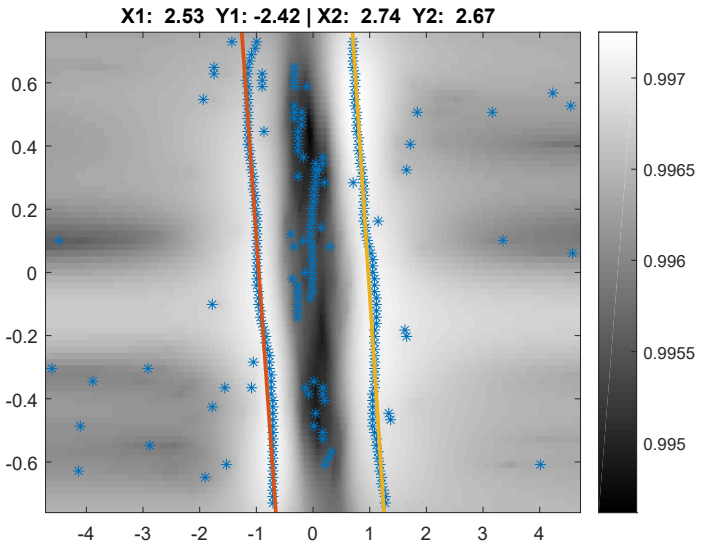


Figure 5.10: *Estimation of multiple sources. r_1' = [2, -2], r_2' = [2, 2], estimation finds [2.54, -2.42], [2.74, 2.67].*

This yields a distance error of $\epsilon = 0.676$ m for r_1', and $\epsilon = 0.998$ m for r_2. Angle error for r_1' is $\phi = 1.273$ ° and for r_2' it is $\phi = 0.741$.
The speakers are moved closer to each other, to r_1' = [2, -1] and r_2' = [2, 1], and the test is repeated. This yields the following ray space in figure 5.11.

[1] Two of the same model as seen in 5.2 are used.

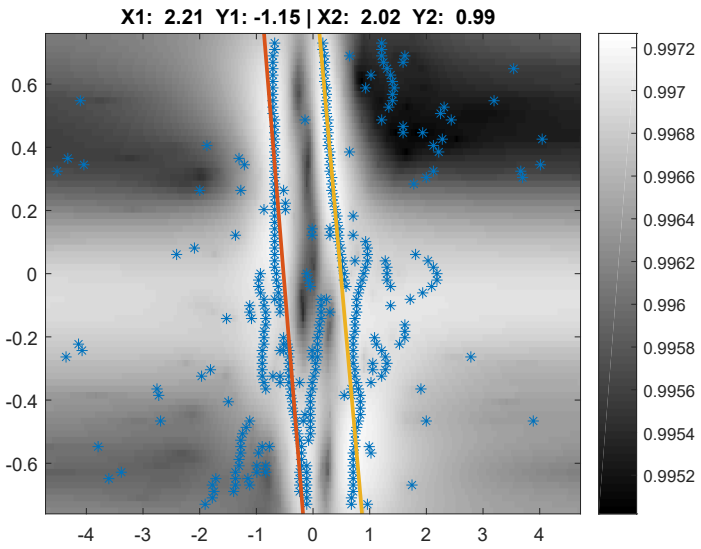


Figure 5.11: *Estimation of multiple sources. r_1' = [2, -1], r_2' = [2, 1], estimation finds [2.21, -1.15], [2.02, 0.99].*

Distance error of $\epsilon = 0.258$ m for r_1', and $\epsilon = 0.022$ m for r_2. Angle error for r_1' is ϕ= 0.926 ° and for r_2' it is $\phi = 0.456$ °.
While the estimation performs better with speakers closer to each other, it is also obvious that more noise is present in the ray space. This follows the expectation found in simulations, where if too close the rays would interfere with each other. Something else to be noted for the estimation in figure 5.11 is that initially the RANSAC did not accept linear patterns among the noise and the distance threshold had to be increased to make it accept the slightly curved lines as linear patterns. The fact that the estimations are better for the closer speakers is likely either by coincidence or due to them being closer to the middle of the microphone array so that each beamformer finds both speakers.

5.3 Listening Room Test

Another test is carried out in a the listening room located in B4-107 at Aalborg University. This room conforms to the IEC-268-13 standard, which defines the 'average living room' with a reverberation time of approximately 0.4 seconds. The room is 7.80 meters long, which allows for tests on a larger distance than in the anechoic chamber. Figure 5.12 shows pictures of the room and set-up.

Figure 5.12: *The listening room.*

The test serves to see how well the RSTM performs in normal, non-anechoic conditions, and to see at how far distances the method still works properly. The same equipment is used as in the anechoic chamber, seen in table 5.2.

Set-Up

The set-up is similar to that of the anechoic chamber as well, although we here only look at different points in distance, moving on the x axis. The five points measured is seen in figure REF, represented by the speaker symbols.

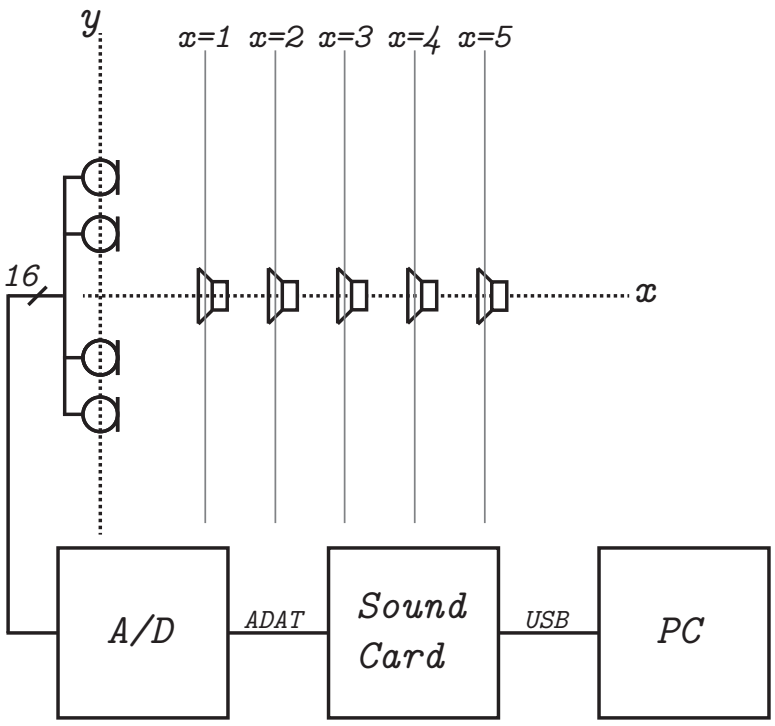


Figure 5.13: *Set-up for the listening room tests.*

The source position is estimated for each of the five positions, using the same settings for parameters for single sources in the anechoic chamber. Figure REF shows the ray spaces of each speaker position.

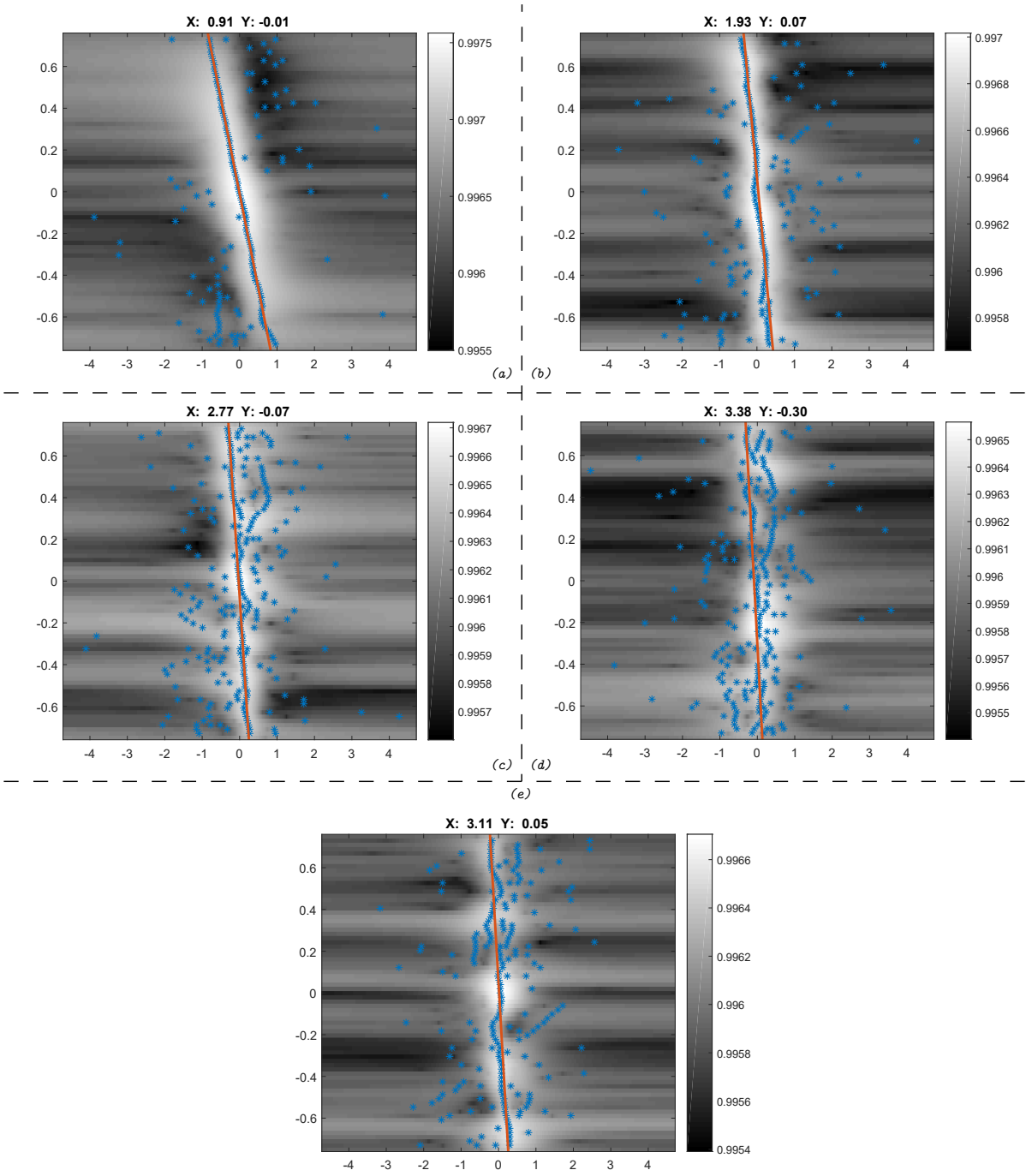


Figure 5.14: *Estimation of speaker positions. (a) r' = [1, 0], estimation finds [0.91, -0.01]. (b) r' = [2, 0], estimation finds [1.93, 0.07]. (c) r' = [3, 0], estimation finds [2.77, -0.07]. (d) r' = [4, 0], estimation finds [3.38, -0.30]. (e) r' = [5, 0], estimation finds [3.11, 0.05].*

The estimations find:

- **(a),** $r' =$ **[1, 0]**: Estimated location in [0.91, -0.01], $\epsilon = 0.091$, $\phi = 0.630°$
- **(b),** $r' =$ **[2, 0]**: Estimated location in [1.93, 0.07], $\epsilon = 0.099$, $\phi = 2.077°$
- **(c),** $r' =$ **[3, 0]**: Estimated location in [2.77, -0.07], $\epsilon = 0.240$, $\phi = 1.448°$
- **(d),** $r' =$ **[4, 0]**: Estimated location in [3.38, -0.30], $\epsilon = 0.689$, $\phi = 4.421°$
- **(e),** $r' =$ **[5, 0]**: Estimated location in [3.11, 0.05], $\epsilon = 1.891$, $\phi = 0.921°$

The test shows that the RSTM estimation starts failing when the distance is beyond three meters, atleast in non-anechoic conditions. On estimating relatively near sources it performs similarly as to in anechoic conditions, with less than 10 cm error for sources at one and two meters distance.

6

Conclusions and Perspectivation

In this thesis we further build the knowledge base on the ray space transform method and its performance in sound field analysis, by acoustic source localization. We delve into the components of the method, discuss these, and establish the link between the components, the method itself, and the real world. The parameter study focuses on single parameters while keeping others static. This provides information as to how each parameter on its own affects the ray space and source localization therein. To expand this study, the authors conclude that a future work studying how changing multiple parameters can be beneficial for certain scenarios, such as sources at very far distances, sources behind other sources, and many simultaneous sources is of great interest.

Following the study of method specific parameters, we carry out a comparison of the RSTM and a more established method in sound-field analysis, the spatial decomposition method. It is found that when studied on as equal terms as found possible by the authors, the methods perform very similar in regards to noise and measurement error, i.e. microphone position error, noisy signals, and phase mismatch. The RSTM outperforms SDM in few of the comparisons: Estimation of sources very close to the microphone array, and type of input signal. In the remainder of comparison variables the SDM performs better, albeit sometimes only slightly. These are: signal incidence angle, number of microphones, and estimation of multiple sources.

Tests are carried to verify the performance of the RSTM with actual measured data. At first an initial test is carried out using less than optimal microphones[1] in non-anechoic conditions. This contributes to attain experience with the practicalities of the method, while giving insight into the performance of the method in sub-optimal conditions. The tests show the method performing according to what the simulations predicted using only eight microphones, see section 4.9. Angle is found for each of the tested source locations while the distance error varies from 0.1 meter to just over a meter. Anechoic chamber tests followed this, using the full 16 microphone array. Tests are performed at different positions in the chamber, covering 13 positions. The estimations show that the simulated results matches those of real measured data in the anechoic chamber. Low distance error[2] and low angle error[3] prove that the method works as predicted, thus verifying the simulations. A final test is performed in a standard living room, conforming to the IEC-268-13 standard, using the 16 microphone array and estimating performance at different distances from source to array. These show that the RSTM perform well at least within a three meter distance in semi-reverberant conditions. Beyond three meters, the incidence angle is still found correctly, but the distance estimation suffers more the further away we move.

To make the RSTM a viable alternative to SDM or other contemporary sound field analysis methods, the authors conclude that a few things need improvement and/or changing[4]:

[1] Here, eight microphones is used while 16 would correspond to the amount used in simulations.
[2] below 20 cm for all cases
[3] below 3° for all cases
[4] These are based on the capabilites of the SDM in sound field analysis.

- Number of microphones and/or microphone geometry. The array as it is, is a 1.5 meter linear array, not suited for many applications since they often require compact solutions to sound field analysis. This might be achieved by either optimization of the method and algorithm, or to change the shape of the array to something more effective. The downside to changing array geometry is that the mathematical aspects will change and likely grow a lot more complex.
- Three dimensional analysis. As it is right now, the method estimates in two dimensions, using the linear array. To achieve three dimensional estimation using the same configuration would require another perpendicular array to the linear array. This also comes down to changing and optimizing array geometry.
- Estimation behind array. While not all applications need the array to be able to see sources behind it, the feature would surely benefit the usage of the method in applications.
- As studied in this thesis, the choosing of parameters have an influence on the output images of the ray space. Depending on the acoustic scenario, and specially in complex room geometries the selection of these parameters can be difficult. In this sense a tuning of this parameters through machine learning can be useful and interesting for a future study.

The previous points all regard the performance in sound field analysis, and while an implementation of using RSTM in spatial filtering and reproduction of sound fields were outside the scope of this thesis, carrying out a similar study on this is to the authors a very interesting subject for future projects. In [2] and [3] the authors claim that the RSTM is very well suited for this purpose, and it might be the area where the RSTM has its biggest potential.

References

[1] S. Tervo, J. Pätynen, A. Kuusinen, T. Lokki, "Spatial Decomposition Method for Room Impulse Responses," in Journal of the Audio Engineering Society 61(1):16-27, Jan. 2013.

[2] L. Bianchi, F. Antonacci, A. Sarti and S. Tubaro, "The Ray Space Transform: A New Framework for Wave Field Processing," in IEEE Transactions on Signal Processing, vol. 64, no. 21, pp. 5696-5706, Nov.1, 1 2016.

[3] D. Marković, F. Antonacci, L. Bianchi, S. Tubaro and A. Sarti, "Extraction of Acoustic Sources Through the Processing of Sound Field Maps in the Ray Space," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 24, no. 12, pp. 2481-2494, Dec. 2016.

[4] F. Borra, L. Bianchi, F. Antonacci, S. Tubaro and A. Sarti, "A robust data-independent near-field beamformer for linear microphone arrays," 2016 IEEE International Workshop on Acoustic Signal Enhancement (IWAENC), Xi'an, 2016, pp. 1-5. doi: 10.1109/IWAENC.2016.7602934

[5] M.A. Fischler, R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," in Communications of the ACM, 1981:381- 395.

[6] T. Ajdler, L. Sbaiz and M. Vetterli, "The Plenacoustic Function and Its Sampling," in IEEE Transactions on Signal Processing, vol. 54, no. 10, pp. 3790-3804, Oct. 2006.

[7] R. A. Kennedy, T. D. Abhayapala and D. B. Ward, "Broadband nearfield beamforming using a radial beampattern transformation," in IEEE Transactions on Signal Processing, vol. 46, no. 8, pp. 2147-2156, Aug 1998.

[8] O. Bretscher, "Linear Algebra with Applications", 5th Edition, Chapter 5, Pearson 2013. ISBN-13: 9780321946553.

[9] R. O. Duda, P. E. Hart. 1972. "Use of the Hough transformation to detect lines and curves in pictures," Commun. ACM 15, 1 (January 1972), 11-15. DOI=http://dx.doi.org/10.1145/361237.361242

[10] Hough, P.V.C. "Method and means for recognizing complex patterns," U.S. Patent 3069654, Dec. 18, 1962.

[11] D. Marković, F. Antonacci, A. Sarti and S. Tubaro, "Soundfield Imaging in the Ray Space," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 21, no. 12, pp. 2493-2505, Dec. 2013. doi: 10.1109/TASL.2013.2274697

[12] A. Hast, J. Nysjö, A. Marchetti (2013). "Optimal RANSAC – Towards a Repeatable Algorithm for Finding the Optimal Set," Journal of WSCG 21 (1): 21–30.

[13] M. Park, B. Rafaely, "Sound-field analysis by plane-wave decomposition using spherical microphone array," The Journal of the Acoustical Society of America 118, 3094 (2005); doi: http://dx.doi.org/10.1121/1.2063108

[14] L. Bianchi, "A unified framework for acoustic scene analysis, synthesis and processing," PhD Thesis in Information Technology from Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), 2016. http://hdl.handle.net/10589/117083.

[15] J. B. Allen, D. A. Berkley. "Image method for efficiently simulating small-room acoustics," Journal of the Acoustical Society of America, vol. 65, no. 4, pp. 943–950, 1979.

[16] E. A. P. Habets, "Room impulse response generator," Technische Univ. Eindhoven, Eindhoven, The Netherlands, Tech. Rep., 2006.

[17] D. Marković "Plenacoustic processing in the ray space: applications to acoustic scene modeling and analysis," PhD Thesis in Information Technology from Politecnico di Milano, Dipartimento di Elettronica e Informazione, 2013.

[18] Christensen, Hald. "B&K Beamforming Technical Review," issue 1, 2004.

[19] J. Dmochowski, J. Benesty and S. Affes, "On Spatial Aliasing in Microphone Arrays," in IEEE Transactions on Signal Processing, vol. 57, no. 4, pp. 1383-1395, April 2009. doi: 10.1109/TSP.2008.2010596

[20] S. Tervo, J. Pätynen, N. Kaplanis, L. Morten, S. Bech, and T. Lokki , "Spatial analysis and synthesis of car audio system and car cabin acoustics with a compact microphone array," J. Audio Eng. Soc. 63(11), 914–925 (2015).

[21] S. Tervo, P. Laukkanen, J. Pätynen, and T. Lokki, "Preferences of critical listening environments among sound engineers," J. Audio Eng. Soc. 62(5), 300–314 (2014).

[22] S. Tervo, J. Saarelma, J. Pätynen, I. Huhtakallio, and P. Laukkanen, "Spatial analysis of the acoustics of rock clubs and nightclubs," Proc. Inst. Acoust. 37(3), 551–558 (2015).

[23] S.V. Amengual Garí, W. Lachenmayr, E. Mommertz, "Spatial analysis and auralization of room acoustics using a tetrahedral microphone," Journal Acoustic Society of America. 2017 Apr;141(4):EL369

[24] C. Knapp, G. Carter, "The Generalized Correlation Method for Estimation of Time Delay," IEEE Trans. Acoust., Speech and Signal Proc., vol. 24, no. 4, pp. 320–327 (1976).

[25] L. Zhang, X. Wu, "On Cross Correlation Based Discrete Time Delay Estimation," IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 4, pp. 981–984 (2005).

[26] E. Adelson, J. Bergen, "The plenoptic function and the elements of early vision," in Computational Models of Visual Processing. Cambridge, MA: MIT Press, 1991, pp. 3–20.

[27] S. Qian, D. Chen, "Discrete Gabor transform," IEEE Trans. Signal Process., vol. 41, no. 7, pp. 2429–2438, Jul. 1993