Room Shape Estimation from Acoustic Echoes using Graph-based Echo Labeling

Ingmar Jager
Room Shape Estimation from Acoustic Echoes
using Graph-based Echo Labeling

by

Ingmar Jager

in partial fulfillment of the requirements for the degree of

Master of Science
in Electrical Engineering

at the Delft University of Technology,
to be defended publicly on Friday September 18, 2015 at 1:45 PM.

Supervisor: Dr. ir. R. Heusdens
Thesis committee:
  Dr. ir. R.C. Hendriks
  Dr. ir. J. Weber
  Dr. N.D. Gaubitch

An electronic version of this thesis is available at http://repository.tudelft.nl/.

CAS-MS-2015-09
Some vision impaired people can hear the shape of a room using acoustic echoes. A computer being able to do the same could benefit applications such as auralization, virtual reality and teleconferencing.

This thesis describes the process of estimating the room shape from acoustic room impulse responses to finding geometry from image sources. As we use multiple microphones, the biggest challenge is to find the echoes from different microphones that correspond to the same image source.

We present a new method to disambiguate the echoes using graph theory. We model combinations of echoes as nodes in a graph. The maximum independent set in the graph yields the disambiguated echoes. The disambiguated echoes are transformed to time-difference-of-arrival data so that we are able to calculate the locations of the sources and image sources in a closed-form fashion. From the estimated sources and image sources we finally infer the room geometry using the image source model.

The experiments, which are limited to simulated shoe box shaped rooms show that we can reliably estimate room shapes within seconds on contemporary hardware. We achieve a sub-centimeter precision on finding the vertices of the room.
This thesis project is done as a completion of the master Electrical Engineering at Delft University of Technology. Before I started the master program with specialisation Signals and Systems I completed the bachelor program Electrical Engineering at the same university.

I always like to try out new tools and techniques. Instead of going with Matlab by default I opted for the Python programming language as a simulation tool. Python combined with the great open source scientific libraries from Scipy [1] and Numpy [2] work as a great replacement for Matlab. The flexible drawing capabilities of Matplotlib [3] have provided most of the figures in this thesis. Finally working in the IPython Notebook [4] environment (Now called Jupyter Notebook) has been a great joy. Being able to use code, markdown + Latex and view the results inline have proven to be a great way to hack away at problems.

Since I have been using all these great open source tools, I would like to return the favor by releasing all the code I wrote for this thesis under the MIT license on GitHub: https://github.com/ijager/Thesis.

Finally I would like to thank my parents for their endless support during my years as a student and Richard Heusdens and Nikolay Gaubitch for their guidance and honest feedback during this project.

Enjoy the read,

Ingmar Jager
Delft, Augustus 2015
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>iii</td>
</tr>
<tr>
<td>Preface</td>
<td>v</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2 Problem Description</td>
<td>3</td>
</tr>
<tr>
<td>2.1 Acoustic echoes</td>
<td>3</td>
</tr>
<tr>
<td>2.2 Image Source Model</td>
<td>5</td>
</tr>
<tr>
<td>2.3 Obtaining acoustic echoes</td>
<td>6</td>
</tr>
<tr>
<td>2.4 Echo Labeling</td>
<td>6</td>
</tr>
<tr>
<td>3 Related Work</td>
<td>9</td>
</tr>
<tr>
<td>3.1 Measuring Room Impulse Responses</td>
<td>9</td>
</tr>
<tr>
<td>3.1.1 Maximum Length Sequence</td>
<td>9</td>
</tr>
<tr>
<td>3.1.2 Time-Stretched Pulse</td>
<td>10</td>
</tr>
<tr>
<td>3.1.3 Sine Sweep Technique</td>
<td>11</td>
</tr>
<tr>
<td>3.1.4 Comparison</td>
<td>12</td>
</tr>
<tr>
<td>3.2 Echo Labeling</td>
<td>12</td>
</tr>
<tr>
<td>3.2.1 Euclidean Distance Matrix</td>
<td>13</td>
</tr>
<tr>
<td>3.2.2 Using the EDM</td>
<td>15</td>
</tr>
<tr>
<td>3.2.3 Multidimensional Scaling</td>
<td>15</td>
</tr>
<tr>
<td>3.3 Room Reconstruction</td>
<td>17</td>
</tr>
<tr>
<td>3.3.1 Filter out higher order image sources</td>
<td>17</td>
</tr>
<tr>
<td>4 Design</td>
<td>19</td>
</tr>
<tr>
<td>4.1 Acquisition of Acoustic Echoes</td>
<td>20</td>
</tr>
<tr>
<td>4.1.1 Translation of Echoes to Distances</td>
<td>21</td>
</tr>
<tr>
<td>4.2 Echo Labeling</td>
<td>23</td>
</tr>
<tr>
<td>4.2.1 Pre-filtering</td>
<td>23</td>
</tr>
<tr>
<td>4.2.2 Subsets of C</td>
<td>24</td>
</tr>
<tr>
<td>4.2.3 Maximum Cliques</td>
<td>24</td>
</tr>
<tr>
<td>4.3 Localization of Image Sources</td>
<td>27</td>
</tr>
<tr>
<td>4.4 Overview</td>
<td>28</td>
</tr>
<tr>
<td>4.5 Room Reconstruction</td>
<td>29</td>
</tr>
<tr>
<td>5 Experiments and Results</td>
<td>31</td>
</tr>
<tr>
<td>5.1 Sampling Rate and Interpolation</td>
<td>31</td>
</tr>
<tr>
<td>5.2 Number of Measurements</td>
<td>32</td>
</tr>
<tr>
<td>5.3 Number of Receivers</td>
<td>33</td>
</tr>
<tr>
<td>5.4 Graph-based Method vs MDS-based Method</td>
<td>34</td>
</tr>
<tr>
<td>5.5 Recommended Configuration</td>
<td>35</td>
</tr>
<tr>
<td>6 Conclusions</td>
<td>37</td>
</tr>
<tr>
<td>6.1 Recommendations</td>
<td>37</td>
</tr>
<tr>
<td>A Pollefeys localization</td>
<td>39</td>
</tr>
<tr>
<td>Bibliography</td>
<td>41</td>
</tr>
</tbody>
</table>
Introduction

Dolphins and bats can hear their environment by producing sounds and listening to the echoes. They use the echo information to navigate. Even some blind people use human echolocation to detect objects in their environment and consequently for acoustic way-finding. Since acoustic echoes bounce off reflective surfaces, the echoes hold information about surfaces in a room. Lately there has been an increasing interest in designing computer algorithms that can measure the shape of a convex room using these acoustic echoes.

Computer algorithms that reconstruct the shape of a room can be advantageous for several applications. In auralization one needs to model the source, the medium and the receiver. The medium can be modeled if you have information about the shape of the rooms in which the source and receiver are placed. For teleconferencing one may want to take into account the reflections of sound, also called reverberations, caused by a room, prior to the excitation of a sound signal. This could be done for both the room on the far end as well as for the room on the near end. Another example which involves both canceling and simulating reverberations is in certain virtual reality applications. One may like to take into account the room shape of the real room in order to generate sounds congruent with reverberations caused by the virtual room. Autonomous robots that operate indoor could directly take advantage of knowing the room geometry as it can be used for mapping and localization.

Several methods for obtaining the room shape from acoustic echoes have been proposed. For example [5] attempts to find the 2D room shape using one room impulse response (RIR). Also in [6] they are limited to the 2D case, while using multiple microphones and sources.

The authors of [7] keep the arrangement of microphones small enough that they can assume that the echoes will cluster together in time. It works in 3D but brings along many restrictions with relation to the microphone and speaker placement and directions.

In [8], Dokmanić et al. describe a way to acquire the room by recording the echoes using five microphones, which can be placed arbitrarily in the room. They only need one source. They utilize the properties of the Euclidean distance matrix and multidimensional scaling to iteratively find the room shape. While this method can be very precise, it is also computationally expensive.

In this thesis we propose a method to obtain the room shape in 3D which is much faster, while maintaining the accuracy of [8]. We will use 5 microphones and at least 2 speakers (or one speaker excited at different locations). The main contribution is an algorithm for efficient echo labeling using graph theory and direct computation of source locations.

We begin by describing the problem in detail while also covering the required background information on acoustics and signal processing in Chapter 2. Then in Chapter 3 the current solutions to the problems are highlighted. A new approach to solve the problem is presented in Chapter 4. This design is evaluated in chapter 5.
Problem Description

Estimating the shape of a room means that we want to find the location of the walls. Obtaining this shape from acoustic echoes comprises roughly three steps. First we need to obtain acoustic echoes from a recorded signal. The echoes hold the distance information to the walls. But we do not know which echoes belong to which wall. Hence the echoes need to be labeled correctly. Using the correctly labeled echoes, we can find image sources and therefore the walls.

This chapter will describe these steps and the challenges involved as well as the background theory. First Section 2.1 will define what we mean by acoustic echoes and why we are interested in them. Then Section 2.2 explains how we can relate the echoes to the wall. Section 2.3 will identify the challenges involved with obtaining acoustic echoes, while Section 2.4 shows why we need to know which echoes came from the same wall.

2.1. Acoustic echoes

Consider an audio source or event positioned at \( s \) and a receiver or microphone at \( r \). Besides that \( s \) and \( r \) are position vectors in three-dimensional space, they are also used to refer to the actual source and microphone at those positions.

When \( s \) produces sound at time \( T_0 \) inside a room, part of the sound will travel directly to the receiver \( r \) and will arrive at time \( t = T_0 + \tau_0 \). Here, \( \tau_0 \) is the time delay for the direct sound to reach the receiver, and it is defined as

\[
\tau_0 = \frac{||s - r||_2}{c},
\]

with \( c \) the speed of sound. In addition, other parts of the sound will first reflect off surfaces in the room before reaching \( r \) at \( t = T_0 + \tau_i \) with \( \tau_i > \tau_0 \). Due to triangle inequality, reflected sound will always arrive later at the source than the direct sound. Sound that has reflected once is called a first order reflection. Second order reflections have bounced off surfaces twice. Figure 2.1 illustrates this example, drawing the direct sound and one first and second order echo.
When the source excites an ideal pulse, all reflections that reach the receiver together compose a room impulse response (RIR). The RIR, which ideally is a train of pulses, each corresponding to an echo, can be expressed as

$$h(t) = \sum \alpha_i \delta(t - \tau_i). \tag{2.1}$$

Here \(\alpha_i\) is the reflection coefficient associated with the \(i\)th reflection. The reflection coefficient attenuates the energy of that particular reflection and depends on the type of surface material, reflection order and distance covered by the sound. Figure 2.2 shows an ideal RIR based on the situation in Figure 2.1.

Any sound emitted in a room will be affected by the room impulse response. At every location in the room the signal \(y(t)\) that will reach the receiver will be the result of the convolution of the original sound signal \(x(t)\) with \(h(t)\)

$$y(t) = x * h(t) = \int x(t) h(t - \tau) d\tau.$$

Note that \(h(t)\) is specific for each location of \(r\) and \(s\), as the reflection times \(\tau_i\) will be different. It is not necessarily unique however.
Since the echoes are closely related to the wall locations, as will be further explained in Section 2.2, we are very interested in obtaining the room impulse response in a particular room.

### 2.2. Image Source Model

The time delays $\tau_i$ which are obtained by measuring the RIR are the propagation times of the emitted sound. These can be coupled to room geometry with the image source model, first implemented in [9]. The image source model allows us to replace reflections by virtual sources mirrored in the surfaces of the room. This is possible as the length of the path that the reflected sound covers is equal to the distance from the corresponding image source to the receiver. The concept of virtual sources is illustrated in Figure 2.3. An image source can be represented in terms of the source and the wall it is mirrored in as

$$s'_i = s + 2(p_i - s, n_i)n_i.$$  \hspace{1cm} (2.2)

$s'_i$ is the image source corresponding to wall $i$, $s$ is the location of the source, $p_i$ is an arbitrary point on wall $i$ and $n_i$ is the unit normal for wall $i$ pointing outward from the room.

For a shoe box shaped room, the image sources form an indefinitely repeating grid as in Figure 2.4. Such a grid is formed by mirroring the image sources itself in walls on the other side of the room and in virtual walls which are repeated as well.

As each virtual image corresponds to a wall, knowing the location of the image sources is equal to knowing the wall configuration. Each delay $\tau_i$ in the RIR can be expressed as the distance between the receiver and the image source divided by the speed of sound $c$:

$$\tau_i = \frac{||s'_i - r||}{c}.$$  \hspace{1cm} (2.3)
2.3. Obtaining acoustic echoes

The signal as depicted in figure 2.2 is measured by exciting an ideal impulse. However an ideal impulse would require infinite bandwidth, and that is not possible in practice. The most naive method to obtain the echoes is to excite a pulse-like sound such as a clap or gunshot and measure the response. This would be good enough if we were just interested in the direct sound. Since we are especially interested in a reliable estimate of the echoes, we need to find a more suitable excitation signal which enables us to estimate the room impulse response. A good signal to noise ratio is important for identifying the peaks in the RIR, but even more important is the accuracy of the estimated time delay for each peak. The time delay is directly related to the distance from the image source to the receiver and thus to the wall location.

2.4. Echo Labeling

Having one microphone and one distance to an image source is not enough to locate the image source. As Figure 2.5 shows, we already need 3 microphones to locate a source in 2D. In 3D we would need at least 4 microphones.
2.4. Echo Labeling

But more microphones do not guarantee that we can find the image sources, as it is ambiguous which echoes we need for locating each image source. For each microphone the room impulse response is recorded. In each RIR we find peaks that correspond to echoes from the walls in the room. However the order in which the echoes arrive at each microphone are not necessarily the same. For example the reflections from two walls might arrive in inverse order at different microphones as is shown in Figure 2.6.

This gives rise to the new challenge of labeling the echoes. We need to find out which sets of echoes belong to the same wall. Where a set of echoes means $M$ echoes, recorded at $M$ different microphones.

$$
\begin{align*}
    r_1 &: \{e_1, e_2, e_3, e_4, e_5\} & r_1 &: \{e_0, e_2, e_3, e_4\} \\
    r_2 &: \{e_1, e_2, e_3, e_4, e_5\} & r_2 &: \{e_0, e_1, e_3, e_4\} \\
    r_3 &: \{e_1, e_2, e_3, e_4, e_5\} & r_3 &: \{e_0, e_2, e_3, e_1, e_4\} \\
    r_4 &: \{e_1, e_2, e_3, e_4, e_5\} & r_4 &: \{e_0, e_3, e_1, e_2, e_4\}
\end{align*}
$$

In this example $r_i$ is the data from microphone $i$ and $e_j$ represent an unlabeled echo, $e_j$ is an echo reflected from wall $j$. There are 4 microphones which all receive 5 echoes, the direct sound and reflections from 4 walls. The colors indicate which echoes form an correct set of echoes, however we
do not actually have that information available at this point. We need to go from the situation on the left side to the situation on the right side, where the echoes are labeled. Note the $e_0$ echoes are all in the correct order already, this is because the direct sound always reaches the microphones first. That the $e_4$ echoes are in the correct order is to show that echoes from the same wall can arrive at the microphones at the same index, but that is just a coincidence.
3 Related Work

This Chapter summarizes work from other literature that is related to the subject of this thesis. First, Section 3.1 describes techniques to measure room impulse responses. Then in Section 3.2 we go over an existing method to disambiguate echo data. Finally a method to deal with higher order image sources is explained in Section 3.3.1.

3.1. Measuring Room Impulse Responses

This section will describe the process of measuring room impulse responses (RIR). The most naive way to measure the room impulse response is to make a impulse like sound, like a gunshot or clap, and record the sound arriving at the microphone. More reliable methods have been proposed which all involve more sophisticated shaped excitation signals. Four of the most suited methods have been compared in [10]: Maximum length sequence, inverse repeated sequence, time-stretched pulse and sine sweep. These methods all work by applying a known input signal and measuring the system’s output signal. The deconvolution of the output signal and the known input signal yields the impulse response. To obtain a high signal-to-noise ratio impulse response, the choice of excitation signal and the deconvolution technique are important.

3.1.1. Maximum Length Sequence

A maximum length sequence (MLS) is a sequence generated by a linear feedback shift register and therefore is a pseudo-random binary sequence. The number of registers in a shift register determines the order of an MLS. The number of samples in one period of an m order MLS signal is: \( L = 2^m - 1 \). Figure 3.1 shows an MLS signal of order 5. A fifth order signal means the period of the signal is \( 2^5 - 1 = 31 \) samples, so the signal in the figure has a length of 2 periods. It was generated using Equation 3.1, where \( s[n] \) is the MLS and \( A \) is the amplitude, with \( A = 1 \).

\[
x[n] = A(-1)^s[n]
\] (3.1)
Figure 3.1: Two periods of a 5th order MLS signal $x[n]$ with amplitude 1

The deconvolution process to estimate the impulse response consists of a circular cross-correlation between the measured output and the known input signal. In order to reduce noise, many measurements are done and averaged. As such we have to assume that the room is a time invariant system.

### 3.1.2. Time-Stretched Pulse

In [11] Suzuki et al. present an optimized version of Aoshima’s time-stretched pulse (AOTSP) [12]. It is a signal of length $N$ that is defined in the frequency domain as

$$X(k) = \begin{cases} \exp(j4\pi mk^2/N^2), & 0 \leq k \leq N/2 \\ X(N - k), & N/2 < k < N \end{cases}$$

(3.2)

where $m$ is an integer that determines the stretch of the OATSP. Figure 3.2 shows $x[n]$, obtained by performing the inverse Fourier transform on $X(k)$. The magnitude of the frequency spectrum is fixed, as Equation 3.2 shows. The time expansion works to increase the signal-to-noise ratio without increasing the nonlinearities introduced by the measurement system. The deconvolution is done with a reverse filter, defined by the reciprocal of $X(k)$ as its magnitude spectrum, that compresses the signal again.
3.1. Measuring Room Impulse Responses

In the same way as with the MLS, we can reduce noise by averaging multiple measurements.

3.1.3. Sine Sweep Technique

The final excitation signal is a swept-sine signal. That is a sinusoidal signal with increasing instantaneous frequency. In [13], Farina showed that an exponential time-growing frequency sweep has the best performance (sometimes called logarithmic sinesweep). This signal is defined as

\[ x(t) = \sin\left( \frac{\omega_0 T}{\ln(\frac{\omega_1}{\omega_0})} \exp\left( \frac{t}{T} \ln(\frac{\omega_1}{\omega_0}) \right) - 1 \right). \] (3.3)

Here \( \omega_0 \) is the frequency the signal starts at and \( \omega_1 \) is the frequency where it will finish its sweep.

Instead of relying on the assumption that the system (the room) is linear and time-invariant, the sine sweep deconvolution can actually separate the impulse response from harmonic distortions. Farina [13] pointed out that the signal coming out of the speaker contains harmonic distortions which may be modeled by the Volterra Kernel [14] as:

\[ w(t) = x(t) \ast k_1(t) + x^2(t) \ast k_2(t) + \ldots + x^N(t) \ast k_N(t), \] (3.4)

where \( k_i(t) \) is the \( i^{th} \) component of the Volterra kernel. Now if we take into account the entire system, including the room impulse response, then the received signal at the microphones can be written as

\[ y(t) = n(t) + x(t) \ast h_1(t) + x^2(t) \ast h_2(t) + \ldots + x^N(t) \ast h_N(t), \] (3.5)

with \( h_i(t) = k_i(t) \ast h(t) \) and additive white Gaussian noise \( n(t) \).

For the deconvolution we need to do the linear convolution with the inverse filter. The inverse filter is obtained by temporally reversing the original excitation signal, and shifting it back to the positive time axis. Furthermore, amplitude modulation is applied to compensate for the different energy generated at low and high frequencies.
3.1.4. Comparison

According to [10], the presented methods are favorable under different circumstances. In an occupied room, the MLS technique is preferred. The drawback is that the system has to be carefully calibrated to obtain optimal results. Non-linearities in the measurement system show up as spurious peaks. These peaks can be avoided by using the time stretched pulse, and remaining nonlinear artifacts can be almost completely eliminated by precise calibration of the measurement system. However, the high value of the optimal output signal level needed to mask out ambient noise, makes it not very suitable for occupied rooms.

In non-occupied rooms, the sinesweep method turns out to be the best method for measuring room impulse responses. Advantages are the perfect and complete rejection of harmonic distortions, and that there is no need to average over multiple measurements. Also no tedious calibrations are needed to obtain very good results.

3.2. Echo Labeling

This section describes a method to disambiguate the acoustic echoes from multiple room impulse responses. The algorithm was presented in [8]. Before we dive into the specifics of the method we need to introduce the concept of the Euclidean distance matrices. In this section all coordinates are 3 dimensional and the room is setup as illustrated in Figure 3.4, which shows the top view of a shoe box shaped room with randomly distributed microphones.
### 3.2.1. Euclidean Distance Matrix

Euclidean distance matrices (EDMs) are a useful tool in signal processing as they hold some interesting properties. An EDM contains all the squared Euclidean distances between all points in an N-dimensional point set \( X \). If \( D \in \mathbb{E}D\mathbb{M} \), the set of all EDMs, then the entries of \( D \) denoted by \( d_{ij} \) are defined as:

\[
d_{ij} = \|x_i - x_j\|^2, \tag{3.6}
\]

with \( x_i, x_j \in X \).

If we use the 5 microphones \( r_1 \) to \( r_5 \) from figure 3.4 as our point set \( X \), we can construct \( D \) as follows:

\[
D = \begin{bmatrix}
    r_1 & r_2 & r_3 & r_4 & r_5 \\
    r_2 & d_{21} & 0 & d_{23} & d_{25} \\
    r_3 & d_{31} & d_{32} & 0 & d_{34} & d_{35} \\
    r_4 & d_{41} & d_{42} & d_{43} & 0 & d_{45} \\
    r_5 & d_{51} & d_{52} & d_{53} & d_{54} & 0
\end{bmatrix}
\]

Note the zeros on the diagonal, as \( d_{ii} \) is always zero. Additionally, it holds that \( d_{ij} = d_{ji} \), making \( D \) a symmetric matrix by construction.

We can define a function that outputs \( D \in \mathbb{E}D\mathbb{M} \) from point set \( X \) as

\[
D(X) = \text{diag}(XX^\top)1^\top + 1\text{diag}(XX^\top)1^\top - 2XX^\top, \tag{3.7}
\]

where \( \text{diag}(\cdot) \) denotes a column vector containing all diagonal entries of the input matrix and \( 1 \) an all ones column vector in \( \mathbb{R}^n \).

Using the function from Equation 3.7 we can obtain an EDM by constructing it. The following definitions are needed in order to describe the necessary and sufficient conditions for an arbitrary matrix to be an EDM.

**Definition 1 (Symmetric hollow subspace[15])** Denoted by \( S^n_h \), the symmetric hollow subspace is a proper subspace of symmetric matrices \( S^n \) with zero diagonal.

\[
S^n_h \overset{\text{def}}{=} \{ A \in S^n | \text{diag}(A) = 0 \}, \tag{3.8}
\]
Definition 2 (Positive semi-definite cone[15]) Denoted by $\mathbb{S}_+^n$, the positive semi-definite cone is the set of all symmetric positive semi-definite matrices of dimension $n \times n$

$$\mathbb{S}_+^n \overset{\text{def}}{=} \{ A \in \mathbb{S}^n | A \succeq 0 \} \quad (3.9)$$

Let $L$ be a geometric centering matrix as

$$L \overset{\text{def}}{=} I - \frac{1}{n} 1 1^T \quad (3.10)$$

with $I$ a $n \times n$ identity matrix.

These definitions lead to the necessary and sufficient conditions declared in Theorem 1.

Theorem 1 (Schoenberg [16])

$$D \in \mathbb{EDM} \iff \begin{cases} -LDL \in \mathbb{S}_+^n \\ D \in \mathbb{S}_+^n \end{cases} \quad (3.11)$$

Another important property of Euclidean distance matrices that we will use later on is called the embedding (or affine) dimension.

Definition 3 (Embedding dimension [17]) If a matrix $D \in \mathbb{R}^{n \times n}$ is an EDM, its embedding or affine dimension is the rank of $X$ with the least rank that generates $D$.

This means that the embedding dimension of an EDM is the dimension of the smallest affine set in $\mathbb{R}^n$ that contains $X$. The requirement of least rank comes from the fact that there are in general infinitely many point sets $X$ that generate $D$. For example, the 1D point set containing 2 points

$$X_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{, with rank 1} \quad (3.12)$$

and 2D point set with 2 points

$$X_2 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \text{, with rank 2} \quad (3.13)$$

both generate

$$D = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (3.14)$$

Theorem 2 (EDM rank vs. embedding dimension [17]) For a Euclidean distance matrix $D \in \mathbb{R}^{n \times n}$ with embedding dimension $r$, we have

$$\text{rank}(D) \leq r + 2 \quad (3.15)$$

Further, $\text{rank}(D) = r + 1$, if and only if the points generating $D$ lie on the relative boundary of an $r$-dimensional hypersphere.

The upper bound stated by equation 3.15 can be proven using the function $D(X)$ of equation 3.7.

$$\text{rank}(D(X)) \leq \text{rank}(\text{diag}(XX^T)1^T) + \text{rank}(1\text{diag}(XX^T)1^T) + \text{rank}(2XX^T) \quad (3.16)$$

$$\leq 1 + 1 + r \quad (3.17)$$
3.2.2. Using the EDM

The properties of the Euclidean distance matrix describe in section 3.2.1 can be used to correctly label echoes. We start by creating a Euclidean distance matrix $D$ from the squared pairwise distances between the microphones in the room.

$$D = \begin{bmatrix}
0 & d_{12} & d_{13} & d_{14} & d_{15} \\
 d_{21} & 0 & d_{23} & d_{24} & d_{25} \\
 d_{31} & d_{32} & 0 & d_{34} & d_{35} \\
 d_{41} & d_{42} & d_{43} & 0 & d_{45} \\
 d_{51} & d_{52} & d_{53} & d_{54} & 0
\end{bmatrix} \quad (3.18)$$

The next step is to select a combination of echoes from the RIRs. From each RIR we pick one echo and we want to see if the distances corresponding to these echoes belong to the same virtual image source. We select vector $e = [e_1, e_2, e_3, e_4, e_5]$ containing squared Euclidean distances $e_i$ obtained from some peak from microphone $i$. Matrix $D$ is then augmented with $e$ by adding it as an additional row and column resulting in $D_{aug}$. If $D_{aug}$ passes the requirements to be an EDM we know that the echoes come from a real point in space, as the Euclidean distance matrix properties do still hold.

$$D_{aug} = \begin{bmatrix}
0 & d_{12} & d_{13} & d_{14} & d_{15} & e_1 \\
 d_{21} & 0 & d_{23} & d_{24} & d_{25} & e_2 \\
 d_{31} & d_{32} & 0 & d_{34} & d_{35} & e_3 \\
 d_{41} & d_{42} & d_{43} & 0 & d_{45} & e_4 \\
 d_{51} & d_{52} & d_{53} & d_{54} & 0 & e_5 \\
e_1 & e_2 & e_3 & e_4 & e_5 & 0
\end{bmatrix} \quad (3.19)$$

To verify that $D_{aug} \in \mathbb{EDM}$, we have to test if Theorem 1 holds. If it does, the combination of echo distances can be used to computed the location of the image source. If $D_{aug}$ is not an EDM, then we can discard the current combination of echoes. Doing this test for all possible combinations of echoes yields all image sources.

Unfortunately, in practice, testing Theorem 1 is not feasible due to noise in the RIRs and errors in the measurements of pairwise microphone distances. Additive noise violates that $-LDL \in \mathbb{S}_+^n$. Instead of checking that $D_{aug} \in \mathbb{EDM}$, we can find the EDM closest to $D_{aug}$ by applying multidimensional scaling.

3.2.3. Multidimensional Scaling

When noise is involved, we will not have $D_{aug}$ but rather $\tilde{D}_{aug}$, a noisy observation matrix, which may not satisfy Theorem 1. We can try to find a matrix $D_{aug} \in \mathbb{EDM}^3$ that is as close as possible to $\tilde{D}_{aug}$ using multidimensional scaling (MDS). Equation 3.20 minimizes the s-stress criterion, which is a metric MDS procedure. It differs from the more common stress majorization in that it is defined on squared distances, hence s-stress. Let $s(D_{aug})$ be the s-stress score of $D_{aug}$.

$$s(\tilde{D}_{aug}) = \min \left\{ \sum_{ij} (D_{aug}[i,j] - \tilde{D}_{aug}[i,j])^2 \right\} \quad \text{s.t. } D_{aug} \in \mathbb{EDM}^3 \quad (3.20)$$

After computing this score for all $D_{aug} \in \mathbb{EDM}^3$, we can sort the scores and select the $D_{aug}$ matrix with the lowest score to obtain the point set which is best related to $\tilde{D}_{aug}$. However, calculating the s-stress score for every possible point in 3D space is not feasible. Instead we can use gradient descent to try to converge to the best possible candidate matrix $D_{aug}$. Consider each point in $\mathbb{R}^3$ as a coordinate vector $x_i = (x_i, y_i, z_i)^T$ and the entries of the distance matrices as $d_{ij}^2 = ||x_i - x_j||^2$, Equation 3.20 can be rewritten as

$$s(\tilde{D}_{aug}) = \min_{x_i, y_i, z_i \in \mathbb{R}} \sum_{ij} \left[ (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 - d_{ij}^2 \right]^2. \quad (3.21)$$

Note that we now need the absolute positions of the microphones, instead of just the pairwise distances. Therefore we need to define point set $X$ which is an $M + 1$ by 3 matrix, consisting of $M$
three dimensional receiver locations and one row for the image source we want to locate. Obviously, we do not know the image source location yet, so an initial position has to be assumed. The locations of the microphones are fixed so if we have \( M = 5 \) microphones we only have to find \( x_{M+1} \), with \( x_i \in X \). Essentially, we will be updating the location of the candidate image source. Then for every new location we calculate the \( d^2_{ij} \) entries, the squared distances between receivers and the candidate image source, to insert in the last row and column of \( D \). Finally, we calculate the s-stress score for that particular \( X \).

The coordinates of the candidate image source \( x_i, y_i, z_i \) need to be updated by \( \Delta x_i, \Delta y_i, \Delta z_i \) respectively. For updating \( x_i \), Equation 3.21 can be augmented as

\[
s(D_{aug})^{(k+1)}_i = \min_{x_i, y_i, z_i, \varepsilon} \sum_{i,j} \left[ (x_i^{(k)} + \Delta x_i^{(k+1)} - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 - d^2_{i,j} \right].
\]  

(3.22)

Where \((\cdot)^{(k)}\) represents the value at iteration \( k \).

The optimal value of \( \Delta x_i^{(k+1)} \) can be found by taking the derivative of \( s(D_{aug})^{(k+1)}_i \) with respect to \( \Delta x_i^{(k+1)} \)

\[
\frac{\delta s(D_{aug})^{(k+1)}_i}{\delta \Delta x_i^{(k+1)}} =
\]

\[
n(\Delta x_i^{(k+1)})^3 + 3 \sum_{j=1}^{n} (x_i^{(k)} - x_j)(\Delta x_i^{(k+1)})^2
\]

\[
+ \sum_{j=1}^{n} [3(x_i^{(k)} - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 - d^2_{i,j}]\Delta x_i^{(k+1)}
\]

\[
+ \sum_{j=1}^{n} [3(x_i^{(k)} - x_j)^3 + (x_i^{(k)} - x_j)(y_i - y_j)^2 + (z_i - z_j)^2 - d^2_{i,j}].
\]  

(3.23)

If we set the cubic function 3.23 equal to zero and solve for \( \Delta x_i^{(k+1)} \) we obtain one to three solutions. The optimal solution among these three is found by computing the s-stress score \( s(D_{aug})^{(k+1)}_i \) (equation 3.22) for each of them.

The same procedure is done for \( y_i \) and \( z_i \) using \( \Delta y_i^{(k+1)} \) and \( \Delta z_i^{(k+1)} \). Each iteration \( x, y \) and \( z \) are updated consecutively. This process is then repeated until \( s(D) \) does not converge any further or until a maximum number of iterations is reached. Since equation 3.20 is not convex, multiple starts at random coordinates can be initiated to increase the chance of finding the global minimum.

The final score is the s-stress score for this particular set of echoes. Now we need to compile an ordered list with scores of each combination of echoes. The coherent echo combinations will end up with a low score, and wrong combinations yield a high score. Sorting these from low to high will aggregate the image sources on the top of the list.
3.3. Room Reconstruction

From the echo labeling algorithm we obtain a list of image sources. Each image source belongs to a boundary that divides space in two half spaces, as we do not know where the reflective surface ends. The list contains both first-order and higher-order image sources. The latter we need to remove from this list. We do this by using the fact that higher-order image sources are “combinations” (explained in section 3.3.1) of lower-order ones[18]. We would like to end up with a list of only the first order image sources. For each source we can then draw a plane on which the wall should reside. The intersections all planes define the shape of the room. This section explains this process in more detail.

Figure 3.5: The wall resides exactly in the middle between the source and image source

3.3.1. Filter out higher order image sources

The location of a first order image source location can be computed as

\[ s'_i = s + 2(p_i - s, n_i)n_i \]  (3.24)

Where \( i \) corresponds with wall \( i \), \( n \) is the unit normal and \( p \) is any point on the wall. In words this is twice the distance from the true source through the wall along the normal unit vector.

In the same way the location of second order image sources can be expressed in terms of the first order image sources. In Figure 2.3, \( s''_{ij} \) is a combination of \( s'_i \) and \( s'_j \). When image source \( s'_i \) is mirrored in the (extended virtual) wall corresponding to image source \( s'_j \), the result is second order image source \( s'_{ij} \).

\[ s''_{ij} = s'_i + 2(p_j - s'_i, n_j)n_j \]  (3.25)

When we check if image source \( s''_{ij} \) is close to the result of mirroring \( s'_i \) in the wall of \( s'_j \) as

\[ ||s'_i + 2(p_j - s'_i, n_j)n_j - s''_{ij}|| < \epsilon \]  (3.26)

with \( \epsilon \) some small number, we know whether \( s''_{ij} \) is a higher order image source.
In this chapter we present a new approach to obtain the room shape from acoustic echoes. We assume that the room is empty and has a shoe box shape, i.e. 4 walls, a ceiling and a floor connected through right angles. Also we assume omni-directional speakers and microphones. First the room impulse responses are interpolated to improve precision of the echoes. The echoes are pre-filtered using properties of the Euclidean distance matrix. Using graph theory we then produce a list of candidate labeled echo sets. From this list we find the correct sets using a method which simultaneously locates (image) sources and microphones from the echo data by means of direct computation. Finally, the image sources reveal the room shape. Each of these steps is elaborated in the sections of this chapter.
4.1. Acquisition of Acoustic Echoes

For a shoe-box-shaped room we expect 6 first order echoes and the direct sound to be recorded by the microphones. Because we assume an empty room, we estimate the room impulse response by using the sinesweep method as described in Section 3.1.3.

For each RIR we need to find the peaks that correspond to the direct sound and the reflections, seven in total. The peaks tell us the time delay from the source to the microphone. Since the sampling rate for microphones usually is \(96000\, Hz\) or lower, we have to deal with limited resolution for this time delay. The error for the delay of a reflection is uniformly distributed within \([-T_s/2, T_s/2]\), with \(T_s\) the sample period. To increase the resolution we interpolate by fitting a one-dimensional smoothing spline to the data. This is illustrated in Figure 4.2. The original signal has a sample rate of 96000 Hz whereas the interpolated signal has an effective sample rate of up to \(100 \times 96\, kHz\). We can see that the shape of the peak becomes much more nuanced and that the time delay for this reflection becomes much more precise with respect to the originally sampled signal. The required precision depends on the application. In Section 5.1 the effect of interpolation is tested.

The direct sound is easily found by finding the maximum value in the impulse response signal. Optionally, this peak can be used for a matched filter to increase SNR so that detection of the reflection peaks in noise is easier. The filter will enhance all parts of the signal that have a similar shape to the first peak. Since the echoes are a delayed and attenuated version of the first peak, they will be amplified, thus making them easier to detect. Consequently, the 6 highest peaks after the direct sound most likely correspond to the first order reflections, see Figure 4.3. As we have \(M\) receivers, we will get \(M\) RIRs. From each RIR we collect 7 time delays in a vector

\[
\tau_m = [\tau_0 \quad \tau_1 \quad \tau_2 \quad \tau_3 \quad \tau_4 \quad \tau_5 \quad \tau_6],
\]

where \(m\) is the microphone index, and \(\tau_i\) are the time delays belonging to the echoes. The time delay is obtained by dividing the sample number corresponding to a peak by the sampling frequency. The first time delay \(\tau_0\) is the time the sound travels the distance of the direct path from source to receiver. The other time delays \(\tau_1 - \tau_6\) correspond to the reflections from the 6 walls.
4.1. Acquisition of Acoustic Echoes

4.1.1. Translation of Echoes to Distances

With the echoes we want to find the location of the image sources. Each peak in the room impulse response is either the time delay from the real source, or from an image source. These time delays, \( \tau_i \), are related to the distance from source to receiver by the speed of sound \( c \) as \( d_i = \tau_i \times c \). However, this is only true when the microphones and the sources are precisely synchronized. If this is not the case, there might be an offset in the impulse response and instead of \( d_i \) we have \( \tilde{d}_i \). For the localization we only need time difference of arrival data which is not affected by this offset. For the pre-filtering however we need exact time of arrival data.

To overcome this issue, we have to subtract this offset from the echo time delays as

\[
\bar{d}_i = d_i - \tilde{d}_i, \tag{4.1}
\]

in which \( \tilde{d}_m \) is the offset for each RIR. We can calculate this offset as

\[
\tilde{d}_m = d_o - ||\hat{x} - r_m||, \tag{4.2}
\]

with \( r_m \) the location of the microphone and \( d_o \) the distance corresponding to the first peak in the RIR and \( \hat{x} \) the estimated position vector for the source. Figure 4.4 shows this offset. The orange line in this figure is where the first peak should be according to the true distance between the source and microphone. For finding this offset we need to estimate the true location \( x \) of the source. Using a closed form solution for multilateration\([19]\), which uses time difference of arrival (TDOA) data and the known microphone positions, we find the source location in a least squares fashion. The next part explains the closed form solution in detail.

Consider \( M \) receivers at positions \( r_i \) and \( M \) time delays \( \tau_i \) which are the time stamps for when a signal from the source at unknown position \( x \) is received. Let us now define distances \( d_i = \tau_i \times c \), with \( c \) the speed of sound. With these definitions we can write the relation between source and receiver as

\[
||x - r_i|| = d_i.
\]

Let \( r_c \) be the first receiver to register a signal event from the source, thus \( ||x - r_c|| = d_c \). The TDOA between the receiver \( r_c \) and the other receivers \( r_i \) is \( (\tau_c - \tau_i) \). Knowing this, we can write

\[
||x - r_i|| = d_i - d_c + ||x - r_c||.
\]
This we can reduce to linear form in a couple of steps.

\[ ||\mathbf{x} - \mathbf{r}||^2 = (d_i - d_c)^2 + 2(d_i - d_c)||\mathbf{x} - \mathbf{r}_c|| + ||\mathbf{x} - \mathbf{r}_c||^2 \]

\[ -2||\mathbf{x} - \mathbf{r}_c|| = (d_i - d_c) + \frac{||\mathbf{x} - \mathbf{r}_c||^2 - ||\mathbf{x} - \mathbf{r}_i||^2}{(d_i - d_c)} \]  \hspace{1cm} (4.3)

Introducing another receiver \( \mathbf{r}_j \) gives us a similar equation

\[ -2||\mathbf{x} - \mathbf{r}_c|| = (d_j - d_c) + \frac{||\mathbf{x} - \mathbf{r}_c||^2 - ||\mathbf{x} - \mathbf{r}_j||^2}{(d_j - d_c)} , \]

which we can equate to Equation (4.4) yielding

\[ (d_i - d_c) + \frac{||\mathbf{x} - \mathbf{r}_c||^2 - ||\mathbf{x} - \mathbf{r}_i||^2}{(d_i - d_c)} = (d_j - d_c) + \frac{||\mathbf{x} - \mathbf{r}_c||^2 - ||\mathbf{x} - \mathbf{r}_j||^2}{(d_j - d_c)} . \]

Expanding the squares gives us a system of linear equations

\[ (d_j - d_c) + \frac{2(r_j^T - r_c^T)\mathbf{x} + r_c^T r_c - r_j^T r_j}{(d_j - d_c)} = (d_i - d_c) + \frac{2(r_i^T - r_c^T)\mathbf{x} + r_c^T r_c - r_i^T r_i}{(d_i - d_c)} , \]

that we can solve as \( a_{ij,c} \mathbf{x} = b_{ij,c} \) where

\[ a_{ij,c} = 2((d_j - d_c)(\mathbf{r}_i - \mathbf{r}_c) - (d_i - d_j)(\mathbf{r}_j - \mathbf{r}_c)), \]

\[ b_{ij,c} = (d_i - d_c)((d_j - d_c)^2 - r_j^T r_j) \]

\[ + ((d_i - d_c) - (d_j - d_c))r_c^T r_c \]

\[ + (d_j - d_c)(r_i^T r_i - (d_i - d_c)^2) . \]

Expressing this as \( \mathbf{A}\mathbf{x} = \mathbf{b} \), we can finally estimate \( \mathbf{x} \) using least squares as

\[ \hat{\mathbf{x}} = \mathbf{A}^+ \mathbf{b} \]
After applying the offset from Equation 4.2 we are left with \( M \) arrays of 7 distances \( d_i \), or an \( M \) by 7 matrix

\[
\tilde{\Delta} = \begin{bmatrix}
d_{11} & \ldots & d_{17} \\
\vdots & \ddots & \vdots \\
d_{M1} & \ldots & d_{M7}
\end{bmatrix}
\] (4.5)

containing all the distance data needed to compute the image source locations. The only problem is that the entries are unordered or unlabeled. Section 4.2 will present a solution to label these entries.

### 4.2. Echo Labeling

Having \( \tilde{\Delta} \), we arrive at the problem of echo labeling. Finding the sets of distances (here called echo combinations) which correspond to each image source. This is a combinatorial problem that grows exponentially with respect to the input size. If we have \( M \) microphones, we are looking for combinations of \( M \) echoes where each echo in one combination is from a different row in \( \tilde{\Delta} \). The number of possible combinations is \( 7^M \) as we have 7 distances for each of the \( M \) receivers. The first observation we can make is that the first echo for each microphone, the set \([d_{11}, d_{M1}]\), always corresponds to the direct sound and thus to the real source. This reduces the number of combinations to \( 6^M \) since we have 6 walls and therefore 6 image sources to identify. The second thing we can do to limit the size of all the distance combinations is to pre-filter the combinations using Theorem 2.

#### 4.2.1. Pre-filtering

Theorem 2 states that the rank of a Euclidean distance matrix (EDM) must have a rank less or equal to 5, given that we are working with 3 dimensional point sets. Let \( R \) be a 3D point set containing the position vectors of the \( M \) microphones, then we can construct an \( M \) by \( M \) EDM \( D \). Matrix \( D \) is then augmented as in equation 3.19 with an arbitrary echo combination from \( \tilde{\Delta} \), forming \( D_{\text{aug}} \).

\[
D_{\text{aug}} = \begin{bmatrix}
d_{r_1 r_1} & \ldots & d_{r_1 r_M} & d_{1[2-7]} \\
\vdots & \ddots & \vdots & \vdots \\
d_{r_M r_1} & \ldots & d_{r_M r_M} & d_{M[2-7]} \\
d_{1[2-7]} & \ldots & d_{M[2-7]} & 0
\end{bmatrix}
\] (4.6)

For every possible combination \([d_{1[2-7]}, \ldots, d_{M[2-7]}]\) we compute the rank of \( D_{\text{aug}} \) based on the method from [20]. Instead of using the default threshold we provide a milder threshold. The rank is determined by calculating the SVD singular values which are considered zero if they are below the threshold. The threshold makes sure that all the correct echo combinations get a rank less than 6, and that most of the wrong echo combinations get a rank of 6 or higher.

The echo combinations that result in a rank \(< 6\) are called candidate echo combinations \( c_i \), which are \( M \)-length vectors. We collect all candidate echo combinations in set \( C \) so that \( c_i \in C \) with \( i \) ranging from 1 to the number of candidate echo combinations. See Figure 4.5 for an overview of the pre-filtering process.

---

**Figure 4.5:** The pre-filter process checks the rank of the augmented EDM.
4.2.2. Subsets of $\mathcal{C}$

After pre-filtering we have obtained set $\mathcal{C}$ containing all the candidate echo sets. Within $\mathcal{C}$ we need to find a subset of candidate vectors that correctly describe the distances from source and image sources to the receivers. This subset contains 6 vectors $c_i \in \mathcal{C}$ that each contain the distances from one image source to all receivers. An important observation we can make here is that these vectors are very unlikely to have elements in common. A sample frequency $f_s = 16kHz$ and a speed of sound $c = 343m/s$ yield a distance of $2.14cm$ per clock tick. Interpolate this with a factor 100, and we have a sub-millimeter spatial resolution for the distances between sources and microphones. This means that we are looking for 6 vectors $c_i \in \mathcal{C}$ that have no elements in common.

For further illustration, we show a simple toy example with 3 microphones and 4 walls in Equation 4.7. Matrix $c_{\text{example}}$ is the collection of candidate echo combinations. Since there are 3 receivers, the candidate vectors have length 3. Because there are 4 walls, there are also 4 image sources and thus the subset should contain 4 candidate echo combinations. The numbers in $c_{\text{example}}$ are artificial and do not represent real distances.

$$c_{\text{example}} = \begin{bmatrix}
10 & 11 & 11 & 11 & 12 & 12 & 14 & 14 \\
20 & 20 & 21 & 22 & 22 & 21 & 24 & 24 \\
30 & 31 & 31 & 31 & 33 & 31 & 31 & 34
\end{bmatrix} \quad (4.7)$$

There are 8 candidate echo sets. Highlighted in blue are the echo sets which together do not have elements in common. The other columns which are colored black are candidate echo combinations that happened to pass the pre-filter but do not represent a correct combination of Euclidean distances. The blue columns comprise the subset we would like to find in an efficient way. There can be more than one set of 4 (or in the 3D shoe box room case, 6) columns with no elements in common. In that case we need to find all these subsets so we can later test which one is the correctly labeled one.

4.2.3. Maximum Cliques

From Section 4.2.2 we learned that we need to find subsets in $\mathcal{C}$ of a certain size with no duplicate elements within the subset. This section will describe how we can use graph theory[21] to solve this problem.

We define a graph as the pair $G = (V, E)$ in which $V$ is a set of vertices of nodes and $E$ is the set of edges that connect the nodes from $V$. A node can represent anything and edges are used to indicate pairwise relations. As such an edge is a 2-element subset of $V$. We will only focus on simple or undirected graphs, in which the nodes have equal weight and the edges have no direction. Graphs are usually illustrated by drawing the nodes as dots or circles and the edges as lines between the nodes. A small example is shown in Figure 4.6, which has 2 nodes, labeled 0 and 1 which are connected by edge ($0, 1$).

Figure 4.6: A graph with 2 nodes and 1 edge

Let us now model collection $\mathcal{C}$ as an undirected graph $G = (V, E)$. Let each candidate echo combination $c_i \in \mathcal{C}$ be a node in $V$. For every two candidates $c_i$ and $c_j$ that have one or more elements in common, we define an edge in $E$. As an example consider the first three echo candidates from
4.2. Echo Labeling

Equation 4.7

\[
\mathbf{C}_{\text{example}} = \begin{bmatrix}
    r_1 & \mathbf{c}_{\text{example}} \\
    r_2 & \mathbf{c}_{\text{example}} \\
    r_3 & \mathbf{c}_{\text{example}}
\end{bmatrix}.
\]  
(4.8)

In this example, candidate 0 has one element in common with candidate 1, and candidate 2 has 2 elements in common with candidate 1, whereas there are no common elements between candidates 0 and 2. Thus we define a node for each of the columns and we define edges \((0, 1)\) and \((1, 2)\). The resulting graph is shown in Figure 4.7.

Figure 4.7: Node 1 is has edges in common with node 0 and node 2.

The same we can do for the original matrix \(\mathbf{C}_{\text{example}}\), which results in the graph as shown in Figure 4.8.

\[
\mathbf{C}_{\text{example}} = \begin{bmatrix}
    r_1 & \mathbf{C}_{\text{example}} \\
    r_2 & \mathbf{C}_{\text{example}} \\
    r_3 & \mathbf{C}_{\text{example}}
\end{bmatrix}.
\]  
(4.9)

Figure 4.8: Representing echo combinations as nodes in a graph. The blue nodes correspond with the blue columns in \(\mathbf{C}_{\text{example}}\).

The subset we want to find, all echo combinations that do not have elements in common, do not share connections in the graph. In graph theory, such a set of nodes no two of which are adjacent, is called an independent set. The size of an independent set is the number of nodes. Hence we need to find independent sets of size 6, if we want to find the image sources of 6 walls.
There can be many independent sets in a graph. By definition, each subset of an independent set is also an independent set. A maximal independent set is a set such that adding any other node forces the set to have an edge and thus makes the set not qualify as an independent set anymore. When an independent set has the largest possible size in a graph, we call it a maximum independent set. Figure 4.9a shows a graph in which the maximum independent set is colored blue whereas Figure 4.9b shows another maximal independent set in the same graph.

Finding a maximum independent set is an NP-hard optimization problem, which means it is unlikely that there exists an efficient algorithm to solve the problem. It can however be solved more efficiently than the $O(n^2 2^n)$ given by a naive brute force method. For example [22] solves the problem in time $O(1.2108^n)$. Since there can be more than one maximum independent set in one graph, we need to find all maximum independent sets. One way to do this is by solving the maximal independent set listing problem. Listing all maximal independent sets also yields all maximum independent sets, since a maximum independent set is also a maximal independent set by definition.

Before we continue we need to introduce the complement of a graph. The complement graph $H$ of a graph $G$ is a graph with the same nodes as $G$ in such a way that nodes in $H$ are adjacent if and only if they are not adjacent in $G$. To transform $G$ into $H$, we need to fill in all missing edges in $G$ and remove all its original edges. So the complement only affects the edges of a graph. An example of a graph and its complement are shown in Figure 4.10. Recall that the blue set in graph $G$ is the maximum independent set. As we can see in Figure 4.10a, the same nodes in $H$ are fully connected. They form a clique. A clique is a complete subgraph. This means that every two distinct nodes in the clique are adjacent.
In the same way as for independent sets we can define a maximal clique as a clique that cannot be extended by adding another node which is adjacent. In other words, a maximal clique is a clique that does not exist as a subset of another clique. The maximum clique is a clique such that there is no larger clique. So the blue clique in Figure 4.10b is the maximum clique. The maximum independent set in graph $G$ corresponds to the maximum clique in graph $H$. Since we need to find all maximal independent sets in a graph, we can also find all maximal cliques in the complement graph.

The maximal clique listing problem is also NP-hard. It has been shown that the worst case time complexity is $O(3^{n/3})$ for an $n$ vertex graph [23]. We will be using the `find_cliques()` function of the NetworkX [24] Python language software package. This implementation is based on the algorithm presented by [25] and the adaptions by [23] and [26]. This function returns a list of all maximal cliques in a graph. From this list we then select all maximum cliques and collect them in set $\mathcal{E}$.

Recall that each clique $\epsilon \in \mathcal{E}$ is a set of candidate echo combinations. Ideally we only find one maximum clique. However when we find more, only one of the cliques is the set of echoes that describe the distances from each image source to the microphones in the correct way, i.e. it is the correctly labeled set of echo combinations.

### 4.3. Localization of Image Sources

Set $\mathcal{E}$ may contain both correct and incorrect sets of echo combinations. For each measurement with one source, only one set can be the correctly labeled one. So we still need to determine which set is the correct one. A fast way to do this is by using the method proposed by Pollefeys [27]. This method is
able to compute the location of both microphones and sources directly from TDOA data using a rank-5 factorization of the input data. The method is fully explained in Appendix A.

The Pollefeys method gives us the coordinates of the receivers and sources given that the input data is a properly labeled set of distance data. If the data is not labeled correctly, the estimated coordinates will be completely wrong. However, the coordinates of the microphones are known. So we can compare the estimated microphone coordinates with the known coordinates to find out whether the input data was correct. If the receiver coordinates turn out to be a match, the (image) source coordinates will be correct as well.

From Section 4.2 we obtained a set of candidate echo sets $\mathcal{E}$. For each source, there is one set that is correct. The Pollefeys method needs at least 10 sources and 5 microphones. Using the image source method, we can consider image sources sources as well. Hence we need to do at least 2 measurements, which yields 2 real sources and 12 image source for a total of $14 \times 5$ distance data. We can construct the input data for the Pollefeys method as

$$\Delta = [S \ E_1 \ E_2],$$

in which $S$ contains the distances from real sources to the microphones, whereas $E_1, E_2 \in \mathcal{E}$. For each combination of 2 sets from $\mathcal{E}$ we compose a $\Delta$ and estimate the receivers and source coordinates. Even though we use only 2 sets from $\mathcal{E}$ at once, we can put more than 2 echo sets in $S$. For $N$ sources we can include $N$ columns of distance information in $S$:

$$S = [S_1 \ldots S_N].$$

After calculating $r_{est}$ and $s_{est}$, the estimations of the receiver and source coordinates, we end up with a list of these estimations together with an error value that is calculated as

$$\text{error} = \frac{||r_{est} - r||_F}{\sqrt{M}},$$

where $|| \cdot ||_F$ is the Frobenius norm, $r$ is a $M$ by 3 matrix containing the known microphone positions, $r_{est}$ are the estimated coordinates and $M$ is the number of microphones. This gives us the average estimation error per receiver in meters. As such, the estimation with the lowest error also yields the best reconstruction of the image sources.

4.4. Overview

Figure 4.13 shows an overview of the most important part of the proposed system, that has been described in Section 4.2 and Section 4.3. The input for this subsystem are the estimated room impulse
responses, called *Measurement data* in the overview. The output consists of the best available estimates of the image source and source locations, here called *Image data*. This output data can then be used to reconstruct the room shape.

### 4.5. Room Reconstruction

There are many ways to reconstruct room geometry from image sources. The method described in this section first finds points on the wall, and then tries to fit the walls through those points. The process starts with a room setup as illustrated in Figure 4.14a. In this case we are using 5 microphones and 6 sources. The image sources and sources are estimated, and the result is shown in Figure 4.14b. We
would expect the estimated sources to be at the same position as the true sources, however sometimes the estimated results are flipped or mirrored. As we will see, this is not an issue as the image sources are affected by the same transformation.

Figure 4.14

(a) Initial room setup
(b) Estimated sources and image sources

From the image sources we can reconstruct the walls. The result for one \( \Delta \) input dataset yields two image sources per wall. But if we use the image sources of multiple results, which all have an error lower than some threshold, we will have more data to reconstruct the walls. Each set of (6) image sources belongs to one real source. Since each wall is located directly between the real source and an image source, we can calculate the midpoints, which are the points on the wall, as

\[ p_{\text{wall}i} = \frac{s_{\text{est},i} + s}{2} \]

with \( s_{\text{est},i} \) the position of image source \( i \) and \( s \) the corresponding source. The wall points are depicted in Figure 4.15a. Apart from the wall points \( p_{\text{wall}i} \), we can also calculate the normal vector for each wall point as

\[ n_{\text{wall}i} = s - s_{\text{est},i}. \]

The normals are used to cluster the wall points, so that the wall points on the same wall are together in one cluster. The normals of the wall points in each cluster are parallel to each other. Consequently, for each cluster of wall points the wall can be estimated by fitting a plane through these points. In Figure 4.15b the walls are estimated in 2D, so instead of a plane a line is fitted through these points in a least squares fashion. The vertices of the room are then simply found by finding the intersections between these lines.

Figure 4.15: Wall points from (a) result in the least square estimates of the walls in (b).
Experiments and Results

This chapter describes the experiments done to quantify the characteristics and performance of the graph-based method proposed in Chapter 4.

The experiments are done with simulated data. A room is defined, and in that room, microphones and sources are placed at random locations. With this information as input, for each receiver-source pair a room impulse response is simulated using room acoustics simulation software for Matlab by [28]. The experiments are run on a MacBook Pro Mid 2012, 2.3 GHz Core i7 processor in Python 3.4.3.

The following symbols will be used throughout this chapter:

$N$  Number of sources  
$M$  Number of receivers  
$L$  Upsampling rate (Interpolation)  
$F_s$  Sampling frequency

Unless otherwise noted, the simulated rooms have dimensions of $8 m \times 7 m \times 5 m$ yielding a volume of $280 m^3$.

The precision of the graph-based algorithm is measured by the room estimation error. This is the distance in meters from the estimated 2D room vertices (as shown in Figure 4.15b) to the true vertices. The computation time is also measured. This is the time from the moment the method receives its input until the vertices are calculated.

5.1. Sampling Rate and Interpolation

In this section we try to get insight in the effect of the sampling rate and interpolation of the room impulse responses. A higher sampling rate should obviously result in a higher resolution for the measured time delays of the received echoes. In addition to varying the sampling rate we also apply interpolation with different upsampling rates on the signals. The room impulse responses with sampling frequencies lower than $48 \ kHz$ are simulated with the Room Impulse Response Generator software by [29] as [28] cannot go lower than $44.1 \ kHz$. For this experiment $N = 8$ and $M = 5$. 
5. Experiments and Results

(a) Sample Frequency and interpolation vs room shape estimation error.

(b) Upsampling versus computation time.

Figure 5.1: A higher upsampling rate decreases the error while the computation time increases linearly.

Figure 5.1a shows us that indeed the estimation error decreases if we use a higher sampling rate. Interpolate of the RIR decreases the estimation error as well, but not to the extend of sampling with a higher frequency. For example, sampling with a sample rate that is 10 times higher produces better results than upsampling 10 times. The interesting thing is that 100 times upsampling does not improve up on the 10 times upsampling. For sampling frequencies lower than 16 kHz the results were not reliable enough to be included in the figure.

In figure 5.1b the effect of interpolation on the computation time is illustrated. The computation time of the upsampling procedure itself increases linearly with the upsampling rate. The echo labeling part of the algorithm is only slightly affected by a higher sampling rate as there are simply more samples to process. The little bump from 32 kHz to 48 kHz is due to the different simulation software used for generating the data for the lower and higher frequency parts.

From these results the recommendation is to use a upsampling rate of 10× when possible as it significantly increases accuracy while the impact on the computation time is minimal. The minimum sampling rate is 16 kHz, however when the application demands sub-centimeter precision a sampling rate of 96 kHz is recommended.

5.2. Number of Measurements

One of the variables we can vary is the number of measurements we do to estimate the shape of the room. Doing a measurement entails placing the source on a random place, exciting a signal and estimating the room impulse response for every microphone.

From Section 4.3 we know that we need at least 2 measurements to have enough data to do the localization. The Pollefeys method needs at least 10 sources and 5 receivers. The method is most accurate when these sources and receivers are placed randomly to minimize the correlation between measured impulse responses. Two measurements yields 14 sources, however, only 2 of them are truly random since the image sources all depend on the source location. Recall that the data we feed to the Pollefeys method is constructed as $Δ = [S \ E_1 \ E_2]$. The $S$ submatrix comprises the distance data for the direct sound, which belong to the real sources. As we know that these are already labeled correctly we can add any number of columns with data from extra measurements to $S$, effectively adding more random data points. Aside from having more sources, we also have more measurement data and thus more labeled data to chose from when we construct $Δ$.

Here we test what the effect is of using more sources on the accuracy of the resulting estimated locations of image sources. We start with the minimum amount of 2 sources, and we increase that amount to 13. Figure 5.2 shows these results. For this experiment $M = 5, F_s = 96 kHz$ and $L = 10$. We measure the performance in relative estimation error. The error for $N = 2$ is the reference and set to 1. For $N > 2$ the errors are relative to this error value. The same method is used for the computation time, where the the time for $N = 2$ is set to 1 again.
Adding sources, including more measurement data, decreases the estimation error as we feed more less-correlated data to the Pollefeys algorithm. This decrease is most apparent when we go from 3 to 4 measurements. The computation time increases more or less linearly. For \( N = 4 \) the error is more than 5 times lower than for \( N = 2 \) while the computation time is about 2.5 times longer. Hence the recommended amount of sources is 4-5.

### 5.3. Number of Receivers

Just like we can vary the number of sources, we can also increase the number of receivers. The minimum number of microphones is 5, so that is where we start, and we move up from there. Figure 5.3 shows the error and computation time for this experiment. For this experiment \( N = 5, F_c = 96 \text{ kHz} \) and \( L = 10 \).

Adding receivers does improve the estimation accuracy (Figure 5.3a), but we can see that the cost of doing so is very high (Figure 5.3b). Using 7 receivers in stead of 5 increases the computation time by a factor 55. This is also the reason why more than 7 receivers is not even feasible to test. The increase in computation time can be attributed to the pre-filtering and echo labeling. With 7 receivers we get an 8 by 8 Euclidean distance matrix to test the rank on, and on top of that, the number of echo combinations the pre-filter has to handle increases exponentially. As such the recommended number of receivers to use at the same time is 5.
5.4. Graph-based Method vs MDS-based Method

This section compares the performance of the MDS-based method from [8] to the graph-based method presented in this thesis. It is difficult to compare these two methods on exactly the same criteria as they work differently. In the ideal case, the MDS-based method only needs 1 measurement as it can find image sources for each wall using only one source. As we know, the graph-based method needs a minimum of 2 sources. The MDS-based method outputs a list of image source coordinates together with their scores. If we are looking for 6 image sources, we should get the 6 images with the best (lowest) scores. However in practice the best 6 results do not always correspond to the 6 true image sources. Sometimes these results contain duplicates when two slightly different echo sets converge to the same image source location. And sometimes the MDS algorithm converges to local minima that do not correspond with actual image sources. So to quantify these results without using a lot of heuristics to improve the results, we will be counting the images that are correctly found and we compute the estimation error for these images as well as the computation time for each set of measurement data (1 set contains the echo data of $M$ receivers and 1 source). To speed up this algorithm we will use the same pre-filtering method as the graph-based method. This can also cause images to be missing from the results when the corresponding echo set did not pass the pre-filter.

For the graph-based method we cannot count the number of sources easily as we need at least two sets of echo data. From the echo candidates $C$ we will only get the cliques of size 6. If the size is less than 6, we discard the entire set of echo data. As such it can happen that there is no result at all, when none of the sources produce labeled distance data. So for this method we will count the datasets that produced a result and compute the image estimation error as well as the computation time for the input data that produce results.

The configuration of the graph-based method is based on the recommendations of the previous sections in this chapter. The number of sources $N = 5$, number of receivers $M = 5$, sample rate $F_s = 96kHz$ and upsampling rate $L = 10$. For the MDS-based method we use the same parameters.

![Figure 5.4: Comparing MDS-based method to Graph-based method for (a) computation time and (b) estimation error](image)

Figure 5.4a shows the computation time for both methods. As we increase the pre-filter threshold value, both methods increase linearly, however the MDS-based has a much steeper slope. A higher threshold results in more echo candidates in $C$, for which the MDS-based method has to do its optimization routine for each $c \in C$. In contrast, the graph-based method has to add more nodes to the graph which results in less increase in computation time.

The estimation errors are compared in Figure 5.4b. The error for the MDS-based method increases for higher thresholds. This can be explained by the fact that more echo candidates with lower precision comprise the input data. This results in more image sources found as can be seen in Figure 5.5a but also in lower average precision. The graph-based method has more or less constant error. That the error is higher for a threshold of 0.05 is because of a design decision in the implementation: When the size of $C$ is too large, the function `find_cliques()` is skipped. This is a trade-off for computation speed. As we can see in Figure 5.5b, the dataset success rate is lower for this threshold value. This means that only about 75% of the datasets have produced usable data, which gives less combinations
5.5. Recommended Configuration

Previous sections in this chapter recommended the parameters from Table 5.1 as a good trade-off between precision and computational costs.

\[ N = 5 \]
\[ M = 5 \]
\[ L = 10 \]
\[ F_s = 96000 \text{ Hz} \]

Table 5.1: Optimal parameters for graph-based method

Running the graph-based algorithm for simulated rooms with volumes of \(120 - 500 m^3\) with the recommended parameters gives us the results shown in Table 5.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean error</td>
<td>0.0235 m</td>
</tr>
<tr>
<td>Error variance</td>
<td>(2.21 \times 10^{-3})</td>
</tr>
<tr>
<td>Minimum error</td>
<td>(1.08 \times 10^{-3}) m</td>
</tr>
<tr>
<td>Average time</td>
<td>2.43 s</td>
</tr>
<tr>
<td>Time variance</td>
<td>0.51</td>
</tr>
<tr>
<td>Minimum time</td>
<td>1.35 s</td>
</tr>
</tbody>
</table>

Table 5.2: Results for estimating room shape with the graph-based method

This means that it is possible to estimate the walls of a room with a precision of 2.35 cm for each end of the wall in 2.43 s. This of course excludes the time it takes to record the measurement data from 5 source locations. To obtain these statistics, 600 simulations were done. A total of 509 simulations ended in a successful estimation of the room shape. The other 81 simulations failed to produce a result. This can be explained with several reasons. The source and receiver locations are randomly generated, when it happens to be the case that the either the receivers or sources are placed more or less along a line, the measurement data will be more correlated, making it harder to find the image sources. Another
reason might be that the prefilter threshold was too strict so that no 6 echo combinations with all unique elements could be found. This can be mitigated by also allowing sets of echo combinations with sizes smaller than 6 to be fed to the Pollefeys algorithm. By doing so, the results of one Pollefeys estimation might miss out on some image sources, however by combining the output of multiple estimations we can still find multiple image sources per wall. This would make the algorithm more robust at a cost of additional computational complexity.
Conclusions

Estimation of room shape from room impulse responses is quite new and not a lot of research has been done yet in this field. The method presented in this thesis is able to estimate the shape of a room in fast and robust way for rooms that have a shoe-box shape. Interpolation of the room impulse responses with a factor 10 has been proven to be a cheap and effective way to increase the estimation accuracy. We have shown that echo labeling using graph theory is feasible on contemporary hardware. The pre-filter makes the computational burden less severe, but we have to be careful not to filter to strictly as false positives during filtering may cause loss of important information. Reliability establishes itself in that we can verify whether or not our results are correct by comparing estimated microphone locations to the well-known microphone locations. In other words, we know when the results are correct. The best trade-off between precision and computational complexity is reached by using 5 microphones and 4-5 sources. More precision can be achieved by adding more sources. But if we add too many sources, we might mask wrong echo combinations because too many correct ones could average out the wrong ones. In 600 simulated experiments we were able to estimate the vertices of the rooms with an average error of $2.35\, cm$ in $2.43\, s$.

6.1. Recommendations

The next step would be to extent the application to rooms with more walls or reflective surfaces. Then the acquisition of the acoustic echoes becomes more critical as having more echoes will heavily affect the computational complexity. So it is important that the amount of false echoes will be limited.

The algorithm could be made more robust by having iteratively adaptive thresholds at a price of additional computational costs.

To make the system more user friendly and practical, it would be a huge improvement if some type of auto calibration algorithm is used to determine the microphone positions automatically. We do not need to know the absolute receiver locations, the pairwise distances are enough. The Pollefeys method may possibly be used for this provided that at least 10 sources are used, however the problem is that the returned receiver positions may be scaled, rotated or flipped.

Another improvement to make the system more practical and faster is to replace the room impulse responses with simple clapping, clicking or popping sounds. As long as the first order reflections are still detectable it is still okay.

Finding the shape of a non-convex room should also be possible using the method from this thesis. By moving around the source, and saving the image source-source pairs, the midpoints describe the outline of the walls, whether they are convex or not.
The Pollefeys method [27] is based on a rank-5 factorization. It needs at least 5 sources and 10 microphones. In a shoe box shaped room, for every source we have 6 image sources, which for this method we can also consider as sources. Therefore, if we use 2 real sources (or do two measurements with one source at different locations) we already have 14 sources as input for this method. Here we will describe the slightly adapted Pollefeys method for at least 5 microphones and 10 sources.

Consider $M$ receivers indexed by $i$, represented as $r_i = [x_i \ y_i \ z_i]^T$ and $N$ sources index by $j$ as $s_j = [x_j \ y_j \ z_j]^T$ where $x$, $y$ and $z$ are spatial coordinates. Using the distance $d_{ij} = ||r_i - s_j||$ between a receiver and source, we can write

$$(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 = d_{ij}^2.$$  \hfill (A.1)

This can be expanded into

$$R_i^T S_j = d_{ij}^2$$  \hfill (A.2)

with

$$R_i = [r_i^T \ r_i -2x_i \ -2y_i \ -2z_i \ 1]^T,$$

$$S_j = [1 \ x_j \ y_j \ z_j \ s_j^T s_j]^T.$$  \hfill (A.3) \hfill (A.4)

We collect all $d_{ij}$ distances into $D$ which is a $M$ by $N$ matrix. From $D$, it is our goal to recover the $R$ and $S$ matrices, which are comprised of the columns $R_i$ and $S_j$ respectively. We start by computing the singular value decomposition (SVD) of $D$

$$U \Sigma V^T = D.$$  \hfill (A.5)

If we then write $\hat{R} = U^T$ and $\hat{S} = \Sigma V^T$, we have $\hat{R}^T \hat{S} = D$, in which $\hat{R}$ and $\hat{S}$ are related to $R$ and $S$ by a transformation matrix $H$ as

$$R^T S = \hat{R}^T H^{-1} \hat{S}.$$  

Now the challenge is to construct $H$ in such a way that the structure of $R$ and $S$ as shown in Equation A.3 and A.4 will be enforced. First we split up $H$ in three matrices as $H = H_Q H_R H_S$. Transformation matrix $H_S$ will ensure that the first row of $S$ will be all ones. We construct it as

$$H_S = \begin{bmatrix} h_S^T \\ 0 \\ 1 \end{bmatrix},$$

where $h_S^T$ can be found by solving the system $h_S^T \hat{S} = 1$. In a similar way we can enforce the last row of $R$ to be ones, by defining

$$H_R = \begin{bmatrix} I \\ 0 \\ h_R \end{bmatrix}^{-1}.$$
This time we find \( \hat{h}_R \) by solving \( \hat{R}^T H_S^{-1} h_R = 1 \). The lasts constraints are quadratic, and they are enforced by \( H_Q \). We can write the constraints on \( S_j \) as

\[
S_j^T B M_j = 0 \text{ with } B = \begin{bmatrix}
0 & 0 & 0 & 0 & -\frac{1}{2} \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
-\frac{1}{2} & 0 & 0 & 0 & 0
\end{bmatrix}.
\]

And therefore \( \hat{S}_j^T H^T B H \hat{S}_j = 0 \).

We can now define \( \hat{S}_j = H_Q H_S \hat{S}_j \) and \( Q = H_Q^T B H_Q \) so that we obtain the following linear equation for determining the coefficients of \( Q \):

\[
\hat{S}_j^T Q \hat{S}_j = 0. \tag{A.6}
\]

The first row of \( \hat{S}_j \) and the last row of \( \hat{R}_i \) should be preserved, therefore we can impose

\[
H_Q = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
. & . & . & 0 & 0 \\
. & . & . & 0 & 0 \\
. & . & . & 0 & 1 \\
. & . & . & . & .
\end{bmatrix}
\text{ and } Q = \begin{bmatrix}
. & . & . & -\frac{1}{2} \\
. & . & . & 0 \\
. & . & . & 0 \\
-\frac{1}{2} & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}.
\]

Since \( Q \) is symmetric, it still has 10 degrees of freedom. Provided we have at least 10 sources, we are able to compute these 10 coefficients linearly using Equation A.6. Then we can construct \( H_Q \) using the entries of \( Q \) as

\[
H_Q = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 0 & K & 0 & 0 \\
0 & -Q_{11} & -2Q_{12} & -2Q_{13} & -2Q_{14}
\end{bmatrix}
\]

with \( K \) the Cholesky factorization of the middle 3 × 3 part of \( Q \). Now we can find

\[
R = (\hat{R}^T H_S^{-1} H_R^{-1} H_Q^{-1})^T \]

\[
S = H_Q H_R H_S \hat{S}
\]

from which we can extract the estimated coordinates of the receivers and (image) sources.


