Jens Cappelen Andresen

Microphone and source localization in rooms using LiDAR-SLAM

Master's thesis in Acoustics
Supervisor: Peter Svensson
July 2020

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems

NTNU
Norwegian University of
Science and Technology

Jens Cappelen Andresen

Microphone and Source Localization In Rooms Using LiDAR-SLAM

Supervisor:
Peter Svensson

Master's thesis in Electronic systems design - Acoustics

Norwegian University of Science and Technology
Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Electronics and Telecommunications

July 6, 2020

Abstract

If an automated measuring system capable of measuring thousands of impulse responses in rooms is to be created a reliable way of localizing the microphones is needed. This thesis investigates if the LiDAR-SLAM technique for localizing the measurement microphones is the way to move forward with this system. It also looks into what some of the possible applications of such a system could be.

Using a version of this system 48 impulse responses were measured in different positions in a room. By identifying different wavefronts combined with time difference of arrival the position of the source and the first-order image sources could be estimated. This project uses microphones in one height resulting in 2D localization of the source. With an expansion of the system 3D localization will be possible.

Along with the results from the measurements a section discussing what changes should be made to the current system, and what further work should be done to make the dream of an automated measurement robot into a reality.

Sammendrag

Hvis et automatisk målesystem som er kapabelt til å måle tusenvis av impulsresponser i rom skal bli utviklet må en pålitelig teknikk for lokalisering av mikrofonene utvikles. Denne masteroppgaven undersøker om bruk av LiDAR-SLAM for lokalisering kan være veien å gå for et slikt målesystem. Noen av de mulige bruksområdene for et slikt system blir også undersøkt.

En versjon av dette systemet ble brukt til å måle 48 impulsresponser i forskjellige posisjoner i et rom. Identifikasjon av ulike bølgefronter kombinert med forskjellene i gangtiden ble brukt til å estimere posisjonen til kilden og førsteordens speilkilder. I dette prosjektet er mikrofonene plassert i én høyde som resulterer i lokalisering av kilden i 2D. Ved en potensiell utvidelse av systemet vil også 3D-lokalisering være mulig.

I tillegg til resultatene fra forsøkene kommer en del hvor mulige endringer til det nåværende system diskuteres, og hva som kan jobbes videre med for å gjøre drømmen om en automatisert målerobot til virkelighet.

Contents

1 Introduction

1.1 Motivation

Measurements are a key part of any scientific research. They are used to back up theories and supporting conclusions. Room acoustical surveys are often limited by examining a relatively small number of positions. Getting the wanted coverage of measurements can be very time-consuming in many cases. In specialized acoustic research like wavefield imaging, it may be necessary to cover large areas in detail to satisfy the Nyquist-Shannon sampling theorem in space [1]. Different solutions for 3D localization systems have been purposed both for indoor and outdoor applications [2][3]. These methods use microphone arrays combined with time difference of arrival analysis to estimate the source position.

To simplify the measurement process an automated system should be developed. There are several different ways of making such a system, but a solution that requires little equipment and has a short setup time is using a LiDAR-based robot combined with a microphone array. The long term goal is to develop a system that is capable of measuring thousands of impulse responses in a room automatically. Using a combination of LiDAR (Light Detection and Range) and SLAM (Simultaneous location and mapping) is a common solution in the field of robotics and should be taken advantage of also in acoustics.

1.2 Objective

This thesis is a continuation of a pilot project where the precision and trueness of the LiDAR-SLAM technique was evaluated, and it was deemed good enough to be continued. To further evaluate what can be done using this method, and what possibilities such an automated measurement system would give, an expanded version of the pilot project was initiated. The same LiDAR will be used to localize microphone positions and from that the arrival times from impulse response measurements will be used to localize the source and first order-image sources. This project uses microphones placed at the same height, thereby giving this iteration of the system the ability to estimate positions in 2D. This can quite easily be expanded to work in 3D using well known methods that, among other things, are used in GPS technology [4].

1.3 Structure

Chapter 2 of this thesis will cover the basic principles of SLAM, the geometric transformation used, a technique for finding source positions, and how image source positions can be calculated. Further, chapter 3 describes the measurement setup and the LiDAR that was used. Also a step by step description of the post-processing in MATLAB. Chapter 4 presents the results from the source estimation, and for the first-order image source estimations. In chapter 5 the results will be discussed along with possible changes that could be made to improve them. There is also a section with suggestions for how this technique can be further developed. Lastly, chapter 6 contains the concluding remarks, summarising the most significant findings of the project.

2 Theory

This chapter presents the theory used in the measurements and analysis of the study. Since a SLAM algorithm is implemented and is used for localization in this project a brief look at how it works will be presented. Further, the geometrical transformation that was used, and lastly the technique for finding source and image source positions are presented.

2.1 SLAM

Simultaneous localization and mapping (SLAM) is the problem of concurrently estimating in real-time the structure of the surrounding world (the map), perceived by moving exteroceptive sensors, while simultaneously getting localized in it [5]. For this project, the SLAM algorithm that was already implemented in the Navigation Toolbox for MATLAB was used. The algorithm tries to optimize the position, $\xi = (\xi_x, \xi_y, \xi_\theta)$. It consists of a (x, y) translation and a rotation, ξ_θ of the LiDAR observations, these are called *scans*. In this project a *scan* refers to a set of distance measurements to all reflecting surfaces within range, in ≈ 1100 directions around the unknown position. Each consecutive scan is matched against the previous few scans using a non-linear optimization which creates what is called a *submap*. Submap construction is the iterative process of aligning scan and submap coordinate frames [6]. The submaps are stored together with the positions ξ and are compared with the world map. For large rooms or if a whole floor is to be mapped the algorithm will attempt loop closure to connect the submaps and align them properly.

2.2 Reference frames and geometry

To compare the coordinates from the output of the SLAM algorithm with the ground truth a translation to the world frame, $\mathcal{W}$ is needed. $\mathcal{F}$ is a Cartesian frame defined with respect to the world frame by a translation vector $\mathbf{t}$ and a rotation angle θ.

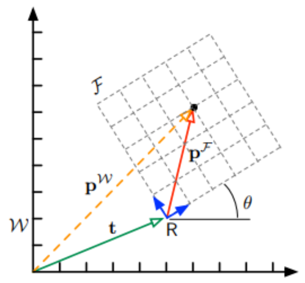


Figure 2.1: The axis marked $\mathcal{W}$ is the world frame, and $\mathcal{F}$ is a Cartesian frame. The figure shows the general principle behind frame transformation in the 2D plane. Figure from [5].

A point in space $\mathbf{p} = [x; y]$ can be expressed in the world frame by a frame-transformation equation.

$$\mathbf{p}^{\mathcal{W}} = \mathrm{R}\mathbf{p}^{\mathcal{F}} + \mathbf{t} \tag{2.1}$$

where R is the rotation matrix associated with the angle θ. R is defined as

$$\mathrm{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

2.3 Locating the source position from impulse responses

The problem of finding the source position from impulse response measurements can be solved by using the arrival times [7]. Assuming there is microphones placed in known positions $\mathbf{x}_i = [x_i, y_i]$, and a source is placed at an unknown position $\mathbf{x}_S = [x_S, y_S]$. The arrival times are related to the source and receiver positions by the following relationship

$$t_i = \frac{1}{c}\sqrt{(x_i - x_S)^2 + (y_i - y_S)^2} \tag{2.2}$$

where c is the speed of sound. If the microphones are placed along $y = 0$ and let the arrival time t_i be a function of x_i equation 2.2 becomes

$$t_i\left(x_i\right) = \frac{1}{c}\sqrt{(x_i - x_S)^2 + y_S^2} \tag{2.3}$$

When plotted $t(x)$ results in a smooth curve with a minimum at $x_i = x_S$. If squared $t(x)$ results in equation 2.3 becoming a quadratic function

$$t_i{}^2\left(x\right) = \frac{1}{c^2}\left[\,(x_i - x_S)^2 + y_S^2\right] = ax^2 + bx + c \tag{2.4}$$

It is then possible to fit a polynomial to the data in a least-square sense, using the known microphone positions x_i and the calculated arrival times squared. Then the minimum of the new polynomial can be found to give an estimate for x_S. This in turn can be used to find an estimate for y_S since the coefficients a, b, c in equation 2.4 can give the minimum value of t^2 for $x = x_S$

$$t_{min}^2 = \frac{1}{c^2}y_S^2 \tag{2.5}$$

Note that this technique can only be used when the microphones are placed in a line, e.i. when only one of the coordinates change. If both coordinates change the localization problem becomes similar to the technique used in modern GPS technology [4].

2.4 First-order image source positions

To locate a first-order image source first the reflection plane must be found. The reflection plane can be defined by the general plane equation $Ax + By + Cz + D = 0$, this can then be normalized such that $A^2 + B^2 + C^2 = 1$. The image source position for a specified source position, $\mathbf{x}_S$, is then given by the following equation [8].

$$\mathbf{x}_{IS} = \mathbf{x}_S - 2t \cdot \mathbf{n} \tag{2.6}$$

where $\mathbf{n}$ is the normal vector to the plane, with normalized plane coefficients the normal vector becomes $\mathbf{n} = [A, B, C]$, and t is the perpendicular distance from the source to the plane,

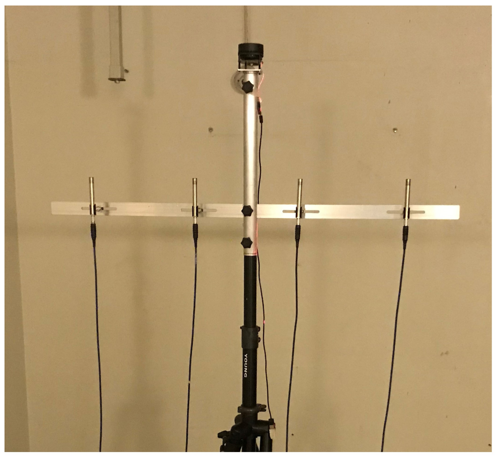
$$t = \mathbf{n} \cdot \mathbf{x}_S + D \tag{2.7}$$

3 Method and Equipment

All data collection for this project was done in a room in the basement of the acoustics lab at NTNU. This room had to be used because the radiation protection coordinator could not confirm that the LiDAR unit was a Class I laser, and deemed it potentially harmful to people. This chapter will present how the measurements were done, the equipment that was used, and give a step-by-step walk-through of the post-processing work in MATLAB. The MATLAB scripts that are used for the post-processing can be seen in appendix C.

3.1 Measurement setup

All the measurements were done in a reverberation chamber in the basement of the acoustics lab at NTNU. The room is 5.874 m by 4.886 m and was mostly empty except for a loudspeaker in one corner and a small table. To make the post-processing easier with regards to estimating the microphone locations the plan was to measure many impulse responses along a line parallel to the shortest walls. A custom made microphone stand was constructed with mounts for four microphones and at the top of the stand a mount for the LiDAR unit that aligned the center of the LiDAR with the center of the microphone stand. The microphones were placed 30 cm apart. To start the measurements a line was marked on the floor using a line laser tool and 12 points each $\approx$ 20 cm apart were marked along this line. The endpoints of this line were measured with a laser measuring tool (Leica DISTO X310) to act as reference points that are used in the post-processing. The microphone stand along with a schematic of the room can be seen in figure 3.1.



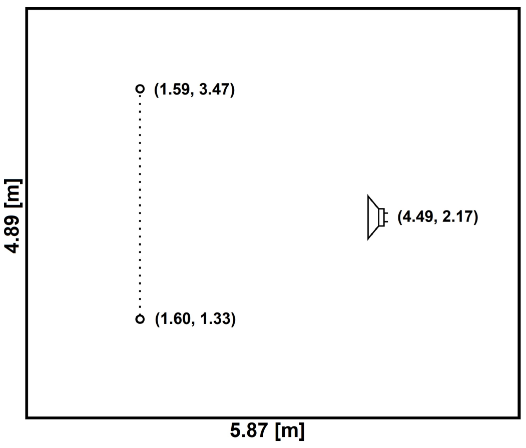


Figure 3.1: Left: Microphone stand with four microphones and the LiDAR mounted on top. Right: Schematic of the room with measured end points and loudspeaker position.

The microphones were connected to an external soundcard via a microphone amplifier and into a PC with EASERA software. A cylindrical loudspeaker with a well defined acoustic center, connected to the PC via an amplifier and the soundcard, was placed on the floor at a randomly selected position at the same height as the microphones, and reference measurements for the loudspeaker position was done. Then the microphone stand was placed in the first position and a multichannel FFT in EASERA was used to measure the impulse responses for all four microphones simultaneously. The output signal to the loudspeaker was a logarithmic sweep ranging from 125 to 20 000 Hz. After the impulse response was measured, the LiDAR unit was used to record nine scans of the room using RoboStudio software in the current position. Then the microphone stand was moved along the line and the same measurement process was repeated for all the marked positions. When moving the microphone stand a line laser tool to check that it was not rotated. This resulted in 48 measured impulse responses and 108 LiDAR-scans to work with in MATLAB. In addition, the temperature was logged before and after doing the measurements. This was done so the correct sound speed could be calculated.

3.2 RPLIDAR A1M8

The RPLIDAR A1M8 was the LiDAR used in this project. It is a low-cost 360° 2D laser scanner (LiDAR) solution developed by SLAMTEC. The LiDAR can perform 360° scans with a maximum range of 12 m. It is based on the laser triangulation ranging principle. The LiDAR emits a modulated infrared laser signal that is then reflected by an object. The reflected signal is sampled by a vision acquisition system in the RPLIDAR A1M8 and is processed to output a distance and an angle value. The datasheet states that the error in distance measurement is less than 1% of the distance.

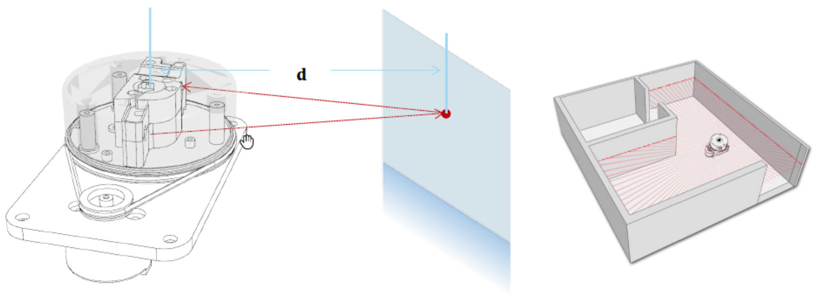


Figure 3.2: Schematic of the RPLIDAR A1 and how it works in a room. [9]

RPLIDAR A1M8 uses a low power (< 5 mW) infrared laser as its light source, and drives it using modulated pulse. The manual states that the laser is a Class I laser. An attempt to test the strength of the laser was done without success. Due to an interlock in the software of the laser, it would not emit if the laser unit was not rotating, this in turn made it difficult to measure its true strength.

The LiDAR uses 3.3V-TTL serial port (UART) as its communication interface and is connected to a PC using a UART to USB bridge. SLAMTEC provides software called RoboStudio, this software gives easy access to data from the LiDAR. The data was then exported to txt-files containing both distance and angle values that could then be imported to MATLAB.

3.3 Post-processing in MATLAB

3.3.1 SLAM

An implementation of SLAM is available as a part of the Navigation Toolbox in MATLAB. It is called lidarSLAM and performs localization and mapping using LiDAR scans. As input, the function needs LiDAR scans in the form of a cell structure. These were made by importing the range and angle values from txt-files. The function then outputs a point-cloud of the matched scans and an estimate of the LiDAR positions, an example of this can be seen in figure 3.3. Each of the blue dots in figure 3.3 in reality represents nine positions that are estimated from each of the nine scans recorded in each position. The mean position of these nine estimates was used as the final estimated position.

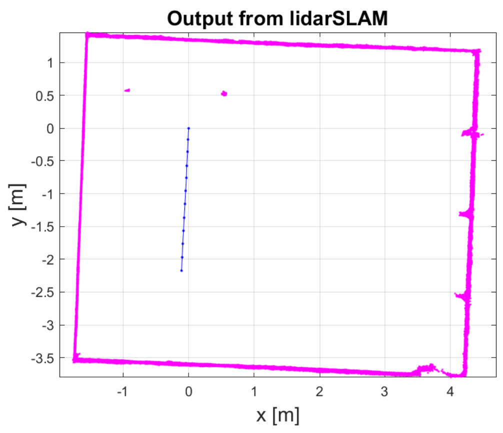


Figure 3.3: Output from the lidarSLAM algorithm, the magenta lines that outline the room are thousands of measured points comprised of multiple scans. The blue dots are the estimated LiDAR positions based on the scans.

3.3.2 Transformation

The next step was to extract data from the point-cloud from the SLAM output, this was done so that the positions of the walls could be estimated too. Then the rotation and transformation needed to be used on the LiDAR-positions to match them with the reference measurements. The rotation matrix R from equation 2.1 was calculated by incrementally increasing the angle θ until the root mean square error between the LiDAR-positions and the reference positions was at a minimum. Figure 3.4 shows the result of the transformation applied on the LiDAR-positions, and the same transform applied to the extracted point-cloud.

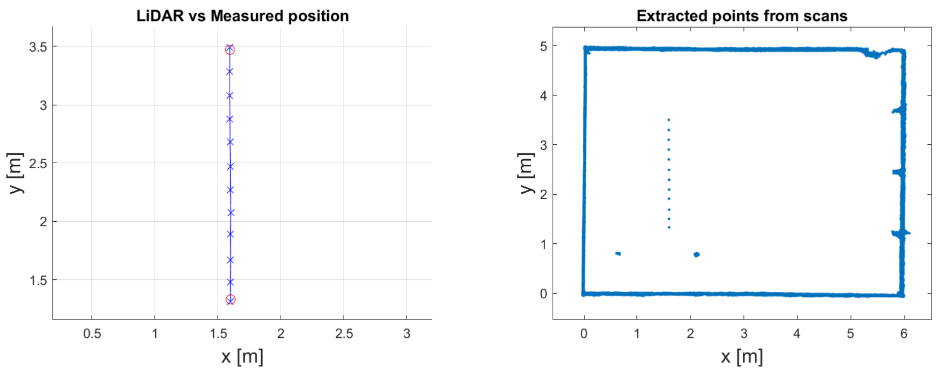


Figure 3.4: Left: The red circles are the reference positions. The blue crosses are the transformed LiDAR-positions. Right: The result of using the same transform on the extracted point-cloud.

3.3.3 Localization of microphone positions

From here the LiDAR-positions were used to find the microphone positions. This was done by applying an offset to the positions in the y-coordinate since the microphones were aligned along the y-axis. This gave a vector containing all the microphone positions, then the positions were sorted in ascending order. Then the impulse responses were imported to MATLAB and put into a matrix. Using the sorted microphone positions the impulse responses could then be sorted according to the sorted microphone positions. This made it possible to plot the impulse responses in a stacked fashion, adding the y-coordinate of the microphone position to the corresponding impulse response. An octave band filter was applied on the impulse response to smooth them out and make the plot more readable. In figure 3.5 the first few wavefronts can be seen clearly. A plot of the unfiltered stacked impulse responses can be seen in appendix A.

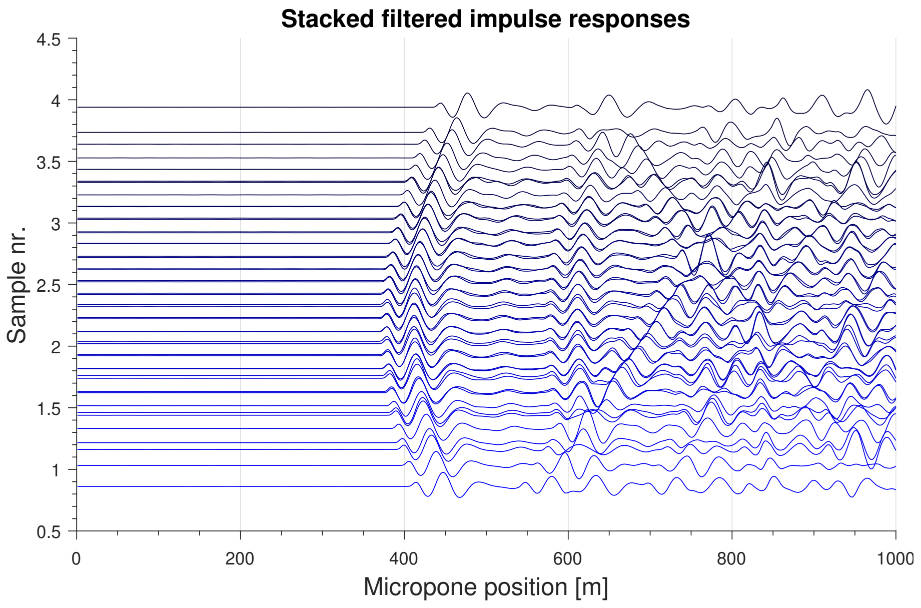


Figure 3.5: Plot showing the stacked impulse responses octave band filtered with the 1000 Hz band.

3.3.4 Identifying wavefronts

To estimate the source position the arrival times of the impulse responses needed to be found. Using a MATLAB function that analyses signals and outputs peak values and the sample number at which they appear a matrix of the sample numbers were created. This matrix was then plotted with respect to the microphone positions as shown in figure 3.6. The first wavefront was easily picked out by hand but then it gets harder to distinguish the wavefronts from each other. A script was developed that plots the peak location matrix and lets the user make a line by clicking in two points in the plot. Then the script picks the peaks closest to the line and the user gives them a wavefront number and peak sample numbers are saved in a matrix. Thereby giving the user the possibility to select visible wavefronts directly from the plot window. In figure 3.6 shows a plot of the original location matrix with respect to the microphone positions, as well as the resulting selected wavefronts from the script.

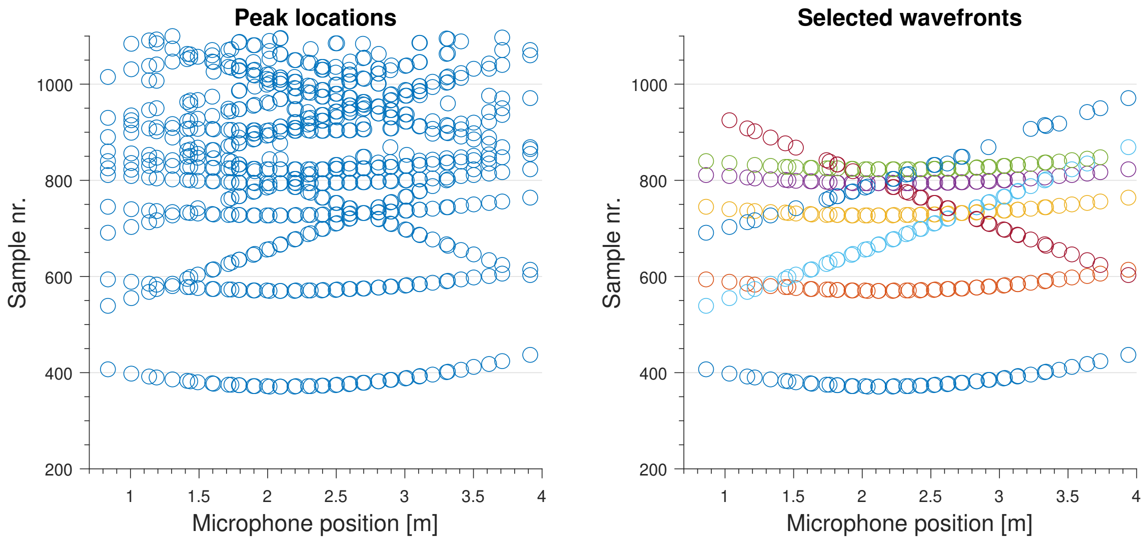


Figure 3.6: Left: Plot showing all the peak locations in the first 1100 samples. Right: Plot of the selected wavefronts, each color represents a different wavefront.

3.3.5 Estimating source position

By utilizing that the arrival sample numbers for each wavefront are known, it was now possible to calculate the arrival times. Using the technique described in section 2.3, and the fact that all the microphones were aligned along the y-axis. Examining each wavefront one at the time the minima of a polynomial of degree two was used to find the y-coordinate of the source, then this is used to find the x-coordinate. An example of the polynomial fit can be seen in figure 3.8, polynomial fits for all the selected wavefronts can be seen in appendix B. Using this technique on the first wavefront, e.g. the direct sound, leads to an estimation of the source (loudspeaker) position. By applying this technique on the other selected wavefronts the image source positions can be estimated. To find out how accurate these estimates were the theoretical positions of the image sources needed to be calculated, and to do this an estimate of the walls plane equations is needed.

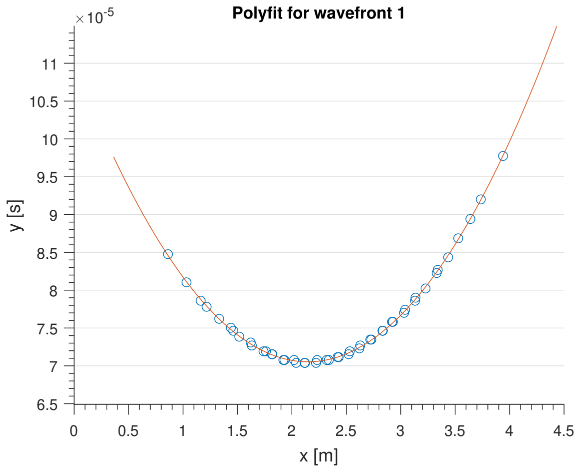


Figure 3.7: Plot showing a polynomial fit for the first wavefront (direct sound).

3.3.6 Calculating first-order image source positions

Based on the extracted point-cloud from the LiDAR-scans, seen in figure 3.4, the corner coordinates were guessed and a linear fit was made for each wall including only the closest points. From the polynomial of the estimated walls, the normalized plane equations could be calculated. These plane equations could then be used as described in section 2.4 to find the image source positions. The calculated positions of the image sources could then be compared to what was estimated by analysing the different wavefronts.

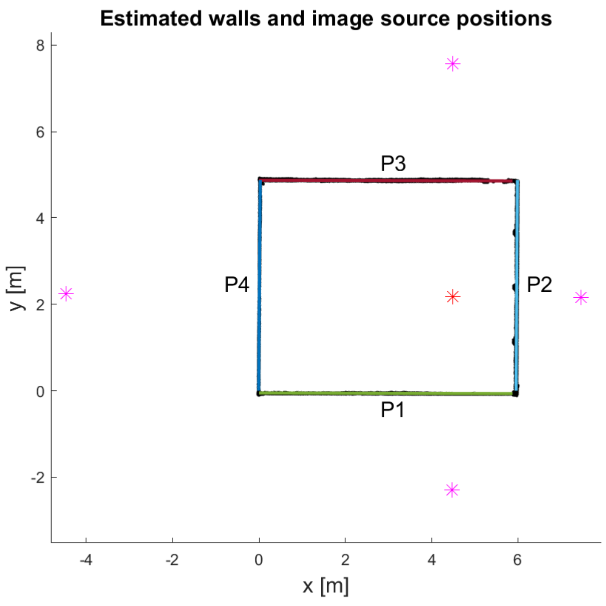


Figure 3.8: The different colored lines are the estimated walls based on the black dots. The red asterisk marks the measured source position and the magenta asterisks marks the calculated image source positions.

4 Results

This chapter presents the results from the measurements and post-processing analysis. This includes the estimated source position and estimated image source positions.

4.1 Estimating the source position

The position of the loudspeaker was the first thing to be estimated. Using the direct sound by looking at the arrival time of the first wavefront a close estimate to the reference measurement was made. With a difference between them of 8 mm in x-direction and 2.2 cm in the y-direction, this corresponds to a geometrical distance of 2.35 cm. Figure 4.1 shows the estimated source position compared to the reference measurement along with the microphone positions and estimated wall positions.

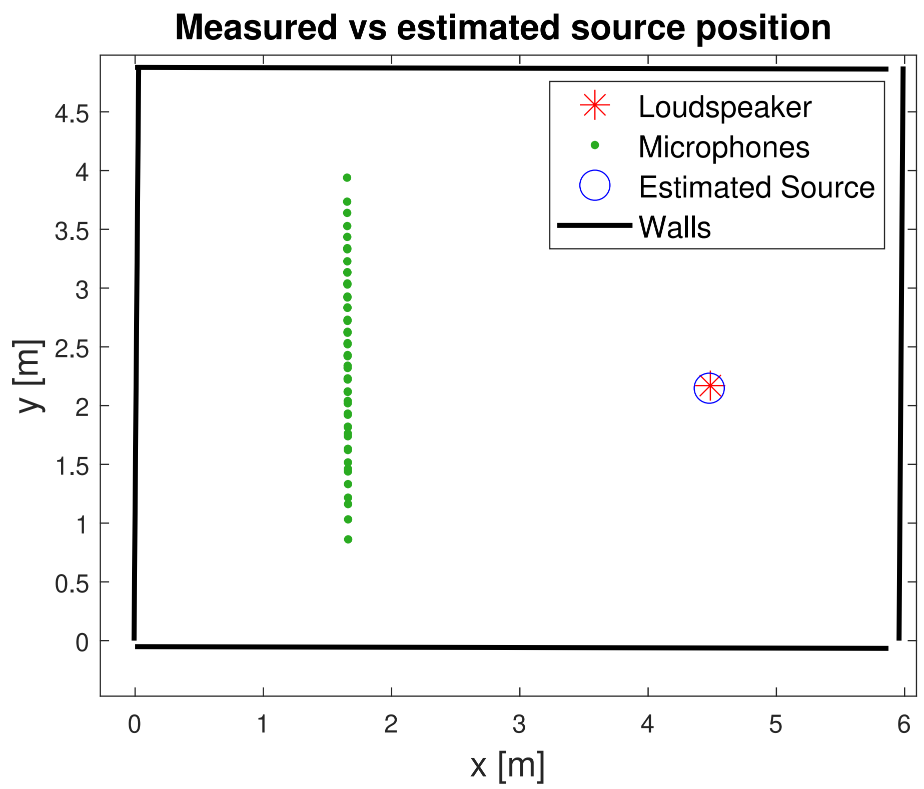


Figure 4.1: The red asterisk is the reference measurement for the source, the blue circle is the estimated source position based on the direct sound, the green dots are the microphone positions and the black lines are the estimated walls.

4.2 Estimating the image source positions

The results from analysing the wavefronts can be seen in figure 4.2, all the estimated image source positions can be seen along with their calculated theoretical counterparts. In table 4.1 a comparison between the estimated and calculated positions, their coordinates, and the difference between them. In the plot three of the blue circles marking source positions do not line up with the calculated ones (asterisks). Because this technique uses time of flight measurements to estimate the source positions, it will only give source positions further away along its x-coordinate. This comes to effect when looking at e.i the blue circle that is on the P2 wall, this is in reality the sound wave that was reflected on the floor. This can be shown by plotting a cross-section of the room as in figure 4.3 since it is known that the image source must be placed somewhere around the surface of a sphere with its center on the microphone position. This is also true for the blue circle to the far right in the plot, it is in reality an estimate of IS_4 as can be seen in the cross-section plot. The estimated image source at $\approx (6, -2)$ is most likely a second-order image source, reflected off P1 and the floor in whichever order.

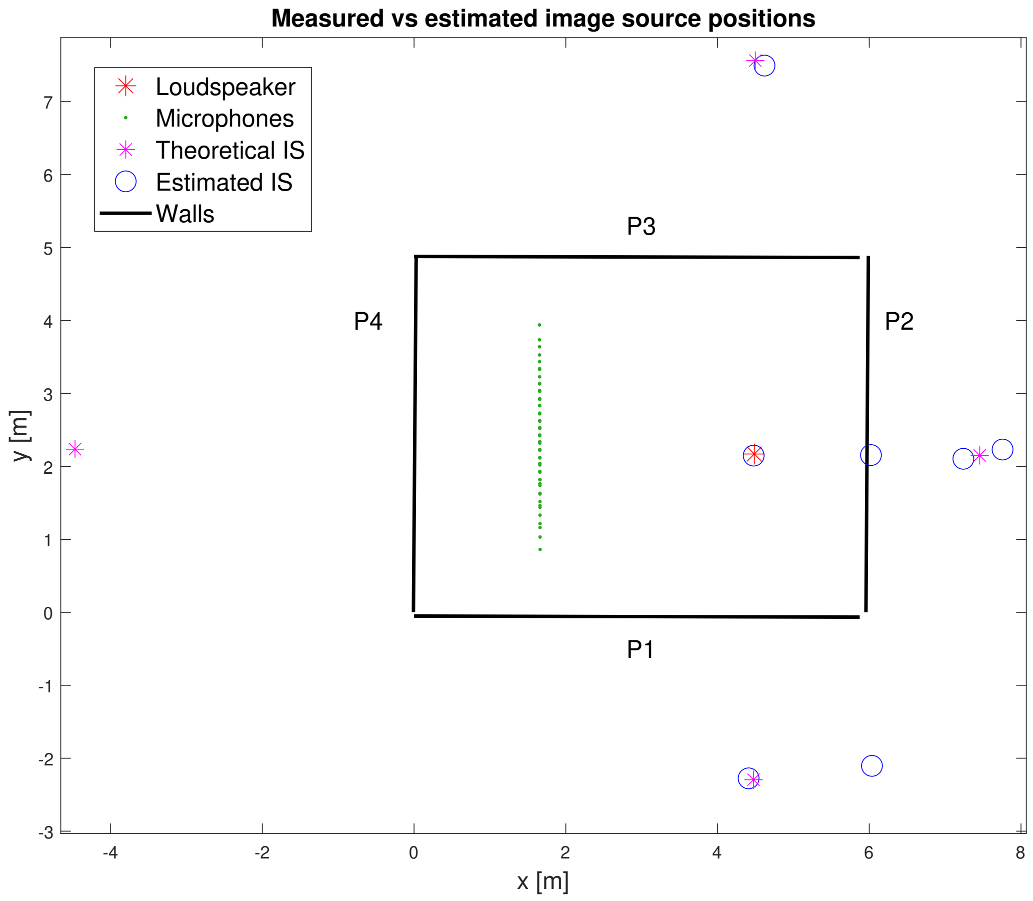


Figure 4.2: Plot showing the estimated image source positions and comparing them to the calculated theoretical positions.

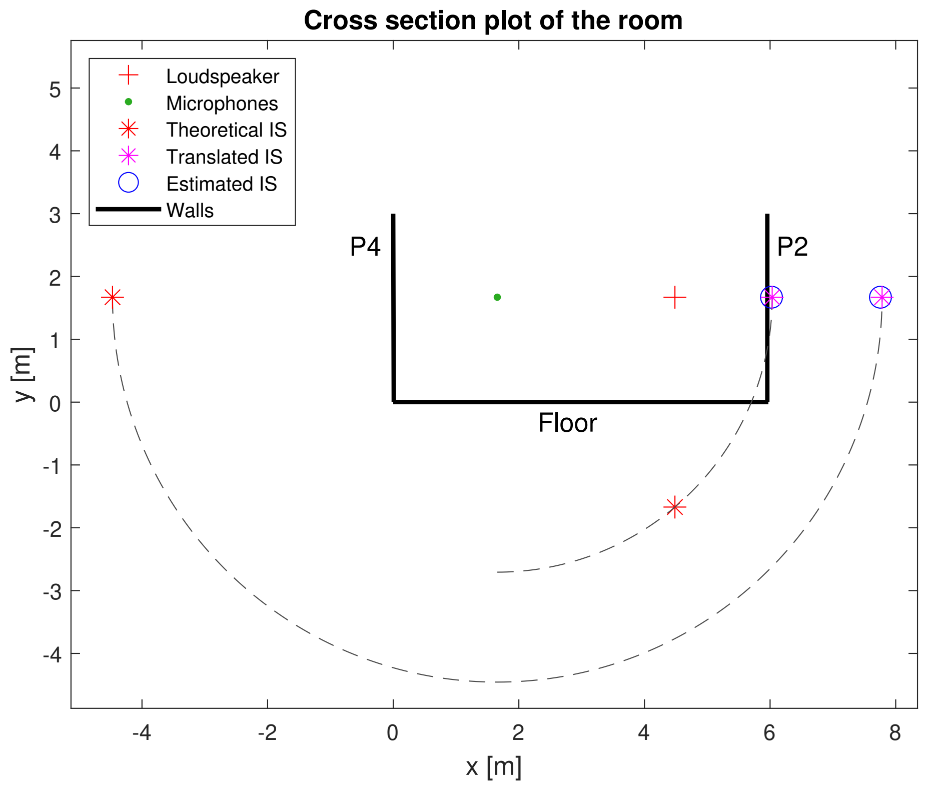


Figure 4.3: Cross section plot showing how the estimated source positions can be translated to fit with their theoretical placement.

From table 4.1 it can be seen that most of the estimates are quite good. The worst estimate is the one for IS_2 with an error in distance of 22 cm, this is most likely due to the error in the wall estimate. $P2$ is the wall that is furthest away from the LiDAR and therefore has the most potential uncertainties in distance measurements. The best estimate was for IS_{floor} which was 1.8 cm away from the theoretical position. This image source estimate is not dependent on the wall estimates. This could mean that the wall estimates are a potential source of the errors and will be discussed later.

Image Source	Estimated pos. [m]	Calculated pos. [m]	Abs. diff. [m]	Error dist. [m]
IS_1	4.410, -2.276	4.474, -2.295	0.064, 0.019	0.067
IS_2	7.241, 2.105	7.456, 2.151	0.215, 0.046	0.220
IS_3	4.619, 7.494	4.498, 7.564	0.121, 0.070	0.140
IS_4	7.757, 2.231	7.783, 2.236	0.026, 0.005	0.027
IS_{floor}	6.023, 2.155	6.033, 2.170	0.010, 0.015	0.018

Table 4.1: Table showing the results from the measurements.

5 Discussion

In this chapter, some of the results are discussed further along with the main sources of error. It will also focus on what could have been done differently and what could be worked on in the future.

5.1 Sources of error

5.1.1 Error in room dimension

The LiDAR measurements give an error in the room dimensions. When examining the wall positions and comparing them to the measurements that were done by hand the estimated walls are further apart than in reality. The walls $P1$ and $P3$ are 4.4 cm longer than the measured room dimension, and walls $P2$ and $P4$ are 8.8 cm longer. This could be the result of the microphone stand being on a slight angle. The results imply that the microphone stand is at an angle of $\approx 10°$ off axis, this seems to be unlikely. As the error was larger in the longest dimension a more probable candidate are errors in the distance measurements of the LiDAR. In the LiDAR datasheet, it is stated that the error should be $> 1\%$ of the distance. This could account for most of the error, but it could also be a combination of the two error sources that come into effect. Bad estimates for the wall positions are quite critical in this case since the wall estimates are used to calculate the image source positions.

5.1.2 Error in reference measurements

As mentioned, in this project several measurements had to be made by hand with a laser measurement tool. This is a source of error that is hard to quantify and could be the biggest source of error when it comes to localizing the source positions. The errors may stem from errors in the laser measurement tool itself, uneven walls, and errors made by the user. The laser tool that was used has an error margin of ± 1 mm per meter, this is a much smaller error margin than for the LiDAR.

5.2 Possible changes and future work

This project would benefit from the possibility to take a lot more different measurements. Testing alternative approaches for microphone positioning, placement in the room, using more than one loudspeaker, etc. Due to the current world situation, the accessibility of measurements was very limited. This made it difficult to do all the wanted tests. Below some of the suggestions for future work on this measurement technique are discussed.

5.2.1 Microphone offset

As can be seen in figure 3.5, many of the impulse responses were measured in almost the same positions. This is a result of the chosen microphone positions on the microphone stand. All four microphones were placed 30 cm apart, this lead to an overlap in many positions. This could be avoided by choosing a different step length between measurements, or by using a different offset. Having a more even coverage of microphones could result in better estimates, as well as easier wavefront identification.

5.2.2 Up-sampling

Using up-sampling on the measured impulse responses could lead to better estimates of source position by getting more accurate arrival times. The problem with this method is that it is not possible to say if the arrival time of the up-sampled signal is more correct than the other. Since there is no way of knowing the true position of image sources using this measurement technique. A way to get better accuracy in the arrival times could be to up the sample rate from 44.1 kHz to e.i. 96 kHz in the initial measurements.

5.2.3 Using the wall estimates

In this version of the measurement system, using LiDAR to find the microphone locations, only the endpoints for the microphone positions were compared to reference measurements. If all the microphone positions were measured by hand as well it might have lead to a more accurate transformation, but this defeats the purpose of using a LiDAR in the first place. As of now only the method of using some reference measurements to transform the LiDAR coordinates to the world coordinates has been tested. Relying on measurements done by hand inevitably leads to errors that are not easily estimated. The future of this measurement system should depend as little as possible on human measurements, potentially eliminating all human factors. A solution that could accomplish this is by getting more accurate estimates of the wall positions. The SLAM algorithm always sets the first scan position as coordinates (0,0), but if the walls are estimated well, an arbitrary corner can be chosen by the user as the origin for the world frame, and the walls could be rotated to match the real walls. Although, this would require a higher grade LiDAR so that the walls can be estimated accurately. The LiDAR used in this project, RPLIDAR A1M8, is a low-cost LiDAR aimed at the consumer market. Its manufacturer, SLAMTEC, offers a wide range of different LiDAR units that could potentially be a better fit for this measurement system. There are of course other manufacturers that could be considered.

So far using a LiDAR to estimate microphone positions have only been tested in an empty room with hard well-defined walls. Further testing in other environments must be done to evaluate how well this technique works in the real world. Especially if the walls are used as the reference to transform the LiDAR coordinates.

5.2.4 Future work

If an automated version of this measurement technique is going to be developed, there are a few things that need to be done. A solution where the microphone stand is placed on a robot that could move around in the room while collecting impulse responses is a long term goal. This requires automatic gathering of LiDAR-data, impulse response measurements, and possibly real-time positioning. SLAMTEC provides an open-source software development kit written in $C++$ that gives the user opportunity to program automated data acquisition from the LiDAR, and real-time SLAM algorithms could be implemented.

In this project, all the microphone positions were kept on one line. To expand the utility of the measurement system it could be developed so that localization of the loudspeaker can be done with more variable microphone positions. The SLAM algorithm outputs not only the estimated position of the LiDAR but also a value representing how much the LiDAR was rotated between each position. If the accuracy of this angle is good enough it can be used to expand the possible movement of the microphone stand, and also how the microphones are attached to the stand.

It is also possible to expand the current microphone setup with microphones in different heights. Thereby giving the opportunity to estimate the source position in 3D, as well as identifying second-order image sources more accurately. This would require the implementation of a different algorithm more like the one used in GPS, but this is a well known method that is used in many fields so that should not be a big problem. In addition, adding a sensor that measures the height of the microphone stand can be easily done.

Further analysing the impulse responses is something that could provide useful information about the rooms acoustic properties. This study focuses mostly on the localization of microphones and sources and only use the arrival times gathered from the impulse responses. By analysing them further and looking at the respective peak values for each wavefront, the reflection coefficients could be calculated for different surfaces in the room.

If the technique is developed further with a robot solution that gives accurate results it could be used for the same purposes as a similar system developed by Witew, Vorländer, and Ning [1]. Instead of a robot moving freely they have built a relatively large metal structure as a guide for a measurement robot. This gives them a high degree of precision when it comes to the robot's movement. This system will be hard to compete with when it comes to precision, but a measurement robot moving freely would require a lot less equipment and setup time. Another thing to add is that a LiDAR-based robot would work best in smaller rooms where the walls are easier to identify, whereas the system that Witew et al. developed was designed for larger rooms like concert halls.

6 Conclusion

If an automated measuring system capable of measuring thousands of impulse responses in rooms is to be created a reliable way of localizing the microphones is needed. In this study a LiDAR combined with a SLAM algorithm was used for microphone localization. The measured impulse responses were then used to estimate the source position along with the first-order image sources.

The estimation of the source position gave a resulting position that deviated by 2.35 cm compared to reference measurements done by hand. Estimation of the first-order image source positions gave varying results. Giving the best estimates for IS_{floor} with an error distance of 1.8 cm, and for IS_4 with an error distance of 2.7 cm. The worst estimates was for IS_2 and IS_3 with errors in distance of 22 cm and 14 cm.

The method used to calculate the theoretical positions of the image sources relied heavily on the LiDAR's ability to correctly measure the distances to the walls. An error of 1.5% in the length of the room, and an error of 0.8% in the width. This is probably the biggest source of error in the study. The reference measurements that had to be made by hand could also effect the accuracy of the estimates.

There are many changes to the measurement system that could yield better results. Using a different microphone offset to get better coverage of measurement points in the room. Another is using a higher sampling rate when doing the initial measurements.

A big disadvantage to the current system is the need for reference measurements. This could be avoided if the walls could be used as a reference instead. Since the LiDAR used in this project gave errors in the room dimensions, the use of a more accurate LiDAR could be the solution.

For future versions of this measurement system an expansion from 2D- to 3D-localization should be implemented. This could be done by adding a sensor measuring the height of the microphones, as well as adding microphones in the z-direction.

To ensure that this system will work well in practical application. It is recommended that the system is tested in different rooms and conditions.

Bibliography

[1] I. B. Witew, M. Vorländer, and N. Xiang, "Sampling the sound field in auditoria using large natural-scale array measurements," *The Journal of the Acoustical Society of America*, vol. 141, no. 3, pp. 300–306, 2017.

[2] P. Guidorzi, F. Pompoli, P. Bonfiglio, and M. Garai, "A newly developed low-cost 3d acoustic positioning system: Description and application in a reverberation room," *Applied Acoustics*, vol. 160, pp. 107–127, 2020.

[3] A. Pourmohammad and S. M. Ahadi, "Real time high accuracy 3-d phat-based sound source localization using a simple 4-microphone arrangement," *IEEE Systems Journal*, vol. 6, no. 3, pp. 455–468, 2011.

[4] S. Bancroft, "An algebraic solution of the gps equations," *IEEE transactions on Aerospace and Electronic Systems*, no. 1, pp. 56–59, 1985.

[5] J. Sola, "Simulataneous localization and mapping with the extended kalman filter," *Lecture note*, 2013.

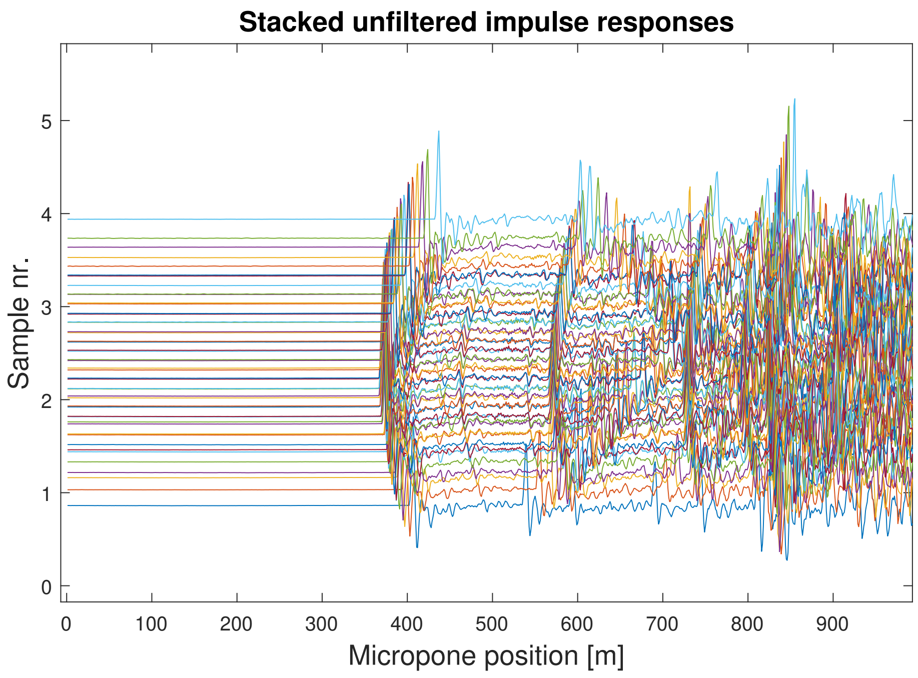
[6] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1271–1278, IEEE, 2016.

[7] U. P. Svensson, "Discussion with peter svensson, (main supervisor)," *Department of electronic systems, NTNU.*

[8] U. P. Svensson, "Geometrical acoustics ++," *Department of electronic systems, NTNU*, pp. 49–50, 2019.

[9] Anon., "Rplidar a1 introduction and datasheet," *Low Cost 360 Degree Laser Range Scanner.*

A Plot of unfiltered impulse responses



B Polynomial fit for selected wavefronts

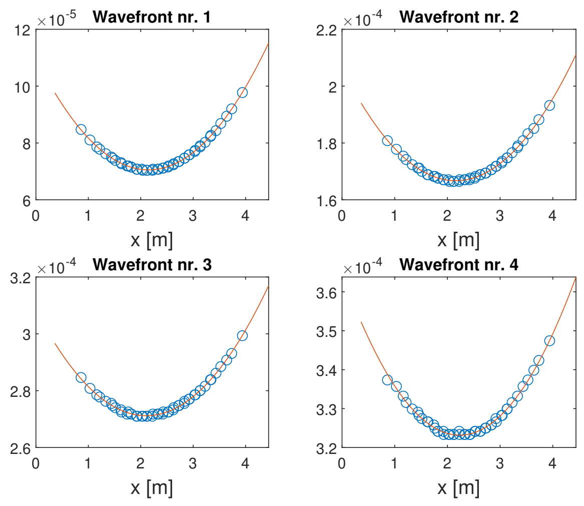


Figure B.1: Polynomial fit for wavefronts 1 to 4.

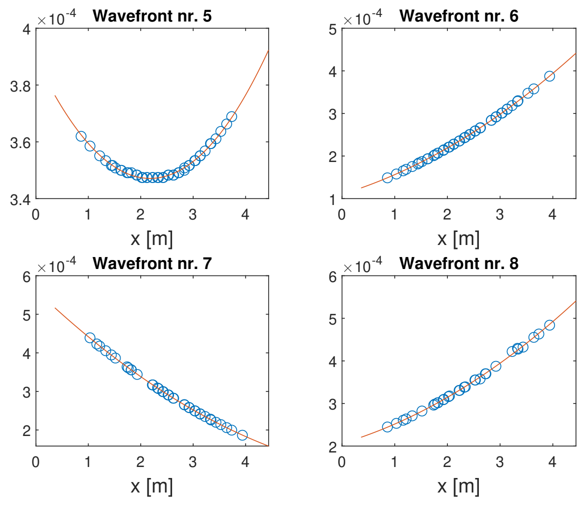


Figure B.2: Polynomial fit for wavefronts 5 to 8.

C MATLAB script

```
% Author: Jens Andresen, 29.06.20

%%
ccc

% Some constants that are used in further calculation

% Measurement 1
% T = 17.3;
% lsp_pos = [5.884-1.304 2.507];
% pos1 = [1.605, 1.280];
% pos12 = [1.595, 4.885-1.415];

% Measurement 2
T2 = 18.51;
pos1 = [1.60, 1.332];
pos12 = [1.595, 4.885-1.415];

% Sound speed
c = 331.3 + 0.606*T2;

fs = 44100;
roomdim = [0 5.874 0 4.886];
lsp_pos = [roomdim(2)-1.3890 2.170];

%% Importing IRs, addIR = 1 to import new data

addIR = 0;
ncut = 1200;

npos = 12;
ir_all  = zeros(ncut,npos*4);

if addIR == 1

for ii = 1:npos

% ir = importdata(sprintf('C:/Users/Jens/Documents/MATLAB/
    Measurements_etx/M0002_S01_R%02d.etx',ii),'\t',22);
ir = importdata(sprintf('C:/Users/Jens/Documents/MATLAB/
    Measurements2_etx/M0001_S01_R%02d.etx',ii),'\t',22);

ircut = ir.data(1:ncut,2:end);
% ircut = [ircut(:,2),ircut(:,1),ircut(:,4), ircut(:,3)];

```

```
ir_all(:,(1+(ii-1)*4:4+(ii-1)*4)) = ircut;


end

irtime = ir.data(1:ncut,1);

save irdata.mat ir_all irtime

else

    load irdata.mat

end
%% import lidarscans

% importing the scan data from txt files , converting them to lidarScan

startpos = 1; endpos = 12; numscan = 9;

addscans = 0;

if addscans == 1

for m = (startpos:endpos)

for k = 1:numscan
% importing the scan data from txt files

% scan = importdata(sprintf('C:/Users/Jens/Documents/MATLAB/lidardata/
    Masterscan/pos%d_%d.txt',m,k),' ',3);
scan = importdata(sprintf('C:/Users/Jens/Documents/MATLAB/lidardata/
    Masterscan2/pos%d_%d.txt',m,k),' ',3);

scan = sortrows(scan.data);
angles = deg2rad(scan(:,1));
ranges = scan(:,2)./1000;
%
% angles = angles(1:3:end);
% ranges = ranges(1:3:end);

scans{k+numscan*(m-startpos)} = lidarScan(ranges,angles);

end
end

% SLAM

maxRange = 5; % meters
resolution = 30; % cells per meter

slamObj = robotics.LidarSLAM(resolution,maxRange);
```

```
slamObj.LoopClosureThreshold = 100;
slamObj.LoopClosureSearchRadius = 8;
slamObj.MovementThreshold = [0 0];

timingvec = zeros(1,numel(scans));

for i=1:numel(scans)
    t0 = clock;
    [isScanAccepted, loopClosureInfo, optimizationInfo] = addScan(
        slamObj, scans{i});
    if isScanAccepted
        fprintf('Added scan %d \n', i);
    end

    timingvec(i) = etime(clock,t0);

        if rem(i,numel(scans)) == 0
%          figure()
%          show(slamObj);
%          figure()
%          show(slamObj.PoseGraph, 'IDs', 'all');
        [scans,poses] = scansAndPoses(slamObj);
        clc
        disp('All scans added')
        end
end



%% Extract lidar data points
figure(30)
show(slamObj);
fig = gcf;

axobj = fig.Children;
dataobj = axobj.Children;

nobj = length(dataobj);

x = [];
y = [];

for ii = 1:nobj

    x = [x; dataobj(ii).XData'];

    y = [y; dataobj(ii).YData'];

end


lidarpointcloud =[x,-y];

```

```
close figure 30

save scandata.mat scans poses timingvec slamObj lidarpointcloud

else
    load scandata.mat

end

%% Plot of the time usage
% figure(3)
% plot(timingvec,'-o')
% grid

%% Standard deviation

xpos = poses(:,1)';
ypos = poses(:,2)';

for i = 1:(endpos-startpos+1)

        mx(1+(i-1)*numscan:numscan+(i-1)*numscan) = mean(xpos((1+(
            numscan*(i-1)):(numscan+numscan*(i-1)))))*ones(1,numscan);

end

for i = 1:(endpos-startpos+1)

        my(1+(i-1)*numscan:numscan+(i-1)*numscan) = mean(ypos((1+(
            numscan*(i-1)):(numscan+numscan*(i-1)))))*ones(1,numscan);

end

for i = 1:(endpos-startpos+1)

    stdxpos(i) = std(xpos(1+(i-1)*numscan:numscan+(i-1)*numscan) - mx
        (1+(i-1)*numscan:numscan+(i-1)*numscan));

end

for i = 1:(endpos-startpos+1)

    stdypos(i) = std(ypos(1+(i-1)*numscan:numscan+(i-1)*numscan) - my
        (1+(i-1)*numscan:numscan+(i-1)*numscan));

end

stdpos = [stdxpos; stdypos]';

%% Ground truth

mpos = [mx(1:numscan:end)' my(1:numscan:end)'];
mpos = [mpos(:,1) mpos(:,2)*-1];
```

```
mposSE = [mpos(1,:);mpos(end,:)];

posscan = [pos1; pos12];

% Transformation
t = [posscan(1,1);posscan(1,2)];
rmsd = [1,1];
numpos = endpos-startpos+1;
a = 0;

for p = 1:100

a = 0;
while rmsd(1)>0.0002*p || rmsd(2)>0.01*p

    a = a + 0.1;

    R = [cosd(a) -sind(a) ; sind(a) cosd(a)];

    mpostr = ((R*mposSE') + t)';

    rmsd = sqrt(sum((posscan - mpostr).^2)/2);

    if a > 10000
        break
    end

end

    if rmsd(1)<0.00001*p && rmsd(2)<0.001*p
            break
    end

end

posmat = ((R*mpos') + t)';

posmat(:,2) = posmat(:,2) - (mpostr(2,2) - posscan(2,2))/2;

lidarpos = posmat;


%% Plot of comaring estimate with ground truth
figure(5)
plot(posscan(:,1),posscan(:,2),'r-o')
hold on
plot(posmat(:,1),posmat(:,2),'b-x')
axis([roomdim])


%% Finding microphone positions (according to lidar) and sorting the
    IRs and offsetmatrices

```

```
offset = [-0.45,-0.15,0.15,0.45];
offsetvec = repmat(offset,1,12)';

replidar = repelem(posmat(:,2),4);
posvec = replidar;

posvec = posvec + offsetvec;

posmatoffset = repmat(posvec',ncut,1);


%% Sorting and plotting the sorted impulse responses

[posvec_sorted,idx] = sort(posvec);

posmatoffset_sorted = posmatoffset(:,idx);

ir_sorted = ir_all(:,idx);

posmat_sorted = repmat(posvec_sorted',ncut,1);


s = repmat(linspace(0,20,48),ncut,1);

figure(6)
plot(oktavbandfilter(ir_sorted,2)+posmat_sorted)

figure(7)
plot(ir_sorted*0.2+posmat_sorted)


%%

lidarposmat = repelem(lidarpos,4,1);
lidarposmat = [lidarposmat(:,1), lidarposmat(:,2) + offsetvec];


d = lidarpos(end,:) - lidarpos(1,:);
e = lidarpos(1,:);


for ii = 1:48

    b = (lidarposmat(ii,2) - e(2)) /d(2);

    yvec(ii) = e(2) + b*d(2);
    xvec(ii) = e(1) + b*d(1);
end


micpos = [xvec', yvec'];

[micposx_sorted,indx] = sort(micpos(:,1),1);
```

```
[micposy_sorted,indy] = sort(micpos(:,2),1);

micpos = micpos(indy,:);

micpos = [micpos(:,1)+0.06  micpos(:,2)];
```

```
% Author: Jens Andresen, 29.06.20

fs = 44100;
ivir = 1:48;
posvec_mic = posvec(ivir);
ncut = 570;

%% Find peaks

ir_peak = ir_all(:,ivir);

npeaks = 20;

peakmat = zeros(npeaks,length(ivir));
locmat = zeros(npeaks,length(ivir));

[posvec_micSort,idx] = sort(posvec_mic);

ir_peakSort = ir_peak(:,idx);

for ii = 1:npos*4

[peaks,locs] = findpeaks(ir_peakSort(:,ii),'MinPeakDistance',3,'
    MinPeakHeight',0.3);

peakmat(1:length(peaks),ii) = peaks;
locmat(1:length(locs),ii) = locs;

end

xmat = posvec_micSort(:,ones(npeaks,1)).';
wavefrontmat = zeros(size(xmat));
wavefrontmat(1,:) = 1;
wavefrontmat(2,5:end-1) = 2;
wavefrontmat(3,1:4) = 2;
wavefrontmat(3,end) = 2;
wavefrontmatvert = reshape(wavefrontmat,npeaks*npos*4,1);
locmatvert = reshape(locmat,npeaks*npos*4,1);
xmatvert = reshape(xmat,npeaks*npos*4,1);
wavefrontcounter = 3;

doclick = 0;

if doclick == 1
    while wavefrontcounter ~=0

    iv = find(wavefrontmatvert == 0 & locmatvert > 0);

    [ivfront] = ptl(xmatvert,locmatvert,iv);

    promt = {'Enter wavefront nr.'};
    dlgtitle = 'Input';
    wavefrontcounter = inputdlg(promt,dlgtitle,1,{num2str(
        wavefrontcounter)});
    wavefrontcounter = cell2mat(wavefrontcounter);
    wavefrontcounter = str2double(wavefrontcounter);

    wavefrontmatvert(iv(ivfront)) = wavefrontcounter;


    end

save wavefrontdata1.mat wavefrontmatvert locmatvert xmatvert
    else
load wavefrontdata1.mat

end

wavefrontmat = reshape(wavefrontmatvert,npeaks,48);


%% Estimating position of the LSP

LSPest = zeros(max(wavefrontmatvert),2);

for ii = 1:max(wavefrontmatvert)

    [~,idc] = find(wavefrontmat == ii);
    tpeak_mic = (locmat(wavefrontmat == ii)-1)/fs;

    p_mic = polyfit(posvec_micSort(idc),tpeak_mic.^2,2);

    posvec_micplot = linspace(min(posvec_micSort)-0.5,max(
        posvec_micSort)+0.5,100);

    t2peakfit_mic = polyval(p_mic,posvec_micplot);

    ySest = -p_mic(2)/(2*p_mic(1));
    xSest = polyval(p_mic,ySest);
    xSest = sqrt(xSest)*c;

    LSPest(ii,:) = [xSest,ySest];

%     figure(ii)

%     plot(posvec_micSort(idc), tpeak_mic.^2,'o',posvec_micplot,
    t2peakfit_mic,'-')
%     title(['Wavefront nr. ',num2str(ii)])

end
LSPest = [LSPest(:,1)+1.6 LSPest(:,2)];
ISallest = LSPest;
ISfloorest = LSPest(2,:);
LSPest(2,:) = [];
ISwall4est = LSPest(3,:);
LSPest(3,:) = [];
```

```
ISunknown = LSPest([3 6],:);
LSPest([3 6],:) = [];

for ii = 1:max(wavefrontmatvert)
[~,idc] = find(wavefrontmat == ii);
figure(max(wavefrontmatvert)+1)
plot(posvec_micSort(idc),locmat(wavefrontmat == ii),'o')
hold on
end
title('Selected wavefronts','FontSize',14)
xlabel('Microphone position [m]','FontSize',14)
ylabel('Sample nr.','FontSize',14)
hold off


figure(max(wavefrontmatvert)+2)
plot(posvec_micSort,locmat,'o')


%% Image sources
% floor reflection - wavefront 2


d2 =  (lsp_pos(1) - 1.6)/2;

ISfloor = [2*sqrt(d2^2 + 1.67^2), LSPest(2,2)];

ISfloordiff = abs(LSPest(2,:) - ISfloor);
```

```
% Author: Jens Andresen, 29.06.20


load scandata.mat
lidarpointcloud = ((R*lidarpointcloud') + t)';

doclick = 0;

if doclick == 1

figure()
h2 = plot(lidarpointcloud(:,1),lidarpointcloud(:,2),'.');

v = ginput(4);
v1 = [v(1,:) 0];
v2 = [v(2,:) 0];
v3 = [v(3,:) 0];
v4 = [v(4,:) 0];



dist1 = point_to_line_distance(lidarpointcloud,v1,v2);
figure()
plot(sort(dist1))
dd = ginput(1);
dd = dd(2);
w1 = lidarpointcloud(find(dist1<dd),:);

dist2 = point_to_line_distance(lidarpointcloud,v2,v3);
figure()
plot(sort(dist2))
dd = ginput(1);
dd = dd(2);
w2 = lidarpointcloud(find(dist2<dd),:);

dist3 = point_to_line_distance(lidarpointcloud,v3,v4);
figure()
plot(sort(dist3))
dd = ginput(1);
dd = dd(2);
w3 = lidarpointcloud(find(dist3<dd),:);

dist4 = point_to_line_distance(lidarpointcloud,v1,v4);
figure()
plot(sort(dist4))
dd = ginput(1);
dd = dd(2);
w4 = lidarpointcloud(find(dist4<dd),:);


save walldata.mat w1 w2 w3 w4 v

else
```

```

    load walldata.mat

end

figure(40)
clf(40)
hold on
plot(w1(:,1),w1(:,2),'k.')
axis equal
plot(w2(:,1),w2(:,2),'k.')
plot(w3(:,1),w3(:,2),'k.')
plot(w4(:,1),w4(:,2),'k.')



pp3 = polyfit(w1(:,1),w1(:,2),1);
pp2 = polyfit(w2(:,2),w2(:,1),1);
pp1 = polyfit(w3(:,1),w3(:,2),1);
pp4 = polyfit(w4(:,2),w4(:,1),1);

ppmat = [pp1;pp2;pp3;pp4];

ssh = linspace(0,roomdim(2),3);
ssv = linspace(0,roomdim(4),3);

ppw1 = polyval(pp1,ssh);
ppw2 = polyval(pp2,ssv);
ppw3 = polyval(pp3,ssh);
ppw4 = polyval(pp4,ssv);




%% Theoretical position of IS

% Using the plane equations for the walls to find theoretical position
    of
% IS.

ppnorm = zeros(4,2);
Dmat = zeros(4,1);

for ii = [1 3]

    nf = sqrt(ppmat(ii,1)^2 + (-1)^2);
    ppnorm(ii,:) = [ppmat(ii,1)/nf -1/nf ];
    Dmat(ii,1) = [ppmat(ii,2)/nf];

end

for ii = [2 4]

```

```
    nf = sqrt(ppmat(ii,1).^2 + (-1)^2);
    ppnorm(ii,:) = [-1/nf ppmat(ii,1)/nf ];
    Dmat(ii,1) = [ppmat(ii,2)/nf];

end

tvec = ppnorm*lsp_pos' + Dmat;

lspposmat = repelem(lsp_pos,4,1);

ISpos = lspposmat - 2*tvec.*ppnorm;


%% plot
figure(41)
clf(41)
h = plot(ssh,ppw1,'r');
set(h(1),'LineWidth',2)
hold on
h= plot(ppw2,ssv,'r');
set(h(1),'LineWidth',2)
h = plot(ssh,ppw3,'r');
set(h(1),'LineWidth',2)
h = plot(ppw4,ssv,'r');
set(h(1),'LineWidth',2)

h = plot(ISpos(:,1),ISpos(:,2),'r*');
h = plot(lsp_pos(1),lsp_pos(2),'m*');
h = plot(LSPest(:,1),LSPest(:,2),'bo');
h = plot(ISwall4est(1),ISwall4est(2),'bs');
h = plot(micpos(:,1),micpos(:,2),'k.');

phivec = [180:360].';
isposcircradius = norm(ISwall4est - [1.657,ISwall4est(2)]);
isposcirc = [1.657,ISwall4est(2)] + isposcircradius*[cosd(phivec) sind(
    phivec)];
h = plot(isposcirc(:,1),isposcirc(:,2),'k--');
set(h(1),'LineWidth',0.2)
h = plot(isposcirc(1,1),isposcirc(1,2),'bo');
xlabel('x-position [m]')
ylabel('y-position [m]')
title('Estimated image source locations compared to the theoretical
    locations ')
axis([-5 9 -4.5 8.5])
text(2.8,-0.4,'P1')
text(6.2,2.5,'P2')
text(2.8,5.3,'P3')
text(-0.8,2.5,'P4')


saveas(gca,'C:\Users\Jens\Documents\Skole\Master\ISvsEst.eps')
saveas(gca,'C:\Users\Jens\Documents\Skole\Master\ISvsEst.png')
hold off
```

```
function d = point_to_line_distance(ptmat, v1, v2)
% pt should be nx3
% v1 and v2 are vertices on the line (each 1x3)
% d is a nx1 vector with the orthogonal distances
v1 = repmat(v1,size(ptmat,1),1);
v2 = repmat(v2,size(ptmat,1),1);
a2 = v1 - v2;
b = ptmat - v2(1:2);
b = [b,b(:,1)*0];
d = sqrt(sum(cross(a2,b,2).^2,2)) ./ sqrt(sum(a2.^2,2));
end
```