Self-Localization of Large Aperture Microphone Arrays

Students Project

Thomas Wilding, BSc

Supervisor: DI Christian Schörkhuber

Graz, April 1, 2015

kunst uni graz

institut für elektronische musik und akustik

Abstract

The aim of this work is the evaluation of sate-of-the-art self-calibration algorithms for synchronized distributed microphone arrays. Here, self-calibration refers to the task of estimating microphone positions assuming unsynchronized source signals emitted from unknown locations. To simplify the whole procedure, it is assumed that the positions of the microphones and of the sources are stationary and estimates of the time-differences-of-arrival (TDOA) are available.

Das Ziel dieser Arbeit ist der Vergleich gängiger Algorithmen zur Selbstkalibrierung von synchronisierten, räumlich verteilten Mikrofonen. Mit Selbstkalibrierung ist hier die Bestimmung der Positionen der Mikrofone unter Verwendung von unsynchronisierten, räumlich verteilten Schallquellen an unbekannten Positionen. Zur Vereinfachung der Positionsschätzung werden die Positionen der Mikrofone und Quellen als stationär angenommen. Des weiteren wird von vorhandenen time-difference-of-arrival (TDOA) Schätzungen ausgegangen.

Contents

1 Introduction

When using microphone arrays in for example hands-free telephoning, they are usually rather small, due to the space reserved for the application. Thus it is easy to find the positions of the used microphone with the accuracy needed for the application when measuring by hand.

When using a large scale microphone array, it can be assumed that measuring the positions of each microphone can be a complicated and time consuming procedure for a single person to undertake, either using a tape-measure or an equivalent device. In some applications, the microphones or sensors may even change their positions over time, which would include the time as another degree of freedom, which will not be treated in this project. Furthermore, microphone arrays are usually deployed in different environments, which also makes it important to know the locations of the microphones. The whole process could be greatly simplified, if there existed a simple method which could achieve the same, or at least sufficiently accurate results.

If not measured by hand, usually the time-of-flight (TOF) from the sources to the sensors (i.e. microphones), which can be reformulated as a cost function and solved using mathematical solvers, is used for finding the distances between microphones and sources. The problem that arises here is, that if the problem is not initialized carefully, it can easily get stuck in a local, rather than the intended global minimum.

The initialization problem is usually solved by introducing additional constraints, such as using sub-arrays [9] which have a certain geometrical structure (linear,...) and reduce the number of unknowns, or by positioning all sources on a plane [6], or using a pyramid shaped structure for the sources surrounding the sensors [13]. When using subarrays, another possibility is to separate the computation process and estimate the shape of the subarrays in the first step, and then use the results obtained by these subarrays to estimate the positions of the sources (mentioned in [4])

1.1 Classification

For the problem of position estimation regarding microphones, there exist a multitude of different algorithms with varying restrictions, advantages and complexities. They could be grouped into classes, based on their mathematical approaches, as follows:

- Multidimensional Scaling (MDS): distance between points needed, used in [10] and [2]
- energy based: assumptions concerning room characteristics (damping, reverberation,...) needed, results are shown in [3] and [16].
- statistics based: using a spatial likelihood function [1] or ML estimations [14]
- TOF/LS: hardly any constraints (mainly number of sources/microphones), no information about room geometry needed, sources usually assumed to be in near-field. Examples can be found in [4, 5, 7, 11, 12, 15].

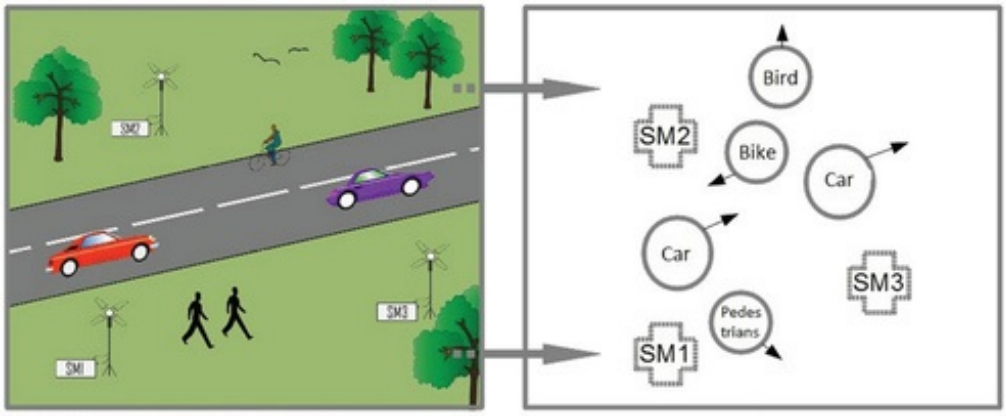


Figure 1 – The general idea of WiLMA for transcription of an acoustic scene (picture taken from [8]).

The easiest way to find the spatial locations of microphones would be to measure the distances and apply a multidimensional scaling algorithm [2, 10], bearing in mind that measuring many microphones with distances greater than comfortably reachable by one person can be a very time consuming process.

A use for such algorithms can already be seen in the WiLMA (**Wi**reless **L**arge Scale **M**ircophone **A**rray) project performed by the IEM [8]. Other useful applications could be search robots with mounted microphones, localizing each other and sound events.

1.2 WiLMA

The aim of the WiLMA project [8] is to analyze an acoustic scene, which could contain any noise/sound source like birds, cars, or speakers in a conference. Using such a technique, it would be possible to investigate new recording techniques, which would adapt to a changing setup of musicians, for example in an orchestra or in theater or opera recordings, or also at conferences for different speaker positions.

WiLMA uses a wireless sensor network based on sixteen sensor modules (SM) which each in turn have the ability to record 4 microphone channels. The sensor modules can thus record the signals from an Oktava 4D-ambient microphone, a first order ambisonics tetrahedral microphone (a setup type which will specifically be examined in Section 5). Using these microphones to record the sound scene results in the benefit, that the transcribed sound scene can be enhanced by additional directional information captured by the Oktava microphones.

To perform all the above tasks, it is essential to know the precise positions of the (in this case) sub-arrays (one sub-array consists of the four capsules of one Oktava microphone), and the orientation of each sub-array, using estimates of the positions of the microphone capsules.

2 Problem Description

Before describing the different existing algorithms, a general description of the problem will be given, together with all the notations needed to describe the whole setup of sources and microphones. Due to the fact that the source/microphone setup is independent to rotation, it is sufficient to find the positions of sources and microphones relative to either one source or one microphone.

The coordinates of the sources and microphones will be defined as $\mathbf{s}_j = [s_{x,j} \;\; s_{y,j} \;\; s_{z,j}]^T$ and $\mathbf{r}_i = [r_{x,i} \;\; r_{y,i} \;\; r_{z,i}]^T$ respectively (in a Cartesian coordinate system). For easier computations, the coordinates can be written as matrices $\mathbf{R} = \{\mathbf{r}_i^T\}$ of size $\mathcal{I} \times 3$ and $\mathbf{S} = \{\mathbf{s}_j^T\}$ of size $\mathcal{J} \times 3$ for microphones and sources respectively.

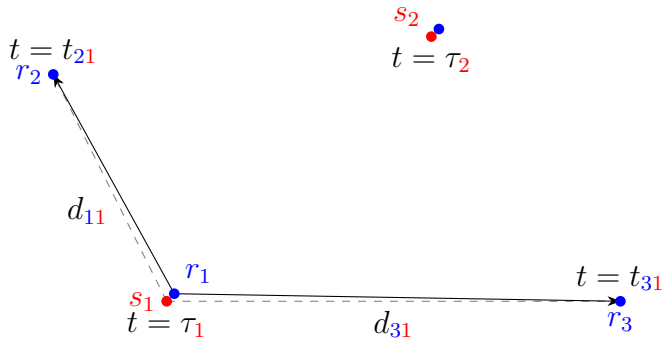
The assumptions that will be used are minimal. The microphones and the sources are assumed to be in the same volume. To be able to determine the time of flights, it is furthermore assumed that all the direct paths between the sources and the microphones are not obstructed by objects that would alter the sound propagation in any way. Furthermore it is assumed that the recordings of the microphone signals are synchronized.

The TOA t_{ij} of source j at microphone i can be found using correlation methods, threshold detection, or any other method. Here is a first crucial point, where noise can possibly be introduced into the measurements, due to the fact that the time of arrival at the microphones could be masked by noise when using threshold detection, or in between two samples when using correlation methods, whereas the latter case might not bet that critical, due to the existence of sub-sample correlation methods. The first reflections that arrive after the direct sound should therefor also not influence the detection of the arrival of the respective sound event.

The sound events can be either played simultaneously, or consecutively. For the simultaneous case it has to be somehow possible to know from which source the sound event originated, for example by using different frequencies for each source. For consecutive sounds, no restrictions concerning the type of sound are applied. In turn, all reverberation of a sound event should have subsided, such that the detection of a new sound event is not influenced by other sound events. The TOD (time of departure of the sound event at source j) will be denoted using τ_j throughout this work.

The estimation of the times of departure would not be needed when using a special source object as proposed by Khanal [11], consisting of a loudspeaker with a microphone mounted as close to the acoustic center as possible to obtain the emission times of the sound events (TODs).

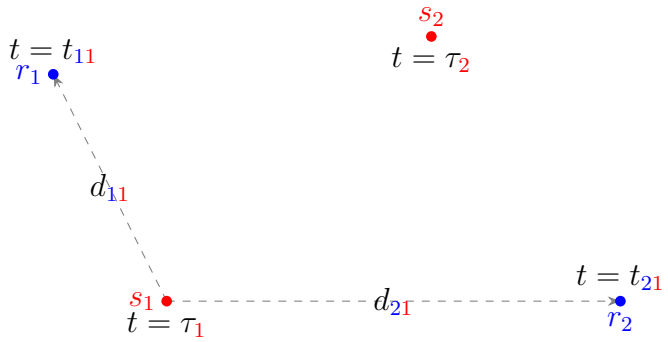
Two 2-dimensional diagrams showing a simplified setup containing all essential values can be seen in figure 2 for the case of using a source object and figure 3 for the general case without a source object.



time: $\tau_1 \approx t_{11} < t_{21} \leq t_{31} < \tau_2$

$r_i \ldots$ microphones	$s_j \ldots$ sources
$\tau_j \ldots$ TOD	$t_{ij} \ldots$ TOA
Assumption:$t_{11} \approx \tau_1$	
$\text{TOF} = t_{ij} - \tau_j$	$d_{ij} = \text{TOF} \cdot \text{c}$

Figure 2 – General setup (simplified to 2D) with a source object proposed by Khanal [11] to obtain the TODs.



time: $\tau_1 < t_{11} \leq t_{21} < \tau_2$

$r_i \ldots$ microphones	$s_j \ldots$ sources
$\tau_j \ldots$ Time of departure (TOD)	$t_{ij} \ldots$ Time of arrival (TOA)
$\text{TOF} = t_{ij} - \tau_j$	$d_{ij} = \text{TOF} \cdot \text{c}$

Figure 3 – General setup (simplified to 2D) used in this and the work of Gaubitch [7], Crocco [4] and others, without a source object and therefore unknown TODs at the sources. In turn, no source object needs to be positioned and clapping or else can be used as source signals.

3 Algorithms

In this section, the different algorithms for solving the problem described in section 2 will be described briefly.

3.1 MDS (Multidimensional Scaling)

Classical Multidimensional scaling provides a solution for the problem of positioning points in a N-dimensional space, and can be used on different problems, not only on metric ones (fitting points into a coordinate system). MDS assumes, that the pairwise distances between all points (in this problem that would be the microphones/sensors and the sources) are known, and then computes the coordinates of the specified points in a non-iterative way. The problem here is, that if not measured by hand, which can be very time consuming, if not at all impossible when sensors and sources are far apart or even in locations not reachable for measuring, the pairwise distances between two sources cannot be obtained by a simple TOF flight measurement alone.

Birchfield [2] describes the classical MDS algorithm and also shows an enhancement thereof, called BCMDS (basis-point classical MDS), which can be used when some of the data needed for performing the MDS algorithm (which would be the case when it is impossible to measure the distances between all points), is missing.

The MDS algorithm for computing the locations of n microphones in a p-dimensional space using the known, pairwise, noisy distances d_{ij} is described by Birchfield as follows:

In case of missing data, basis points with known locations are used, which could be speakers attached to a rigid structure where the distances can be measured or are known already. The distances can be acquired by TOF measurements or using a tape measure. The MDS and BCMDS algorithms can both be seen as an easier alternative to nonlinear optimization.

In the method proposed by Ji [10], local maps (described as relative positions of a number of sensors/sources) are used, which are then fused together to acquire the global positions of the microphones. Therefore some nodes are needed, which are sensors with known locations, assuming that the rest of the sensors are at unknown positions. The more anchors are used, the better the approximation of the true positions of the sensors in the connected local maps is acquired. The sensors are furthermore assumed to be connected with each other wirelessly, which also introduces another restriction, that not all sensors can communicate with each other, due to the fact that the radio signals are attenuated by propagation.

A big benefit of the method proposed by Ji is, that the area where the sensors are located can be anisotropic, meaning that also obstacles which shield the sensors from each other (missing data) do not pose a problem. A little drawback is the fact that the proposed method only estimates 2D coordinates (although it could be extended to 3D easily, according to Ji).

1. Construct the squared-distance matrix $\mathbf{D} = \{d_{ij}^2\}$
2. Compute the inner product matrix $\mathbf{B} = -\frac{1}{2}\mathbf{JDJ}$, with $\mathbf{J} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, with $\mathbf{1}$ being a vector consisting of ones.
3. Decompose $\mathbf{B}$ as $\mathbf{B} = \mathbf{V}\Lambda\mathbf{V}^T$, where Λ is a diagonal matrix $\Lambda = \text{diag}\{\lambda_i\}$ containing the eigenvalues of $\mathbf{B}$ in decreasing order, and $\mathbf{V}$ is a matrix containing respective eigenvalues $\mathbf{V} = [\mathbf{v}_1, \cdots, \mathbf{v}_n]$
4. Extract the first p eigenvectors (p is the dimension of the space, i.e. $p = 3$ for 3D-coordinates).
5. The microphone coordinates are now a $n \times p$ matrix $\mathbf{X} = [\mathbf{p}_1, \cdots, \mathbf{p}_n]^T = \mathbf{V}_p\Lambda_p^{\frac{1}{2}}$.

Figure 4 – The classical Multidimensional scaling algorithm as describe by Birchfield in [2]

3.2 Energy based

An example for an energy-based algorithm for the position estimation is presented by Chen in [3], which is based on the work by Zicheng [16]. The big advantage here is, that the synchronization of the different microphones does not have to be that strict, which results in a probably easier setup for ad-hoc microphone arrays without much planning. The general problem setup that was tested in [3] was with human speakers and laptops as sensors, using the built in microphones and a network connection between these. The improvement compared to [16] is, that the speakers are no longer assumed to be in the same positions as the microphone, which was an unlikely scenario to begin with.

While leading to reasonable results, a major drawback of this algorithm concerning the problem dealt with here (described in 2), is the fact, that speakers and sensors were assumed to be on the same plane, thus resulting only in a 2D map of the positions, which is not suitable for the problem proposed in Section 2.

3.3 Spatial Observability Function (SOF) or Maximum Likelihood (ML)

An ML style approach is proposed by Aarabi in [1], and used for orientation alignment by Valente [14]. Aarabi introduces a spatial observability Function (SOF), together with the assumption that the positions of the sound sources are known, which is used as an estimator for the positions of the microphone arrays, together with their orientation, which is an interesting addition, thinking of the fact that this could maybe improve the positioning of the sensors when using the Oktava microphones, after estimating the source positions, which can then be used to refine the microphone array positions and orientation.

The algorithm uses the sources sending out signals to find the most likely positions of the microphone arrays with respect to the sound events produced by the sources, which are assumed to be located at a fixed location. The SOFs can be derived using different methods, for example array steered response power.

The SOFs for each microphone array are assumed to be scaled to values between 1 and 0, with values close to 1 representing a likely position with a sound source. Again, only a 2D problem is solved, which leaves three values to estimate: x and y coordinate, and the orientation as angle Θ.

The SOFs are used to find a measure for the overall observability (OSO) of a source at all discrete spatial locations in an analyzed area. The OSO value can furthermore be used to define an instantaneous measure in case the SOFs are changing rapidly, which is called the Q-observance measure (QOSO). The QOSO then can be used to find the probability that the spatial likelihood function (SLF) for certain microphone location and orientation gives the actual sensor probability. The SLF can be found using for example cross correlation techniques, the results of which (the quality of the SLF) directly effects the results of the array localization.

The fact that the sources should be known plus the fact that the derivation of the SOFs is more complex than the TOF based algorithms, also deemed this implementation undesirable.

3.4 Time-of-flight (TOF) based or Least Squares (LS)

The time of flight (TOF) based algorithms are all based on a formula equivalent to

$$t_{ij} - \tau_j = \frac{||\mathbf{r}_i - \mathbf{s}_j||}{c}, \tag{1}$$

which connects the TODs τ_j with the measured TOAs t_{ij} to obtain the TOFs, and the source and microphone coordinates, which solve upper equation. A solution can be obtained by solving the non-linear least squares problem

$$\hat{\mathbf{r}}_i, \hat{\mathbf{s}}_j = \underset{\mathbf{r}_i, \mathbf{s}_j}{\operatorname{argmin}} \left\{ \left(t_{ij} - \tau_j - \frac{||\mathbf{r}_i - \mathbf{s}_j||}{c} \right)^2 \right\}, \tag{2}$$

with a possible additional variable ϵ_i, representing the onset times and internal delays by each individual microphone, resulting in

$$\hat{\mathbf{r}}_i, \hat{\mathbf{s}}_j = \underset{\mathbf{r}_i, \mathbf{s}_j}{\operatorname{argmin}} \left\{ \left(t_{ij} - \tau_j - \frac{||\mathbf{r}_i - \mathbf{s}_j||}{c} - \epsilon_i \right)^2 \right\}. \tag{3}$$

While leading to a more realistic representation of the overall problem, the additional ϵ_i complicates the computation, because the onset times for each microphone are in general not identical. Crocco [4, 5] assumes that the TOAs and TODs are known, thus assuming that the TOFs are known.

Similar to Crocco [4, 5], the Free Source Method (FrSM) proposed by Khanal [11] needs known TOFs, but also gives a possible solution, describing a special source object with a sensor attached in the acoustic center (of the loudspeaker), thus eliminating the need to compute the times of departure (TODs) at the sources (the start times of the sound events). Using these special sources, the only problem arising would be the fact that they are not perfectly coincident, thus a small time correction (distance from the source to the reference microphone attached to it) would be needed.

Due to the fact that for TOF based position estimation very few constraints lead to rather good results, a TOF based approach was implemented. The used approaches, [4, 7, 12] and [5] will be describe in section 4.

4 Implementation

In figure 5 an overview over the problem is given, whereas only the TOD estimation and the Position estimation algorithm were implemented, assuming that the TOAs were already retrieved with an error in the magnitude of 4 samples and more at a sample rate of 48kHz. Although two Algorithms were implemented, only the second one was used, due to the fact that the first one, proposed by Pollefeys [12] was yielding results that were not accurate enough to be used in the position estimation stage.

This might be caused by the fact that only the TOD estimation part was implemented, and not the computation of the positions, which was different from the one proposed by Crocco. Still, the final results achieved by Pollefeys [12] were similar to those achieved by Gaubitch [7] and Crocco [4, 5].

A complete program would also need the first two stages, recording and TOA estimation. The recoding stage would need to perform a synchronized recording of all the microphone signals for each source. The TOA estimation stage would then need to derive the TOAs from these recordings. These TOAs are then fed to the TOD estimation stage, which tries to find the TODs fitting best to the estimated TOAs. The last stage performs the actual estimation of the source and microphone coordinates from the TOFs (the difference between the respective TOAs and TODs).

4.1 TOD estimation

For the TOD estimation two algorithms were tested, one proposed by Pollefeys in [12] and another proposed by Gaubitch in [7]. The one by Pollefeys was chosen because it requires very little computation and no iterations, but unfortunately did not lead

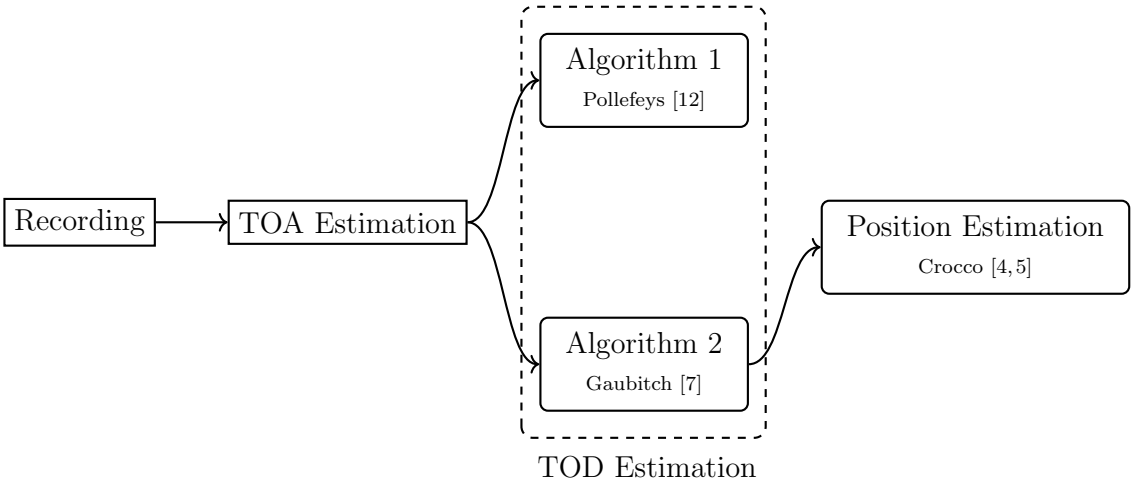


Figure 5 – Overview over the different blocks that need to be implemented for acquiring an estimate of microphone and source positions.

to satisfying results (it is still described briefly). The results of the second algorithm were used for the computations.

4.1.1 Algorithm 1

The solution proposed by Pollefeys uses a vector/matrix representation of equation 1 (without the ϵ_i), using the following vectors for the locations of the j-th source and the i-th microphone respectively,

$$\mathbf{s}_j = \begin{bmatrix} x_j & y_j & z_j \end{bmatrix}^T \tag{4}$$

$$\mathbf{r}_i = \begin{bmatrix} X_i & Y_i & Z_i \end{bmatrix}^T, \tag{5}$$

which were then inserted into equation 1 results

$$t_{ij} = \frac{||\mathbf{r}_i - \mathbf{s}_j||}{c} + \tau_j \tag{6}$$

$$c^2(t_{ij} - \tau_j)^2 = ||\mathbf{r}_i - \mathbf{s}_j||^2 \tag{7}$$

$$c^2(t_{ij} - \tau_j)^2 = (x_j - X_i)^2 + (y_j - Y_i)^2 + (z_j - Z_i)^2 \tag{8}$$

$$c^2(t_{ij}^2 - 2t_{ij}\tau_j + \tau_i^2) = \mathbb{S}_j^T \mathbb{R}_i \tag{9}$$

$$c^2(t_{ij}^2 - 2t_{ij}\tau_j) = \mathbb{S}_j^T \mathbb{R}_i - c^2\tau_j^2 \tag{10}$$

The right side of equation 8 can be shortened to the form of $\mathbb{S}_j^T \mathbb{R}_i$, by expanding the square terms and using $\mathbb{S}_j$ and $\mathbb{R}_i$ of the following form

$$\mathbb{S}_j = \begin{bmatrix} \mathbf{s}_j^T \mathbf{s}_j & -2x_j & -2y_j & -2z_j & 1 \end{bmatrix}^T \tag{11}$$

$$\mathbb{R}_i = \begin{bmatrix} 1 & X_i & Y_i & Z_i & \mathbf{r}_i^T \mathbf{r}_i \end{bmatrix}^T, \tag{12}$$

from which the TODs can be retrieved by writing equation 10 in the form of

$$\mathbf{T} = \mathbf{A} + \mathbf{DB} = \begin{bmatrix}\mathbf{I} & \mathbf{D}\end{bmatrix} \begin{bmatrix}\mathbf{A} \\ \mathbf{B}\end{bmatrix}, \tag{13}$$

where $\mathbf{T} = \{t_{ij}^2 - 2t_{ij}\tau_j\}$, $\mathbf{A} = \{t_{ij}^2\}$, $\mathbf{B} = \{-2t_{ij}\}$ and $\mathbf{D}$ is a diagonal matrix $\mathbf{D} = \operatorname{diag}\{\tau_j\}$, which is unknown. Due to the fact that the two matrices $\mathbb{S}$ and $\mathbb{R}$, containing the vectors $\mathbb{S}_j$ and $\mathbb{R}_i$ respectively, are of rank 5, also the product of these two matrices has to be of rank 5, and thus also $\mathbf{T}$ is of rank 5. Using the fact that the first row of $\mathbb{R}$ contains only ones, a linear combination of five rows has to exist to achieve that. Thus, any five rows of the matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{D}$ can be used to build the matrices $\bar{\mathbf{A}}$,$\bar{\mathbf{B}}$ and $\bar{\mathbf{D}}$

$$\bar{\mathbf{A}} = \begin{bmatrix}\mathbf{A}_{i_1}^T & \mathbf{A}_{i_2}^T & \mathbf{A}_{i_3}^T & \mathbf{A}_{i_4}^T & \mathbf{A}_{i_5}^T\end{bmatrix}^T \tag{14}$$

$$\bar{\mathbf{B}} = \begin{bmatrix}\mathbf{B}_{i_1}^T & \mathbf{B}_{i_2}^T & \mathbf{B}_{i_3}^T & \mathbf{B}_{i_4}^T & \mathbf{B}_{i_5}^T\end{bmatrix}^T \tag{15}$$

$$\bar{\mathbf{D}} = \begin{bmatrix}\mathbf{D}_{i_1}^T & \mathbf{D}_{i_2}^T & \mathbf{D}_{i_3}^T & \mathbf{D}_{i_4}^T & \mathbf{D}_{i_5}^T\end{bmatrix}^T \tag{16}$$

which then will be inserted into

$$\mathbf{c}^T \begin{bmatrix}\mathbf{I} & \bar{\mathbf{D}}\end{bmatrix} \begin{bmatrix}\bar{\mathbf{A}} \\ \bar{\mathbf{B}}\end{bmatrix} = \mathbf{X} \begin{bmatrix}\bar{\mathbf{A}} \\ \bar{\mathbf{B}}\end{bmatrix} = \begin{bmatrix}1 & \cdots & 1\end{bmatrix} \tag{17}$$

where $\mathbf{c}^T$ is a vector that produces the vector containing only ones on the right hand side. The matrix $\mathbf{X}$ in equation 17 then can be used to compute the TODs τ_{j_k} according to

$$\tau_{j_k} = \frac{X_{k+5}}{X_k}, \tag{18}$$

The matrix $\mathbf{X}$ can be recovered by inverting the matrix $\begin{bmatrix}\bar{\mathbf{A}} & \bar{\mathbf{B}}\end{bmatrix}^T$, which leads to the results for the times of departure for five sources at a time. The rows of the matrices $\mathbf{A}$ and $\mathbf{B}$ can (and have to be used more than once when the number of sources is not divisible by 5) be used more than once to build the matrices $\begin{bmatrix}\bar{\mathbf{A}} & \bar{\mathbf{B}}\end{bmatrix}^T$, but that does not lead to additional information.

Unfortunately, this approach did not lead to results that were accurate enough to use them with the position estimation stage. This could either be caused by the fact that the position estimation performed by Pollefeys uses a different approach than the one that was used for the position estimation in this work.

Furthermore, the number of sources for this approach is also given by the matrices $\mathbb{R}$ and $\mathbb{S}$, which have to be of full rank (i.e. $\operatorname{rank}\{\mathbb{R}\} = \operatorname{rank}\{\mathbb{S}\} = 5$)

4.1.2 Algorithm 2

Compared to the TOD estimation by Pollefeys in section 4.1.1, Gaubitch uses the complete equation 1 for the computation of the TOFs from each source to each microphone, with ϵ_i containing the onset times δ_i for each microphone and additional measurement noise ν_{ij}.

$$t_{ij} = \frac{||\mathbf{r}_i - \mathbf{s}_j||}{c} + \tau_j + \delta_i + \nu_{ij} \tag{19}$$

The measurement noise is assumed to be zero for the derivations, but will be included in the simulations. The onset times are the times before the first source sound event starts, which will also be assumed to be equal, which is only true for the case of perfectly synchronized microphones.

Pollefeys uses a matrix notation for equation 19, which results in matrices containing the coordinates of the sources and the microphones

$$\hat{\mathbf{R}}, \hat{\mathbf{S}} = \underset{\mathbf{R},\mathbf{S}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{\mathcal{I}} \sum_{j=1}^{\mathcal{J}} \left(\frac{||\mathbf{r}_i - \mathbf{s}_j||}{c} - t_{ij} \right)^2 \right\}, \tag{20}$$

where the matrices $\mathbf{R}$, $\hat{\mathbf{R}}$, $\mathbf{S}$ and $\hat{\mathbf{S}}$ have the form

$$\mathbf{R} = \begin{pmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \vdots \\ \mathbf{r}_i^T \\ \vdots \\ \mathbf{r}_{\mathcal{I}-1}^T \\ \mathbf{r}_{\mathcal{I}}^T \end{pmatrix} \quad \hat{\mathbf{R}} = \begin{pmatrix} \hat{\mathbf{r}}_1^T \\ \hat{\mathbf{r}}_2^T \\ \vdots \\ \hat{\mathbf{r}}_i^T \\ \vdots \\ \hat{\mathbf{r}}_{\mathcal{I}-1}^T \\ \hat{\mathbf{r}}_{\mathcal{I}}^T \end{pmatrix} \quad \mathbf{S} = \begin{pmatrix} \mathbf{s}_1^T \\ \mathbf{s}_2^T \\ \vdots \\ \mathbf{s}_j^T \\ \vdots \\ \mathbf{s}_{\mathcal{J}-1}^T \\ \mathbf{s}_{\mathcal{J}}^T \end{pmatrix} \quad \hat{\mathbf{S}} = \begin{pmatrix} \hat{\mathbf{s}}_1^T \\ \hat{\mathbf{s}}_2^T \\ \vdots \\ \hat{\mathbf{s}}_j^T \\ \vdots \\ \hat{\mathbf{s}}_{\mathcal{J}-1}^T \\ \hat{\mathbf{s}}_{\mathcal{J}}^T \end{pmatrix}, \tag{21}$$

where matrices or vectors with a hat are the estimated coordinate vectors, or the matrices containing the respective estimated coordinate vectors.

To estimate the onset times for the sound events (assuming $c = 1$ for simplification), the right hand side of equation 20 can be expanded to

$$\mathbf{r}_i^T \mathbf{r}_i + \mathbf{s}_j^T \mathbf{s}_j - 2\mathbf{r}_i^T \mathbf{s}_j = t_{ij}^2 + \tau_j^2 + \delta_i^2 - 2(t_{ij}\tau_j + t_{ij}\delta_i - \delta_i\tau_j), \tag{22}$$

then, subtracting upper equation 22 with $i = 1$ inserted leads to

$$\mathbf{r}_i^T\mathbf{r}_i - \mathbf{r}_1^T\mathbf{r}_1 - 2(\mathbf{r}_i - \mathbf{r}_1)^T\mathbf{s}_j = t_{ij}^2 - t_{1j}^2 + \delta_i^2 - \delta_1^2 - 2(t_{ij}(\delta_i + \tau_j) + \\ + 2t_{1j}(\delta_1 + \tau_j) + 2(\delta_i - \delta_1)\tau_j, \tag{23}$$

and by again subtracting the last equation 23 with $j = 1$ leads to

$$-2(\mathbf{r}_i - \mathbf{r}_1)^T(\mathbf{s}_j - \mathbf{s}_1) = t_{ij}^2 - t_{1j}^2 - t_{i1}^2 + t_{11}^2 \\ - 2\delta_i(t_{ij} - t_{i1}) + 2\delta_1(t_{1j} - t_{11}) \\ - 2\tau_j(t_{ij} - t_{1j}) + 2(\delta_i - \delta_1)\tau_j \tag{24}$$

The final equation 24 can be expressed as

$$-2\bar{\mathbf{R}}\bar{\mathbf{S}}^T = \mathbf{T} + \mathbf{A}(\mathbf{p}) + \mathbf{\Gamma} = \hat{\mathbf{T}}, \tag{25}$$

where $\mathbf{T} = \{t_{ij}^2 - t_{1j}^2 - t_{i1}^2 + t_{11}^2\}_{i,j}$ with $i = 2, ..., \mathcal{I}$ and $j = 2, ..., \mathcal{J}$ contains the squared $\mathbf{\Gamma}_{ij} = \{2(\delta_i - \delta_1)\}_{ij}$, and $\mathbf{A}(\mathbf{p}) = \mathcal{C}^{-1}_{\mathcal{I}-1\times\mathcal{J}-1}\{\mathbf{W}\cdot\mathbf{p}\}$, where $\mathcal{C}^{-1}_{\mathcal{I}-1\times\mathcal{J}-1}$ is equivalent to the Matlab command `A_p = reshape(W*p,I-1,J-1)`. $\hat{\mathbf{T}}$ would contain the TOFs relative to the first sound event.

The vector $\mathbf{p}$ is the vector containing the onset times δ_i for each microphone and the times of departure τ_j for each source, which are relative to the TOD of the first source $\tau_1 = 0$ (which is assumed to be zero). The whole p-vector has the form $\mathbf{p} = \begin{bmatrix}\delta_1 & \delta_2 & \cdots & \delta_{\mathcal{I}} & \tau_2 & \tau_3 & \cdots & \tau_{Jj}\end{bmatrix}$. Due to the assumption that all the onset delays are equal, the vector $\mathbf{p}$ can be shortened by using only a single δ, which helps enforcing the equality of all the delays such that $\delta = \delta_1 = \delta_2 = ... = \delta_{\mathcal{I}}$. The used vector $\mathbf{p} = \begin{bmatrix}\delta & \tau_2 & \tau_3 & \cdots & \tau_{Jj}\end{bmatrix}$, which also reduces the computational time for the estimation of the microphone and source locations, by forcing the individual δ_i to be equal.

Using the fact that the multiplication of the two matrices containing the microphone and source coordinates $\bar{\mathbf{R}}\bar{\mathbf{S}}^T$ results in a rank-3 matrix, also the right side of equation 25 has to be of rank-3. Thus, rank approximation of the right hand side can be used to achieve a LS solution for the estimation of the vector $\mathbf{p}$, $\hat{\mathbf{p}}$. This estimated vector can be used to compute an estimate of the matrix $\hat{\mathbf{T}} = \mathbf{T} + \mathbf{A}(\hat{\mathbf{p}}) + \hat{\mathbf{\Gamma}}$.

A rank approximation can be performed by using a singular value decomposition, truncating the diagonal matrix containing the singular values, such that only the three largest singular values (the matrix turns into a 3×3 matrix) remain.

The rank approximation is then used to minimize cost function

$$\hat{\mathbf{p}}^{(n+1)} = \underset{\hat{\mathbf{p}}^{(n)}}{\operatorname{argmin}} \left\{ ||\mathbf{E}(n) - \mathbf{A}(\hat{\mathbf{p}}^n) - \hat{\mathbf{\Gamma}}||_F^2 + \lambda||\hat{\mathbf{T}}^{(n)}||_F^2 \right\}, \tag{26}$$

iteratively, where the error matrix $\mathbf{E}^{(n)} = \tilde{\mathbf{T}}^{(n)} - \mathbf{T}$, with $\tilde{\mathbf{T}}^{(n)}$ being the rank-3 approximation of the matrix $\mathbf{T}$ at iteration n. λ is a Lagrange multiplier, which, according to Gaubitch [7], has to be set to zero for the solution to fully converge. This has to be done after the update $\hat{\mathbf{T}}^{(n)}$ from one iteration to the next falls below a threshold (the change is computed using the Froebenius norm $||\hat{\mathbf{T}}^{(n)}||_F$). After setting λ to zero, a least squares solution can be used to compute $\hat{\mathbf{p}}$ iteratively,

$$\hat{\mathbf{p}}^{(n+1)} = \mathbf{W}^{+}\mathbf{e}^{(n)}, \tag{27}$$

with $\mathbf{W}^{+}$ being the pseudo inverse of $\mathbf{W}$, and $\mathbf{e}^{(n)}$ being the vectorized Matrix $\mathbf{E}^{(n)}$, which would be the inverse $\mathcal{C}\{\cdots\}$-operator, which represents the `reshape()` function of Matlab. When using the LS equation for the optimization, a certain number of iterations should be used as stop condition. Another possibility would be to again compute the Froebenius norm of the matrix $\hat{\mathbf{T}}^{(n)}$.

4.2 Position Estimation

The estimation was reformulated into a closed form solution in [4, 5] which does not require an iterative approach. This was shown by Crocco by introducing the additional constraint that one source and one microphone have to coincide (this can be relaxed a little for a better applicability as shown in section 5). These coinciding microphones are then used to define the coordinate origin, and all the other positions will be in respect to these two. The solution shown by Crocco will be described in this section.

Crocco starts with the equation for the TOF from the j-th source to the i-th microphone

$$||\mathbf{r}_i||^2 + ||\mathbf{s}_j||^2 - 2\mathbf{r}_i \cdot \mathbf{s}_j = d_{ij}^2 \tag{28}$$

which can be transformed into a bilinear form with the same steps as shown in section 4.1.1 and finally results in

$$-2\mathbf{R}\mathbf{S}^T = \mathbf{D}, \tag{29}$$

which is the same as equation 25, with $\mathbf{D} = \hat{\mathbf{T}} \cdot c^2$ and $\hat{\mathbf{T}}$ being the matrix containing the corrected TOFs, and $\mathbf{D}$ containing the squared distances $\{\mathbf{D}\}_{ij} = d_{ij}^2 - d_{1j}^2 - d_{i1}^2 + d_{11}^2$ for the ij-th element of the matrix $\mathbf{D}$, with i denoting the row index and j the column index.

Again, equation 29 can be solved using singular value decomposition and the fact, that $\mathbf{D}$ has to have rank-3, caused by $\mathbf{R}$ and $\mathbf{S}$ being of rank-3.

$$\mathbf{D} = \mathbf{UVW} \tag{30}$$

In the case of no measurement noise, the diagonal matrix $\mathbf{V}$ containing the singular values, should only contain three singular values which are not zero. For the real case with measurement noise, rank approximation has to be used again (truncating the matrix containing the singular values). Equation 30 is then used to derive a cost function that should be minimized with respect to a certain invertible matrix $\mathbf{C}$, from which the matrices $\mathbf{R}$ for the microphone coordinates and $\mathbf{S}$ for the source coordinates can be retrieved. In equation 30, the matrix $\mathbf{C}$ can be inserted, which would result in the following equations for $\mathbf{R}$ and $\mathbf{S}$

$$\mathbf{D} = \mathbf{UCC}^{-1}\mathbf{VW} = -2\mathbf{RS}^T \tag{31}$$

where

$$\mathbf{R} = \mathbf{UC} \tag{32}$$
$$-2\mathbf{S}^T = \mathbf{C}^{-1}\mathbf{VW} \tag{33}$$

holds.

These new definitions of $\mathbf{R}$ and $\mathbf{S}$ can be inserted into the cost function obtained in the rest of this section, using equation 28

$$\underset{\mathbf{r}_i,\mathbf{s}_j}{\operatorname{argmin}} \left\{ \sum_{i=1}^{\mathcal{I}} \sum_{j=1}^{\mathcal{J}} \left(||\mathbf{r}_i||^2 + ||\mathbf{s}_j||^2 - 2\mathbf{r}_i\mathbf{s}_j - d_{ij}^2 \right)^2 \right\}, \tag{34}$$

which leads to

$$\underset{\mathbf{r}_i,\mathbf{s}_j}{\operatorname{argmin}} \left\{ \sum_{i=2}^{\mathcal{I}} \sum_{j=2}^{\mathcal{J}} \left(||\mathbf{r}_i||^2 - ||\mathbf{r}_1||^2 + ||\mathbf{s}_j||^2 - 2(\mathbf{r}_i - \mathbf{r}_1)\mathbf{s}_j - d_{ij}^2 + d_{1j}^2 \right)^2 \right\}, \tag{35}$$

by subtracting the equation for $i = 1$ and then to

$$\underset{\mathbf{r}_i,\mathbf{s}_j}{\operatorname{argmin}} \left\{ \sum_{i=2}^{\mathcal{I}} \sum_{j=2}^{\mathcal{J}} \left(||\mathbf{r}_i||^2 - 2(\mathbf{r}_i - \mathbf{r}_1) \cdot (\mathbf{s}_j - \mathbf{s}_1) - 2 \cdot r_{i,x} \cdot s_{1,x} - d_{ij}^2 + d_{1j}^2 \right)^2 \right\}, \tag{36}$$

by expanding the term

$$-2(\mathbf{r}_i - \mathbf{r}_1)\mathbf{s}_j = -2(\mathbf{r}_i - \mathbf{r}_1) \cdot (\mathbf{s}_j - \mathbf{s}_1) - 2 \cdot (\mathbf{r}_i - \mathbf{r}_1) \cdot \mathbf{s}_1 \tag{37}$$

$$= -2(\mathbf{r}_i - \mathbf{r}_1) \cdot (\mathbf{s}_j - \mathbf{s}_1) - 2 \cdot r_{i,x} \cdot s_{1,x} \tag{38}$$

and inserting the constraints (mentioned above) that the first source and microphone with coordinates

$$\mathbf{s}_1 = \begin{bmatrix} s_{1,x} \\ s_{1,y} \\ s_{1,z} \end{bmatrix} = \begin{bmatrix} s_{1,x} \\ 0 \\ 0 \end{bmatrix} \tag{39}$$

$$\mathbf{r}_1 = \begin{bmatrix} r_{1,x} \\ r_{1,y} \\ r_{1,z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{40}$$

are concident up to a shift along the x-axis by $t_{1,x}$.

To achieve the closed form solution, also $s_{1,x} = 0$ is assumed. The cost function can thus be reduced to

$$\underset{\mathbf{r}_i, \mathbf{s}_j}{\operatorname{argmin}} \left\{ \sum_{i=2}^{\mathcal{I}} \sum_{j=2}^{\mathcal{J}} \left(||\mathbf{r}_i||^2 - 2(\mathbf{r}_i - \mathbf{r}_1) \cdot (\mathbf{s}_j - \mathbf{s}_1) - d_{ij}^2 + d_{1j}^2 \right)^2 \right\}. \tag{41}$$

Now the vectors for the coordinates of the sources and microphones can be replaced by those obtained using the SVD (equation 32 and 33), to achieve the cost function depending on only the 3×3-matrix $\mathbf{C}$ to minimize,

$$\underset{\mathbf{C}}{\operatorname{argmin}} \left\{ \sum_{i=2}^{\mathcal{I}} \sum_{j=2}^{\mathcal{J}} \left(\{\mathbf{UC}\}_{i,x}^2 + \{\mathbf{UC}\}_{i,y}^2 + \{\mathbf{UC}\}_{i,z}^2 + \{\mathbf{UVW}\}_{ij} - d_{ij}^2 + d_{1j}^2 \right)^2 \right\}. \tag{42}$$

Due to the fact that the matrix $\mathbf{C}$ is a real square matrix, a QR-decomposition can be performed in it, resulting in a $\mathbf{C} = \mathbf{Q}\mathfrak{R}$. The rotation matrix $\mathbf{Q}$ can be chosen to be the identity matrix $\mathbf{I}$ (a rotation of the resulting coordinates does not effect the relative positions of microphones and sources) and an upper triangular matrix $\mathfrak{R}$ of the form

$$\mathfrak{R} = \begin{pmatrix} r_1 & r_2 & r_3 \\ 0 & r_4 & r_5 \\ 0 & 0 & r_6 \end{pmatrix}, \tag{43}$$

which has to be obtained by minimizing the cost function equation 42, with $\mathbf{C} = \mathbf{Q}\mathfrak{R} = \mathbf{I}\mathfrak{R} = \mathfrak{R}$ inserted.

The final cost function is

$$\underset{\mathfrak{R}}{\operatorname{argmin}} \left\{ \sum_{i=2}^{\mathcal{I}} \sum_{j=2}^{\mathcal{J}} \Big(\{\mathbf{U}\mathfrak{R}\}_{i,x}^2 + \{\mathbf{U}\mathfrak{R}\}_{i,y}^2 + \{\mathbf{U}\mathfrak{R}\}_{i,z}^2 + \{\mathbf{UVW}\}_{ij} - d_{ij}^2 + d_{1j}^2 \Big)^2 \right\}. \tag{44}$$

which can be expanded into

$$\hat{\mathfrak{R}} = \underset{\mathfrak{R}}{\operatorname{argmin}} \sum_{i=1}^{\mathcal{I}} \sum_{j=1}^{\mathcal{J}} \{(r_1 u_{i1})^2 + (r_1 u_{i1} + r_4 u_{i2})^2 + (r_3 u_{i1} + r_5 u_{i2} + r_6 u_{i3})^2 \tag{45}$$

$$+ \{\mathbf{UVW}\}_{ij} - d_{i+1,j+1}^2 + d_{1,j+1}^2\}^2, \tag{46}$$

and simplified by using the following vectors and matrices,

$$\xi_i = \begin{bmatrix} u_{i1}^2 & u_{i2}^2 & u_{i3}^2 & 2u_{i1}u_{i2} & 2u_{i1}u_{i3} & 2u_{i2}u_{i3} \end{bmatrix}^T \tag{47}$$

$$\mathbf{f} = \begin{bmatrix} r_1^2 + r_2^2 + r_3^2 & r_4^2 + r_5^2 & r_6^2 & r_2 r_4 + r_4 r_5 & r_3 r_6 & r_5 r_6 \end{bmatrix}^T \tag{48}$$

$$\mathbf{k} = \begin{bmatrix} k_{1,1} & k_{2,1} & \ldots & k_{\mathcal{I}-1,1} & k_{1,2} & \ldots & k_{\mathcal{I}-1,\mathcal{J}-1} \end{bmatrix}^T \tag{49}$$

$$\mathbf{\Xi} = \begin{bmatrix} \xi_1 & \xi_2 & \ldots & \xi_{\mathcal{I}-1} \end{bmatrix}^T \tag{50}$$

$$\mathbf{P} = \begin{bmatrix} \mathbf{\Xi} \\ \mathbf{\Xi} \\ \ldots \\ \mathbf{\Xi} \end{bmatrix} \tag{51}$$

with

$$k_{i,j} = -\{\mathbf{UVW}\}_{ij} + d_{i+1,j+1}^2 - d_{1,j+1}^2. \tag{52}$$

The resulting linear least squares problem in $\mathbf{f}$ is

$$\hat{\mathbf{f}} = \underset{\mathbf{f}}{\operatorname{argmin}} \left\{ |\mathbf{P}\mathbf{f} - \mathbf{k}|^2 \right\} \tag{53}$$

can be solved easily by using the pseudo inverse,

$$\hat{\mathbf{f}} = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{k}. \tag{54}$$

from which the elements of the triangular matrix $\mathfrak{R}$ can be retrieved as follows.

$$r_6 = \pm\sqrt{f_3} \qquad r_5 = f_6/r_6$$
$$r_4 = \pm\sqrt{f_2 - r_5^2} \qquad r_3 = f_5/r_6$$
$$r_2 = (f_4 - r_3/r_5)/r_4) \qquad r_1 = \pm\sqrt{f_1 - r_2^2 - r_3^2}$$

Due to the square roots, eight different variations of $\mathfrak{R}$ can be retrieved, which only differ in terms of a rotation. Thus any result for $\mathfrak{R}$ can be chosen which results in the local coordinates for microphones and sources. The global coordinates could be recovered by using the microphone array to determine the direction of additional sources which are positioned on the global axis.

5 Simulation Results

In the following section, the results obtained by the implementations of the algorithms mentioned above will be presented and analyzed. The two simulation cases implemented were the case solvable by the closed form solution by Crocco [4] in section 5.1, and the more general solution, on which Gaubitch [7] focuses, in section 5.2, but which is also described by Crocco [4, 5].

To test the algorithms, different shapes of microphone and the source setups can be created for use in Matlab:

- `croc`: original setup used by Crocco [4].
- `rand`: random Gaussian distributed x-,y- and z-coordinates within a certain volume for sources and microphones.
- `octa`: dimensions of the Oktava microphones as used in the WiLMA project (sec.1.2) are used to group four microphones together, no assumptions of the phase of the received microphone signals are included, the centers of each Oktava microphone are at Gaussian random distributed x-, y- and z-coordinates.

For all cases, the TOFs between microphones and sources are estimated using the method proposed by Gaubitch in [7].

The parameters needed for creating the setups are the $x_{m/s}$,$y_{m/s}$ and $z_{m/s}$-Dimensions of the Volume in which the microphones (subscript m) and sources (subscript s) are distributed, and a possible displacement x_d, y_d and z_d of the Volume of the sources with respect to the microphones. Both Volumes in which the x,y and z-coordinates of the microphones/sources are created uniformly distributed, are centered around zero.

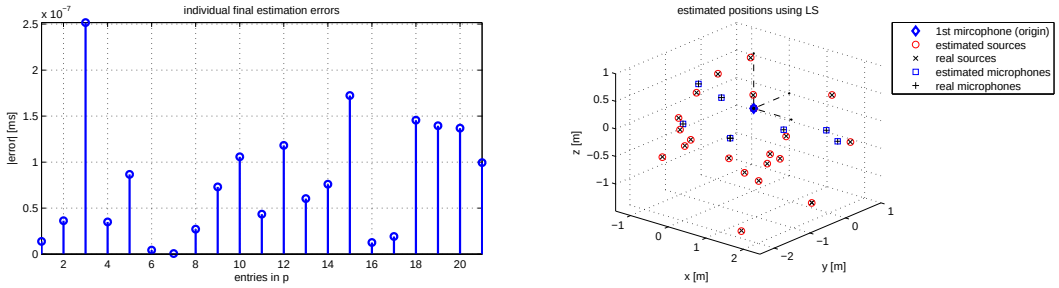
In all simulation plots, the same conventions where used. A box □ represents the estimates of the microphones and + the real microphone position. Similarly, a

circle ○ was chosen for the source position estimates, and × for the real microphone positions, thus using the more accurate symbols for the real positions and the symbols introducing some kind of uncertainty to the results for the estimated positions.

5.1 Special Case [4]

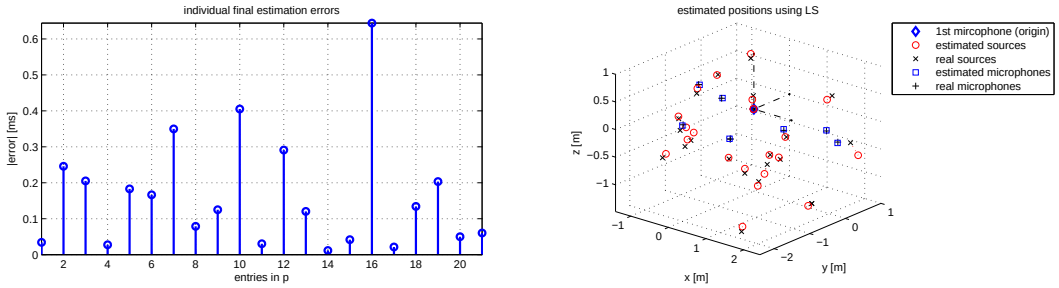
The first simulations that were performed used the same microphone and source setup as the one used by Crocco, which can be downloaded from Alessio Del Bue's Website [1]. The results for those coordinates can be seen in the following plots, figure 7 and 6, to have a reference for the quality of the implementation. In general, ϵ_p represents the error of the vector $\mathbf{p}_{est}$, which contains the onset time $\delta = \delta_1 = \delta_2 = ... = \delta_{\mathcal{I}}$ and the TODs τ_j for $j = 1...\mathcal{J}$, with respect to the optimal result $\mathbf{p}_{opt}$, representing the actual solution of the setup ($\boldsymbol{\epsilon}_p = |\mathbf{p}_{opt} - \mathbf{p}_{est}|$).

To simulate possible errors when estimating the TOAs, uniformly distributed noise was added to the TOAs in samples. For the simulations, the values $\tau_0 = 30$ and $p_i = (i-1) \cdot 3$ (a sound event every 3 seconds) were used, which should then be found by the algorithm estimating $\mathbf{p}$.



(a) ϵ_p (b) estimated positions in [m]

Figure 6 – **setup:** `croc`, **noise:** 0 samples.



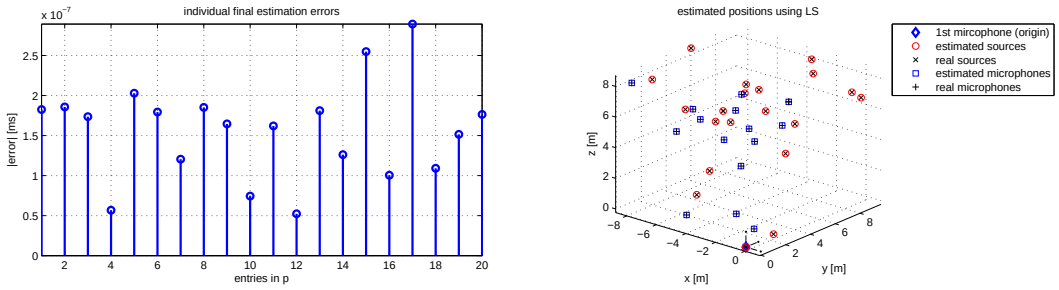
(a) ϵ_p (b) estimated positions in [m]

Figure 7 – **setup:** `croc`, **noise:** 2 samples.

1. `http://users.isr.ist.utl.pt/~adb/code/`

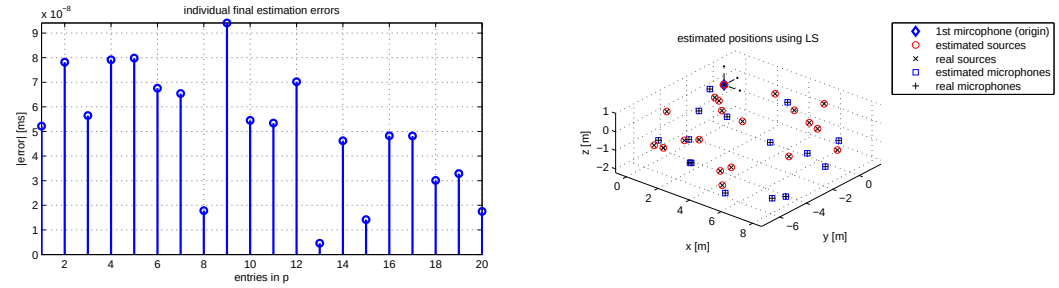
Simulations using positions from [4]

As can be seen in figure 7, despite of the significantly larger error with $\boldsymbol{\epsilon}_p$ in the magnitude of 0.7 ms for the worst case, the positions of the microphones and the sources are still estimated accurately. The only really noticeable difference between figure 7 and figure 6 (both using Croccos original data) are the slightly off-center estimated positions of the sources, whereas the estimated positions for the microphones are well centered around the signs representing the real positions.



(a) ϵ_p (b) estimated positions in [m]

Figure 8 – **setup:** `rand`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 10 \times 10 \times 10\ m$.



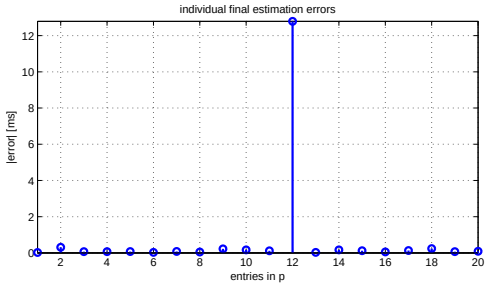
(a) ϵ_p (b) estimated positions in [m]

Figure 9 – **setup:** `rand`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 10 \times 10 \times 3\ m$.

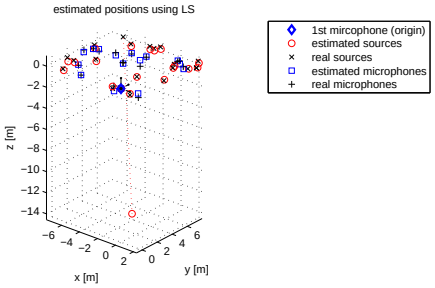
Simulations for congruent array shapes

In figures 8-11 the results for different microphone/source setups where the volumes in which the microphones and sources are places are congruent can be seen, always with 0 samples noise. The achieved results were similar for all different volume sizes, although it can be seen in fig. 10 that the convergence of $\mathbf{p}_{est}$ cannot be assumed in general, as shown in fig.10/a, where the TOD estimate of the 11th source is off by over 12 $ms \rightsquigarrow 4.08\ m$, resulting in a source estimate being very far off the real position. Additionally, fig. 10/c shows that the error worsened towards the end of the iteration of the TOD-estimation.

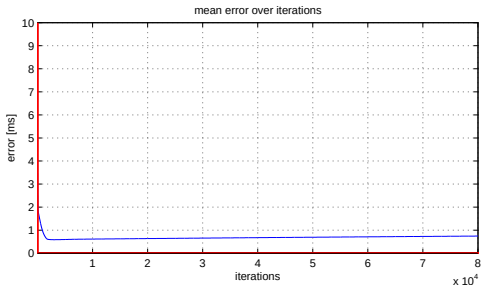
Comparing this with the results of the simulation using what could be seen as a long narrow "corridor" in fig. 11, the results of the accurate estimates obtained for the



(a) ϵ_p

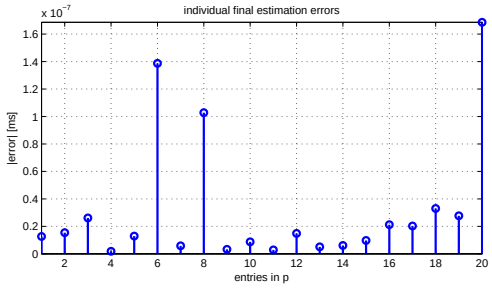


(b) estimated positions in [m]

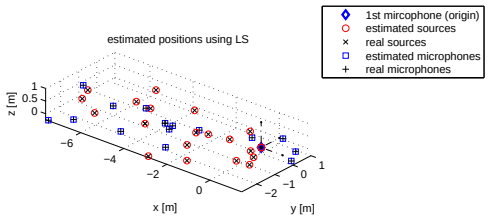


(c) average error of the estimtated **p** vector.

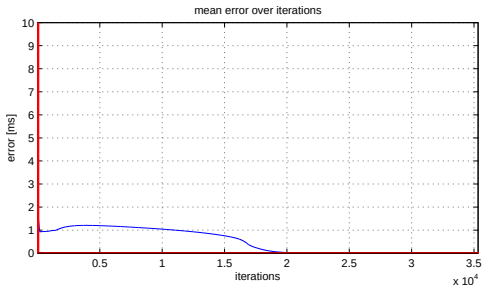
Figure 10 – **setup:** `rand`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 10 \times 10 \times 3\ m$.



(a) ϵ_p

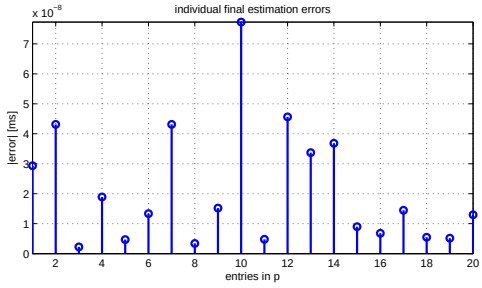


(b) estimated positions in [m]

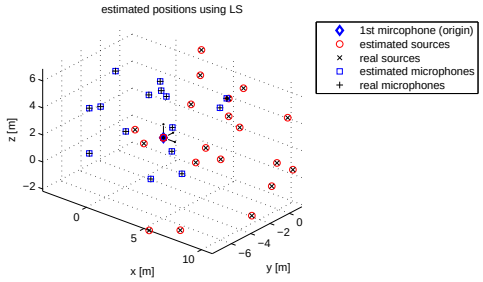


(c) average error of the estimtated **p** vector.

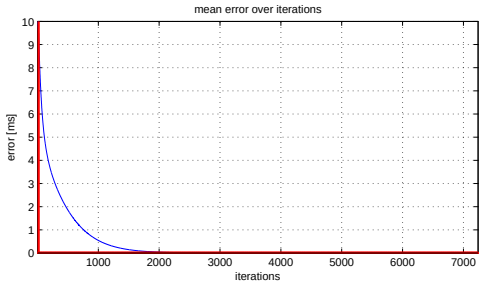
Figure 11 – **setup:** `rand`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 10 \times 3 \times 1\ m$.



(a) ϵ_p

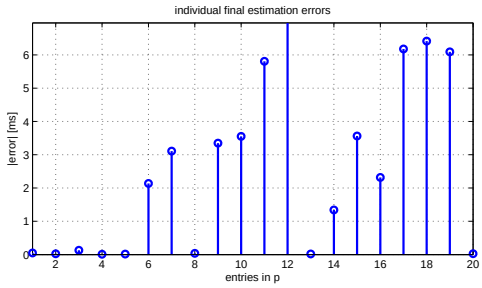


(b) estimated positions in [m]

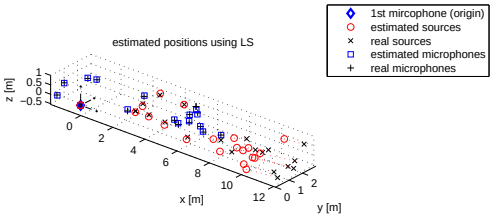


(c) average error of the estimtated **p** vector.

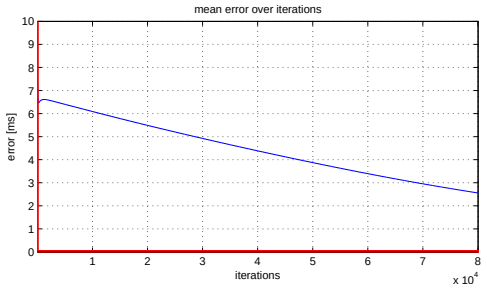
Figure 12 – **setup:** `rand`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 10 \times 3 \times 1\ m$, $x_{disp} = 5\ m$.



(a) ϵ_p

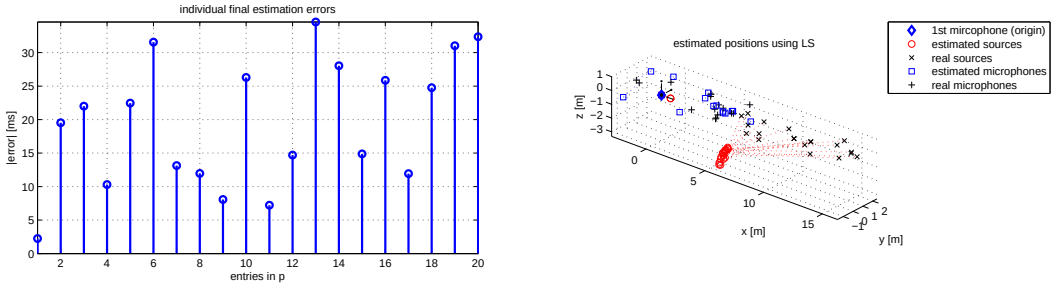


(b) estimated positions in [m]



(c) average error of the estimtated **p** vector.

Figure 13 – **setup:** `rand`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 10 \times 3 \times 1\ m$, $x_{disp} = 5\ m$.



(a) ϵ_p (b) estimated positions in [m]

Figure 14 – **setup:** `rand`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 10 \times 3 \times 1\ m$, $x_{disp} = 5\ m$.

cubic volume show an interesting convergence behavior of the TODs, which decrease in accuracy before actually converging and were at times worse than the ones which could not reach convergence. Thus it can be assumed that the instantaneous error at a certain iteration might not give information about the convergence behavior. It can furthermore be seen that the chosen number of iterations has direct influence of whether good results can be achieved: too few iterations might let the algorithm stop before converging, whereas too much iterations might lead to a deterioration of the results, all of which can be expected when dealing with any optimization problem.

Simulations for non-congruent array shapes

In figures 12-14 the results for not-congruent volumes (the displacement was 5 m in x-direction, resulting in 50% overlap of sources and microphones) in which microphones and sources are distributed are shown, again for different volume dimensions (microphone and source volumes have the same dimensions for every plot). Already in fig. 12 it can be seen that the displacement has no real effect on the estimated positions for the cubic room, but a slowing effect on the speed of convergence when estimating the TODs (compare fig. 12/c and 13/c). By further increasing the displacement, the positions of the sources cannot be estimated in a meaningful way, but the estimated microphone positions seem somewhat inclined to be in the vicinity of the real positions.

Simulations using Oktava microphones

The next set of simulations was performed to determine whether it would be possible to locate the Oktava microphones that are used with in the WiLMA project mentioned in section 1.2. What the reader has to note here is that only the ability to estimate microphones that are so close together was examined, without using any of the directional information that could be extracted from the Oktava microphone signals. Furthermore, using additional information from the Oktava microphones would also require an adaption of the used algorithms, which will later be discussed in section 7.

The used model of a possible Oktava microphone with the four capsules indicated by a dot at the edges and the center of the capsules by a dot in the origin can be seen in fig. 15. The positions of the capsules are just estimates derived from a picture, but since no real microphone recordings were used perfectly accurate positions of the capsules were not deemed necessary.

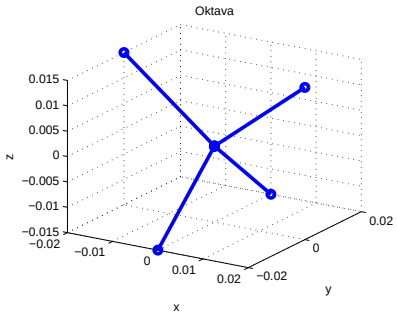
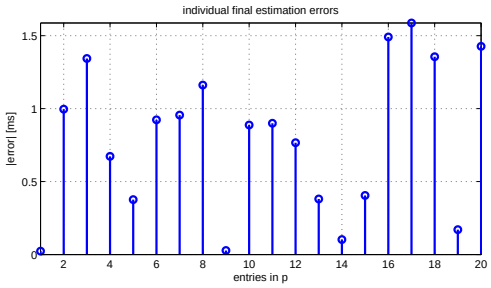
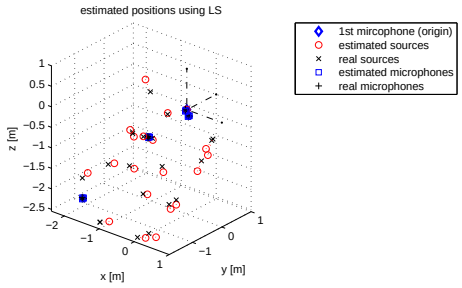


Figure 15 – Model of the Oktava 4D Ambient microphone used for the Matlab simulations in the `octa`-setup (dimensions in [m]).

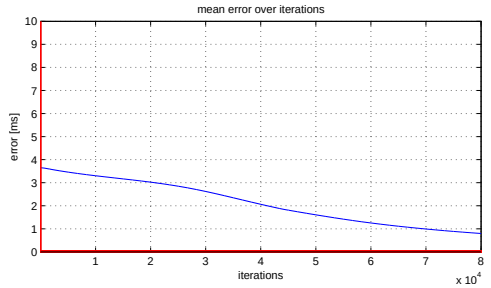
The results in fig. 16 show that also if microphones are grouped closely together, the positions can be estimated with accuracy, with the results being very close to the real positions. The source estimates are again a little off concerning the real position, although there seems to be room for improvement with the mean of ϵ_p showing a clear downward trend before the maximum for the iterations is needed.



(a) ϵ_p



(b) estimated positions in [m]



(c) average error of the estimated **p** vector.

Figure 16 – **setup:** `rand`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 3 \times 3 \times 3\ m$.

Random and pseudo-random $\mathbf{p}_{\text{init}}$

Concerning $\mathbf{p}$, an initialization $\mathbf{p}_{init}$ has to be found to begin the optimization process with. Here, four ways of initializing the elements of $\mathbf{p}$ were implemented:

- informed
- zero
- random, elements are Gaussian random numbers ranging from 0 to $2 \cdot J$
- pseudo-random, the first elements is a Gaussian random number from 0 to 1, the other elements are random integers from 0 to $2 \cdot J$ plus some Gaussian noise.

The sorting for the pseudo-random $\mathbf{p}_{\text{init}}$ was intended to achieve a similar initialization compared to the real $\mathbf{p}$ vector, which contains the ascending TODs τ_i of the sources with respect to the TOD of the first source τ_1. The range for the random values of the last two initializations of $\mathbf{p}$ are based on the assumption that sound events might occur at intervals of approximately 2 seconds, which could be adapted in case of a final implementation.

For the informed initialization, the smallest TOA was assumed to be the δ (onset time), and the difference between this δ and the next smallest TOA (for each source) is assumed to be the interval of the sound events, which results in a starting point that adapts to the surroundings. For random intervals between the sources, an initialization for the whole $\mathbf{p}_{init}$ could be found by looking at the spacing between the TOAs at one microphone (for the free-field case, i.e. no reflections). This informed initialization was used for the simulations, first because it led to the best results, and second because it does not need any additional information or assumption (only the TOAs), and would therefore also be the variant of choice for a possible real-world implementation.

5.2 General Case

If the special case of first source and microphone being coincident cannot be fulfilled, it is interesting to solve the general formulation of the problem, according to equation 42, using a nonlinear least-squares solver (NLSQ). A problem here will be the influence of the initialization of the 3×3 matrix $\mathbf{C}$ on the convergence in general and the convergence time, although no tests of this were performed in this work.

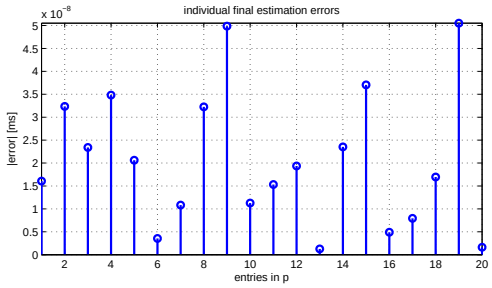
In the cases shown here, $\mathbf{C}$ was initialized with Gaussian random values, ranging from 0 to 1, based on converged solutions. Another initialization that was tried was based on the dimensions of the room, because some kind of dependency of the maximum values of the was observed, but this type of initialization was not pursued any further, because no real improvement was apparent. Another possibility would be to use an eye matrix, being equivalent to a scalar multiplication with 1.

The problem with the closed form solution is the assumption that the first source and the first microphone have to be coincident. By increasing this distance, also the mean position error (MPE) of the microphones and the sources increase, although

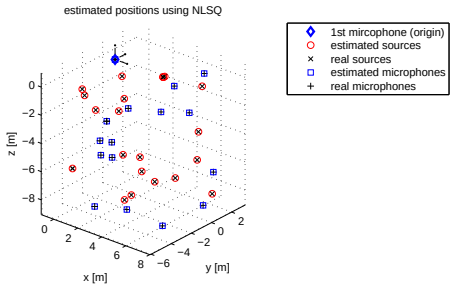
the MPE of the sources increases by much more than the one of the microphones. Thus it might be a possibility to use the closed form solution if the positions of the sources are not important, probably because they are only used for initialization purposes (clapping,...) thus not containing needed information. In the following section, the results for the general solution using a NLSQ solver will be shown, often compared to the solution obtained as a closed form solution (LS), with the knowledge that the LS solution will always contain an error. The comparison is used to obtain information about the robustness of the LS algorithm with respect to the coincidence condition.

5.2.1 Simulations for congruent array shapes

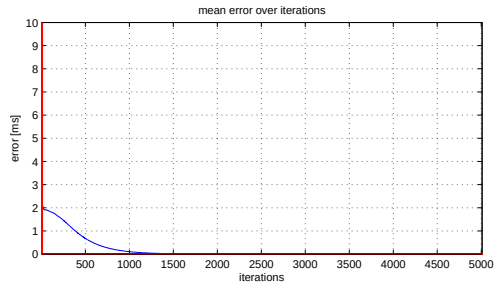
In this section, the same simulations as for the special solution are performed, but this time for $d_x = 4\ m$, in differently shaped volumes (seen in figure 17, 18, 19 and 17) and with more or less overlap of the volume of the sources and microphones. The results that are achieved are similar to those obtained by the LS algorithm, but what should be noted is that this time noise levels of 0 and 10 *samples* were used, to increase the visible changes. Additionally, a comparison of the MPEs of the LS and the NLSQ solution are compared in one plot, to give some measure of the quality improvement when using the correct algorithm.



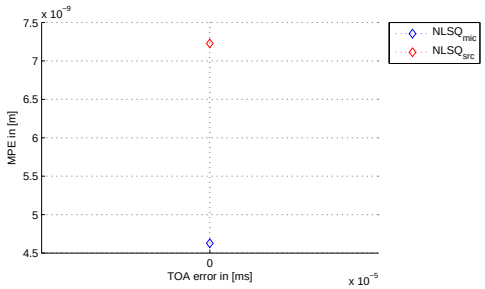
(a) ϵ_p



(b) estimated positions in [m]



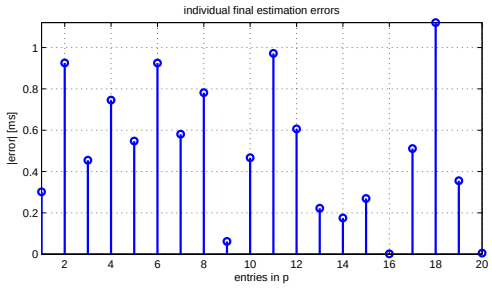
(c) average error of the estimated **p** vector.



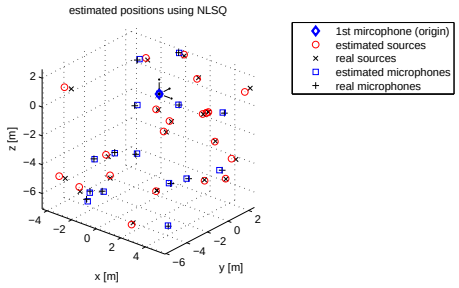
(d) MPE of the NLSQ solution.

Figure 17 – **setup:** `rand`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 10 \times 10 \times 10\ m$.

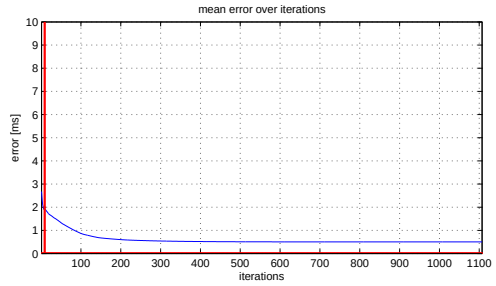
Especially when looking at figure 20/c, it can again be seen that the convergence of the TOD estimation can not be guaranteed. The estimated TODs clearly decrease,



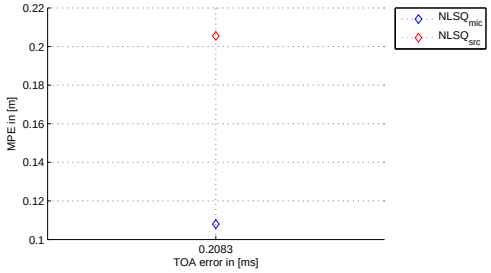
(a) ϵ_p



(b) estimated positions in [m]

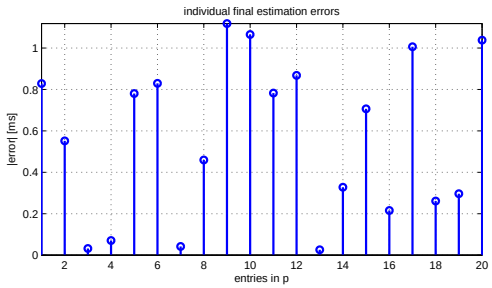


(c) average error of the estimated **p** vector.

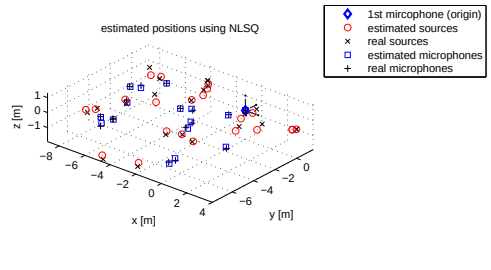


(d) MPE of the NLSQ solution.

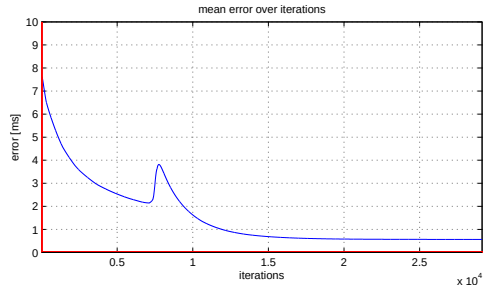
Figure 18 – **setup:** `rand`, **noise:** 10 samples, **size:** $x_R \times y_R \times z_R = 10 \times 10 \times 10\ m$.



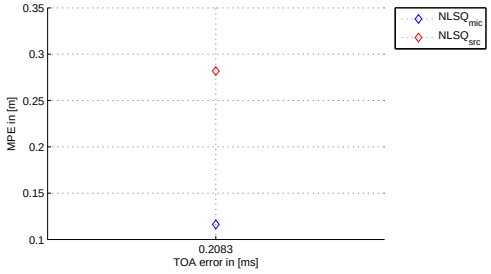
(a) ϵ_p



(b) estimated positions in [m]

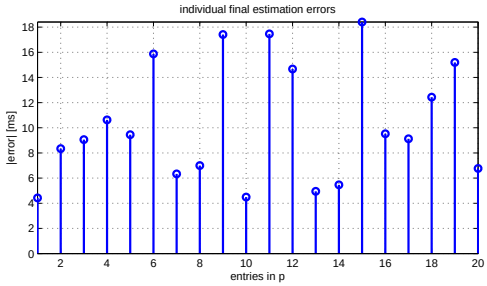


(c) average error of the estimated **p** vector.

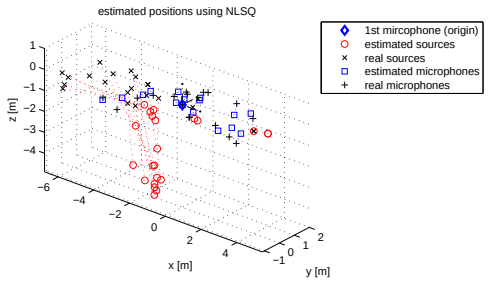


(d) MPE of the NLSQ solution.

Figure 19 – **setup:** `rand`, **noise:** 10 samples, **size:** $x_R \times y_R \times z_R = 10 \times 10 \times 3\ m$.

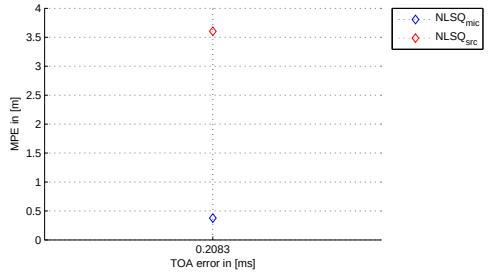


(a) ϵ_p



(b) estimated positions in [m]

(c) average error of the estimated **p** vector.



(d) MPE of the NLSQ solution.

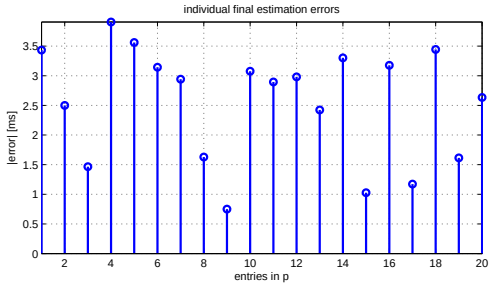
Figure 20 – **setup:** `rand`, **noise:** 10 samples, **size:** $x_R \times y_R \times z_R = 10 \times 3 \times 1\ m$.

but the NLSQ solution still achieves results, although this was not always the case for this shape of volumes.

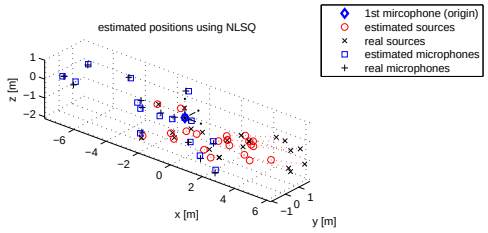
What seems to be the case here, is that when the directions from the sources to the microphones become more and more parallel, the performance of the algorithms decreases rapidly, thus calling for a reduction of the dimensions of the problem to a 2-dimensional estimation problem. The 2D solution could then again be rotated arbitrarily in the 3-dimensional space to find the best fit for the given setup, thus being restricted only in such a way, that sources and microphones should be close to or on a plane in the 3-dimensional space.

5.2.2 Simulations for non-congruent array shapes

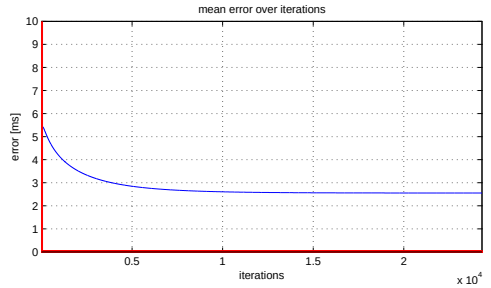
The simulations for non-congruent array shapes for a TOA error of 10 *samples* did not converge consistently, therefor the error was decreased to 5 *samples* and the volumes enlarged in the z dimension to $x_R \times y_R \times z_R = 10 \times 3 \times 3\ m$, which resulted in a more stable performance. The displacement in this case was 5 m in x direction, resulting in 50% overlap (depending on the distribution of sources and microphones). What is again apparent when looking at fig. 21/a, is the fact that the sources are estimated much worse than the microphones (fig. 21/d).



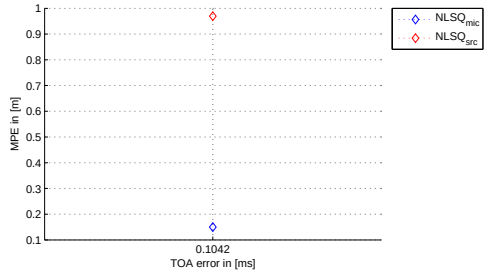
(a) ϵ_p



(b) estimated positions in [m]



(c) average error of the estimated **p** vector.



(d) MPE of the NLSQ solution.

Figure 21 – **setup:** `rand`, **noise:** 5 samples, **size:** $x_R \times y_R \times z_R = 10 \times 3 \times 3$ m and a displacement in x direction by 5 m.

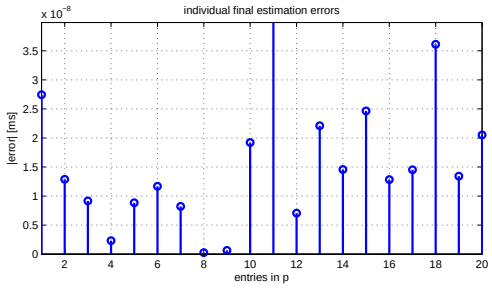
5.2.3 Simulations using Oktava microphones

When simulating Oktava microphones, more microphones (8 Oktava microphones) were used for these simulations. Without TOA error the estimation works perfectly, resulting in an error in the magnitude of 10^{-8} (figure 22), which is essentially zero. When increasing the error to 4 *samples* (seen in fig.23), although the centers of the Oktava microphones are not estimated very accurately, the error behavior seems to be similar to the one without the Oktava microphones, although the error when using Oktava microphones is slightly larger.

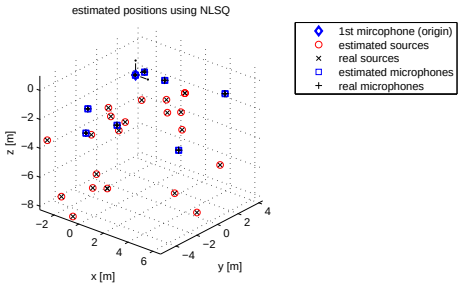
5.2.4 Mean position error and computation time comparison

To determine the quality of the estimated positions, the measure of the mean position error (MPE) will be introduced here, which can be used to describe the estimated positions of sources and microphones with a single value. When plotting the MPE for more setups, the standard deviation will be added, to further increase the information shown in the plots. The MPE can be computed by

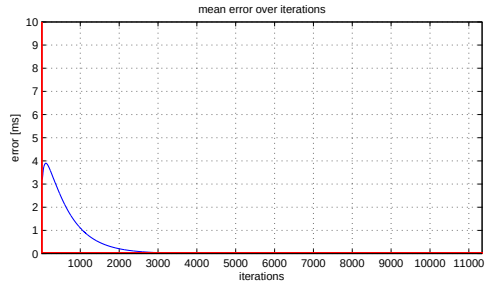
$$\mathrm{MPE} = \frac{1}{N}\sqrt{\sum_{n=1}^{N}\left(\sum_{i=1}^{D}\left(\mathrm{pos}_{i,\mathrm{true}} - \mathrm{pos}_{i,\mathrm{est}}\right)^2\right)}, \tag{55}$$



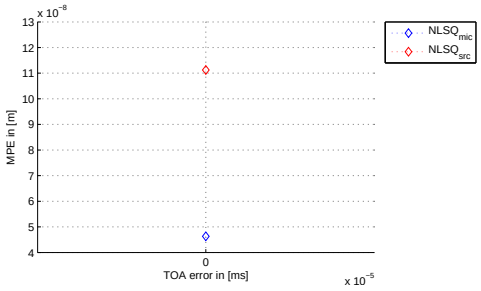
(a) ϵ_p



(b) estimated positions in [m]

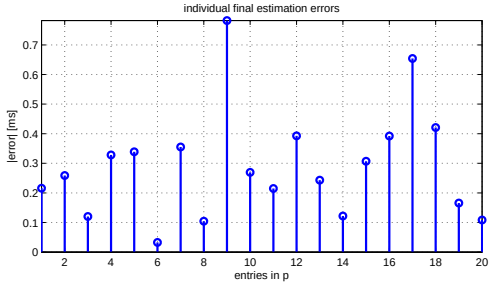


(c) average error of the estimated **p** vector.

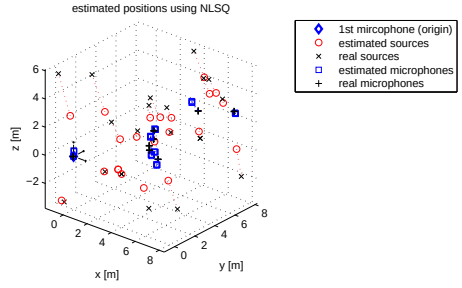


(d) MPE of the NLSQ solution.

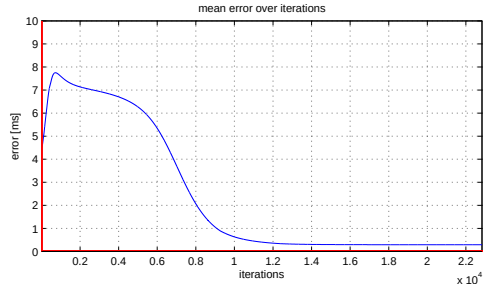
Figure 22 – **setup:** `okta`, **noise:** 0 samples, **size:** $x_R \times y_R \times z_R = 10 \times 10 \times 10\ m$.



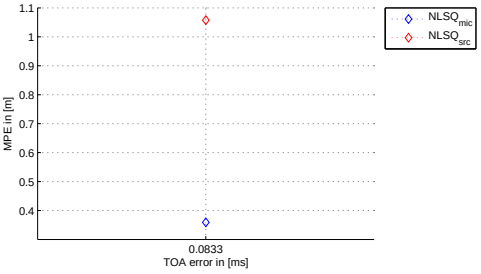
(a) ϵ_p



(b) estimated positions in [m]



(c) average error of the estimated **p** vector.



(d) MPE of the NLSQ solution.

Figure 23 – **setup:** `okta`, **noise:** 4 samples, **size:** $x_R \times y_R \times z_R = 10 \times 10 \times 10\ m$.

where N is the number of sources or microphones, depending on the MPE needed, and D is the dimension of the coordinates (3 in this case). When using more setups, the mean of the MPEs of each setup has to be computed, using

$$MPE_{\text{mean}} = \frac{1}{\#\text{Setups}} \sum_{S=1}^{\#\text{Setups}} MPE_S, \tag{56}$$

where MPE_S is the mean position error of setup S and #Setups the number of setups evaluated. When using more setups to compute the MPE, only setups that led to real coordinates were used.

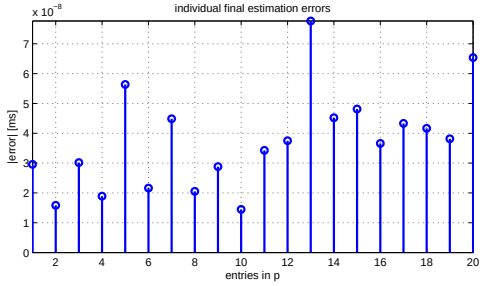
In figure 24 a comparison of the normal LS based position estimation, which relies on the first source and microphone being coincident, and the solution obtained by the NLSQ algorithm can be seen. Figures 24(a,c,d) show the estimated results with perfect TOAs (no noise), representing the best possible estimation of the positions, figures 24(b,d,f) show the results with 4 samples TOA noise. The setup to estimate was not ideal for the LS estimation, with the coincidence condition not being fulfilled ($d_x = 0.5\ m$), but the LS algorithm still leads to results for the estimates of the microphones, when comparing the 3D plots of the LS and NLSQ results in fig.24, and the MPEs in figure 25/a. The sources are usually estimated worse than the microphones, independent of using the LS or NLSQ algorithm. Still, the overall performance of the LS algorithm decreases very fast for increasing d_x (with constant volume dimensions), as expected because of the ignored breach of the coincidence condition.

Due to the fact that the computation time of the NLSQ solution can be even lower than the time needed for the LS algorithm (fig. 25/b), a use of the LS algorithm is generally not merited when the coincidence condition is not full filled.

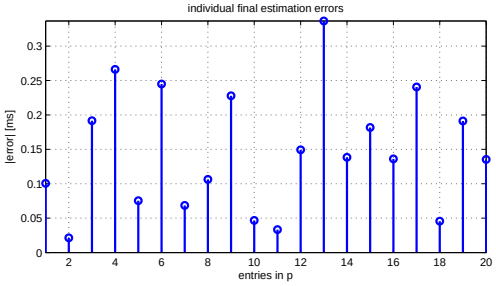
Still, the maximum of the computation time when using the NLSQ algorithm is usually larger than the constant computation time of under 1 second of the LS problem (fig. 25(b)). Still, the NLSQ solution has a much lower MPE for the all estimated positions.

In figure 26 the MPEs of 10 setups for each noise level, all fulfilling the coincidence condition $d_x = 0\ m$ can be seen. When comparing the MPEs, it can be seen that the NLSQ solution yields better results for all simulated noise levels, and when comparing the NLSQ results to those of figure 25, not much change in the MPE can be observed, with the MPEs being linearly (or weak quadratic) dependent on the TOA error.

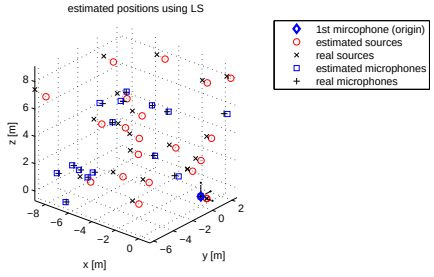
When looking at much higher TOA errors (0 to 90 samples, equal to 0 to $\approx 2\ ms$) simulated in figure 27, the MPE is still linearly (or weak quadratic) dependent on the TOA error, with slowly increasing standard deviations for all plotted MPEs. For the special and the general case, the standard deviations are slowly increasing towards higher TOA errors, together with an increase in computation time for the algorithm estimating the TODs, seen in both fig. 27 for the LS and fig. 28 for the



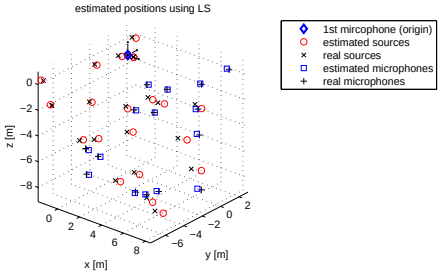
(a) **noise**=0 samples, **p** vector



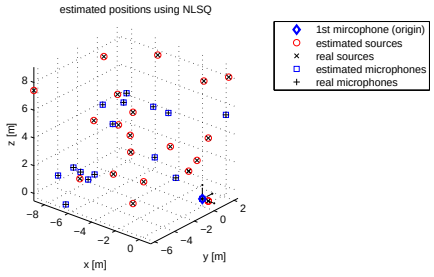
(b) **noise**=3 samples, **p** vector



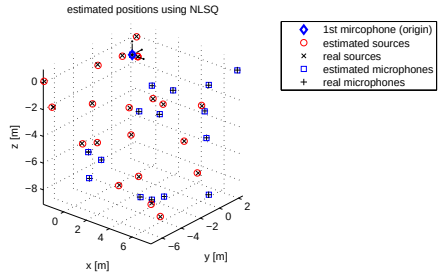
(c) **noise**=0 samples, estimated positions using LS, dimensions in [m].



(d) **noise**=3 samples, estimated positions using LS, dimensions in [m].

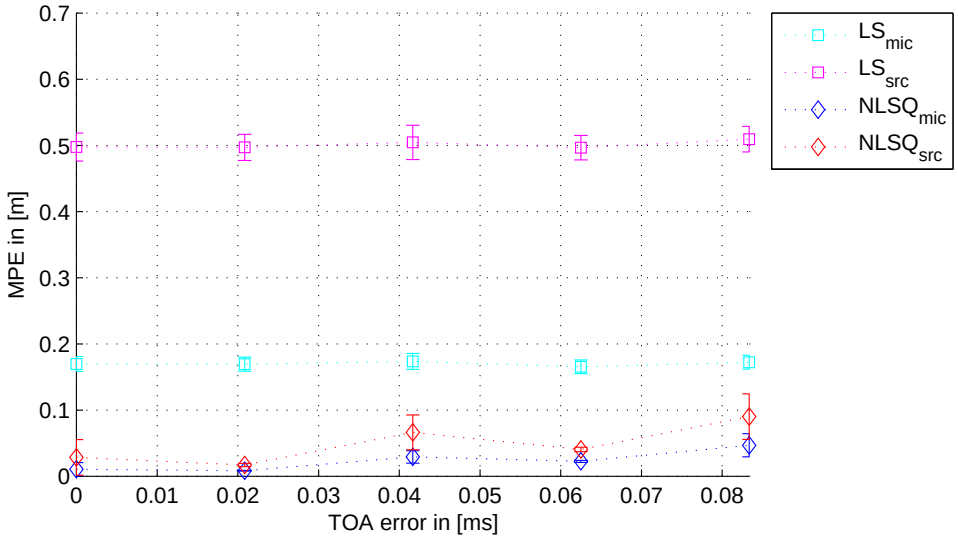


(e) **noise**=0 samples, estimated positions using NLSQ, dimensions in [m].

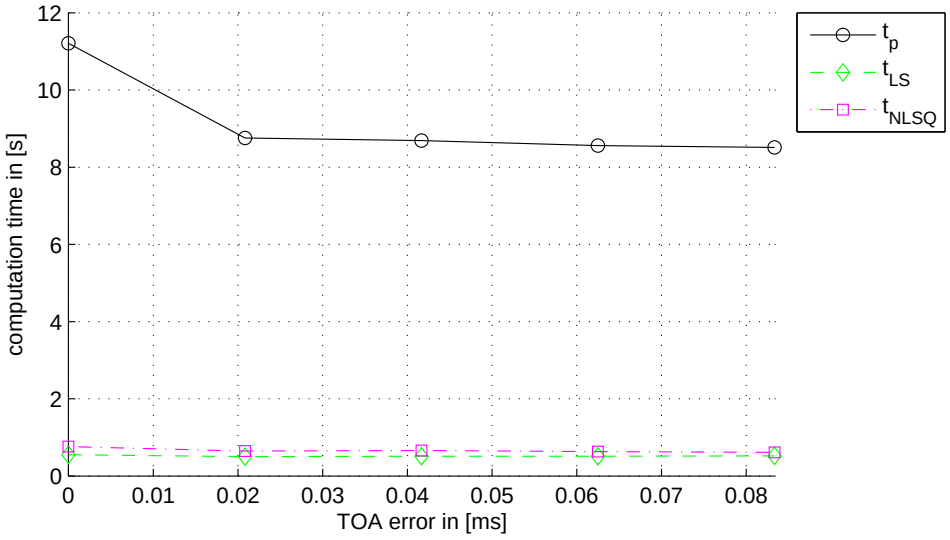


(f) **noise**=3 samples, estimated positions using NLSQ, dimensions in [m].

Figure 24 – For the simulation data of the above plots (a-f) $d_x = 0.5\ m$ was used, with different noise levels. Comparing (c) to (e) and (d) to (f) illustrates the improvement achieved by the NLSQ algorithm compared to the LS algorithm. The size of the volume was $x_R \times y_R \times z_R = 10 \times 10 \times 10$ for all above simulations.



(a) MPE for different TOA errors for a random setup



(b) computation times needed for the different algorithms

Figure 25 – Different MPEs for the respective solver over the TOD error ranging from 0 to 4 *samples* (0 to 0.083 *ms* for a sample rate of 48kHz), with a volume size of $x_R \times y_R \times z_R = 3 \times 3 \times 3$ and $d_x = 0.5$. On average the computation time of the NLSQ is slightly higher than the LS computation, whereas the MPE of the NLSQ is much better.

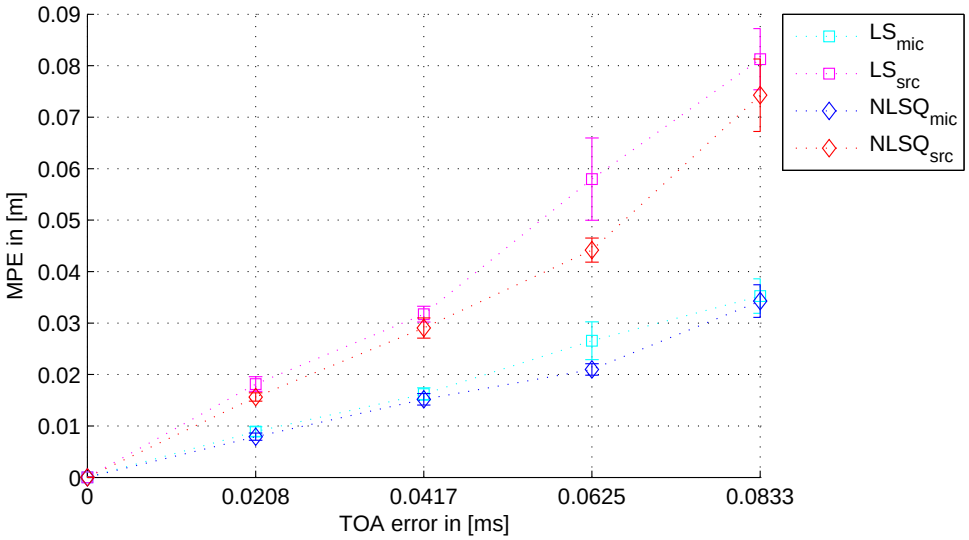
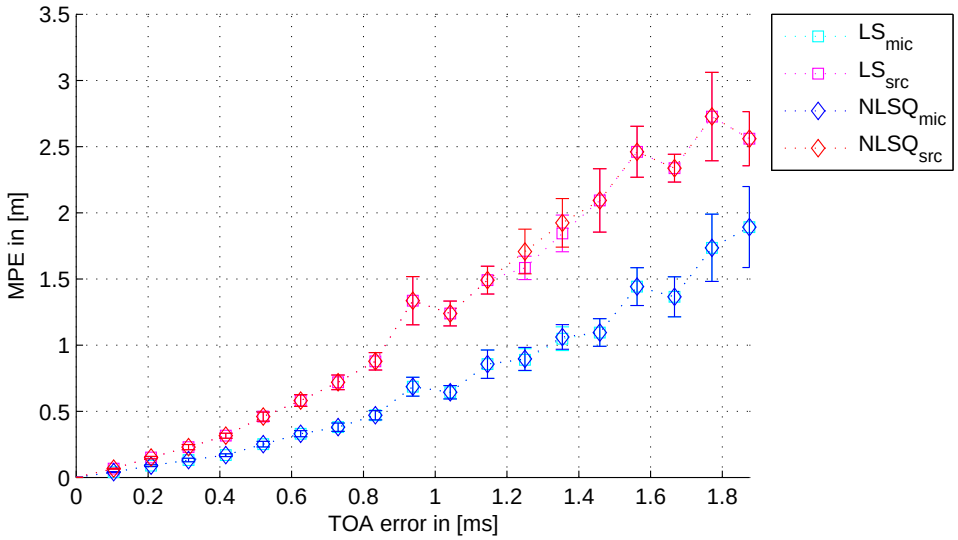


Figure 26 – MPE over 10 setups ($x_R, y_R, z_R = 10\ m$, $\mathcal{I} = 16$, $\mathcal{J} = 20$), TOA error ranging from 0 to 4 samples, with the coincidence condition fulfilled. The MPE seems to be linearly dependent of the TOA error.

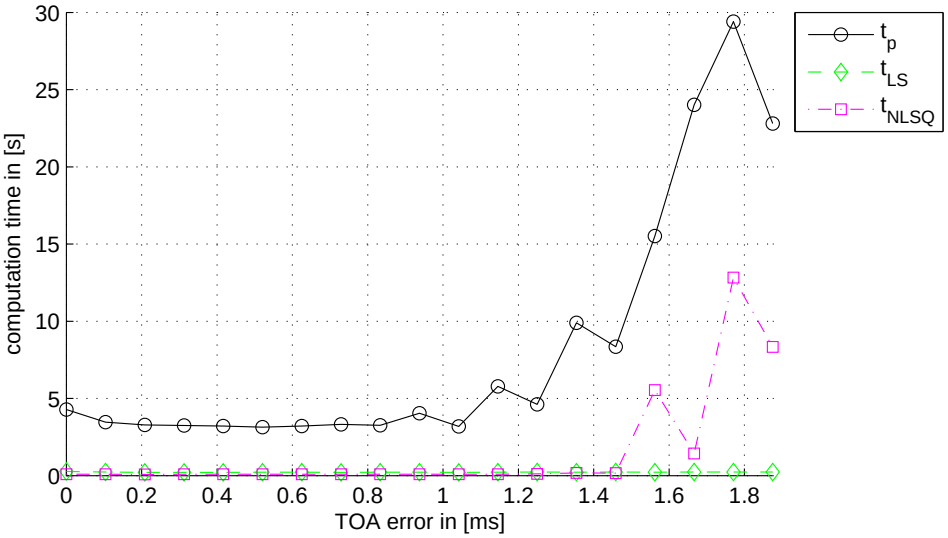
NLSQ algorithm. What also becomes apparent is the fact that the NLSQ algorithm is also likely to suffer an increased computation time. Still, the NLSQ results for the special and the general case are in the same magnitude, with the general case seeming to be a little more inclined to vary more when comparing different setups (i.e. larger variance).

As expected, without the coincidence condition being fulfilled, the MPE of the source positions when using the LS algorithm (which would require the coincidence condition to be fulfilled) is much larger than for the NLSQ algorithm.

It should be noted, that if a few setups could not be estimated correctly, they were removed from the evaluations to be able to determine the quality of the correctly estimated results. This did not occur very often, but more often for higher noise levels, especially for the plots simulating TOA errors up to 90 *samples*.

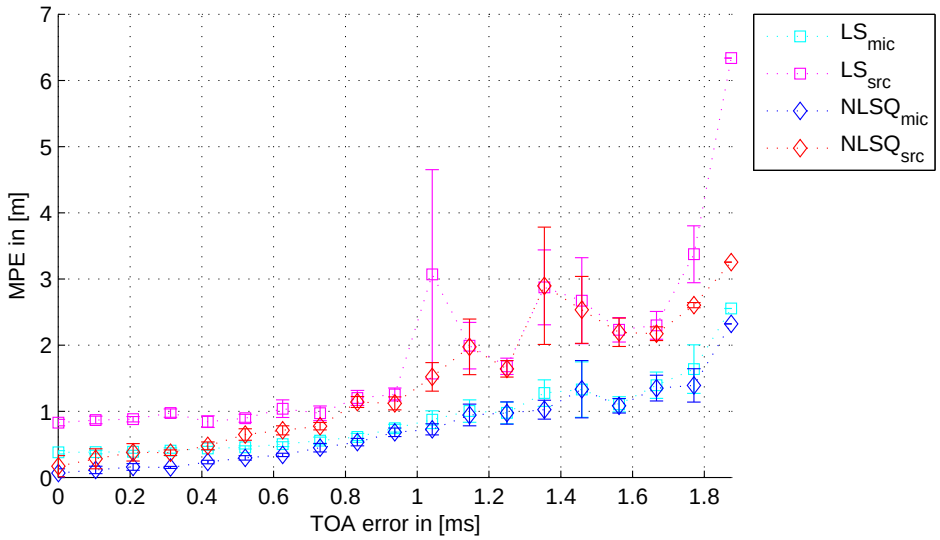


(a) MPE for different TOA errors for a random setup

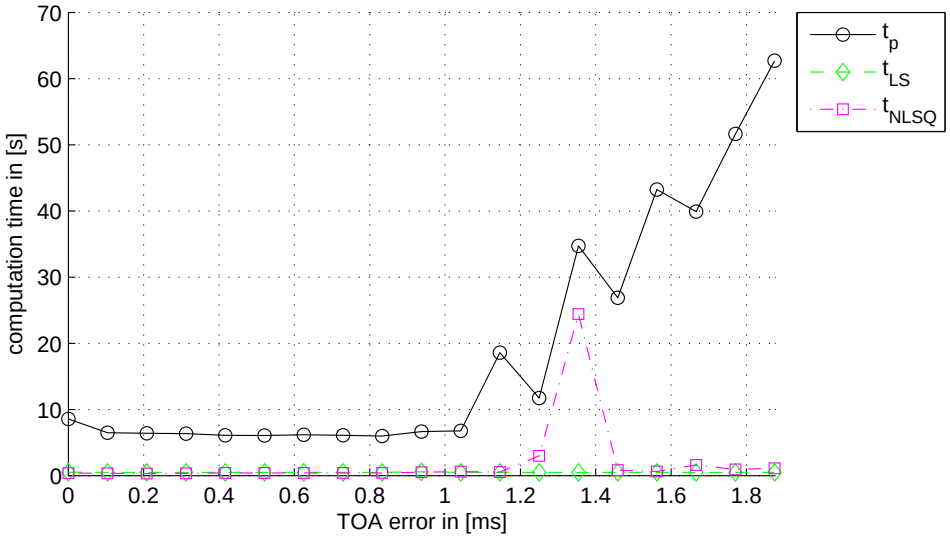


(b) computation times needed for the different algorithms

Figure 27 – Computation times of 10 setups ($x_R, y_R, z_R = 10\ m$, $\mathcal{I} = 10$, $\mathcal{J} = 21$), with TOA error ranging from 0 to 90 samples, with the coincidence condition fulfilled. The computation time can vary rather much.



(a) MPE for different TOA errors for a random setup



(b) computation times needed for the different algorithms

Figure 28 – Computation times of 10 setups ($x_R, y_R, z_R = 10\ m$, $\mathcal{I} = 10$, $\mathcal{J} = 21$), with TOA error ranging from 0 to 90 samples, with the coincidence condition fulfilled. The computation time was not corrected, and, as can be seen, can vary rather much.

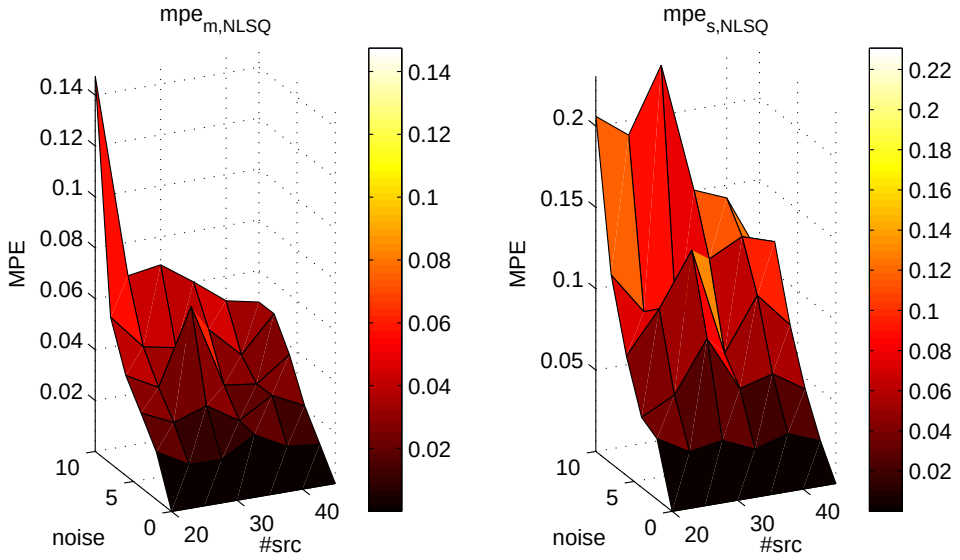


Figure 29 – MPE for a each setup (for every source number/noise level combination only one simualtion was performed), with $x_R, y_R, z_R = 10\ m$, $d_x\ = 1\ m$,$\mathcal{I} = 16$, with TOA error ranging from 0 to 10 samples in steps of 2 samples, for different number of source numbers (#src). As can be seen, the source positions estimated by the NLSQ algorithm have a higher MPE (0.1 to 0.2 m), compared to the microphone MPEs ranging from 0.06 to 0.14 m. A general decreasing trend of the MPE for microphones and sources when increasing the number of sources can be observed.

6 Conclusion

As can be seen from the results in the previous chapters, it is in fact possible to obtain the relative positions (under certain restrictions) of an unknown microphone array and of the sources used to calibrate the microphone array simultaneously with reproducible results concerning the mean position error (MPE). The MPE furthermore seems to be dependent on the error of the TOA detection, with errors of up to 20 samples still leading to a reasonable MPE of under 0.5 m for microphone and source estimates in a $10 \times 10 \times 10$ room.

When comparing the general solution (NLSQ solver) to the special solution (source and microphone coincidence using LS solver), it was shown that the LS solver is more inclined to produce better results for the microphone positions when the coincidence condition is not fulfilled (no microphone and source coincident), but still to much worse results than the NLSQ solver with for in general the same computation times. The computation time on the other hand was shown to be dependent on the TOA error magnitude (for LS and NLSQ).

What can be said about the results in general is, that microphones and sources can be estimated much better, when they are evenly distributed within each other, and when the volumes in which they are distributed are not very long and flat.

What also becomes obvious is an increase of the accuracy of microphone and source

positions alike when increasing the number of sources. Therefore the NLSQ algorithm can be assumed to yield more and more accurate results when increasing the numbers of sources used to calibrate the arrays.

Concerning the simulations, only a rather low number of trials was used when computing an average MPE over samples (10 at maximum), due to the long computation times that would be needed for a 500 trials, a number used in the paper by Crocco [4, 5].

7 Future Work

Additional work could be done concerning a possible use of the above algorithms for the WiLMA array, which was shown to correspond to a much more difficult simulation environment. Furthermore, directional information of the Oktava microphones can be used to acquire additional information concerning the distribution of the sources.

Furthermore, also the general possibility of increasing the accuracy by fixing certain microphone and source constellations is interesting. Future implementations might also include the possibility of using moving source, moving microphones or an overall microphone/source constellation that is not fixed and changing shape constantly, increasing the need for a fast solution to minimize the lag of the estimated system with respect to the actual system at a given time.

Probably the first step that should be done would be the real implementation of the system, to examine the relation between the simulations and real tests. For such an implementation, some problems need to be countered first: the TOAs need to be extracted as good as possible, with respect to echoes or masking of the sound events arriving at the microphones (missing data), and a way to find the orientation of the estimated positions with respect to the real positions has to be determined. This could for example be done when using moving sources (walking person, car,...) which can be assumed to move in a certain plane (xy/xz/yz-plane).

When trying to locate moving sources, it would be interesting to determine the needed interval of updates to be able to track a source moving around.

Another interesting area of use for such an array would be the measuring of a room, concerning its size,geometry and impulse response.

References

[1] P. Aarabi. Self-localizing dynamic microphone arrays. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 32(4):474–484, 2002.

[2] S.T. Birchfield and A. Subramanya. Microphone Array Position Calibration by Basis-Point Classical Multidimensional Scaling. *Speech and Audio Processing, IEEE Transactions on*, 13(5):1025–1034, 2005.

[3] Minghua Chen, Li-Wei He Zicheng Liu, and P. Chou. Energy-Based Position Estimation of Microphones and Speakers for Ad Hoc Microphone Arrays. In *Applications of Signal Processing to Audio and Acoustics, 2007 IEEE Workshop on*, pages 22–25. IEEE, 2007.

[4] Marco Crocco, Alessio Del Bue, Matteo Bustreo, and Vittorio Murino. A closed form solution to the microphone position self-calibration problem. In *ICASSP*, pages 2597–2600. IEEE, 2012.

[5] Marco Crocco, Alessio Del Bue, and Vittorio Murino. A Bilinear Approach to the Position Self-Calibration of Multiple Sensors. *IEEE Transactions on Signal Processing*, 60(2):660–673, 2012.

[6] D. Dobler, G. Heilmann, and M. Ohm. Automatic detection of microphone coordinates. *Proc. Berl Beamforming Conference*, February 2010.

[7] Nikolay D. Gaubitch, W. Bastiaan Kleijn, and Richard Heusdens. Auto-localization in ad-hoc microphone arrays. In *ICASSP*, pages 106–110. IEEE, 2013.

[8] IEM. WiLMA, June 2014. `http://wilma.kug.ac.at`.

[9] P.D. Jager, M. Trinkle, and A. Hashemi-Sakhtsari. Automatic microphone array position calibration using an acoustic sounding source. In *Industrial Electronics and Applications, 2009. ICIEA 2009. 4th IEEE Conference on*, pages 2110–2113. IEEE, 2009.

[10] Xiang Ji and Hongyuan Zha. Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 4, pages 2652–2661 vol.4. IEEE, 2004.

[11] Sarthak Khanal, Harvey F. Silverman, and Rahul R. Shakya. A Free-Source Method (FrSM) for Calibrating a Large-Aperture Microphone Array. *IEEE Transactions on Audio, Speech & Language Processing*, 21(8):1632–1639, 2013.

[12] Marc Pollefeys and David Nistér. Direct computation of sound and microphone locations from time-difference-of-arrival data. In *ICASSP*, pages 2445–2448. IEEE, 2008.

[13] J.M. Sachar, H.F. Silverman, and W.R. Patterson. Position calibration of large-aperture microphone arrays. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 2, pages II–1797–II–1800. IEEE, 2002.

[14] S.D. Valente, M. Tagliasacchi, E. Antonacci, P. Bestagini, A. Sarti, and S. Tubaro. Geometric calibration of distributed microphone arrays from acoustic source correspondences. In *Multimedia Signal Processing (MMSP), 2010 IEEE International Workshop on*, pages 13–18. IEEE, 2010.

[15] Burgess Yubin Kuang, Torstensson S., and Astrom A. A complete characterization and solution to the microphone position self-calibration problem. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 3875–3879. IEEE, 2013.

[16] Zhengyou Zhang Zicheng Liu and Chou Li-Wei He. Energy-Based Sound Source Localization and Gain Normalization for Ad Hoc Microphone Arrays. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 2, pages II–761–II–764. IEEE, 2007.