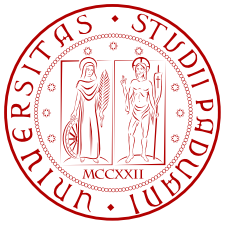


University of Padova
Department of Information Engineering (DEI)
Intelligent and Autonomous Systems Laboratory (IAS-Lab)

Robot Audio Localization using Directions Of Arrival

Author:

Riccardo Levorato

Supervisor:

Enrico Pagello

PhD Degree in Computer Engineering

Padova, July 2015

There are two directions in a road.
I choose the opposite one.

To Germana, Umberto and Alice

Abstract

Nowadays there is an intensive study on the functionalities and capabilities of the robots in order to achieve the goal of building a human-like autonomous robot. In the robotics field, a lot of research has focused on video analysis but the contribution on the audio robot skills is much weaker. This thesis explores the Audio Source Localization problem (ASL) using only the Directions of Arrival (DOAs) estimated by Acoustic Sensors mounted on Mobile Robots connected in a Network. The use of the acoustic DOAs instead of the analysis of the whole audio signal is an innovative method thought for robotics purposes that introduces a new perspective of the audio-video synergy using video sensors for the robot localization in the environment in order to do the extrinsic audio sensor calibration. It is also proposed a new algorithm for localizing the 2D/3D position of an acoustic source using a Gaussian Probability over DOA approach (GP-DOA). Several simulation tests varying the errors over the DOA estimations and over the positions of the robots and the acoustic sources are done. Real test results using Microsoft Kinects as DOA sensors mounted on mobile robots and Aldebaran's NAO robots within the ROS framework, show that the algorithm is robust, modular and the approach can be easily used for robotics applications.

Sommario

Al giorno d'oggi vi è uno studio intensivo sulle funzionalità e le capacità dei robot al fine di raggiungere l'obiettivo di costruire un robot autonomo simile all'uomo. Nel campo della robotica, la ricerca si è concentrata sull'analisi video, ma il contributo delle competenze audio dei robot è molto pi debole. Questa tesi esplora il problema della Localizzazione di Sorgenti Sonore - Audio Source Localization (ASL) - usando solo le Direzioni di Arrivo - Directions of Arrival (DOAs) - stimate da Sensori Acustici montati su Robot Mobili connessi in rete. L'utilizzo di DOAs acustici al posto dell'analisi dell'intero segnale audio è un metodo innovativo pensavo per scopi robotici che introduce una nuova visione della sinergia audio-video usando i sensori video per la localizzazione del robot nell'ambiente per calibrare i parametri estrinseci dei sensori audio. Viene altresì proposto un nuovo algoritmo per localizzare la posizione 2D/3D di una sorgente sonora usando un approccio probabilistico basato su Gaussiane e applicato alle DOA - Gaussian Probability over DOA approach (GP-DOA). Sono stati fatti diversi test in simulazione variando l'errore sulle stime delle DOA e l'errore sulle posizioni dei robot e delle sorgenti acustiche. Test reali usando le Microsoft Kinect come sensori DOA montate su robot mobili e robot Aldebaran NAO all'interno del framework ROS, dimostrano che l'algoritmo è robusto, modulare e che l'approccio può essere facilmente usato per applicazioni robotiche.

List of Publications

List of publications in chronological order:

1. Filippo Basso, **Riccardo Levorato** and Emanuele Menegatti. *Online Calibration for Networks of Cameras and Depth Sensors*. OMNIVIS: The 12th Workshop on Non-classical Cameras, Camera Networks and Omnidirectional Vision - 2014 IEEE International Conference on Robotics and Automation (ICRA 2014);

2. **Riccardo Levorato** and Enrico Pagello. *Probabilistic 2D Acoustic Source Localization Using Direction of Arrivals in Robot Sensor Networks*. In Davide Brugali, Jan F. Broenink, Torsten Kroeger, and Bruce A. MacDonald, editors, Simulation, Modeling, and Programming for Autonomous Robots, volume 8810 of Lecture Notes in Computer Science, pages 474-485. Springer International Publishing, 2014;

3. **Riccardo Levorato**, Emanuele Menegatti and Enrico Pagello. *Audio DOA Sensor Evaluation using Video as Ground Truth for Microsoft Kinect*. Italian Workshop on Artificial Intelligence and Robotics - AIRO 2014. Workshop of the XIII AI*IA Symposium on Artificial Intelligence. December 2014;

4. **Riccardo Levorato** and Enrico Pagello. *DOA Acoustic Source Localization in Mobile Robot Sensor Networks*. In Autonomous Robot Systems and Competitions, 2015. ICARSC 2015. IEEE International Conference on, pages 71-76, April 2015;

5. Filippo Basso, **Riccardo Levorato**, Matteo Munaro and Emanuele Menegatti. *A distributed calibration algorithm for color and range camera network*. Chapter in Springer Book on ROS, edition of the book series "Studies in Systems, Decision and Control";

6. **Riccardo Levorato** and Enrico Pagello. *Acoustic Source Localization for Robotic Networks*. Chapter in Springer Book on ROS, edition of the book series "Studies in Systems, Decision and Control".

Contents

List of Figures

List of Tables

Chapter 1

Introduction & State of the Art

1.1 Introduction

Audio for robotics is a field that is not well explored as the vision for robotics. Probably it happens because acoustic skills are considered less important than visual ones in the every day life. It is a matter of fact that the visible light frequency range is very important for the living beings (e.g. for navigation purposes, object recognition, etc.) but it cannot cover some particular events which can be detected analysing only the acoustic signals (e.g. horn machines, exploding bombs, barking dogs, etc.). Considering only audio skills, we can retrieve also other important information about the environment (e.g. classification of sounds, room shape construction using echoes and reflections, directions of arrival of the sound sources) but, on the other hand, this information is not always enough to localize properly the acoustic source position in the environment. For example, estimating the distance of an acoustic source requires further skills that take into consideration also sound reflections and echoes (i.e. bats echolocation) [2]. Hence, there is the need for fusing data coming from different sensors in order to have more information of the surroundings. In our robotics field we imagine that a robot should find itself in a map, hear some sounds and fuse these data to find what and where the acoustic source is. Moreover, if we deal with connected robots, we can in addition share the data of each robot with the others in order to have a better understanding of the environment.

1.2 State of the Art

The current state of the art offers lots of works that refers to Audio Source Localization (ASL) (i.e. for a detailed and complete overview look up the PhD Thesis of Pertilä [1] and Salvati [3]). The majority of these works focuses on various techniques that use the signals coming from all the microphones and estimate the position of the sources fusing all these data. In a robotics environment, this approach cannot be applied because the collection of all the audio signals in a unique master node for the calculations can cause a network block and, over all, synchronizing all audio signals in time [4] could be very challenging.

In the acoustic field, a DOA sensor consists specifically of an array of at least two microphones. Common tested acoustic sources talking people [5], gun shots and human screams [6], clapping hands and so on. There are various techniques for calculating the DOA of an acoustic source such as Angle of Arrival (AoA), Time Difference of Arrival (TDOA), Frequency Difference of Arrival (FDOA), and other similar techniques [1][3].

Recently there was an increasing use of sensor networks, that can share audio and video data in a cooperative way, achieving more precise knowledge of the environment. In an Audio Sensor Network (ASN), by knowing the position and pose of each DOA sensor, it is possible to better estimate the position of the acoustic source by sharing and synchronizing the DOA estimations [5][7][8]. Following this idea, Hawkes et Al. [9] proposed an analytic Weighted Least Square method (WLS-DOA) that minimizes the distance of the estimated point to the estimated DOA line and achieves good results in 3D space. In his PhD thesis, Pertilä [1] also focused on 2D DOA-Based Localization and introduced a new approach that tried to eliminate the sensors with bad acoustic DOA estimations starting from the WLS-DOA method (Robust WLS-DOA (RWLS-DOA)) [9].

Although it was shown that this method performed better than WLS-DOA in outlier situations, I found that it is error prone, because in some situations it can discard the DOA sensors that are better than others (see sections 3.4 and 5.3.1 for a complete analysis).

In the approach of this thesis, proposed for the first time in [10], I use all the DOA sensors and I propose a new way of thinking that is based on the probability error over the angle of the DOA estimation. The difference with all the WLS-DOA approaches is that our method minimizes the angle (instead of the distance) of the estimated point to the estimated DOA. Furthermore this new approach is not affected by the angle ambiguity problems of the WLS-DOA approaches. Another contribute is the new concept of Audio-Video data fusion. More specifically, Audio Localization will

be strictly correlated to Video Calibration: working with mobile robots it would be very hard to calibrate at each instant time the relative position of all microphones of all moving robots with non-invasive techniques (i.e. using an acoustic high-frequency periodic signal as audio reference). For this reason, using SLAM techniques and knowing the relative position of the microphones with respect to the visual sensors, will help robots to share also audio information.

In a Mobile Robot Sensor Network (Figure 1.1), to avoid bottleneck problems sharing audio-video data among robots, I proposed [10] a method that shares only the pose of the robot and the DOAs of the acoustic sources heard introducing the GP-DOA approach that uses a probabilistic model of the sensor for finding the acoustic source position that outperforms the state-of-art WLS-DOA method. Starting from the GP-DOA approach, it is also proposed a faster algorithm for calculating the maximum probability point using an adapted binary search.

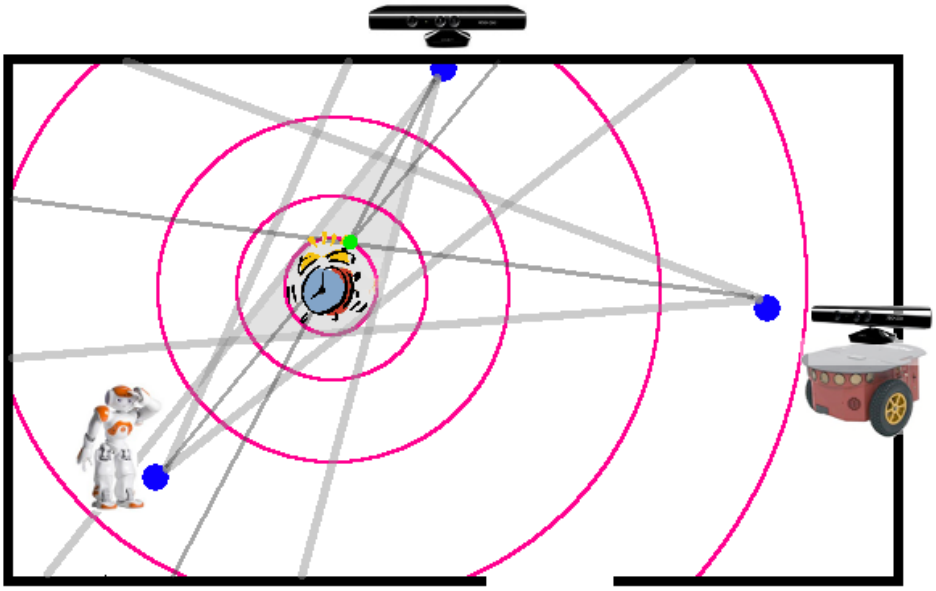


Figure 1.1: 2D Acoustic Source Localization Problem in an indoor environment. In the example there are a fixed Microsoft Kinect, a Pioneer 3-AT equipped with a Microsoft Kinect and a NAO robot that cooperate for localizing the position of an alarm clock.

Chapter 2

DOA Source Localization Problem

2.1 Introduction

The definition of the generic DOA Source Localization Problem (DOA-SL) can be logically divided into two different sets:

- DOA Single-Source Localization (DOA-SSL);
- DOA Multiple-Source Localization (DOA-MSL).

The set representation of the above separation regarding the definition of the DOA-SL problem is shown in Figure 2.1. It clearly shows that the DOA-SSL problem can be seen as a particular case of the DOA-MSL problem. Note that this representation gives an hint to the approach for managing DOA-MSL problems. As a matter of facts, a DOA-MSL problem can be considered as an extension of the DOA-SSL one, thought as the base case of the problem, and will focus mainly on the algorithms that select the best solutions among all possible DOA-SSL solutions.

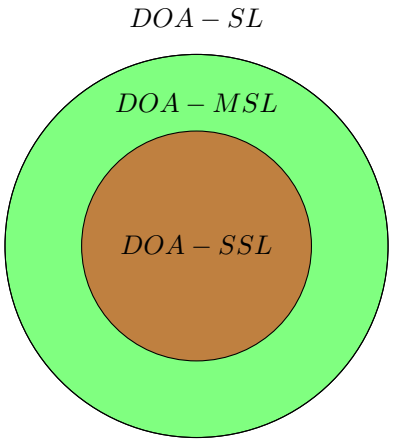


Figure 2.1: Set representation of the division of the DOA Source Localization problem (DOA-SL) definition into DOA Single-Source Localization (DOA-SSL) and DOA Multiple-Source Localization (DOA-MSL).

The DOA-SSL problem was described in [9] and [1]. On the other hand the DOA-MSL problem was analysed in [11].

In section 2.3 it is proposed the definition of the three-dimensional (3D) DOA-MSL case as the DOA-SSL problem definition can be derived from the former considering only one source.

2.2 Direction of Arrival representation

The Direction of Arrival (DOA) is just an half-line that can exist both in a two-dimensional space (2D) and in a three-dimensional space (3D).

For representing a DOA in the 2D space it is needed the DOA initial point $\mathbf{s} \in \mathbb{R}^2$ and the angle of the direction $\phi \in [-\pi, \pi]$ with respect to the *world* reference system, that it is commonly called **azimuth**

On the other hand, in order to represent the DOA in the 3D space using the spherical coordinate system, it is needed a three-dimensional initial point $\mathbf{s} \in \mathbb{R}^3$, the azimuth $\phi \in [-\pi, \pi]$ and another angle $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, called **elevation** as shown in Figure 2.2.

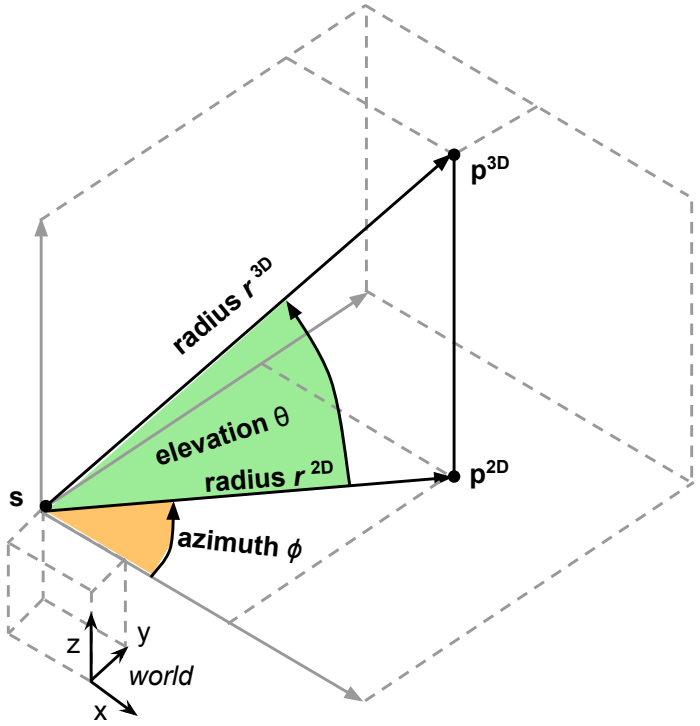


Figure 2.2: DOA spherical representation in the 3D space with azimuth ϕ, elevation θ, radius r and the initial point $\mathbf{s}$. $\mathbf{p^{2D}}$ and $\mathbf{p^{3D}}$ are respectively the 2D and 3D points that lie on the DOA.

Both in 2D and in 3D spaces, any generic point $\mathbf{p}$ that lies on the DOA can be identified by using the **radius** r that is the norm of the vector $\mathbf{sp}$:

$$r = ||\mathbf{sp}|| \tag{2.1}$$

The Cartesian coordinates of the point $\mathbf{p}$ with respect to the point $\mathbf{s}$ (labeled as $\mathbf{p_s}$) can be calculated from the spherical coordinates using azimuth ϕ, elevation θ and radius r with the following one-shot formula[1]:

$$\mathbf{p_s} = spherical\_2\_cartesian(\phi, \theta, r) = \begin{cases} x = r * \cos(\theta) * \cos(\phi) \\ y = r * \cos(\theta) * \sin(\phi) \\ z = r * \sin(\theta) \end{cases} \tag{2.2}$$

Finally the point $\mathbf{p}$ with respect to the *world* (labeled as $\mathbf{p}_{world}$) is calculated by adding the coordinates of the initial point $\mathbf{s}$:

$$\mathbf{p}_{world} = \mathbf{s} + \mathbf{p_s} \tag{2.3}$$

[1] http://it.mathworks.com/help/matlab/ref/sph2cart.html

2.3 Problem statement

Let S be a set of DOA sensors, with $|S| = N_S \geq 2$ and let R be a set of sources, with $|R| = N_R \geq 1$. In the acoustic field, a DOA sensor $s_i \in S$, $i \in [1, N_S]$, consists in a microphone array that can estimate different DOAs belonging to one or more different generic audio sources $r_j \in R$, $j \in [1, N_R]$, with respect to the intrinsic reference system of the DOA sensor s_i.

Let $\mathbf{T}^{s_i} = [s_i^x, s_i^y, s_i^z]^T$ and $\mathbf{R}^{s_i}$ represent respectively the 3D position and the 3D rotation matrix of the i-th DOA sensor s_i with respect to the Cartesian reference system *world* in $\mathbb{R}^3$. The DOA estimation of the source r_j computed by DOA sensor s_i in spherical coordinates it is composed by two angles: the azimuth $\phi_{i,j} \in [-\pi, \pi]$ and the elevation $\theta_{i,j} \in [-\frac{\pi}{2}, \frac{\pi}{2}]$.

The problem consists in finding the closest estimation of the acoustic source positions $\mathbf{r}_j = [r_j^x, r_j^y, r_j^z]^T \in \mathbb{R}^3$, $\forall j \in [1, N_R]$, assuming that all $\mathbf{T}^{s_i}$ and $\mathbf{R}^{s_i}$ are known *a priori* and all $\phi_{i,j}$ and $\theta_{i,j}$ are estimated using a DOA estimation algorithm (Figure 2.3).

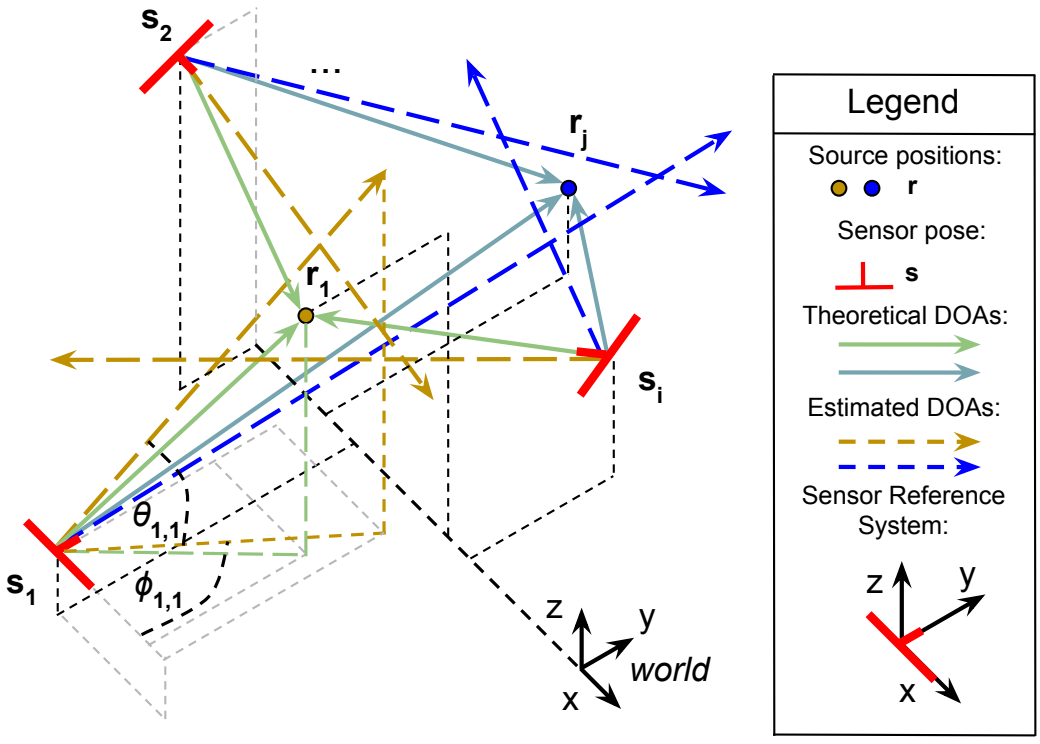


Figure 2.3: 3D DOA Multi-Source Localization Problem (3D DOA-MSL). In this Figure are shown three sensors and two sources but there are theoretically no upper limits for the number of the sensors and the number of the sources.

As it may not seem at a first sight, the problem is very complex and has a lot of variables that have to be managed. From now on, in order to deal with a simplified version of the problem, the following assumptions are considered:

1. all DOA sensors are connected together within a network supported by a framework that permits to share their poses and the DOA estimations that can both be error-prone. Furthermore all DOA estimations from different DOA sensors are synchronized and grouped by the framework;

2. only true positive DOA estimations are considered: all the sensors can detect all the coexisting sources and estimate their DOA if and only if they are present;

3. the precision of all $\phi_{i,j}$ and $\theta_{i,j}$ depends on the accuracy of each DOA sensor and on the DOA estimation algorithm used. Echoes and sound reflection effects are considered to be already managed by the DOA estimation algorithm.

Assumption 1 allows to solve the problem also in the real world: the inter-connection among sensors, the data sharing and the synchronization within a common framework help to focus only on the DOA-SL problem.

Assumption 2 restricts the number of the cases that can occur in the real world: this assumption allow to initially analyse the problem from a basic case that doesn't have to deal with false positive or true negative estimations, preventing from dealing with the evaluation and eventual discard of some wrong estimations.

Finally, assumption 3 leaves the problem of the DOA estimation to the DOA estimation algorithms. This very large topic will not be deeply analysed because the thesis will focus mainly on an higher level of programming for robotics, considering the DOA estimation a simple signal coming from a generic DOA sensor.

Chapter 3

Common Approaches

3.1 Introduction

In the ideal case, with perfect DOA estimations, all DOAs from each sensor will intersect in the same point and the source could be localized by simply finding the single intersection point of two of them.

Unfortunately, in the real world or in any realistic simulation, due to estimation errors, the DOA estimations are not be perfect and hence the intersections do not coincide.

In the state of the art there are some approaches that take this issue into consideration and try to solve the DOA-SSL problem in different ways:

1. Centroid (C-DOA);
2. Weighted Least Square (WLS-DOA);
3. Robust Weighted Least Square (RWLS-DOA);
4. Grid Based (GB-DOA);
5. Gaussian Probability (GP-DOA).

All these approaches will be described in the following. The comparisons among their behaviours and performances will be discussed in Chapters 5 and 6 using simulated tests and real experiments.

3.2 Centroid

Let O be the set containing all the existing intersection points between all the pairs of the DOA estimations:

$$\mathbf{O} = \{\mathbf{o}_{(i,j)} \in \mathbf{R}^{dim} : 1 \leq i, j \leq N_s \wedge i \neq j \wedge \mathbf{o}_{(i,j)} = \mathbf{o}_{(j,i)}\}, \tag{3.1}$$

where *dim* can be 2 or 3 in the 2D or the 3D case respectively.
Analysing the 2D DOA-SSL problem (Figure 3.1), the first idea that comes to mind for estimating the location of the source $\mathbf{r}$ is based on taking the mean point among the intersections of the pairs of DOAs. The centroid $\hat{\mathbf{r}}_C$ is simply the mean of the points in the set O; it minimizes the sum of squared Euclidean distances between itself and each point in the set O [12] [13]:

$$\hat{\mathbf{r}}_C = \frac{\sum_{\mathbf{o}_{(i,j)} \in O} \mathbf{o}_{(i,j)}}{|O|} \tag{3.2}$$

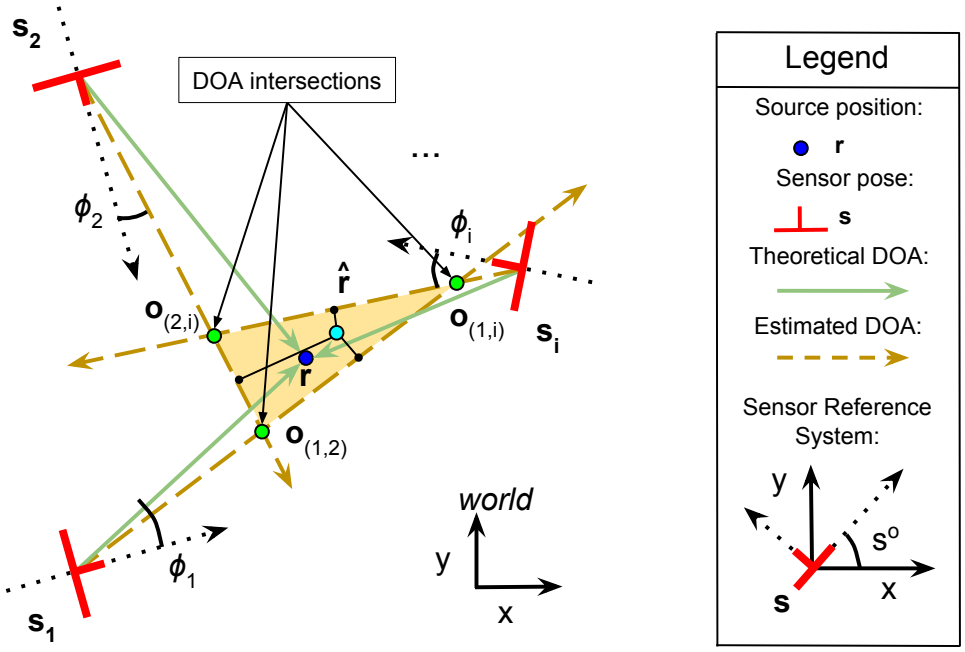


Figure 3.1: Example of 2D DOA-SSL problem with centroid solution. The green points are the intersections between pairs of DOAs which are also the vertexes of a triangle. The cian point $\hat{\mathbf{r}}$ is the centroid of that triangle.

This approach has unfortunately some drawbacks: in the 2D case, the intersection of two lines exists only if and only if the two lines are not parallel. In this particular case, the intersection point is not considered and so the cardinality of the set O, that is commonly $|O| = \binom{N_s}{2}$, has to be reduced.

In the 3D case, the intersection between two lines exists if only if the two lines are complanar and not parallel. If two DOAs r and s are skew lines, $\mathbf{o}_{(i,j)}$ can be calculated as the medium point of the smallest segment that links r and s using the same concept for calculating the distance between two skew lines in the 3D space (Figure 3.2).

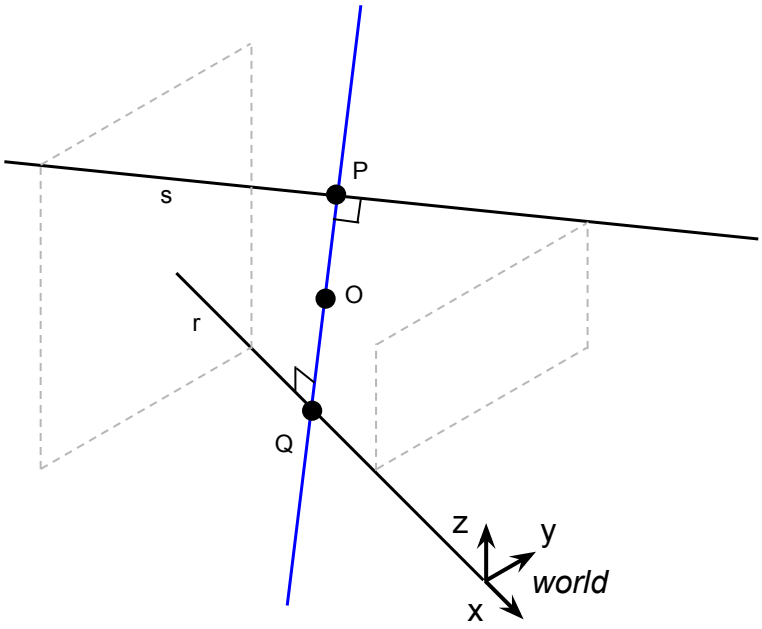


Figure 3.2: Explanation for the centroid approach in the 3D space with two skew lines.

This approach was introduced by Griffin et al. in [11] where they extended this concept by removing the less reliable intersection points. In other words, they deleted from the set O the intersection points that were too far from the centroid $\hat{\mathbf{r}}_C$.

The algorithm has a computational complexity of $\Theta(\binom{N_s}{2}) = \Theta(N_s^2)$.

3.3 Weighted Least Square

In [9], Hawkes et Al. proposed an analytic Weighted Least Square method (WLS-DOA) that minimizes the distance of the estimated point to the estimated DOA line. Here it is reported the detailed definition of the solution.

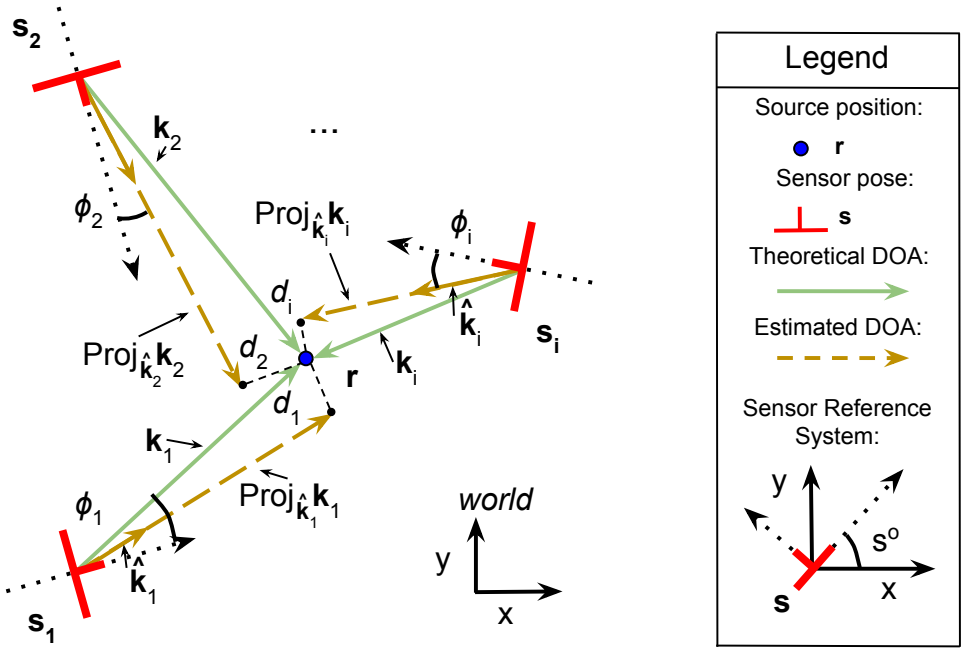


Figure 3.3: Explanation for the WLS approach.

Let the i-th DOA estimation be defined as:

$$\mathbf{k}_i = \mathbf{r} - \mathbf{s}_i \tag{3.3}$$

The closest point from the line defined by the measurement vector $\mathbf{k}_i$ to the source position is written as:

$$\mathbf{s}_i + Proj_{\hat{\mathbf{k}}_i}\mathbf{k}_i \tag{3.4}$$

where the projection vector is defined as:

$$Proj_{\hat{\mathbf{k}}_i}\mathbf{k}_i = \mathbf{k}_i(\mathbf{r} - \mathbf{s}_i)\hat{\mathbf{k}}_i \tag{3.5}$$

Note that the hypothetical source direction vector $\mathbf{k}_i$ in Equation 3.5 is projected onto the actual DOA measurement vector $\hat{\mathbf{k}_i}$. The distance from the real source position $\mathbf{r}$ to this closest point is:

$$d_i = ||\mathbf{s}_i + Proj_{\hat{\mathbf{k}}_i}\mathbf{k}_i - \mathbf{r}|| \tag{3.6}$$

which is squared and summed to form the weighted least squares estimation:

$$\hat{\mathbf{r}} = \underset{\mathbf{r}}{argmin} \sum_{i=1}^{N_s} d_i^2 w_i = \underset{\mathbf{r}}{argmin} \sum_{i=1}^{N_s} ||\mathbf{s}_i + \hat{\mathbf{k}_i}^T(\mathbf{r} - \mathbf{s}_i)\hat{\mathbf{k}}_i - \mathbf{r}||^2 w_i \tag{3.7}$$

where w_i is a weight associated with the accuracy of each DOA estimation vector $\hat{\mathbf{k}}_i$. The localization criterion 3.6 is illustrated in Figure 3.3. Equation 3.7 can be rewritten as:

$$\hat{\mathbf{r}} = \underset{\mathbf{r}}{argmin} \sum_{i=1}^{N_s} \left(-2\mathbf{s}_i^T(\mathbf{I} - \hat{\mathbf{k}}_i\hat{\mathbf{k}_i}^T)\mathbf{r} + \mathbf{r}^T(\mathbf{I} - \hat{\mathbf{k}}_i\hat{\mathbf{k}_i}^T)\mathbf{r}\right) w_i \tag{3.8}$$

where terms not related to $\mathbf{r}$ are omitted. Differentiating 3.7 with respect to the source position $\mathbf{r}$ and setting the result to zero gives the following equation:

$$2\sum_{i=1}^{N_s}(\mathbf{I} - \hat{\mathbf{k}}_i\hat{\mathbf{k}_i}^T)(\mathbf{s}_i - \hat{\mathbf{r}})w_i = 0 \tag{3.9}$$

Note that the second derivative should be confirmed positive semi-definite to ensure that the point is actually the minimum and not the maximum. The closed-form solution can be then written with the following one-shot formula giving to the approach a computational complexity of $\Theta(1)$:

$$\hat{\mathbf{r}}_{WLS} = \left[\left(\sum_{k=1}^{N_s} w_k\right)\mathbf{I} - \hat{\mathbf{K}}\mathbf{W}\hat{\mathbf{K}}^T\right]^{-1}\mathbf{A}\mathbf{w} \tag{3.10}$$

where $\mathbf{w} = [w_1, w_2, \ldots, w_{N_s}]^T$ are the weights, $\mathbf{W} = diag(\mathbf{w})$,
$\mathbf{I}$ is the identity matrix, $\hat{\mathbf{K}} = [\hat{\mathbf{k}}_1, \hat{\mathbf{k}}_2, \ldots, \hat{\mathbf{k}}_{N_s}]^T$,
and $\mathbf{A} = \left[(\mathbf{I} - \hat{\mathbf{k}_1}\hat{\mathbf{k}_1}^T)\mathbf{s}_1, (\mathbf{I} - \hat{\mathbf{k}_2}\hat{\mathbf{k}_2}^T)\mathbf{s}_2, \ldots, (\mathbf{I} - \hat{\mathbf{k}}_{N_s}\hat{\mathbf{k}}_{N_s}^T)\mathbf{s}_{N_s}\right]$.

3.4 Robust Weighted Least Square

This approach was introduced first by Pertilä in [14] and starts from the concept that some DOA estimations might not be originated from the same source, causing a biased result. A single outlier DOA estimation may reduce the efficiency of the WLS method. The RWLS approach tries to detect the outlier estimations with the following procedure.

Let the power set $\mathcal{P}(S)$ be the set of all the subsets of S:

$$\mathcal{P}(S) = \{\{\emptyset\}, \{s_1\}, \ldots, \{s_1, s_2\}, \ldots, \{s_{N_s-1}, s_{N_s}\}, \{s_1, s_2, s_3\}, \ldots, S\} \quad (3.11)$$

A subset of the powerset with three or more DOA sensor and their corresponding DOA measurements is noted by S_n. There are N_{3+} of such subsets, where $N_{3+} = \sum_{i=3}^{N_S} \binom{N_s}{i}$.

The Average Distance Criterion (ADC) from a specific subset n can be written in this way [14]:

$$ADC(n) = \frac{1}{|S_n|} \sum_{k \in S_n} d_{k,n} \quad (3.12)$$

where $n = 1, \ldots, N_{3+}$ is the subset index, $|S_n|$ is the cardinality of subset S_n, and $d_{k,n}$ is the distance contributed by the k-th sensor in the subset S_n using criterion of Equation 3.6. The subset that has the minimum ADC is selected for the solution:

$$\tilde{n} = \underset{n}{argmin} ADC(n) \quad (3.13)$$

$$\hat{\mathbf{r}}_{RWLS} = \hat{\mathbf{r}}_{\tilde{n}} \quad (3.14)$$

The complexity of the algorithm is $\Theta(N_{3+}) = \Theta(\sum_{i=3}^{N_S} \binom{N_s}{i})$.

3.5 Grid based Approach

Griffin et Al. in [15] considered a 2D grid-based (GB-DOA) method that is an approach based on discretizing the area of interest into a grid of N points, and then finding the grid point whose DOAs match most closely the estimated DOAs. Moreover, since the measurements are angles, it is used an Angular Distance as a more proper measure of similarity. The grid is thus an $(S \times N)$ matrix Φ, whose elements, $\Phi_{i,n}$ are the DOAs from the i-th sensor to the n-th grid point.

Localizing a single source with this method is done in the following way: we need to find the grid point whose DOAs match most closely the estimated DOAs. However, in order to properly compare angles, Griffin defined an angular distance function $A(X, Y)$ whose output is limited to the range $[0, \pi]$. An elegant, if somewhat inefficient, implementation of $A(X, Y)$ is given by:

$$A(X, Y) = 2\sin^{-1}\left(\frac{|\exp(jX) - \exp(jY)|}{2}\right) \tag{3.15}$$

With this defined, the problem may then be expressed as:

$$\breve{n} = \underset{n}{argmin} \sum_{i=1}^{N_s} \Big[A(\phi_i, \Phi_{i,n})\Big]^2 \tag{3.16}$$

The source position estimate $\hat{\mathbf{r}}_{GB}$ is simply given as the coordinates of the $\breve{n}$-th grid point:

$$\hat{\mathbf{r}}_{GB} = \hat{\mathbf{r}}_{\breve{n}} \tag{3.17}$$

As a source may be located anywhere within the area of interest, it is be clear that the resolution of the grid, determined by the number of grid points N, will influence the position estimation error. However, increasing N to decrease the position estimation error will also increase the complexity of the algorithm that is $\Theta(N)$. A good compromise is to use an iterative algorithm that starts with a coarse grid, and once the best grid point is found, a new grid centered on this point is generated, with a smaller spacing between grid points, but also a smaller scope. Then the best grid point in the new grid is found. This may be repeated until the desired accuracy is obtained, while keeping the complexity under control. A drawback of this method is that all the sensors are considered to have the same error and it is not possible to set some weights. This issue will be resolved by the proposed GP-DOA approach.

Chapter 4

Gaussian Probability Approach

4.1 DOA Single-Source Localization

The Gaussian Probability over DOA approach was first introduced in [10] for the 2D DOA-SSL problem. This section will refer only to the 2D DOA-SSL case. Hence it will be considered only one acoustic source $r \in R$ with $|R| = 1$ at time and the obvious index $j = 1$ will be omitted (i.e. ϕ_{i1} becomes ϕ_i). As we now limit the problem to the two-dimensional case, all the sensors are considered to be complanar and so for the DOA estimation will be taken into consideration only the azimuth ϕ_i (Figure 4.1).

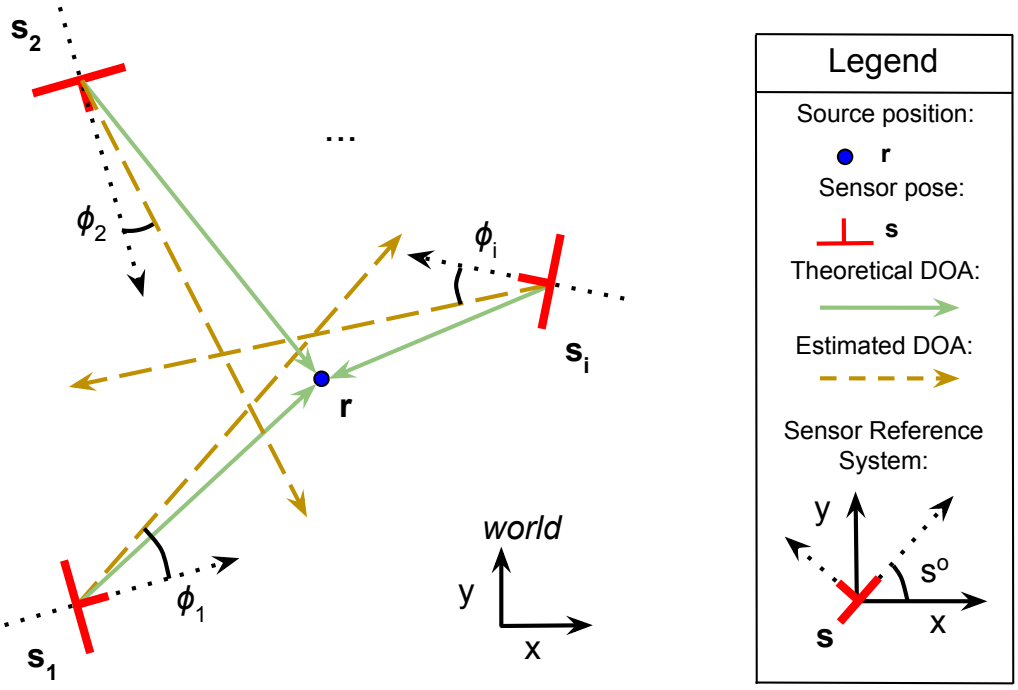


Figure 4.1: 2D DOA Single-Source Localization Problem (2D DOA-SSL).

The approach is based on the simple fact that each real DOA estimation ϕ_i has an intrinsic error that depends mainly on the accuracy of the i-th DOA sensor. This error can be modelled as a Gaussian probability error in the angle domain with zero mean and variance σ_i using only values in the range $\phi_i \in [-\pi, \pi]$. Hence the angular probability sensor model $\mathcal{M}_i$ is defined as follows:

$$\mathcal{M}_i \sim \mathcal{N}(0, \sigma_i)_{[-\pi,\pi]} = \frac{1}{\sigma_i\sqrt{2\pi}} e^{-\frac{\phi_i^2}{2\sigma_i^2}} \quad , \quad \phi_i \in [-\pi, \pi] \tag{4.1}$$

At this step, as for the Grid-Based approach, it is needed a change of domain from the angular domain to the Cartesian coordinate system $G = (n \times n) \in \mathbb{Z}^2$. G is as a square spatial 2D grid with a fixed spatial *range* [m] and a fixed precision parameter *prec* [m] such that $n = \frac{range}{prec}$.

The angle $\beta_i^{\mathbf{q}}$ that considers $\mathbf{T}^{s_i} = [s_i^x, s_i^y]^T$ as vertex and it is included between a first line that passes through $\mathbf{T}^{s_i}$ and $\mathbf{q}$ and a second line given by the axis s_i^y is calculated for each generic point $\mathbf{q} = [q^x, q^y]^T \in G$.
s_i^o is the orientation angle of s_i with respect of the world reference system and it is calculated using the rotation matrix of the sensor $\mathbf{R}^{s_i}$ (Figure 4.2):

$$\beta_i^{\mathbf{q}} = atan2(q^y - s_i^y, q^x - s_i^x) - s_i^o \quad , \quad \mathbf{q} \in G \wedge i \in [1, N_s] \tag{4.2}$$

where the function $atan2$ is the arctangent function with two arguments:

$$atan2(y, x) = 2 \arctan \frac{\sqrt{x^2 + y^2} - x}{y} \quad , \quad atan2(y, x) \in (-\pi, \pi) \tag{4.3}$$

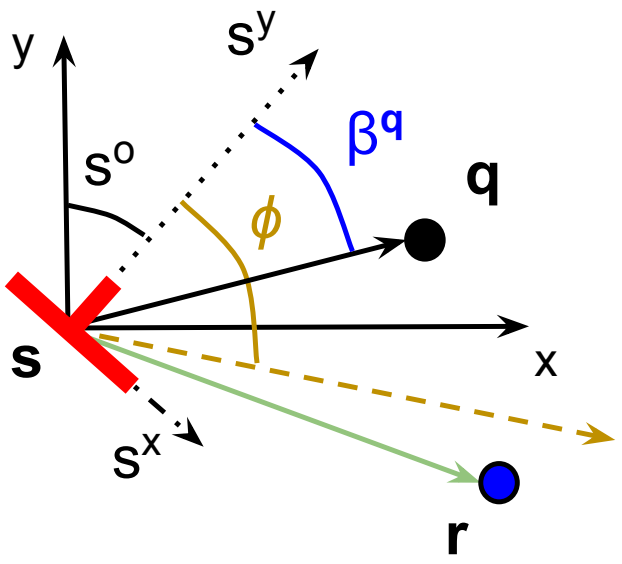


Figure 4.2: Representation of all considered angles in a DOA sensor.

The probability $\mathcal{G}_i$ in the grid G is given by evaluating in the angular probability sensor model $\mathcal{M}_i$ the difference between the angle $\beta_i^{\mathbf{q}}$ and the DOA estimation ϕ_i for each point $\mathbf{q}$:

$$\mathcal{G}_i = \mathcal{M}_i(\beta_i^{\mathbf{q}} - \phi_i) = \frac{1}{\sigma_i\sqrt{2\pi}} e^{-\frac{(\beta_i^{\mathbf{q}} - \phi_i)^2}{2\sigma_i^2}}, \quad i \in [1, N_s] \ \wedge \ \forall\, \mathbf{q} \in G \tag{4.4}$$

All results of the operations among angles in Equations (4.2) and (4.4) take a value in the range $[-\pi, \pi]$. The graphical representation of a single DOA sensor estimation probability $\mathcal{G}_i$ over the Cartesian plane can be seen in Figure 4.3.

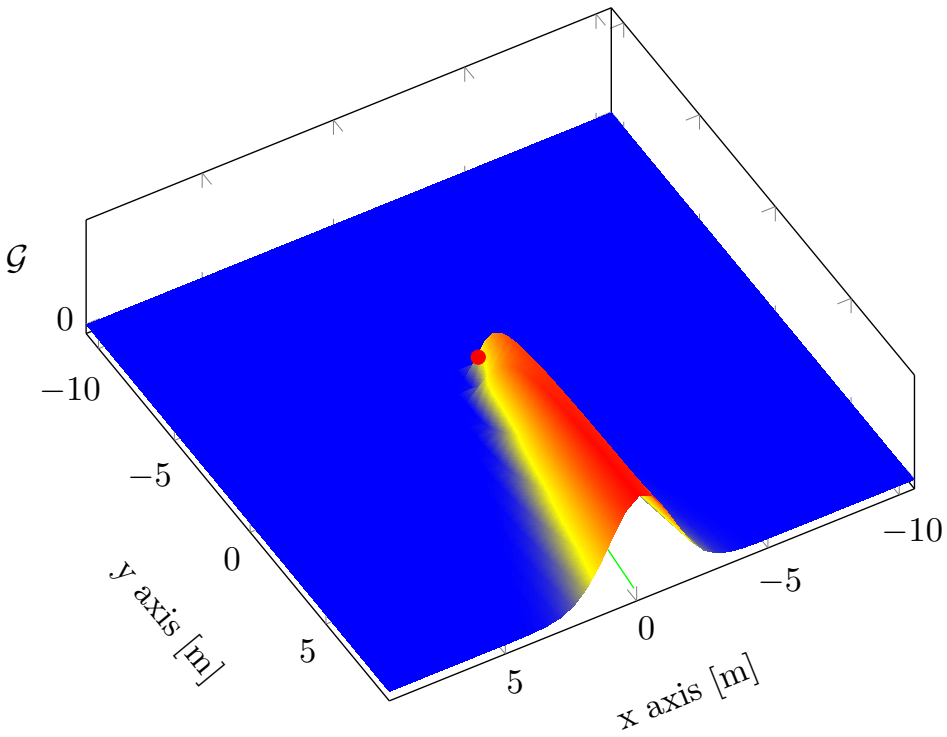


Figure 4.3: Graphical representation of a single DOA sensor estimation probability $\mathcal{G}_i$ over the 2D Cartesian plane G. Red and blue regions have higher and lower likelihoods respectively.

At this point all the $\mathcal{G}_i$ are multiplied point-wise for each $\mathbf{q} \in G$ obtaining the multiplication of all the probabilities of all sensors $\mathcal{G}$ in the 2D space domain (Figure 4.4).

$$\mathcal{G} = \prod_{i=1}^{Ns} \mathcal{G}_i = \prod_{i=1}^{Ns} \left[\frac{1}{\sigma_i \sqrt{2\pi}} e^{-\frac{(atan2(q^y - s_i^y, q^x - s_i^x) - s_i^o - \phi_i)^2}{2\sigma_i^2}} \right] \quad , \quad \forall\, \mathbf{q} \in G \tag{4.5}$$

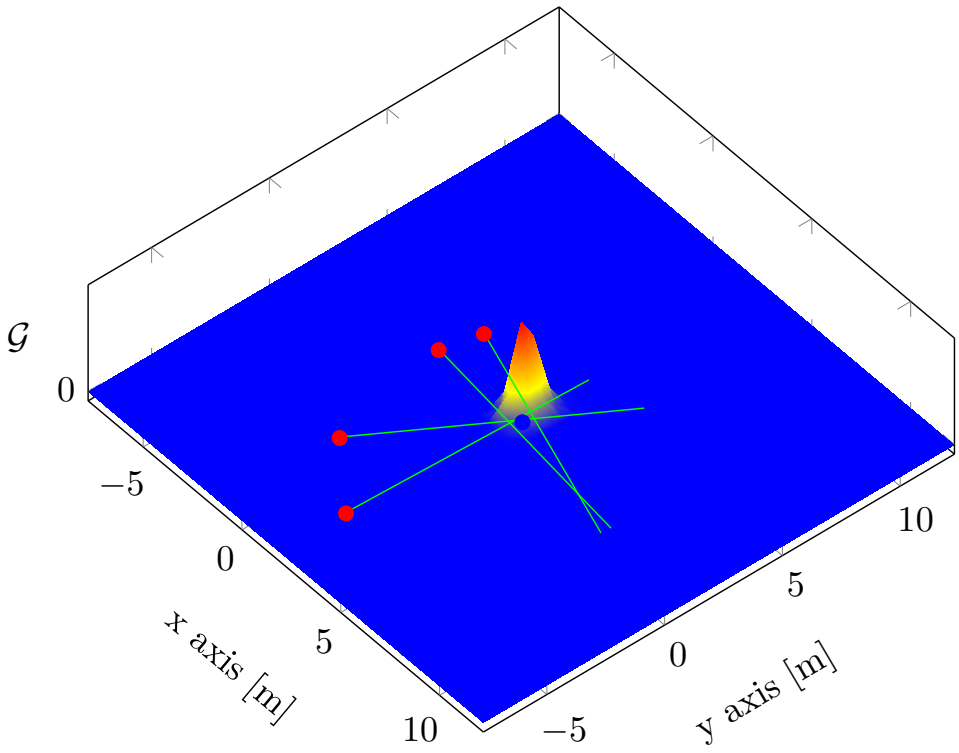


Figure 4.4: Example of Probability $\mathcal{G}$ over G with four DOA sensors.

Finally, the point in G with the maximum value of $\mathcal{G}$ is the estimation of the solution $\hat{\mathbf{r}} \in G$ with the proposed Gaussian Probability over DOA approach:

$$\hat{\mathbf{r}}_{GP} = \underset{G}{\operatorname{argmax}}\ \mathcal{G} \tag{4.6}$$

In Equation (4.5) the product is used instead of the sum for the fact that if the likelihoods from different DOA sensors are independent, the intersection of sets equals their product, as stated in [1].

It is worth noting that even though the probability axiom $P(\Omega) = 1$ is no longer satisfied in Equation (4.1), the omission of this axiom will not compromise the correctness of the procedure: dealing with multiplications - and not with sums - of probabilities (see Equation (4.5)), all unused values can be omitted because are not useful and because the multiplication of numbers $\in [0, 1]$ still takes a value $\in [0, 1]$. On the other hand it is important to set offline the experimented variances σ_i in order to represent correctly each angular probability DOA sensor model $\mathcal{M}_i$.

4.2 Algorithm Analysis and Optimization

The algorithm showed in section 4.1 has clearly a computational complexity of $\Theta(n^2)$ because it has to calculate all the probabilities in $G = (n \times n) \in \mathbb{Z}^2$ in order to find the maximum value in a two-dimensional matrix. In the following it is described a smarter and faster approach proposed for the first time in [16] for searching for the maximum value in G that analyses and takes in consideration the shape of $\mathcal{G}$.

Observing the shapes of the probability density function $\mathcal{G}$ (e.g. Figure 4.4) it seems that $\mathcal{G}$ is a Gaussian probability function. Unfortunately, each $\mathcal{G}_i$ is no longer a Gaussian like $\mathcal{M}_i$ although $\mathcal{G}_i = \mathcal{M}_i(\beta_i^{\mathbf{q}} - \phi_i)$. The reason is due to the formula that calculates $\beta_i^{\mathbf{q}}$ which translates the values from the Cartesian domain to the angle domain. This change of domain causes the loss of the property for $\mathcal{G}_i$ of being a Gaussian probability function, preventing from using the well-known property that states that multiplication of Gaussians is still a Gaussian [17]. On the other hand, in the majority of the cases, it can be noted that $\mathcal{G}_i$ is unimodal, hence it has only one absolute maximum:

Definition 1 *A function $f(x)$ is called **unimodal** if for some value m, it is monotonically increasing for $x \leq m$ and monotonically decreasing for $x \geq m$. In that case, the maximum value of $f(x)$ is $f(m)$ and there are no other local maxima.*

This property occurs when the DOA estimations are quite accurate. But if we are not under this specific condition it can happen that the maximum is no longer absolute. Fortunately, tests with increasing error over the DOA estimations show that this unpleasant event happens very rarely and in most of that cases the solution provided by the new search algorithm is still good (see Section 5.3.2). Before explaining the details of this algorithm I suggest a funny riddle:

The Top of the Unimodal Mountain

Suppose there is a robot-climber that needs to reach the top of an unimodal mountain. The robot always knows which are its geographical position and altitude - thanks to a compass and to an altimeter - but it doesn't know where is the top because there is too much fog. Fortunately the robot can move safely to any geographical position it wants only with its special jet-pack. Which is the method that let the robot reach the top of the unimodal mountain with the smallest number of uses of its special jet-pack?

The solution of this funny riddle is not very immediate so let's first concentrate on a single dimension of a n meters long mountain. If we start from the foot of the mountain (e.g. from the left) and "walk" towards the climb (on the right), we could find the top in $\Theta(n)$ time. If the fuel of the jet-pack is limited and the top of the mountain is on the top right, the robot couldn't reach the top of the mountain.

Hence, in a smarter way, we have to take advantage of the unimodal property of the mountain by applying a special Unimodal Binary Search (UBS) that takes also into consideration the slope of the mountain: for each point we explore using the binary search, we will explore also the adjacent point (i.e. the one on the left) and measure their respective altitudes in order to decide in which direction to jump, in a binary search way. This method will reasonably take $\Theta(\log_2 n)$ time.

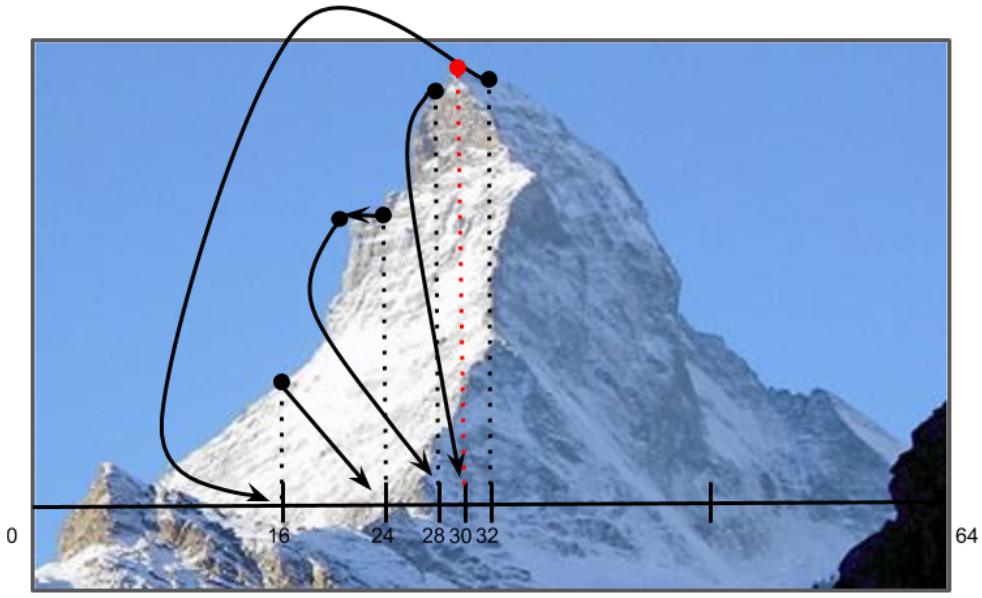


Figure 4.5: Example of a solution of the Top of the Unimodal Mountain riddle using the Unimodal Binary Search (UBS) approach.

Before going on, let's see now a complete simple example in one dimension: the unimodal mountain in Figure 4.5 is long $n = 64 = 2^l$ where $l = log_2 n = 6$ is the number of levels.
In the following there are all steps for reaching the top of the mountain:

1. Level $l = 6$: first the robot checks the altitude of position 32 (2^{l-1}) and its adjacent on the left (31) and sees that the top should be on the left.

2. Level $l = 5$: the robot jumps to the left to the position 16 (32 - 2^{l-1}) and checks also position 15.

3. The robot realizes that the altitude in 16 is higher than the one in 15 hence it jumps right to position 24 (16 + 8) (Level $l = 4$).

4. When the robot checks position 23 it sees that it has the same altitude than in position 24. So it starts walking on the left till it finds a different altitude.

5. Since in position 20 there was a lower altitude, the robot jumps right to position 28 (24 + 4) (Level $l = 3$). Note that the robot can "walk" on the left only for 2^{l-1} position because the other left positions are already been checked or can be discarded using the previous information.

6. At Level $l = 2$ the robot jumps to position 30 (28 + 2) where it has to check positions 29 and 31 because the robot is at the end of the algorithm.

7. Finally, positions 29 and 31 give minor altitudes so the robot can state that the top of the mountain is in position 30.

For the two-dimensional unimodal mountain case it can be used a two-level Unimodal Binary Search: the first level is a UBS over the rows; for each visited row the approach applies another UBS (second level) that finds the position of the maximum point of that row always in a UBS way. With this procedure we reach the top of the mountain in $\Theta((\log_2 n)^2) = \Theta(\log_2^2 n)$ time instead of the first naïve method that took $\Theta(n^2)$: a very big amount of time and fuel saved!

Working with probabilities and floating-point numbers it can happen that two adjacent points have the same probability (see step 4 of the example). In this case the above algorithm cannot decide where to jump. A solution of this problem is, remembering the mountain, to "walk" always in the same direction until we find a different altitude. So if there are plain areas that are all $m < n$ long the overall complexity of the algorithm is $\Theta((m + \log_2 n)^2) = \Theta(m^2 + \log_2^2 n + m \cdot \log_2 n)$. Since the experimental tests were always in the $m \ll n$ condition, the computational time is still $\Theta(\log_2^2 n)$.

4.3 3D DOA Source Localization

For the 3D space extension there are some differences that must be analysed. For the explanation of the approach we need to discretize the 3D space introducing a cubic 3D spatial grid $G = (n \times n \times n) \in \mathbb{Z}^3$ with a fixed spatial *range* [m] and a fixed precision parameter *prec* [m] such that $n = range/prec$ and $\mathbf{q} = [q^x, q^y, q^z]^T \in G$. We now know from the formalization of the problem that every sensor s_i has its own pose that is composed by a position in the *world* defined as $\mathbf{T}_i^s = [s_i^x, s_i^y, s_i^z]^T$ and an orientation $\mathbf{R}_i^s$.

Taking this into consideration, at this step we suppose that the sensor s_i detected a sound and estimated the DOA calculating the azimuth ϕ_i and the elevation θ_i. For using the probabilistic model $\mathcal{M}$ of Equation 4.1, we need a further step that consists in calculating the angle α_i between the estimated DOA and the generic point $q \in G$, that are vector $\mathbf{s}_i\mathbf{p}$ and the vector $\mathbf{s}_i\mathbf{q}$ respectively, with the following formula (see Figure 4.6):

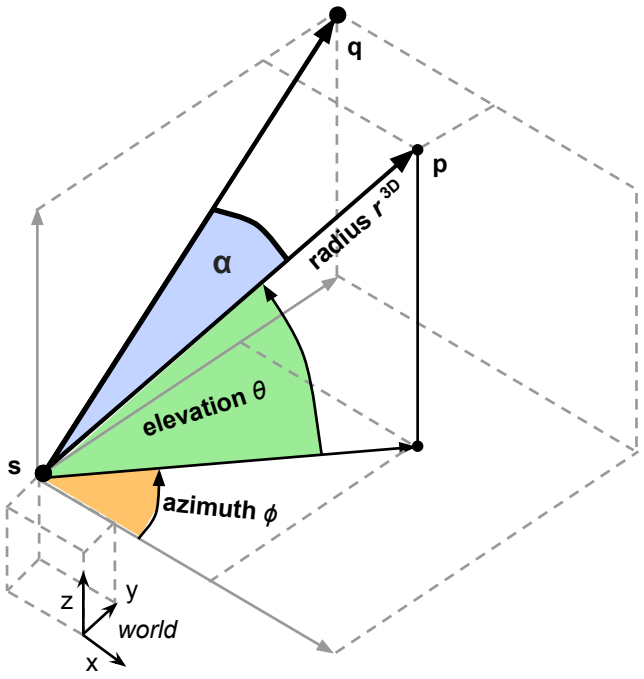


Figure 4.6: Angle α for the 3D model $\mathcal{M}$.

$$\alpha_i = \frac{acos(\mathbf{s}_i\mathbf{p} \cdot \mathbf{s}_i\mathbf{q})}{||\mathbf{s}_i\mathbf{p}|| * ||\mathbf{s}_i\mathbf{q}||} \quad , \quad \alpha \in [0, \pi] \tag{4.7}$$

As stated in assumption 3, each real DOA estimation has an intrinsic error that depends only on the accuracy and precision of the i-th sensor. This error can be modelled as a Gaussian probability error in the angle domain with zero mean and variance σ_i using only values in the range of $\alpha \in [0, \pi]$. So the angular probability sensor model $\mathcal{M}_i$ of Equation 4.1 is redefined as follows:

$$\mathcal{M}_i \sim \mathcal{N}(0, \sigma_i)_{(-\pi,\pi]} = \frac{1}{\sigma_i\sqrt{2\pi}} e^{-\frac{\alpha_i^2}{2\sigma_i^2}} \quad , \quad \alpha_i \in [0, \pi] \tag{4.8}$$

For graphical reasons the shape of the probability model cannot be shown. The three-dimensional model can be imagined thinking that the Gaussian is equally distributed in all three directions. So Equation 4.5 becomes:

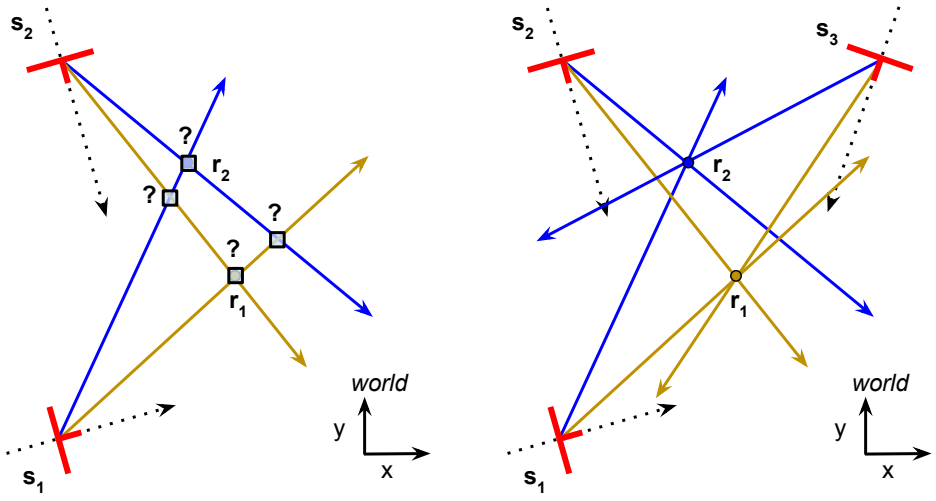
$$\mathcal{G}(\mathbf{q}) = \prod_{i=1}^{Ns} \mathcal{G}_i(\mathbf{q}) = \prod_{i=1}^{Ns} \left[\frac{1}{\sigma_i\sqrt{2\pi}} e^{-\frac{\left(\frac{acos(\mathbf{s}_i\mathbf{p}\cdot\mathbf{s}_i\mathbf{q})}{||\mathbf{s}_i\mathbf{p}||*||\mathbf{s}_i\mathbf{q}||}\right)^2}{2\sigma_i^2}} \right] \quad , \quad \mathbf{q} \in G \tag{4.9}$$

and from now on the algorithm follows the same procedure described in Section 4.1.

The algorithm showed has clearly a computational complexity of $\Theta(n^3)$ because it has to calculate all the probabilities in $G = (n \times n \times n) \in \mathbb{Z}^3$ in order to find the maximum value in a three-dimensional matrix (brute force). Using the algorithm explained in Section 4.2 it is possible to reach the same results of the brute force algorithm in $\Theta(\log^3 n)$ time.

4.4 DOA Multiple-Source Localization

As said before, dealing with multiple coexisting acoustic sources makes the DOA-MSL problem more complicated than the DOA-SSL one because it introduces the problem of selecting and estimating among the same sensor the DOA estimation that refers to a specific source. In the limited field given by the initial assumption 2, we know that each sensor estimates n sources; so we can say that $n = N_R$ and then we will search for all the N_R different solutions. In the following it will be considered the case with $N_S \geq 3$ because the case $N_S = 2$ arise the problem of selecting the right DOA intersections (Figure 4.7).



(a) abc (b) abc

Figure 4.7: Problem of selecting the right DOA intersection. In the 2D space, with only two sensors it is not possible to guess where are the location of two sources (a). There is a need of a third sensor in order to give a solution (b).

In the GP-DOA approach, multiplying all $\mathcal{G}_{ij}$ in a same probability map $\mathcal{G}$ and trying to find the N_R maxima would not be a correct approach because DOA estimations of different sources would mutually interfere and the probability likelihood $\mathcal{G}$ would no longer be unimodal.

So the proposed approach collects and calculates all possible combinations of DOA estimations, one for each sensor, that in this case are $N_R^{N_S}$. Unfortunately, calculating all the possible solutions takes an exponential computational time of $\Theta(N_R^{N_S})$ that is the main drawback of the approach. After collecting all the possible solution, the algorithm sorts the solutions by the

value of the probability $\mathcal{G}(\hat{\mathbf{r}}_j)$ for the GP-DOA approach and by the Average Distance Criterion (ADC) of Equation 3.12 for the WLS-DOA approach. The solution cannot be the first N_R sorted solutions because it can happen that, due to the intrinsic errors of the estimations, two or more of the top N_R solutions are calculated using the same DOA estimation $\phi_{i,j}$. So it is extracted only the first solution (the most probable) and then the solution set is pruned by eliminating all the solutions that were calculated used at least one of the DOA estimations used for calculating the most probable solution. At this step, the solution set is still sorted and the current first solution, that is the second most probable solution among the available ones, is extracted. Iterating this procedure N_R times we will find out all N_R estimations.

The Multiple-Source GP-DOA Selection Algorithm 1 takes in input the solution set S and outputs the estimated positions of the N_R sources. The Solution set S contains the following information for each of the $N_R^{N_S}$ solutions:

- estimated source position in the 2D/3D space;
- probability p of the estimated source position calculated using the GP-DOA approach;
- index of the DOA estimation used for calculating the estimated source position for each DOA sensor.

Algorithm 1 Multiple-Source GP-DOA Selection Algorithm.

```
1: procedure FIND_MULTIPLE_SOURCES_SOLUTIONS(S)
2:     S ← sort(S, p)
3:     for j ∈ [1, N_R] do
4:         r_j ← top(S)
5:         for k ∈ [1, |S|] do
6:             if S_k used a DOA estimation already used in r_j then
7:                 delete S[k] from S
```

Chapter 5

Validation with Simulation

5.1 Introduction

Simulation was an helpful tool for comparing the localization performances of the presented approaches for solving the DOA-SL problem before implementing them for the real experiments. The simulation scenarios consisted in groups of N_S DOA sensors positioned in a virtual room G. It was also assumed that the room G was without objects or obstructions and was always a perfect square/cube of side n. The sound reflections over the wall were not considered as previously assumed in assumption 3. At each iteration, all sensors' positions and orientations were positioned randomly in G. For each test iteration the acoustic sources were also positioned randomly in G. For all simulations it has been assumed that all sensors were similar so they had the same probability error over the DOA estimation. Hence, for simulation purposes, each DOA estimation was modified from the real one using the probability model $\mathcal{M}_i$ with the same σ_{i_sim} for all sensors. Each simulation test was repeated t_{sim} times for each group of sensors. The cardinality of the sensors' group N_S was varied in the range $[3, N_{S\_max}]$. The case with $N_S = 2$ was not considered for the reason that all the approaches give the same solution that is the trivial intersection of the only two existing DOAs - if the DOAs are not parallel. Simulation code was implemented in MATLAB [18] using the Parallel Toolbox to speed up the simulation time.

5.2 Simulation Tests

In the following are listed all the simulations done in chronological order:

1. Simulation for testing the accuracy and precision of the basic GP-DOA algorithm versus the other state-of-art approaches;
2. Simulation for testing the accuracy and precision of the fast GP-DOA algorithm;
3. Simulation for testing the accuracy and precision of the fast GP-DOA algorithm augmenting only the error over the DOA estimations;
4. Simulation for testing the accuracy and precision of the fast GP-DOA algorithm augmenting only the error over the position of the DOA sensors;
5. Simulation for testing the accuracy and precision of the fast GP-DOA algorithm in a 3D scenario;
6. Simulation for testing the accuracy and precision of the Multiple-Source GP-DOA approach in a 3D scenario.

The order of the simulations follows the research steps. First of all it was tested the accuracy and precision of the GP-DOA approach that was proposed for the first time in [10]. Next there was the need to evaluate the performances of the fast GP-DOA [16] and to see if the new theoretic approach could be used without a loss of performance and most of all to check if the binary search approach would not reach local maxima in the majority of the cases, making the approach unusable. Once the GP-DOA Fast approach resulted to be robust to local maxima, I tested the behaviour of the algorithm inserting two errors separately: the error over the DOA angle and the error over the position of the DOA sensors in order to see if the approach could be used in a robotic environment. Finally I extended the algorithm to the 3D scenario adding also the possibility to have multiple sound sources at the same time.

The localization performance metric used was the Distance Error (DE) that is the distance between the estimated source position and the real source position in meters [m] [1]. The validation metrics calculated were the mean and variance values of all DEs. For each simulation type I considered the following parameters:

- *range* [m] - the side of the square/cube of the grid G;
- σ_{k_sim} - the variance of the DOA sensor Gaussian model;
- N_{S_max} - the maximum number of sensors in the environment;
- t_{sim} - the number of repetitions of the simulation test;
- GP-DOA Basic *prec* [m] - the precision of the grid G for the GP-DOA Basic approach;
- GP-DOA Fast 1 *prec* [m] - the precision of the grid G for the GP-DOA Fast 1 approach;
- GP-DOA Fast 2 *prec* [m] - the precision of the grid G for the GP-DOA Fast 2 approach;
- $e_{\phi_i}^{max}$ [degrees] - the maximum error of the angle given by the DOA estimations;
- e_{s_i} [m] - the maximum error of the position of the sensors in G.

Table 5.1 summarizes all the parameters' choices for each simulation test.

# Simulation	1	2	3	4	5	6
range [m]	15	10	10	10	10	10
σ_{k_sim}	0.1	0.1	0.1	0.1	0.1	0.1
N_{S_max}	20	20	20	20	20	8
t_{sim}	10^3	10^5	10^5	10^5	10^5	10^2
GP-DOA Basic *prec* [m]	10^{-3}	10^{-2}	/	/	/	/
GP-DOA Fast 1 *prec* [m]	/	10^{-2}	/	/	/	/
GP-DOA Fast 2 *prec* [m]	/	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}
$e_{\phi_i}^{max}$ [degrees]	5	5	[5:5:50]	5	5	5
e_{s_i} [m]	0	0	0	[0:0.1:1]	0	0

Table 5.1: Simulation Parameters used for each different type of simulation.

5.3 Simulation Results

In this section are listed and commented all the results from all the simulation tests.

5.3.1 GP-DOA vs. Other state-of-art approaches

The parameters used are for the first simulation are listed in Table 5.2.

$range$ [m]	σ_{k_sim}	N_{S_max}	t_{sim}	GP $prec$ [m]	$e_{\phi_i}^{max}$ [deg]	e_{s_i} [m]
10	0.1	15	10^3	10^{-3}	5	0

Table 5.2: Parameters used for tests in Simulation 1.

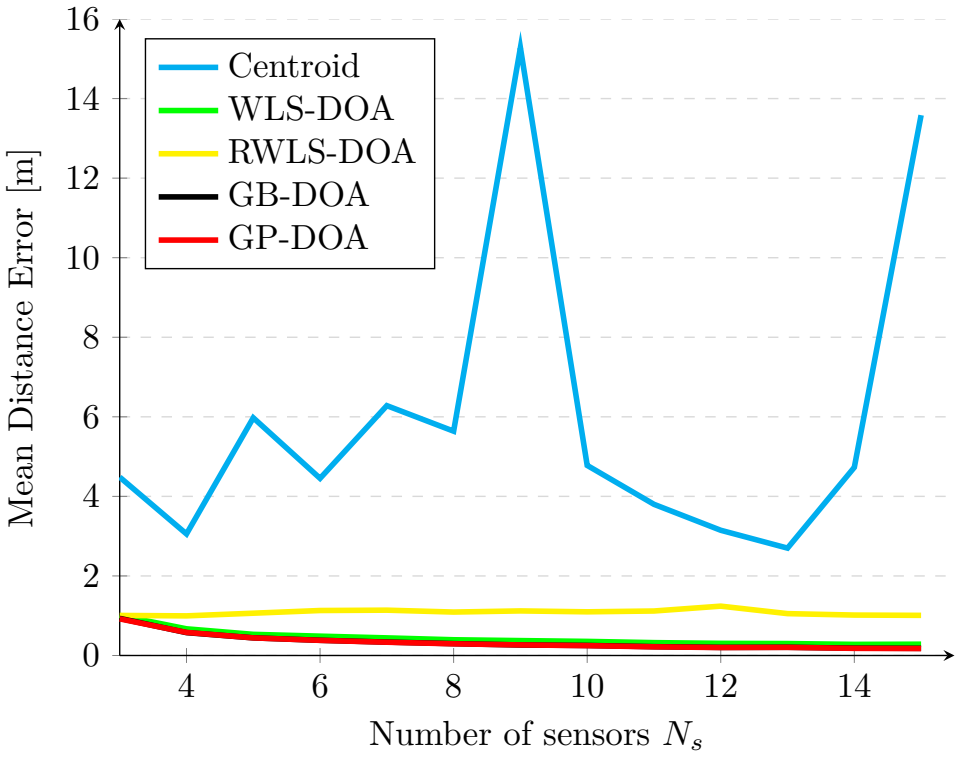


Figure 5.1: Mean error comparison among all DOA approaches.

The Figure 5.1 shows the results of the mean values of the Distance Error for the first simulation test with one thousand iterations. It is clear that the Centroid method is not a good approach because it has not a stable mean.

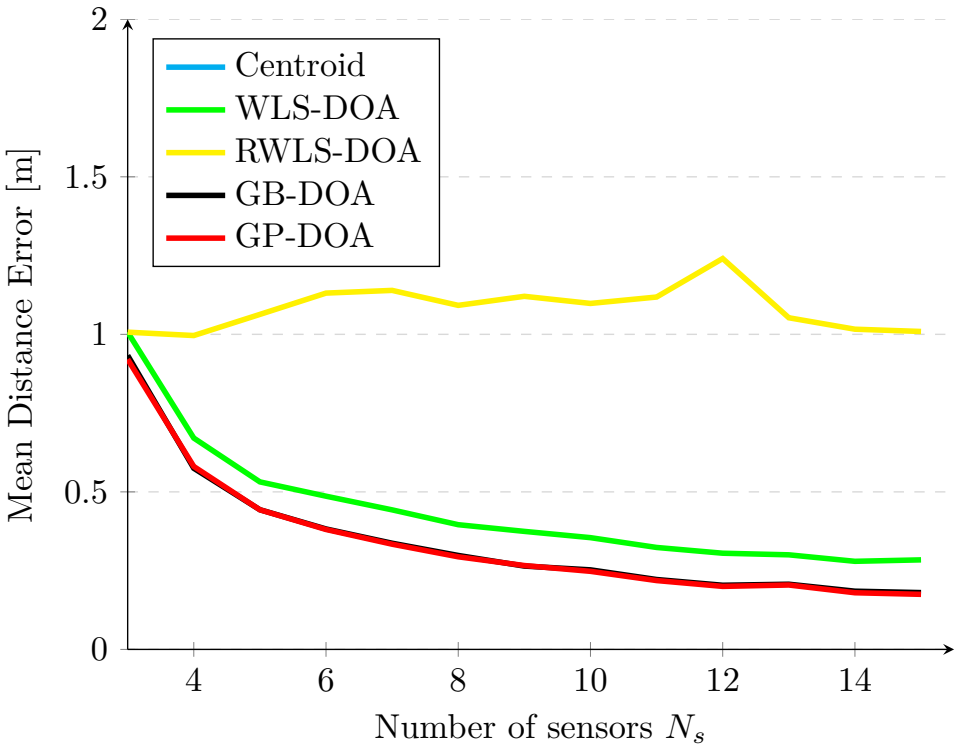


Figure 5.2: Zoom of the Mean error comparison among all DOA approaches.

In Figure 5.2 there is a zoom of the previous figure that give focus on the behaviour of the other approaches. The Robust WLS-DOA approach (RWLS-DOA) is almost stable with an error of about one meter. The remaining three approaches have a different behaviour while the number of sensors augments. The mean of the Distance Error for the WLS-DOA, the GB-DOA and the GP-DOA behaves similarly and the approaches give better results with more sensors. The GP-DOA approach is a grid-based approach as the GP-DOA and differs from the latter with the formula for calculating the angular distance function. As a matter of facts the behaviour and the slope are very similar but the GP-DOA approach reaches slightly better results.

Looking at the results of the variance of the Distance Error in Figure 5.3 we can see that the Centroid method is not plotted because its variance values were too big. The RWLS-DOA method has a big variance that reflects the mean error values and for these reasons cannot be considered properly robust.

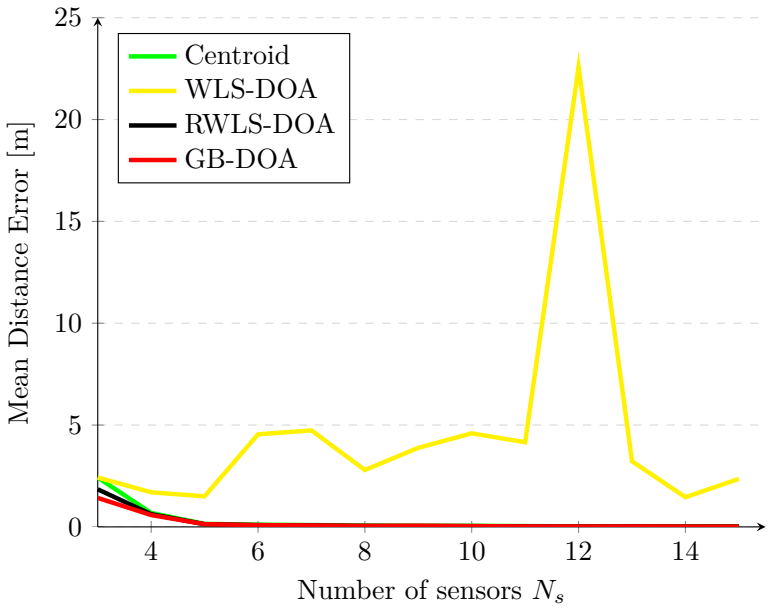


Figure 5.3: Variance error comparison among all DOA approaches.

If we look to the zoom figure of the variance Distance Error in Figure 5.4 we see that the remaining three approaches have a similar behaviour that again underlines the fact that the precision augment while the number of sensors grows.

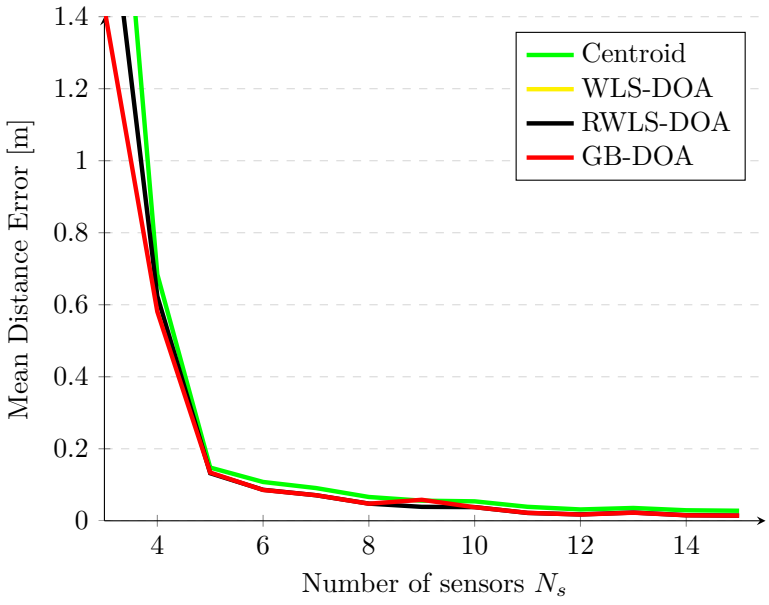


Figure 5.4: Zoom of the Variance error comparison among all DOA approaches.

RWLS-DOA approach failures

Although it was shown in [14] that the RWLS-DOA method performed better than WLS-DOA in outlier situations, I found that it is error prone, because in some situations it can discard the DOA sensors that are better than others. A proof of the failure of this method can be easily given if we consider the example in Figure 5.5 were we can see that the Robust WLS-DOA solution (in magenta) is very far from the real one (in blue). On the other hand the WLS-DOA solution (in green) and GP-DOA approach (in red) solutions are closer to the real solution. This is due to the fact that the DOA estimations of sensors s_1, s_2 and s_3 intersect very close to each other (near the RWLS-DOA solution), not considering that only sensors s_1 and s_5 are the only good estimations and the others s_2, s_3 and s_4 are outliers. So, it is not advisable to discard any of the sensors because the real position of the source is unknown and it is not possible to detect which DOA sensors are outliers if they have the same probability error to be outliers. Furthermore, the simple WLS-DOA approach can also lead to errors because a WLS-DOA estimation is considered as a line without a precise direction. As an example, in Figure 5.5, even if the DOA estimation of sensor s_2 is pointing towards the south-east, the WLS-DOA approach considers also the north-west direction (RWLS-DOA fails also for this reason).

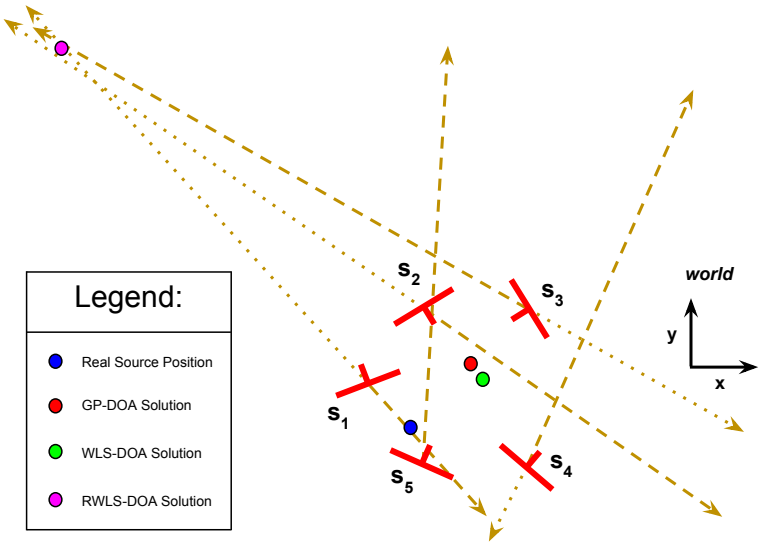


Figure 5.5: Example of failure of the Robust DOA-Based Localization proposed in [1].

GP-DOA vs. WLS-DOA

In this subsection it is analysed in further detail the difference between the GP-DOA and the WLS-DOA approaches because they resulted to be the best among all approaches. In Figure 5.6 we see that GP-DOA performs better in mean than WLS-DOA approach and the error diminishes as the number of sensors grows in the environment. Conversely, the WLS-DOA variance performs better than GP-DOA variance (Figure 5.7) that means that GP-DOA has an higher accuracy but lower precision with respect to the WLS-DOA approach.

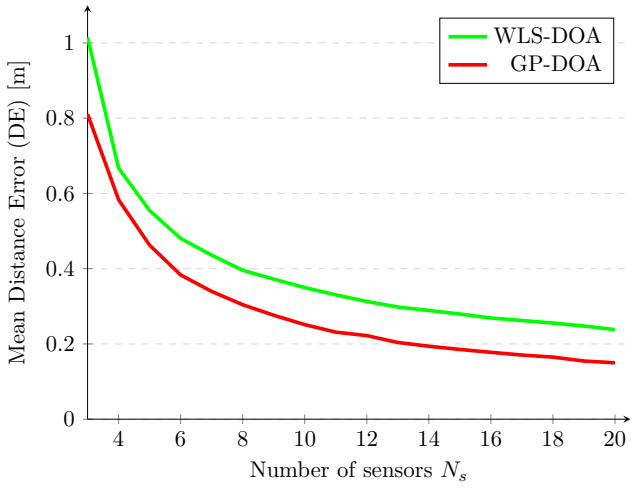


Figure 5.6: Mean error comparison among GP-DOA and WLS-DOA.

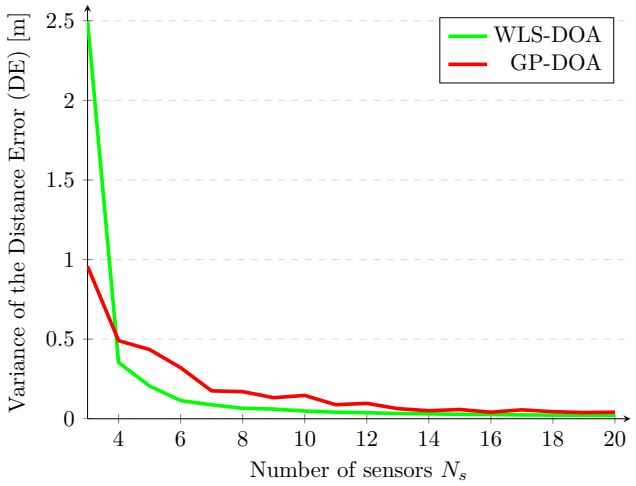


Figure 5.7: Variance error comparison among GP-DOA and WLS-DOA.

5.3.2 GP-DOA Basic vs. GP-DOA Fast

In Figures 5.8 and 5.9 we can see the results of the mean and the variance of the second simulation test. Here I used two different values of the precision *prec* for the Grid G for testing the performance of the GP-DOA Fast approach. Looking first at the mean in Figure 5.8, we can see that both GP-DOA Fast approaches are better than the WLS-DOA method. Although having the same precision of the grid of the GP-DOA Basic approach, GP-DOA Fast 1 is less precise and less accurate, as it can be seen also looking at the variance of Figure 5.9. This can be addressed to the fact that the shape of G is not always unimodal and so the algorithm found a local maximum. Augmenting the precision of the grid to 10^{-4} meters, it allows to reach practically the same results obtained with the GP-DOA Basic method both in mean and in variance but with much less time. The parameters used are listed in Table 5.3.

range [m]	10
σ_{k_sim}	0.1
N_{S_max}	20
t_{sim}	10^5
GP-DOA Basic *prec* [m]	10^{-2}
GP-DOA Fast 1 *prec* [m]	10^{-2}
GP-DOA Fast 2 *prec* [m]	10^{-4}
$e_{\phi_i}^{max}$ [degrees]	5
e_{s_i} [m]	0

Table 5.3: Parameters used for tests in Simulation 2.

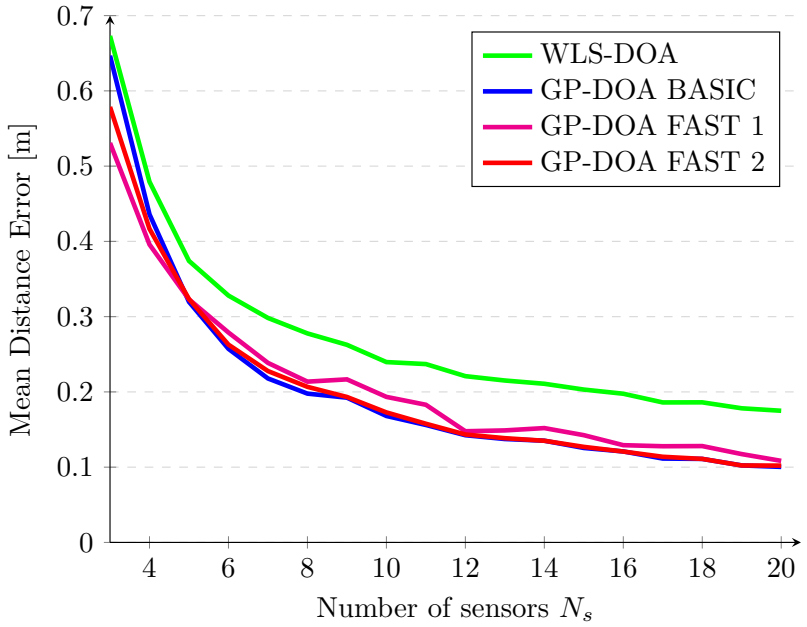


Figure 5.8: Mean error comparison among GP-DOA Basic, GP-DOA Fast 1 and 2 and WLS-DOA approaches.

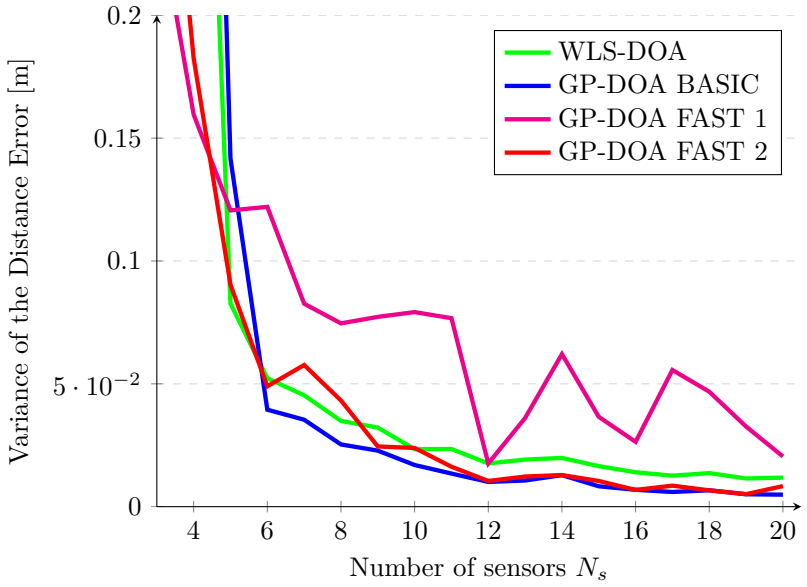


Figure 5.9: Variance error comparison among GP-DOA Basic, GP-DOA Fast 1 and 2 and WLS-DOA approaches.

5.3.3 Varying the DOA Angle Error

In this simulation I varied the Maximum Angle Error over the DOA estimation $e_{\phi_i}^{max}$ between 5 and 50 degrees with a step of 5 degrees to see the behaviour of the GP-DOA Fast compared with the WLS-DOA approach. The simulation results, shown in Figure 5.10, reveal that GP-DOA Fast always outperforms the WLS-DOA approach[1]. Moreover the source estimation error seems to be linearly dependent on the angle error and the different slopes show that the GP-DOA approach is more robust over the error of the DOA angle than the WLS-DOA approach. Finally, N_S maintains its inverse proportionality with respect of the source estimation error.

range [m]	σ_{k_sim}	N_{S_max}	t_{sim}	GP *prec* [m]	$e_{\phi_i}^{max}$ [deg]	e_{s_i} [m]
10	0.1	20	10^5	10^{-4}	[5:5:50]	0

Table 5.4: Parameters used for tests in Simulation 3.

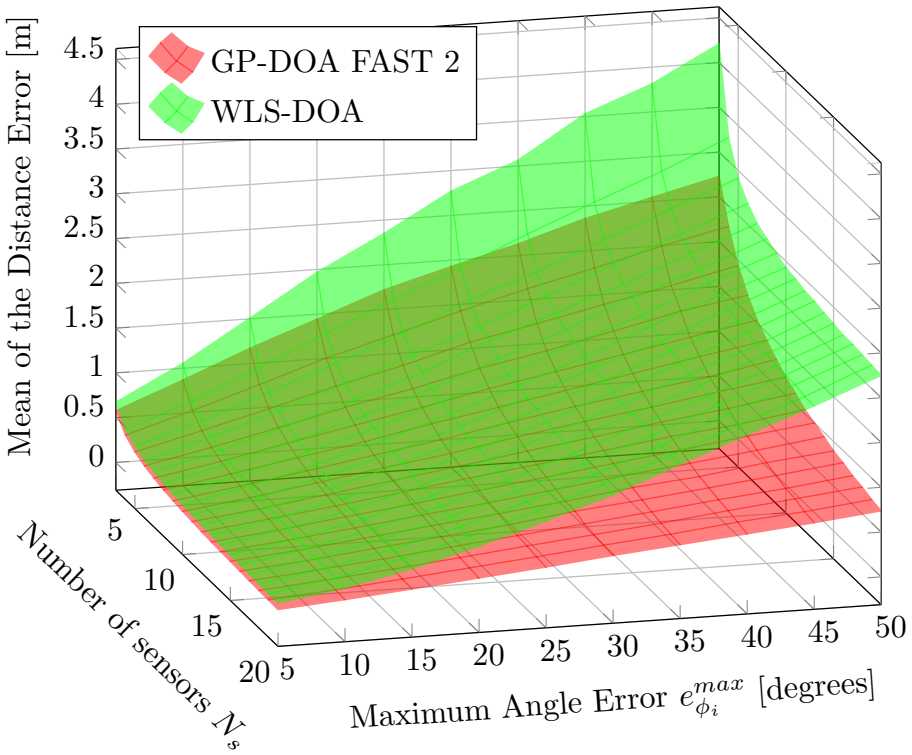


Figure 5.10: Simulation results of GP-DOA Fast 2 and WLS-DOA approaches varying the Maximum Angle Error over the DOA estimation $e_{\phi_i}^{max}$.

[1]The variance chart is omitted because it is similar to the mean one.

5.3.4 Varying the DOA Sensor Position Error

The results varying the Maximum Position Error over the DOA sensors $e_{s_i}^{max}$ are reported in Figure 5.11[2]. It can be easily noted that WLS-DOA starts to outperform the GP-DOA approach when the estimation of the position of the sensors in the map has a maximum error of 0.4-0.5 [m]. This threshold can be used in a real environment in order to select the approach to be used for ASL purposes. Since we are dealing with small rooms, a robot position error of 0.5 [m] can be reasonably considered as a borderline for actual SLAM algorithms so it would be reasonable to use the GP-DOA approach.

range [m]	σ_{k_sim}	N_{S_max}	t_{sim}	GP *prec* [m]	$e_{\phi_i}^{max}$ [deg]	e_{s_i} [m]
10	0.1	20	10^5	10^{-4}	5	[0:0.1:1]

Table 5.5: Parameters used for tests in Simulation 4.

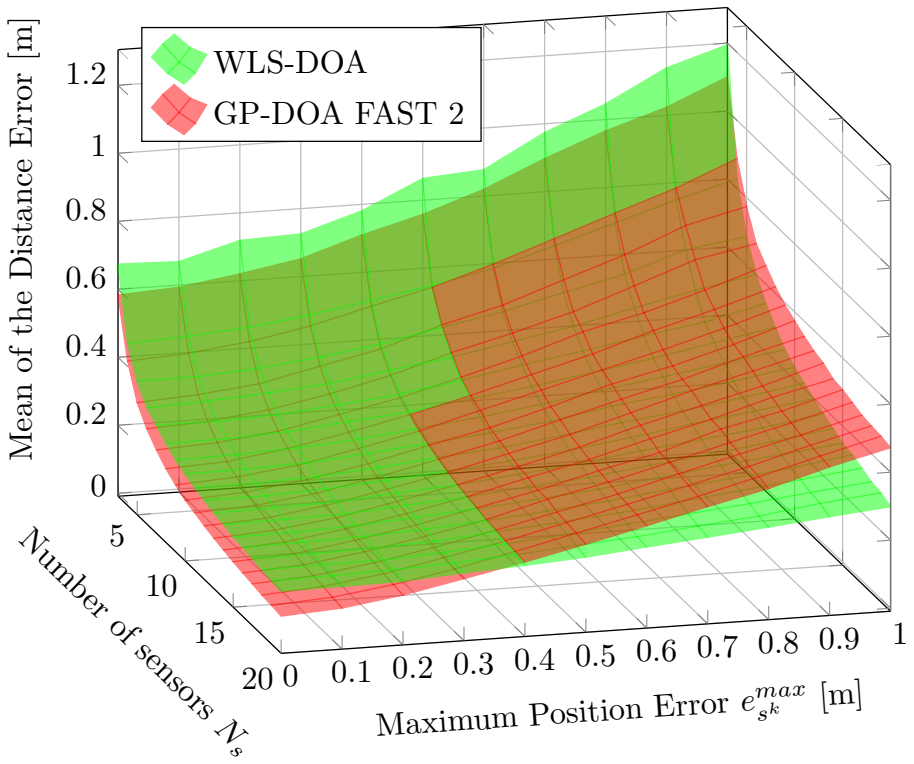


Figure 5.11: Simulation results of GP-DOA Fast 2 and WLS-DOA approaches varying the Maximum Position Error over the DOA sensors $e_{s_i}^{max}$.

[2]The variance chart is omitted because it is similar to the mean one.

5.3.5 GP-DOA in 3D

In this simulation I extended the research also in the z axis. A MATLAB example is shown in Figure 5.12. The results are shown in Figures 5.13 and 5.14 and the behaviour is the same as in the 2D scenario as expected.

range [m]	σ_{k_sim}	N_{S_max}	t_{sim}	GP *prec* [m]	$e_{\phi_i}^{max}$ [deg]	e_{s_i} [m]
10	0.1	20	10^3	10^{-3}	5	0

Table 5.6: Parameters used for tests in Simulation 5.

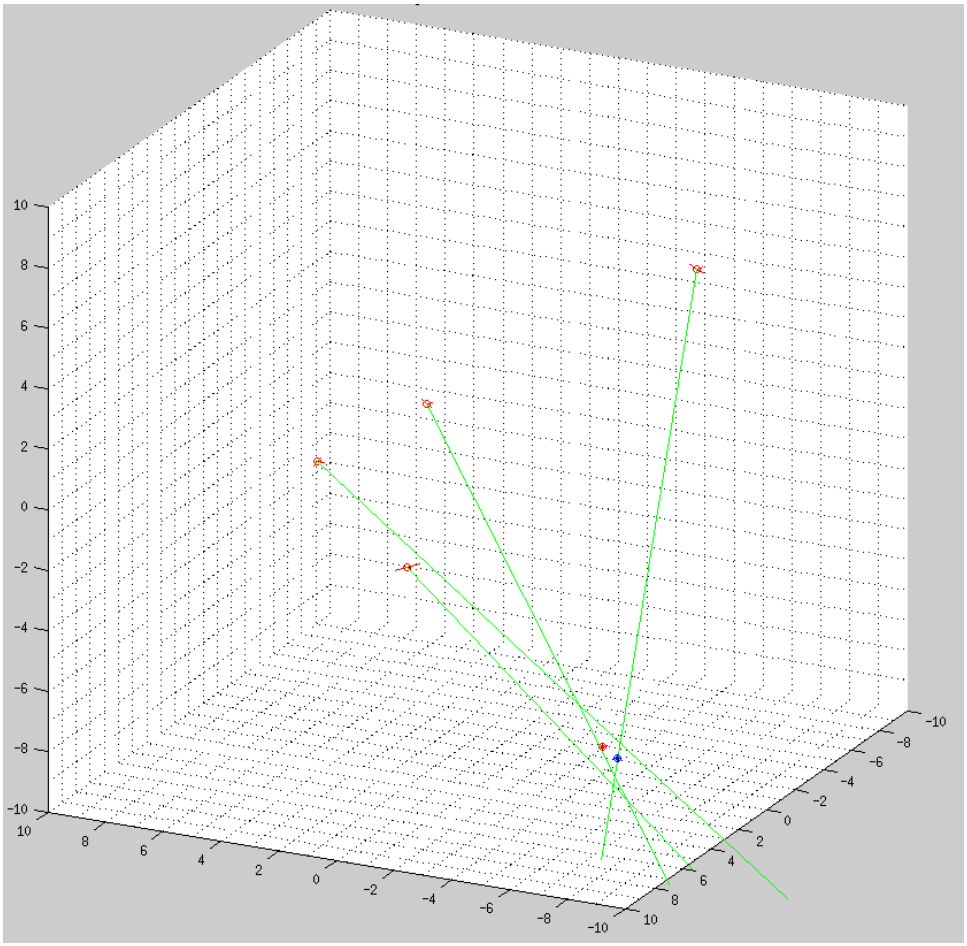


Figure 5.12: Example of 3D simulation in MATLAB. The blue point is the real source position and the red point is the GP-DOA estimation.

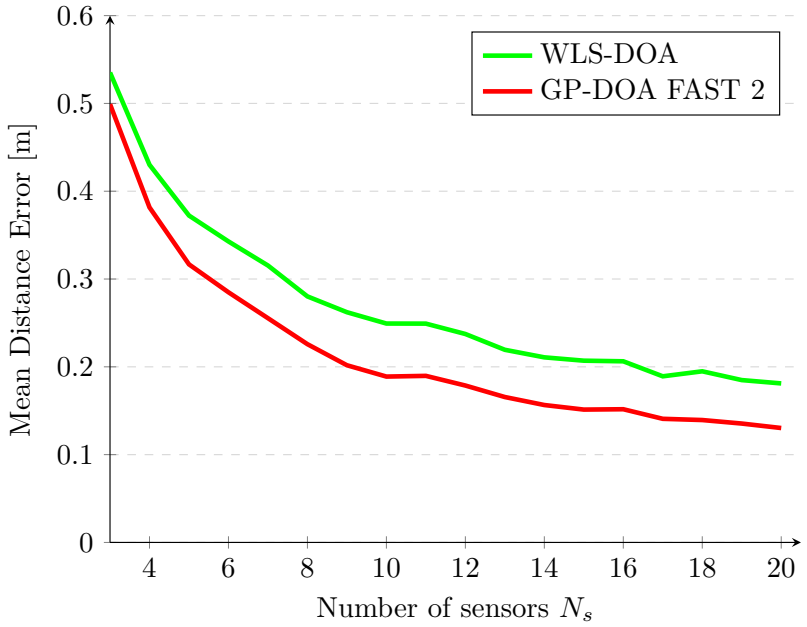


Figure 5.13: Mean error comparison among GP-DOA Fast 2 and WLS-DOA approaches in the 3D space.

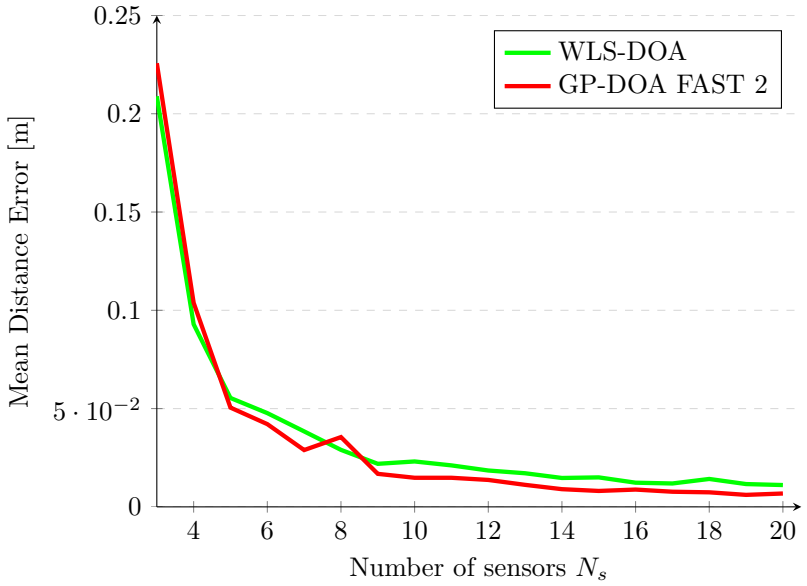


Figure 5.14: Variance error comparison among GP-DOA Fast 2 and WLS-DOA approaches in the 3D space.

5.3.6 3D Multiple-Source GP-DOA approach

In Figure 5.15 we can see an example of the 3D simulation with three sources.

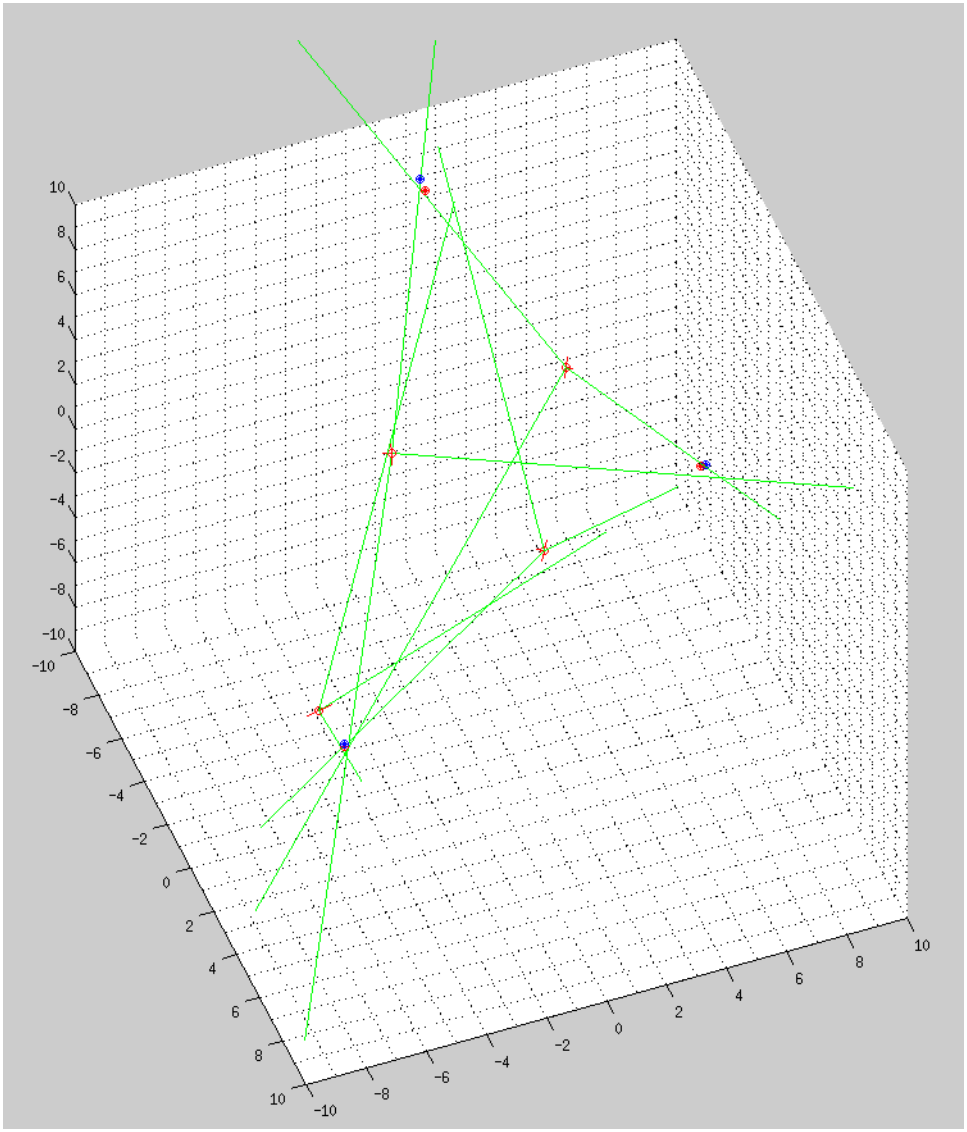


Figure 5.15: Example of 3D simulation with multiple sources in MATLAB. The blue points are the real sources positions and the red points are the GP-DOA estimations.

The parameters used for the last simulation test are listed in Table 5.7.

range [m]	σ_{k_sim}	N_{S_max}	t_{sim}	GP *prec* [m]	$e_{\phi_i}^{max}$ [deg]	e_{s_i} [m]
10	0.1	8	10^2	10^{-3}	5	0

Table 5.7: Parameters used for tests in Simulation 6.

Localizing up to three sources at time and applying the approach described in Algorithm 1 revealed that the GP-DOA approach is more robust than the WLS-DOA while the number of sources augments5.16.

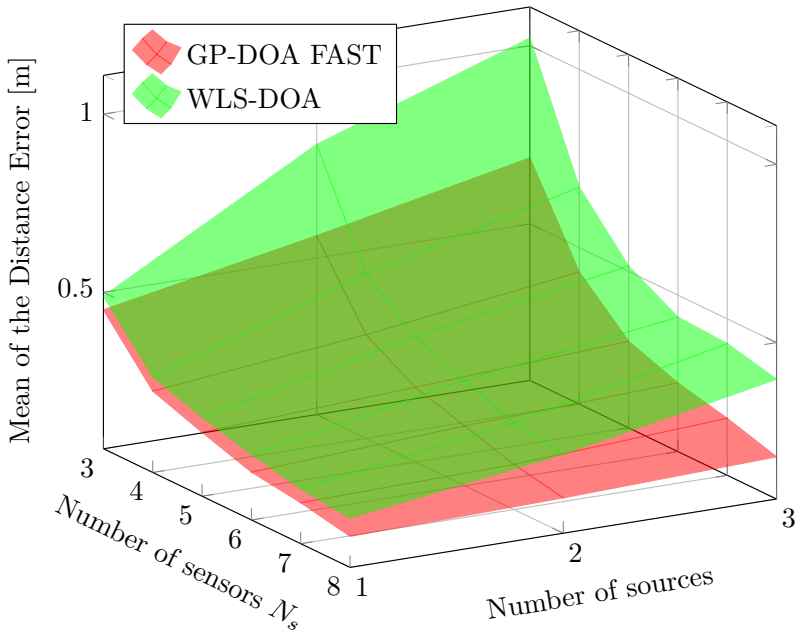


Figure 5.16: Mean error comparison among GP-DOA Fast and WLS-DOA approaches in the 3D space with multiple sources.

Chapter 6

Validation with Real Tests

6.1 Introduction

A real DOA sensor is a microphone array with at least two microphones. The DOA can be calculated taking into account the Time Difference of Arrival (TDOA) of the acoustic source among the signals of the microphones. The number and the position of the microphones in the space is significant. For the real tests I used mainly the Microsoft Kinect for Xbox 360[1] (old model) as DOA sensor. Microsoft Kinect has one RGB-camera, a 3D depth sensor and four microphones positioned as shown in Figure 6.1[2]. Each DOA estimation came from *HARK* software[3] implemented in a ROS package with an error of $\pm 0,0873$ [rad] $= 5°$ [19]. For the sensors extrinsic calibration, I used the ROS *multisensor_calibration* software[4] [20] developed in our laboratory that helped us to easily calibrate and find the extrinsic parameters among Kinect RGB-cameras. Starting from this calibration, I translated each RGB-camera reference point to the reference point of the DOA estimation given by *HARK* with respect to the reference system *world*. The reference point of the DOA estimation given by *HARK* is exactly the center point over the x axis of the Kinect as shown in Table 6.1.

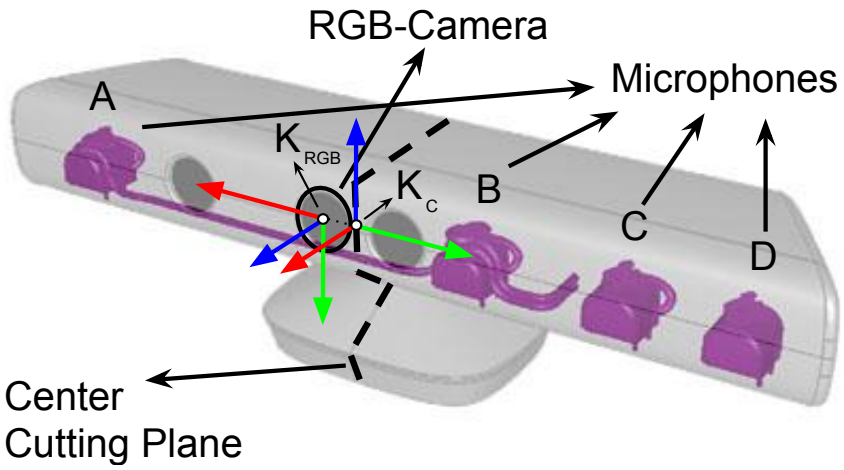


Figure 6.1: Kinect sensors - Microphones (A-D) and RGB-camera - positions in 3D. Kinect RGB Reference System is shown with red arrows. The black dashed line represents the center of the Kinect with respect to the x axis.

[1] http://www.xbox.com/kinect

[2] Note: all figures has the following well-known common notation for 3D axes: x axes are in red, y axes are in green and z axes are in blue.

[3] http://winnie.kuis.kyoto-u.ac.jp/HARK

[4] https://github.com/iaslab-unipd/multisensor_calibration

Mic A [m]	K_{RGB} [m]	K_C [m]	Mic B [m]	Mic C [m]	Mic D [m]
-0.1150	-0.0140	0	0.0350	0.0750	0.1150

Table 6.1: Kinect sensors - Microphones (A-D) and RGB-camera - positions over x axis.

For the reason that the four microphones of the Kinect are all positioned on its K^x_{RGB} axis (see Figure 6.1 and Table 6.1), a DOA estimator that analyses the four audio signals can mathematically give only the rotation angle (azimuth) of the plane with the normal perpendicular to the K^y_{RGB} axis (i.e. the normal can be thought as belonging to the plane created by K^x_{RGB} and K^z_{RGB} axes), taking the K^z_{RGB} axis as zero axis and K_C as zero point for the angle estimation. For example, if the sound source is in front of the device and its position belongs to the plane created with the point K_C and the two vectors K^y_{RGB} and K^z_{RGB}, then the estimated azimuth should be $\phi = 0$ degrees. Note that in this way it is impossible to understand if the sound comes from the front or behind the device (Figure 6.2), like in the well-known Cone of Confusion problem.

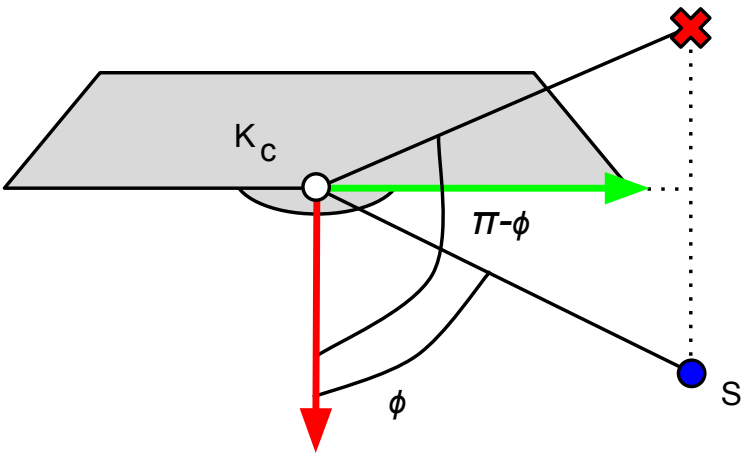


Figure 6.2: Example of limitation of the Kinect DOA sensor. S is the right available source position, the red crossed position is the wrong available position.

In this Figure we can see that the Kinect DOA sensor cannot decide which of the two available solutions is the right one. For simplicity, only the front positions are considered as possible solution, limiting the DOA sensor range to $-\frac{\pi}{2} \leq \phi \leq \frac{\pi}{2}$.

6.2 Sensor Accuracy Estimation Test

6.2.1 Introduction

Evaluating the accuracy and the precision of a real sensor can be sometimes uneasy especially for audio DOA sensors. The main problem is to find a reliable ground truth in the audio domain. This issue is often simply overcome by measuring the position of the acoustic source in the environment relatively to the sensor by hand measurements (i.e. with a meter and a goniometer). This approach is used for example by Pavlidi et al. in [21], where the sound source angle of both active male speaker and static speakers was measured in a previous step. A different approach can use the video signal and associate it to the audio signals to infer more information. Hershey and Movellan in [22] gave an application of this topic associating facial moves to the sound in order to localize the sound source; Siracusa et al. [23] proposed a method for speaker localization and head focus using facial tracking with a Kinect-like sensor. Our approach uses a visual checkerboard that allows to give with high precision the 3D pose and position of a RGB-camera. The choice of the use of this type of ground truth was thought to be used in robotics applications that use video features for localizing robots with Simultaneous Localization and Mapping (SLAM) algorithms.

This section will explain all the details of the experiment done for estimating the Kinect DOA sensor accuracy and precision. For the DOA estimation I used the code provided by HARK [19], that implements the adaptive beamforming algorithm called Multiple Signal Classification (MUSIC) [24] and can work with Kinect under the Robotic Operating System (ROS) [25]. In order to calculate the error of the DOA estimation given by HARK I set up the following experiment that used the RGB image calibration as ground truth: in a white square in the center of a $4 * 3$ checkerboard CB (with $CB_{sq\_sz} = 0.675$ [m]) printed over a cardboard, a circular frame was removed in order to create a hole where it was possible to insert an acoustic speaker (Figure 6.3). Using the OpenCV checkerboard finder [26], I determined the position and pose of the RGB sensor K_{RGB} of the Kinect with respect to the reference corner of the checkerboard CB_{RGB} (Figure 6.1).

During tests, an audio signal with pre-registered stereo speech noises was played by an acoustic speaker positioned behind the checkerboard with the center of the speaker in CB_{SPK}. The 3D point CB_{SPK} is calculated from CB_{RGB} taking in consideration that belongs to the checkerboard plane (with $CB_{sq\_sz} = 0.675$ [m]) and is in the center of one of its white squares shown in Figure 6.3. For avoiding multiple audio sources I did the experiment totally in silence and played it only over the channel of the stereo speaker that

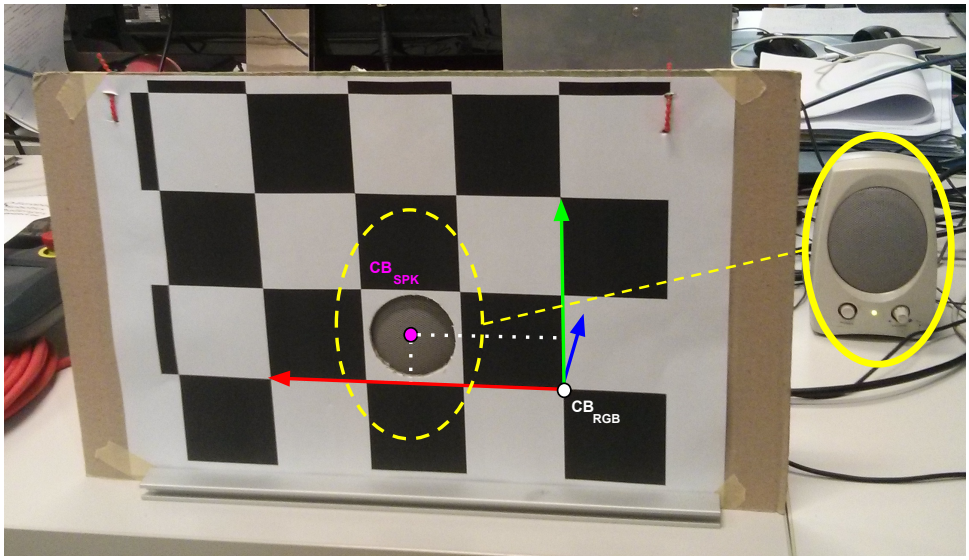


Figure 6.3: CB is the checkerboard used for sensor accuracy estimation. CB_{RGB} is the reference frame of the checkerboard found by the OpenCV checkerboard finder. CB_{SPK} is the center of the speaker and belongs to the checkerboard plane.

was behind the checkerboard. During the experiment, the reference system composed by the checkerboard and the speaker was leaved always in the same position, with CB^y_{RGB} (the y axis of the checkerboard) perpendicular to the ground. Only the pose and position of the Kinect were changed. In order to keep the K^y_{RGB} axis always perpendicular to the ground, I fixed the Kinect on a wheeled cart and moved only the cart. For each noise detection, HARK estimates the direction of arrival (DOA) of the sound with a granularity of 5 degrees.

At each new sound source detection, each DOA estimation ϕ_i and the corresponding position and pose estimation (K^R_{RGB}, K^T_{RGB}) of the Kinect with respect to the checkerboard CB_{RGB} were associated. After collecting data moving the device in different pose and positions, it was calculated the azimuth error e_ϕ considering the OpenCV checkerboard finder as ground truth. This choice for the ground truth is supported by a previous research that proved that the error over K^R_{RGB} is less than 0.1 degrees and the error over K^T_{RGB} is about 2 cm and increases depending on the distance of the RGB sensor from the checkerboard [20]. e_ϕ was computed by calculating the angle between the norms of the following two planes (Figure 6.4):

1. **Estimated Plane** P_{est} - plane generated by:
   - K_C point;
   - K_C^y axis;
   - $\overline{K_C S}$[5] vector calculated rotating K_C^z over K_C^y by α_i.
2. **Ground Truth Plane** P_{gt} - plane generated by:
   - K_C point;
   - K_C^y axis;
   - $\overline{K_C CB_{SPK}}$ vector.

$$e_\phi = \arccos\left[\frac{P_{est}^N}{\|P_{est}^N\|} \bullet \frac{P_{gt}^N}{\|P_{gt}^N\|}\right] \tag{6.1}$$

where P_{est}^N and P_{gt}^N are the normals of P_{est} and P_{gt} respectively.

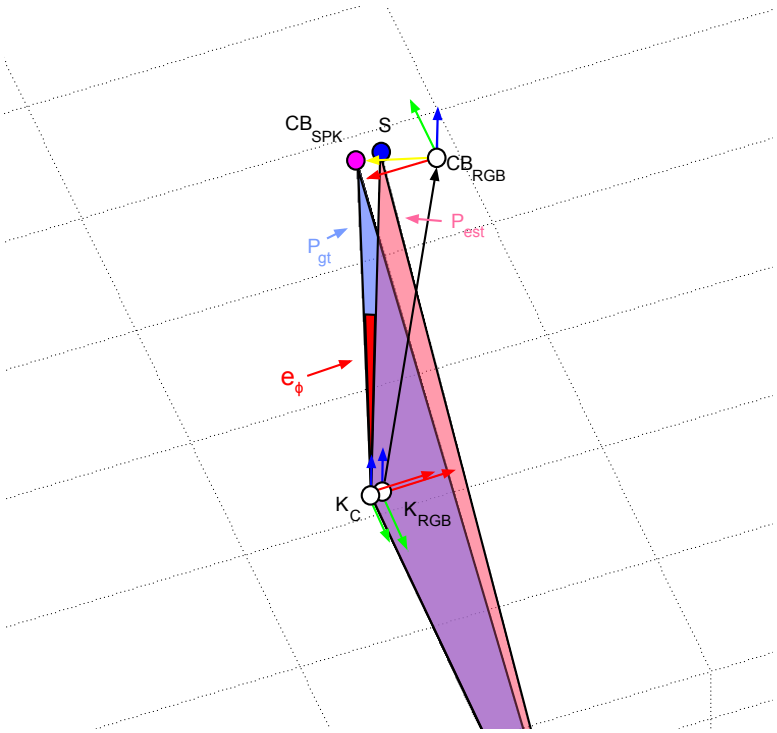
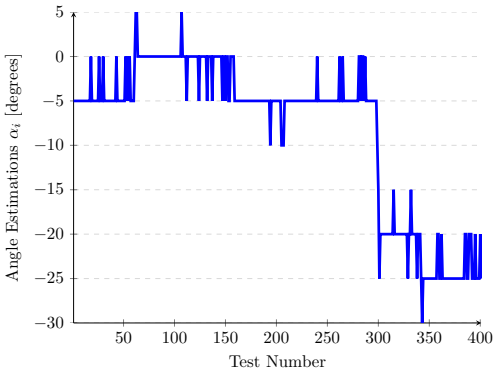


Figure 6.4: Angle error e_ϕ between the Estimated Plane P_{est} and the Ground Truth Plane P_{gt}.

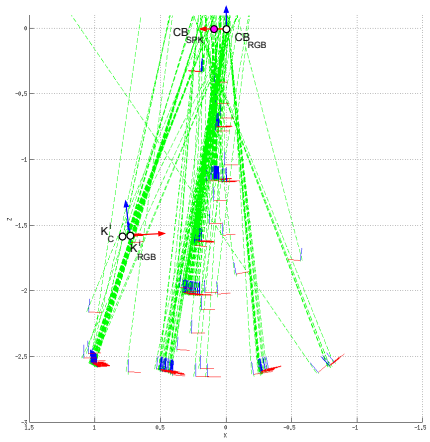
[5]Note that the position of S is unknown and cannot be estimated. S is an arbitrary point that belongs to the DOA and it is plotted only for understanding reasons.

6.2.2 Experimental Results

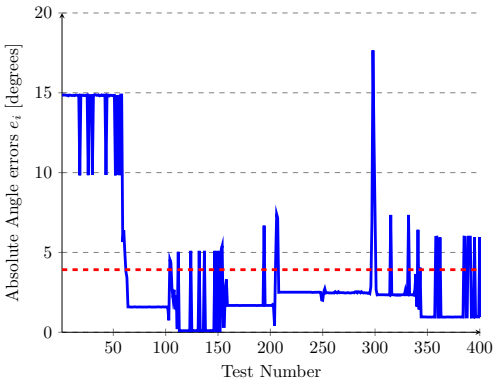
For the experiment I used a Microsoft Kinect for Xbox 360 [27] plugged into a laptop with i7-4700MQ CPU @ 2.40GHz 8 and 11,5 GiB RAM running under Linux (Ubuntu 12.04). Audio signals were processed by HARK [19] and video signals were analysed by an OpenCV application [26] used also in [20]. The audio-video synchronization and signal processing was done within ROS [25]. Test were done for $n = 400$ different positions of the Kinect with respect to the checkerboard. Results are shown in Figures 6.5(a-d) and 6.6.



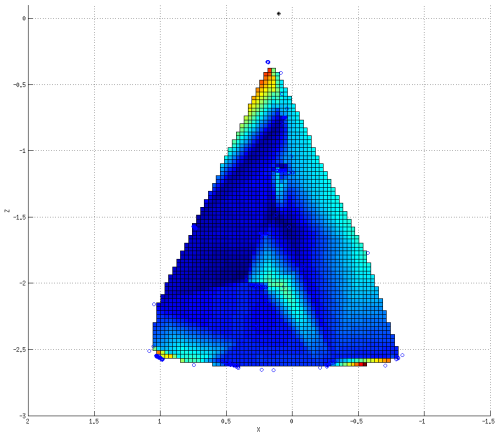
(a) Angle estimations α_i in degrees.



(b) 2D sensor and checkerboard positions.



(c) Absolute error for each ϕ_i. The red dotted line is the mean angle error $m_{e_\phi} = 3.9$ in degrees.



(d) Absolute error 2D distribution for each ϕ_i.

Figure 6.5: Test Results with different positions of the Kinect with respect to the checkerboard.

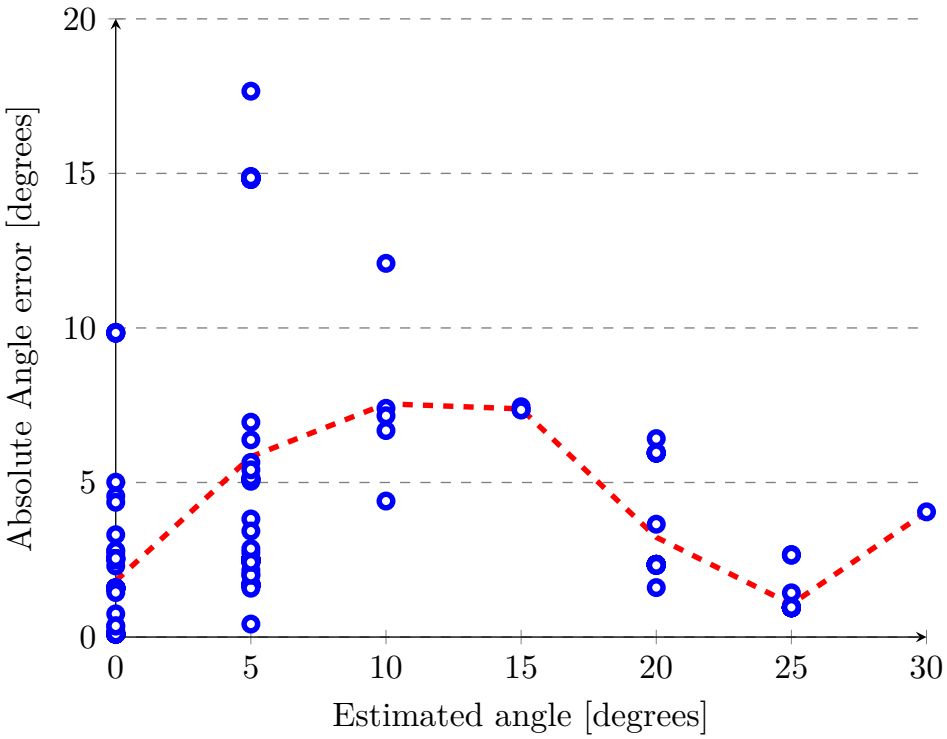


Figure 6.6: Absolute angle errors ordered by the angle estimation classes. The red dotted line links the mean errors for each estimated angle class.

In Figure 6.5(a) are plotted all the ϕ_i for $i \in (1, n)$. All the positions assumed by the Kinect with respect to the checkerboard are well visible in Figure 6.5(b). The absolute value of e_ϕ of each position is showed in 6.5(c) and the red dotted line is the mean angle error $m_{e_\phi} = 3.9$ in degrees. The variance of the error is $\sigma^2_{e_\phi} = 21.15$ in degrees. In Figure 6.5(d) is plotted the distribution of e_ϕ in the 2D space. Notice that, for granularity reasons, the mean error m_{e_ϕ} of about 5 degrees corresponds to an error of only one value of the available angles. Figure 6.5(d) shows that the error isn't affected by the distance of the sound source with respect to the sensor. Furthermore, Figure 6.6 shows that the error isn't affected by the angle of the sound source with respect to the sensor (it doesn't varies to much augmenting ϕ). Finally we saw that the accuracy of the DOA estimation is highly corrupted if the sensor is moving. These last two statements allows to associate this error to the Gaussian probability error in the angle domain with zero mean and variance of Equation 4.1.

6.3 Tests

6.3.1 Fixed 2D DOA sensors

The first real test consisted in setting three Kinects in a line with the positions and orientations shown in Table 6.2 and clapping hands in different positions. A total of twenty different positions with $r^x \in [-1, 1]$, $r^y \in [0.5, 2]$ and a step of 0.5 [m] are visible in Figure 6.9. Another simulation was done using the same parameters used in the real test in order to compare the results. For the GP-DOA simulation I used $prec = 0.01$ and $t_{sim} = 1000$ and the mean of the number of samples collected for each real test number is 35.

Sensor Type	k	s_k^x [m]	s_k^y [m]	s_k^o [rad]
Kinect	1	-1.05296	0.897504	0
Kinect	2	0	0	0
Kinect	3	0.0314011	0.0343866	0

Table 6.2: Sensors positions and orientations in real test.

Looking at real tests results and its related simulation results shown in Figures 6.7 and 6.8 it is possible to see that G-DOA performs always better than WLS-DOA both in mean and in variance and that real results does not differ so much to the ones of its relative simulation. This can be addressed to the fact that in this specific case there are only three sensors and in the simulation results of Figures 5.6 and 5.7 G-DOA performed always better than WLS-DOA in the case $N_s = 3$. It is evident that errors becomes bigger as the distance between the acoustic source and the sensors grows. This is visible looking at results of Test Numbers [1, 5, 11, 16, 20] in Figure 6.7, 6.8 and 6.9 that correspond to lateral positions in G. This phenomenon reflects the fact that the distance between the real DOA and a position $\mathbf{q}$ increases as $\mathbf{q}$ moves away from the vertex of a fixed angle. Qualitative results for the second real test showed that the algorithm performed well and the main problems arose when the SLAM algorithm didn't estimate well the robot pose.

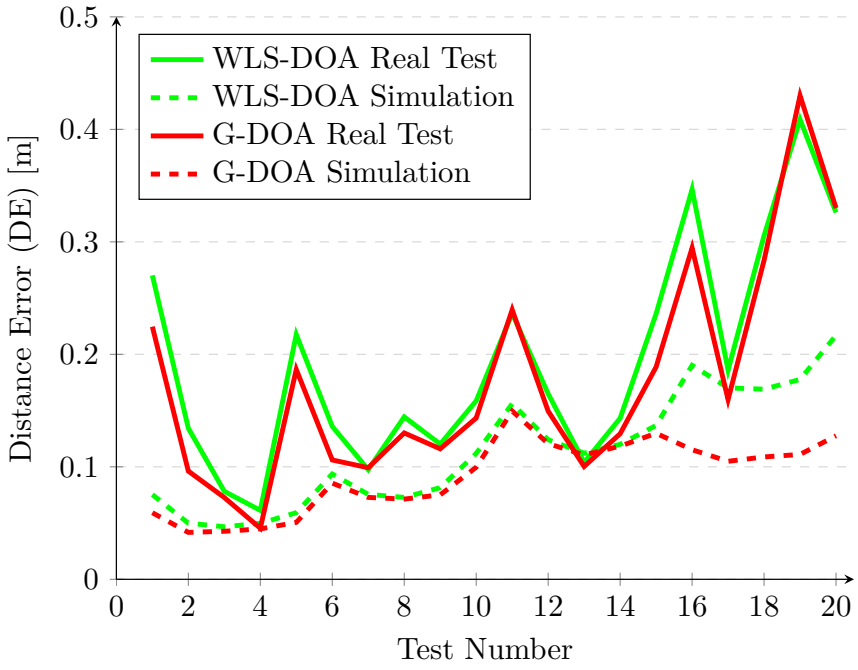


Figure 6.7: Mean values of the Real test results of G-DOA vs. WLS-DOA.

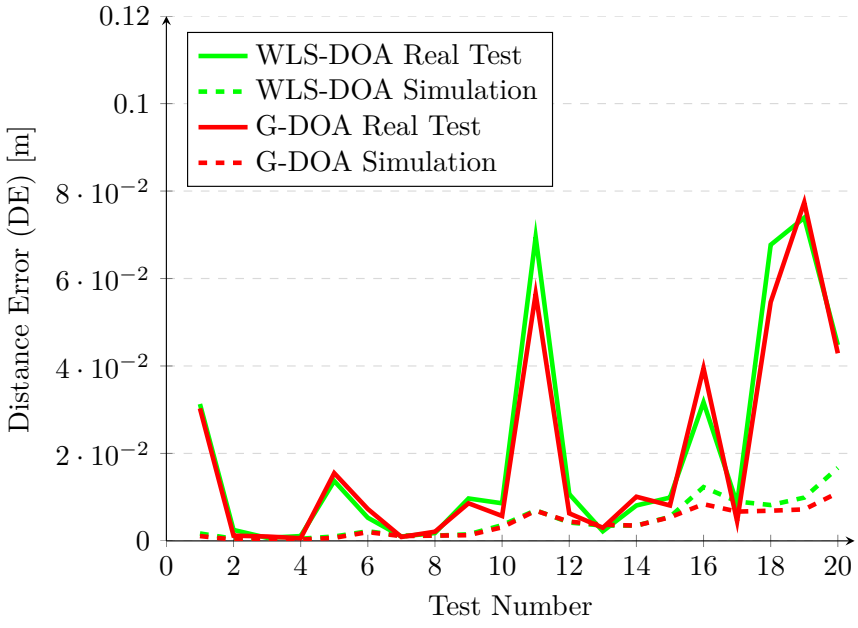


Figure 6.8: Variance values of the Real test results of G-DOA vs. WLS-DOA.

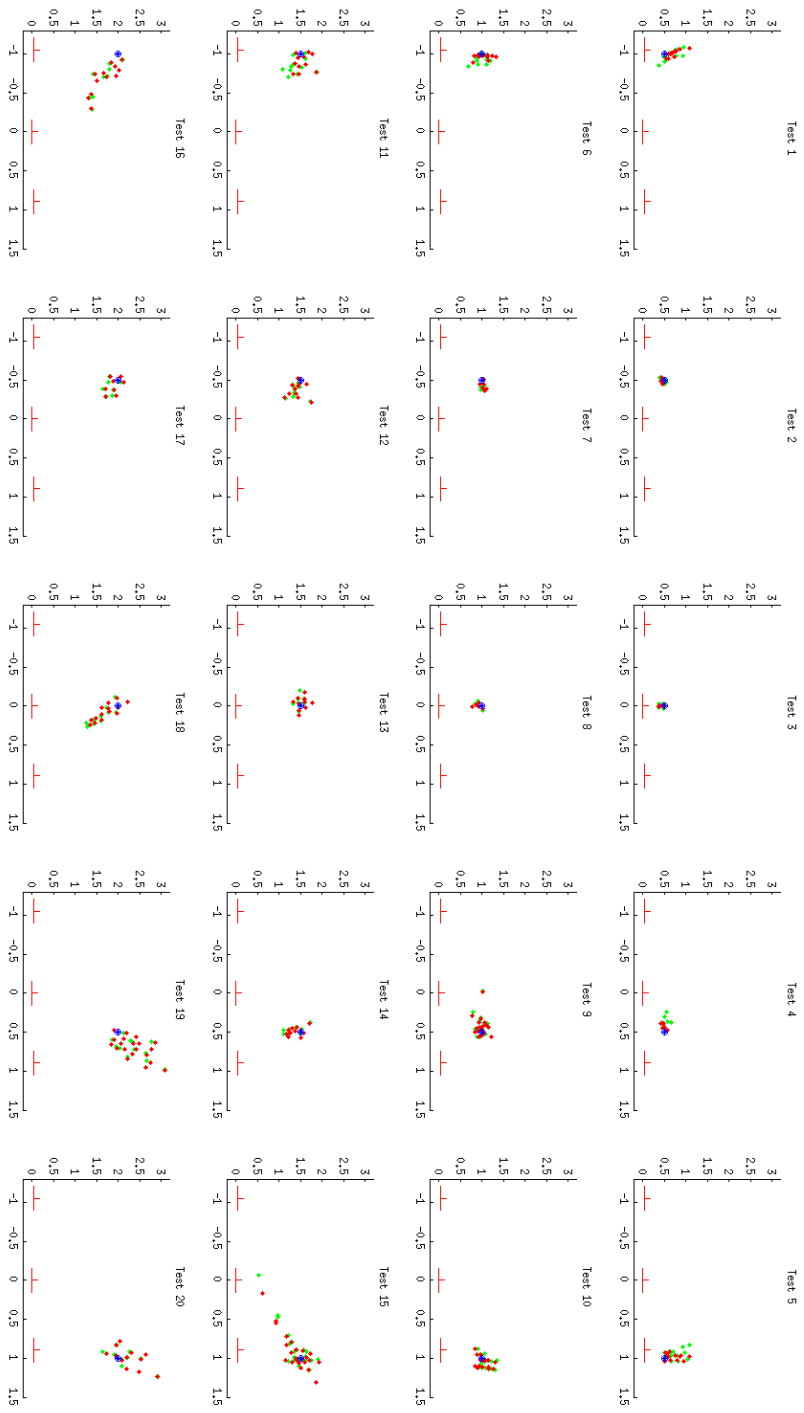
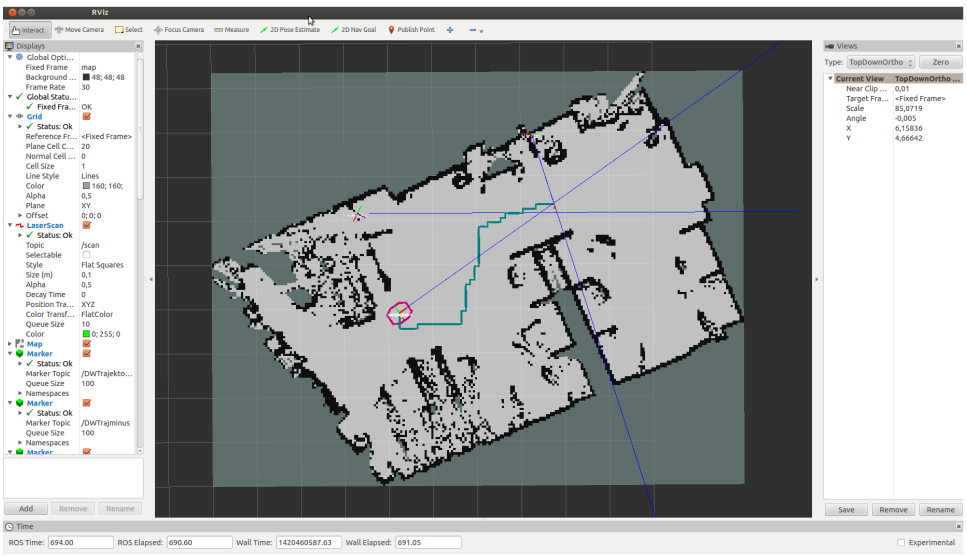


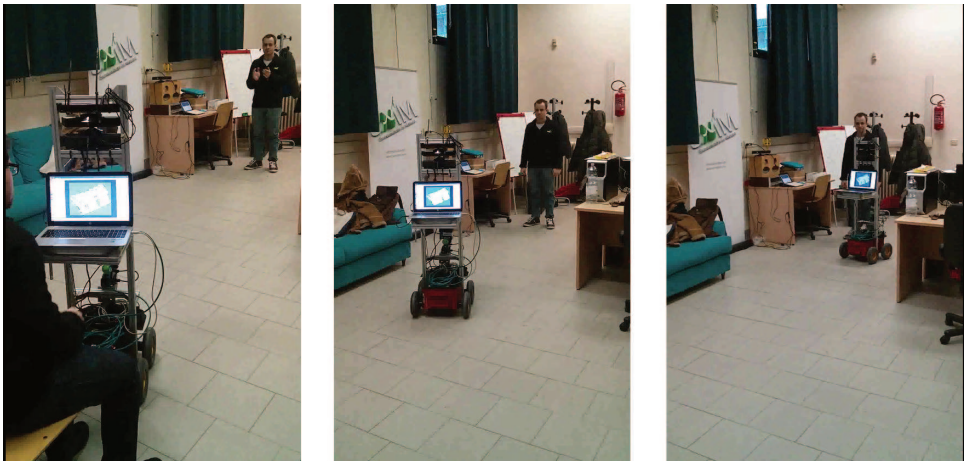
Figure 6.9: Real Tests results. WLD-DOA and G-DOA position estimations are in green and red respectively. Source and sensors positions are in blue and red respectively. The axis scale is measured in meters.

6.3.2 Mobile 2D DOA sensors

In the second test I used three Microsoft Kinect as DOA sensors, with one of them mounted over a Pioneer 3-AT with a laser scanner. The real test consisted in having two fixed kinects and moving the Pioneer in the environment towards the clapping hands position (Figure 6.10)[6].



(a) Audio Localization with map in Rviz



(b) Screenshots of the video

Figure 6.10: Real experiment with three Microsoft Kinect as DOA sensors, with one of them mounted over a Pioneer 3-AT with a laser scanner.

[6]A short video can be seen at: https://www.youtube.com/watch?v=TVAQ-sFSpF8

6.3.3 Fixed 3D DOA sensors

Real Testing of the 3D DOA-ASL has been done with two different experiments using two different kind of DOA sensors:

- Aldebaran NAO robot;
- Microsoft Kinect.

NAO

This experiment was done using two Aldebaran NAO robots and using the built-in *ALSoundLocalization* method. This method uses the microphones on the head of the robot to localize in the impulsive sounds. The position of the four microphones allows to calculate the both azimuth and elevation of the acoustic source heard (Figure 6.11).

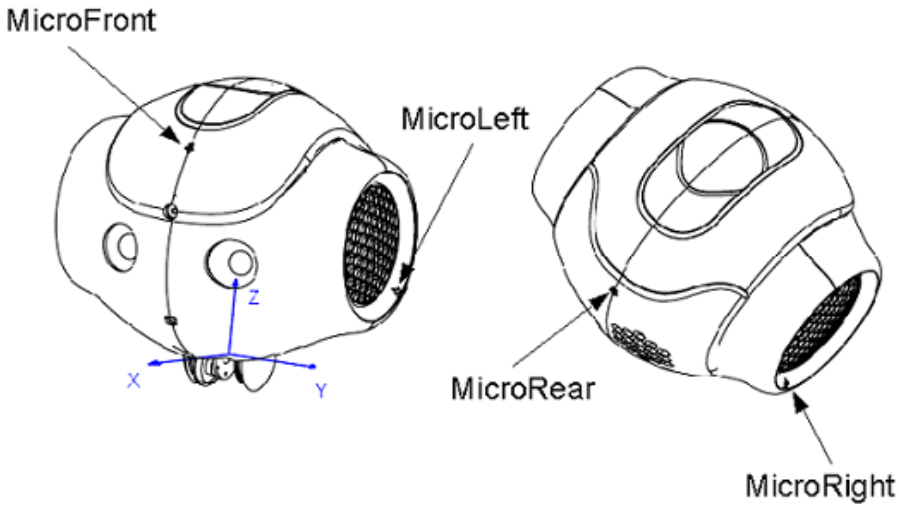


Figure 6.11: Position of the four microphones in a NAO v4 robot.

The native procedure is not very precise (error of about 15 degrees), especially with the elevation, because the position of the microphones does not permit to reach better results. In this experiment the robots were first calibrated using their cameras and a checkerboard and a human clapped the hands in different 3D positions near the robots.

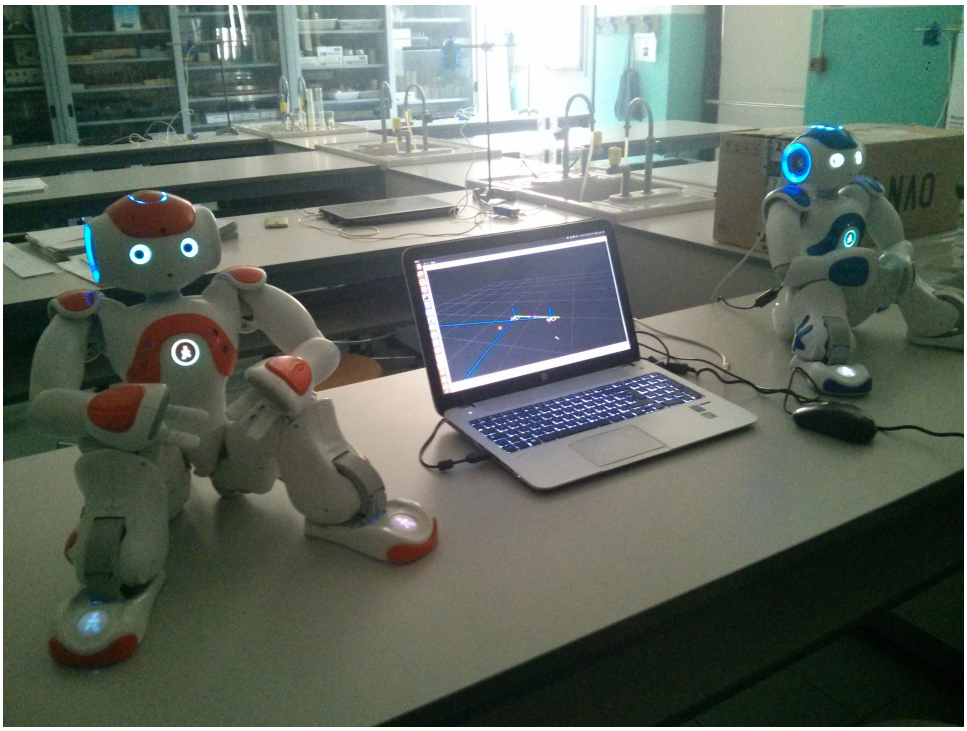


Figure 6.12: 3D DOA-ASL using two Aldebaran NAO robots. the point in red in the computer's screen shows that the source comes approximately near the camera that took the photo.

Unfortunately it has been impossible to collect data of the experiment because the blue NAO was lent for few hours, but the results visible on the screen show that the algorithm worked well also with the inaccurate DOA estimator of the NAO robots[7].

[7] A short video can be seen at: https://www.youtube.com/watch?v=vsTvc-AEdJU

Kinect 3D

In order to have a good and cheap 3D DOA sensor I used a combination of two Microsoft Kinects: the old model under the Kinect one model flipped of 90 degrees clockwise (Figure 6.13). The old model was used for calculating the azimuth and the new model was used for the elevation. In order to have right results with the combination of the two values, the initial point of the DOA was considered to be in the center of the new model K_C.

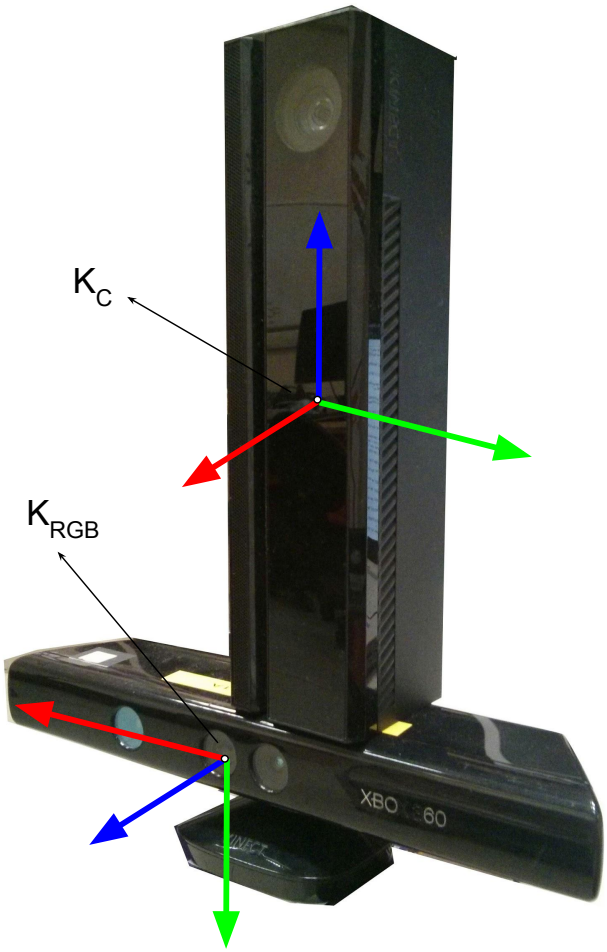


Figure 6.13: 3D Kinect DOA sensor using a Microsoft Kinect (old model) and a Kinect One (new model).

The experimental set up was based on two 3D Kinect DOA sensors (4 Kinects) positioned as show in Figure 6.14. After the video calibration of the four Kinects, the experiment consisted in setting a loud speaker playing a continuous speech noise in 27 different 3D positions in the nearby environment (see Figure 6.15). In Table 6.3 there is the number of data collected with a mean of about 43 DOA estimation for each source position.

Figure 6.14: 3D DOA-ASL using four Microsoft Kinect setup.

Source position number	1	2	3	4	5	6	7	8	9
Number of estimations	47	50	41	46	39	40	47	43	50
Source position number	10	11	12	13	14	15	16	17	18
Number of estimations	59	46	43	50	50	39	39	45	40
Source position number	19	20	21	22	23	24	25	26	27
Number of estimations	44	40	37	44	41	42	55	41	40

Table 6.3: Number of source estimations for each source point.

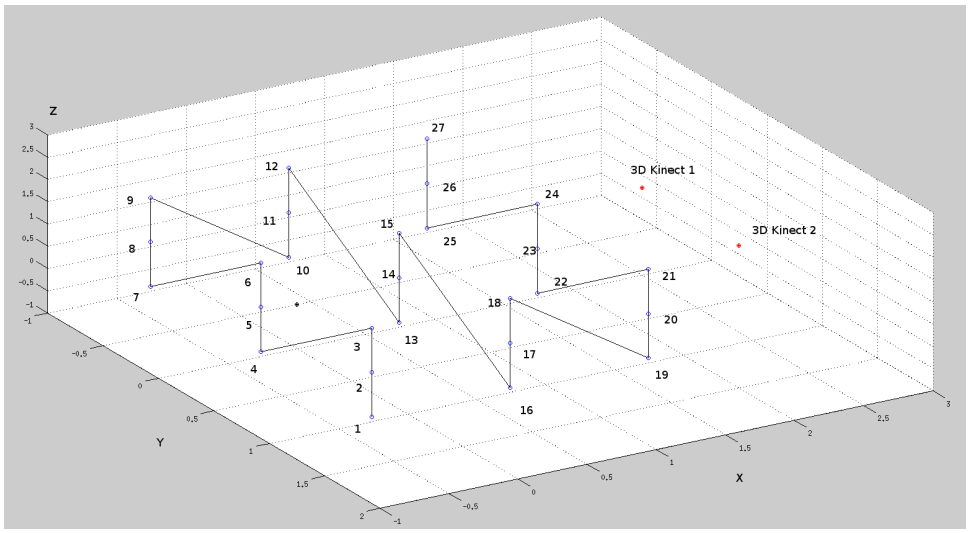


Figure 6.15: 3D DOA-ASL experiment source positions.

Results are shown in Figure 6.16. As it can be noticed, the Mean Distance Error is high with respect to the first real experiment. This is due mainly for two reasons: first of all the distance of the sound source from the sensors was augmented (up to more than three meters); second, the test was done using only two 3D DOA sensors instead of three or more. As we saw from the simulation results, more is the number of DOA sensors and better is the source estimation. The sources that are positioned in the middle between the two DOA sensors have better estimations (e.g. positions 4, 5, 13, 14, 23).

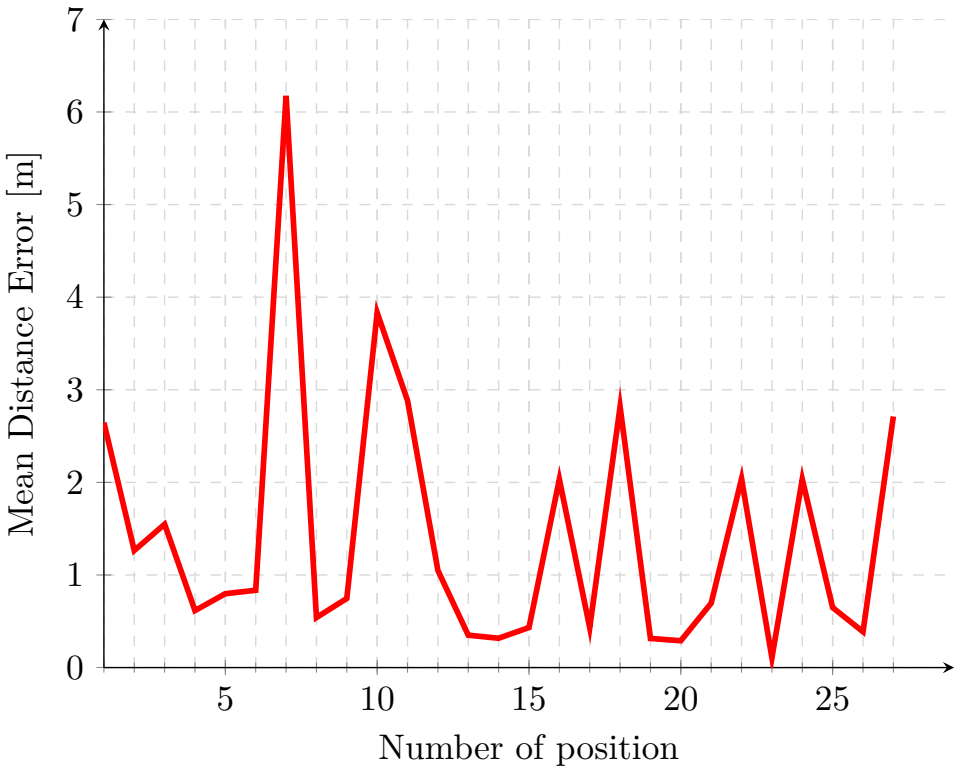


Figure 6.16: Results of the 3D DOA-ASL experiment.

Chapter 7

Conclusions

This thesis dealt with the problem of finding the source position of acoustic sounds using the DOAs calculated by the DOA sensors mounted on robots. In the field of robotics this approach was never been used and it bring to the state of the art a new way of using the audio sensors mounted on the robots. So, in order to avoid bottleneck problems sharing audio-video data among robots, I proposed a method that shares only the pose of the robot and the DOAs of the acoustic sources heard. A new grid-based probabilistic algorithm is also proposed that, instead of the GB-DOA approach, can apply different weights to the DOA sensors precision. The algorithm has also been fastened heavily after a deep study of the problem assisted by the simulation tests. The approach has been extended for 3D audio localisation and for multiple audio sources. Simulation tests prove that this new method gives better results than all the other state of the art DOA-approaches. In conjunction with the spreading use of the Robot Operating System (ROS), I proposed a new way of thinking the audio for robotics: use video sensors for robot audio localization. Finally, real tests with Microsoft Kinect devices as DOA sensors mounted on a mobile robot and with two Aldebaran's NAO robots show that the algorithm can be used in real time with a ROS package with encouraging results.

Appendix A

ROS Package documentation

A.1 Why ROS?

The choice of using the Robot Operating System [25] for developing all the features explained in the thesis is due to the following factors:

- simplification of sensors signal connections in a network;
- easy data sharing;
- robust data synchronization;
- abstraction of the concept of device and no further driver implementation;
- modular approach using logical nodes;
- orientation towards the large community development and reuse.

Undoubtedly the ROS framework is a very useful tool because it avoids wasting time with other issues that does not properly concern the field of robotics (e.g. network issues, driver compatibilities, etc.).

A.2 Package Download

I implemented a C++ ROS package that can be downloaded here:

`https://github.com/iaslab-unipd/DOA_acoustic_source_localization.git`

The package is written in order to be extended with other features like 3D Audio Localization and Multiple Sources Localization. The basic version allows to estimate only one acoustic source at time. A map of the environment can be linked if the user has a fixed reference.

A.3 Installation and Required Packages

The current version of the ROS package works only under **Ubuntu 12.04 (Precise)** with **ROS Hydro Medusa** (Figure A.1).

Figure A.1: ROS Hydro Medusa.

The reason is due to the HARK-Kinect ROS package I used for the implementation that is compatible only with the above configuration. After the installation of the ROS environment, the HARK-ROS package should be installed first, following the instruction from the HARK site[1] and being careful to select the Ubuntu 12.04 (Precise) section. Then the HARK-KINECT package should be installed from the instruction of the HARK site[2] and tested with a real Microsoft Kinect plugged in. The user must verify that the packages listed in Table A.1 are installed with the correct version. Other versions of the following packages are not recommended. The packages signed with an asterisk (*) must be installed with a version that is different from the current one and can be downloaded from the repository of our package or from the HARK repository[3].

[1] http://www.hark.jp/wiki.cgi?page=HARK+Installation+Instructions

[2] http://www.hark.jp/wiki.cgi?page=HARK-KINECT+Installation+Instructions+%28as+a+USB+recording+device%29

[3] http://www.hark.jp/harkrepos/dists/precise/non-free

Package name	Version
harkfd*	2.0.0.6582
harkfd-dev*	2.0.0.6582
harktool4*	2.0.0.5427
flowdesigner-0.9.1-hark	2.1.0
hark-designer	2.1.0
hark-kinect	1.2.0.4440
hark-music	2.1.0
hark-ros-hydro	2.0.0.6430
harktool4-cui	2.1.0
julius-4.2.3-hark	2.1.0
julius-4.2.3-hark-plugin	2.1.0
libhark-netapi	2.1.0
libharkio-dev	2.1.0
libharkio1	2.1.0
libharkio2	2.1.0
libharkio2-dev	2.1.0

Table A.1: HARK Packages.

For installing the *DOA_acoustic_source_localization* package do the following:

1. copy *DOA_acoustic_source_localization* package in your */rosbuild* directory;
2. *cd ./your_rosbuild_path/rosbuild*
3. *rosws set DOA_acoustic_source_localization* and then answer "yes"
4. $\sim$ */.bashrc*
5. *roscd DOA_acoustic_source_localization*
6. *make*

If there are no errors, the package is correctly installed.

A.4 Input Parameters

In this part are detailed all available input parameters:

- **is_simulation**: type *boolean*. If set to "*true*" it disables the subscription to the topic containing the DOA estimations published by the HARK-Kinect ROS nodes. In this mode the DOA estimation data can only be set in a file.
- **data_simulation_file_path**: type *string*. Absolute path of the file containing the angles (in degrees) of the DOA estimations of the sensors. Each row contains an angle and the value in the first row is assigned to the first sensor, the second row is assigned to the second and so on. Example of file "*/home/username/workspace/ros/angles_sim.txt*" containing three angles:

  ```
  90
  -45
  120
  ```

- **n_sound_DOA_sensors**: type *int*. Number of DOA sensors in the network.
- **audio_signal_power_threshold**: type *double*. Threshold that helps selecting the acoustic sources with a minimum signal power. Common values go from 25 to 40. A good choice is, for example, 35.
- **algorithm_type**: type *int*. There are three available algorithms:

  1. GP-DOA fast
  2. GP-DOA slow
  3. WLS-DOA

  WLS-DOA solution is always calculated (one shot formula) and it is used as starting point for the GP-DOA research and refinement. If there are only two DOAs in the 2D space - trivial problem - all the three algorithms give reasonably the same solution, so it is calculated only the WLS-DOA one because it is the fastest.

- **precision_grid**: type *double*. Precision in meters of the grid. Common values are 0.01 or 0.001 for GP-DOA Basic and 0.001 or 0.0001. Note that the precision value of the grid can be very small if used with the GP-DOA fast algorithm because of its high reduction of the computational time with respect to the slower one (see Section 4.2).

- **range**: type *double*. Exploration range in meters. Note that the center point of the grid G is the WLS-DOA solution.
- **tf_world_name**: type *string*. Name of the reference frame to which all the poses are referred. Example: */world*
- **tf_solution_name**: type *string*. Name of the reference frame of the published solution. Example: */sound_source_estimated_position*
- **rviz_DOA_line_lenght**: type *double*. Length in meters of the lines that start from the DOA sensor and represent the DOA.
- **sensor_3D_pose_topic_** $<$ **i** $>$: type *string*. Name of the topic of the i-th-DOA sensor pose.
- **DOA_topic_** $<$ **i** $>$: type *string*. Name of the topic of DOA estimation of the i-th-DOA sensor.

A.5 Simulation Tests

For testing the package in simulation mode the user has first to set the pose (position and orientation) of the DOA sensors. It can be done by modifying the launch file that publishes the tranformations (*tf*). Here is an example:

```
<node pkg="tf" type="static_transform_publisher" name="DOA_sensor_1"
args="1 2 0 0 0 0 1 /world /DOA_sensor_1 100" />
```

where the DOA_sensor_1 is in the position $s_1^{x,y,z} = [1, 2, 0]$ and the orientation is written in quaternions $s_1^o = [0001]$. Note that the program takes into consideration the DOA in the xy-plane and a DOA with 0 degrees is considered to be on the x axis (see Figure 6.1). During the simulation tests the user can vary the values of the data contained in the $data\_simulation\_file$; after saving the file, the package will update the values and will show the new solution.

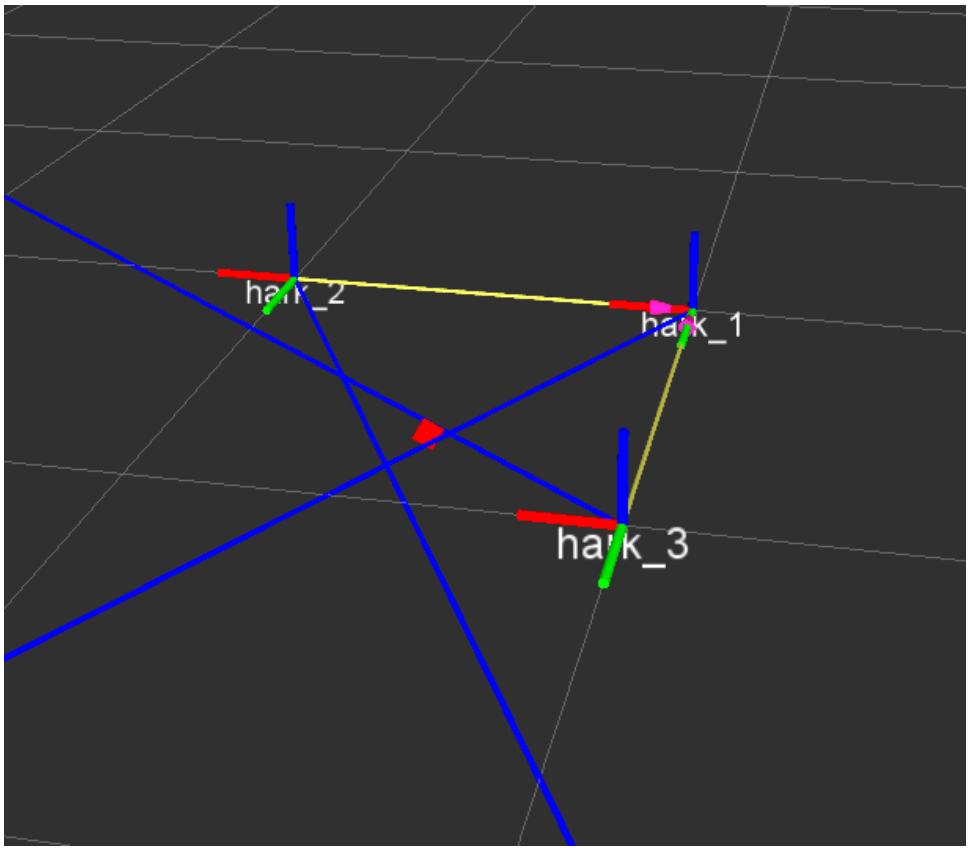


Figure A.2: Simulation of 2D DOA-ASL in ROS using three DOA sensors in *rviz* (ROS). The red point is the source position estimation.

A.6 Real Tests

Real tests need the calibration of the DOA sensors. Doing it manually can be a very long process and usually is error prone. Finding the extrinsic calibration parameters between DOA sensor is not so easy because the user has to measure the position of all the microphones and then calculate the position of the DOA starting point. As we deal with mobile robots, this procedure cannot be done considering only the microphones because the robot changes its pose and consequently the pose of its microphones. For this reason ,it can be helpful to use the vision sensors that normally do the Simultaneous Localization And Mapping (SLAM) that update the pose of the robot and provide a link between video and audio sensors. In order to achieve this aim I used another ROS package that allows to calibrate multiple vision sensors, a project that I started when I was in Paris and then developed in collaboration with my laboratory [4]. I used this package with the RGB images coming from the Microsoft Kinect.

A.6.1 Microsoft Kinect configuration

Using HARK with Microsoft Kinect is very simple but there are some details to take into consideration. First of all, once the RGB calibration is done, we need to consider the following transform that refers to Figure 6.1 and Table 6.1:

```
<node pkg="tf" type="static_transform_publisher" name="hark_to_rgb"
args="-0.0140 0 0 -0.5 0.5 -0.5 -0.5 /kinect_rgb /hark 100" />
```

Second we need to link each kinect RGB sensor to the right DOA HARK estimation. This procedure is very important and must be done paying attention. HARK ROS files contain all the information that HARK needs to localize the sounds. These data are collected in files with extension ".n". In our package I provided also some example files that can be modified for the user needs. The parameters that the user has to modify are substantially three. One is the *plughw* value that refers to the audio hardware to which the Kinect is associated. This value can be found in the computer by typing

```
cat /proc/asound/cards
```

and looking to the number of the device assigned to the kinect. Here is an example of output:

[4] https://github.com/iaslab-unipd/multisensor_calibration

```
0 [MID  ]: HDA-Intel - HDA Intel MID
   HDA Intel MID at 0xd3710000 irq 47
1 [PCH  ]: HDA-Intel - HDA Intel PCH
   HDA Intel PCH at 0xd3714000 irq 48
2 [Audio]: USB-Audio - Kinect USB Audio
   Microsoft Kinect USB Audio at usb-0000:00:14.0-2.1, high speed
```

so in the file ".n" you have to put *plughw* : 2. Note that if you plug more than one Kinect to the same computer, the *plughw* will increase and the unique way to associate the right HARK signal to the right Kinect is to plug one Kinect at time, being careful to note the *plughw* value in order to set the right topic in the launch files. The second parameter that the user needs to modify is the $TOPIC\_NAME\_HARKSOURCE$ inserting the name of the topic of the DOA. Finally the user have to modify the A_MATRIX parameter of the node $node\_LocalizeMUSIC\_1$ and put the absolute path of the file *kinect_loc.dat* that contains the information for localizing sounds in the kinect.

Summarizing, the steps for a real test are the following:

1. plug the usb cable of the Kinects to the computer one at time and label them with a name or a number that links to the *plughw* number;
2. calibrate the Microsoft Kinects and find the extrinsic parameters;
3. launch *roscore*;
4. launch the ROS transform publishers and visualize them in rviz;
5. modify the .n files and the launch file of the ROS *DOA_acoustic_source_localization* package being careful to assign the correct topics to the correct values;
6. launch the .n files that start to output the DOAs from the Kinects;
7. launch the ROS *DOA_acoustic_source_localization* package.

Bibliography

[1] P. Pertilä. *Acoustic Source Localization in a Room Environment and at Moderate Distances - [PhD Thesis]*. Tampere University of Technology, 2009.

[2] D. R. Griffin. *Listening in the dark: the acoustic orientation of bats and men*. Yale University Press, 1958.

[3] D. Salvati. *Acoustic Source Localization Using Microphone Arrays - [PhD Thesis]*. Department of Mathematics and Computer Science - University of Udine, 2012.

[4] Darren B. Ward, E.A. Lehmann, and R.C. Williamson. Particle filtering algorithms for tracking an acoustic source in a reverberant environment. *Speech and Audio Processing, IEEE Transactions on*, 11(6):826–836, Nov 2003.

[5] M. Omologo and R. De Mori. *Spoken Dialogue with Computers*, volume chap. Acoustic Transduction. Academic Press, 1998.

[6] G. Valenzise, L. Gerosa, M. Tagliasacchi, E. Antonacci, and A. Sarti. Scream and gunshot detection and localization for audio-surveillance systems. In *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on*, pages 21–26, Sept 2007.

[7] Parham Aarabi. The fusion of distributed microphone arrays for sound localization. *EURASIP Journal on Advances in Signal Processing*, 2003(4):860465, 2003.

[8] J. H. Di Biase, H. F. Silverman, and M. Brandstein. *Microphone Arrays: Signal Processing Techniques and Applications*, volume chap. Robust localization in reverberant rooms. Springer, 2001.

[9] M. Hawkes and Arye Nehorai. Wideband source localization using a distributed acoustic vector-sensor array. *Signal Processing, IEEE Transactions on*, 51(6):1479–1491, June 2003.

[10] Riccardo Levorato and Enrico Pagello. Probabilistic 2d acoustic source localization using direction of arrivals in robot sensor networks. In Davide Brugali, JanF. Broenink, Torsten Kroeger, and BruceA. MacDonald, editors, *Simulation, Modeling, and Programming for Autonomous Robots*, volume 8810 of *Lecture Notes in Computer Science*, pages 474–485. Springer International Publishing, 2014.

[11] A. Griffin and A. Mouchtaris. Localizing multiple audio sources from doa estimates in a wireless acoustic sensor network. In *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*, pages 1–4, Oct 2013.

[12] A. Ali S. Asgari R.E. Hudson K. Yao D. Estrin H. Wang, C. E. Chen and C. Taylor. Acoustic sensor networks for woodpecker localization.

[13] A. Ledeczi, G. Kiss, B. Feher, P. Volgyesi, and G. Balogh. Acoustic source localization fusing sparse direction of arrival estimates. In *Intelligent Solutions in Embedded Systems, 2006 International Workshop on*, pages 1–13, June 2006.

[14] P. Pertilä and M. Parviainen. Robust speaker localization in meeting room domain. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, pages IV–497–IV–500, April 2007.

[15] A. Griffin, A. Alexandridis, D. Pavlidi, and A. Mouchtaris. Real-time localization of multiple audio sources in a wireless acoustic sensor network. In *Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European*, pages 306–310, Sept 2014.

[16] R. Levorato and E. Pagello. Doa acoustic source localization in mobile robot sensor networks. In *Autonomous Robot Systems and Competitions (ICARSC), 2015 IEEE International Conference on*, pages 71–76, April 2015.

[17] K. B. Petersen and M. S. Pedersen. The matrix cookbook, nov 2012. Version 20121115.

[18] MathWorks®. Matlab. http://www.mathworks.com/products/matlab/.

[19] K. Nakadai, T. Takahashi, H. G. Okuno, H. Nakajima, Y. Hasegawa, and H. Tsujino. Design and implementation of robot audition system

"hark"; open source software for listening to three simultaneous speakers. *Advanced Robotics*, 24(5-6):739–761, 2010.

[20] F. Basso, R. Levorato, and E. Menegatti. Online calibration for networks of cameras and depth sensors. *OMNIVIS: The 12th Workshop on Non-classical Cameras, Camera Networks and Omnidirectional Vision - 2014 IEEE International Conference on Robotics and Automation (ICRA 2014)*, 2014.

[21] D. Pavlidi, A. Griffin, M. Puigt, and A. Mouchtaris. Real-time multiple sound source localization and counting using a circular microphone array. *Audio, Speech, and Language Processing, IEEE Transactions on*, pages 77–80, Oct 2013.

[22] John Hershey and Javier Movellan. Audio-vision: Using audio-visual synchrony to locate sounds. In *Advances in Neural Information Processing Systems 12*, pages 813–819. MIT Press, 2000.

[23] Michael Siracusa, Louis-Philippe Morency, Kevin Wilson, John Fisher, and Trevor Darrell. A multi-modal approach for determining speaker location and focus. In *Proceedings of the 5th International Conference on Multimodal Interfaces*, ICMI '03, pages 77–80, New York, NY, USA, 2003. ACM.

[24] C.W. Therrien. *Discrete Random Signals and Statistical Signal Processing*. Number v. 1 in Prentice-Hall signal processing series. Prentice Hall, 1992.

[25] Robot operating system (ros). http://www.ros.org/.

[26] G. Bradski. The opencv library. *Dr. Dobb's JOURNAL of Software Tools*, 2000.

[27] ©Microsoft. Kinect. http://www.xbox.com/kinect.

Acknowledgements

First of all I want to acknowledge my girlfriend Alice for staying next to me and for supporting me morally during these years of my PhD. I also obviously acknowledge my professors Enrico Pagello and Emanuele Menegatti for giving me the chance to achieve this prestigious degree. A particular thank goes to all the people that worked with me in the IAS-LAb: Stefano Ghidoni, Stefano Michieletto, Matteo Munaro, Luca Tonin, Roberto Bortoletto, Elisa Tosello, Nicolò Boscolo, Morris Antonello, Marco Carraro and my two classmates Mauro Antonello and Filippo Basso, with whom I studied together for the exams of the PhD school and wrote some articles. Special thanks go to Salvatore Maria Anzalone and Stéphane Michelet that worked with me and hosted me at the Institut des Systèmes Intelligents et de Robotique (ISIR) of the Université Pierre et Marie Curie (UPMC) in Paris and with whom I had a fantastic stay in Paris. Finally, I acknowledge all the people that helped me in the experiments, in reviewing my articles and in finding the research topic of my PhD thesis.