POLITECNICO
MILANO 1863

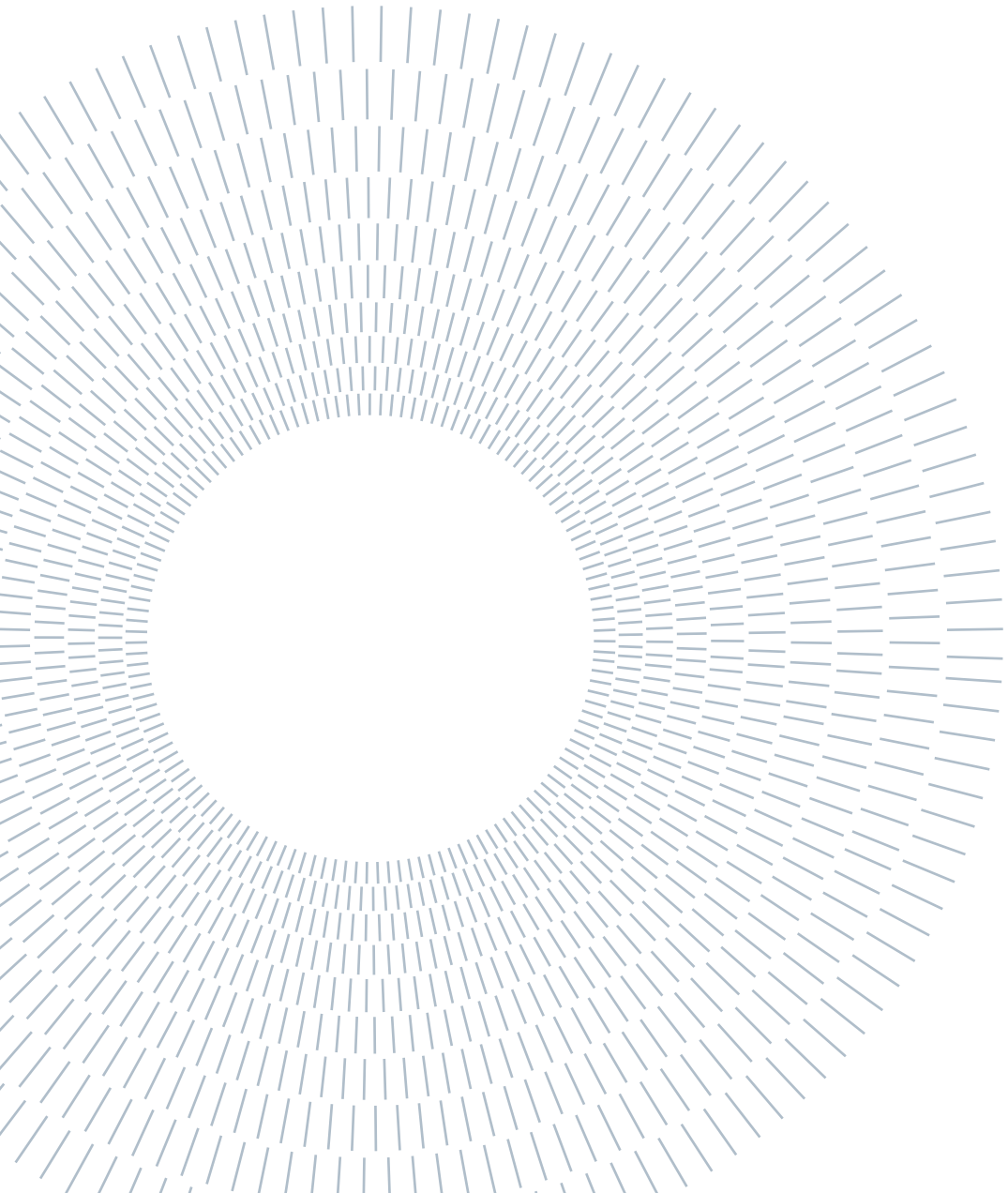
SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

RO-BAT: A bat-inspired approach on mobile robot navigation using Direction of Arrival estimation

Tesi di Laurea Magistrale in
Music and Acoustic Engineering - Ingegneria Acustica e Musicale

Author: **Alberto Doimo**

Student ID: 103376
Advisor: Prof. Fabio Antonacci
Co-advisors: Dr. Thejasvi Beleyur, Prof.Dr.-Ing. Heiko Hamann,
Dr. Andreagiovanni Reina
Academic Year: 2023-24



Abstract

This thesis presents the development of a passive echolocation solution for autonomous swarm robotics, as the initial contribution to the RO-BAT project, whose aim is to study how animals or robots can use echolocation when the environment is populated by several individuals emitting numerous, overlapping signals, the so-called "Cocktail Party Problem". On the one side, biologists are still exploring how bats echolocate effectively within dense swarms, distinguishing between environmental and conspecific cues. On the other side, an effective echolocation system could help robotics to improve active sensing, as current technologies like RADAR and infrared (IR) badly scale in dense environments.
The first phase of the RO-BAT project corresponds to the study conducted in this thesis. The focus is on the passive aspect of sound-based localisation, which provides a foundation for future developments in active echolocation. Passive echolocation enables a robot to locate external sound sources without emitting its own signals. For this thesis, I designed, assembled, and tested a small bio-inspired robotic platform called "ro-bat," optimized for real-time Direction of Arrival (DOA) estimation to perform obstacle avoidance in multi-agent environments. The ro-bat features Micro-Electro-Mechanical Systems (MEMS) microphones arranged in a custom microphone array, paired with the MCHStreamer sound card and a Raspberry Pi single-board computer for onboard signal processing. I implemented and evaluated three DOA algorithms — Generalised Cross-Correlation with Phase Transform (GCC-PHAT), Steered Response Power with Phase Transform (SRP-PHAT), and Multiple Signal Classification (MUSIC) — and integrated them with the ro-bat's navigation strategy to balance accuracy, computational efficiency, and responsiveness on a resource-limited platform.
Experiments were conducted in controlled laboratory conditions using the Thymio II robot as a mobile platform. Each algorithm was tested in various obstacle configurations to evaluate localisation accuracy and error distribution. Results showed successful obstacle avoidance with two algorithms (GCC-PHAT and SRP-PHAT), while MUSIC performed poorly on the tested hardware due to high computational demands. Overall, GCC-PHAT performed better than the other algorithms, being the fastest and most responsive one. However, this configuration, while reliable, had limited detection accuracy

at side angles due to construction and array choices.
My ro-bat has become a representative platform for the bio-inspired robotics research conducted at the Centre for the Advanced Study of Collective Behaviour (CASCB), leading me to deliver numerous demonstrations of the platform at various events, including my participation in ICRA@40.

Keywords: sound source localisation, direction of arrival, MEMS, swarm robotics, collective behaviour

Abstract in lingua italiana

Questa tesi presenta lo sviluppo di una soluzione di ecolocalizzazione passiva per la robotica autonoma in sciami, come contributo iniziale al progetto RO-BAT. Questo progetto ha l'obiettivo di studiare come gli animali o i robot possono usare l'ecolocalizzazione quando l'ambiente è popolato da numerosi segnali sovrapposti, un problem chiamato "Cocktail Party Problem". Da un lato, i biologi stanno ancora studiando come i pipistrelli ecolocalizzino in modo efficace all'interno di sciami densi, distinguendo tra segnali ambientali e segnali dei conspecifici. Dall'altro, un sistema di ecolocalizzazione efficace potrebbe aiutare la robotica a migliorare il rilevamento attivo, poiché le tecnologie attuali come il RADAR e l'infrarosso (IR) risultano inadeguate in ambienti densi.
Qesta prima fase del progetto RO-BAT corrisponde allo srudio condotto in queta tesi. L'attenzione è focalizzata sull'aspetto passivo della localizzazione basata sul suono, che fornisce una base per futuri sviluppi nell'ecolocalizzazione attiva. L'ecolocalizzazione passiva consente a un robot di individuare sorgenti sonore esterne nello spazio senza emettere segnali propri. In questa tesi, ho progettato, assemblato e testato una piccola piattaforma robotica bio-ispirata chiamata "ro-bat", ottimizzata per la stima della Direction of Arrival (DOA) in tempo reale, per evitare gli ostacoli in ambienti multi-agente. Il ro-bat è dotato di microfoni Micro-Electro-Mechanical Systems (MEMS) disposti in un array microfonico costruito ad-hoc, collegato alla scheda audio MCHStreamer e a un Raspberry Pi per l'elaborazione del segnale. Ho sviluppato e testato tre algoritmi DOA — Generalised Cross-Correlation with Phase Transform (GCC-PHAT), Steered Response Power with Phase Transform (SRP-PHAT) e Multiple Signal Classification (MUSIC) — e li ho integrati con il controllo di navigazione del ro-bat, bilanciando accuratezza, efficienza computazionale e reattività su una piattaforma con risorse limitate.
Gli esperimenti sono stati condotti in condizioni di laboratorio controllate utilizzando il robot Thymio II come piattaforma mobile, con ognuno degli algoritmi testato in varie configurazioni di ostacoli per valutare l'accuratezza della localizzazione e la distribuzione dell'errore. I risultati hanno mostrato una efficace capacità di evitare gli ostacoli con due algoritmi(GCC-PHAT e SRP-PHAT), mentre il MUSIC ha avuto prestazioni scarse sull'hardware testato a causa delle elevate esigenze computazionali. Complessivamente, il

GCC-PHAT ha mostrato prestazioni migliori rispetto agli altri algoritmi, risultando il più veloce e reattivo. Tuttavia, la configurazione testata, pur essendo affidabile, presentava una precisione limitata nella risoluzione laterale dell'angolo a causa di scelte costruttive e dell'array microfonico.
Il mio ro-bat è diventato una piattaforma rappresentativa per la ricerca in robotica bio-ispirata condotta presso il Centre for the Advanced Study of Collective Behaviour (CASCB), portandomi a realizzare numerose dimostrazioni della piattaforma in vari eventi, inclusa la mia partecipazione a ICRA@40.

Parole chiave: localizzazione di sorgente sonora, direzione di arrivo, MEMS, robotica degli sciami, comportamento collettivo

Acknowledgements

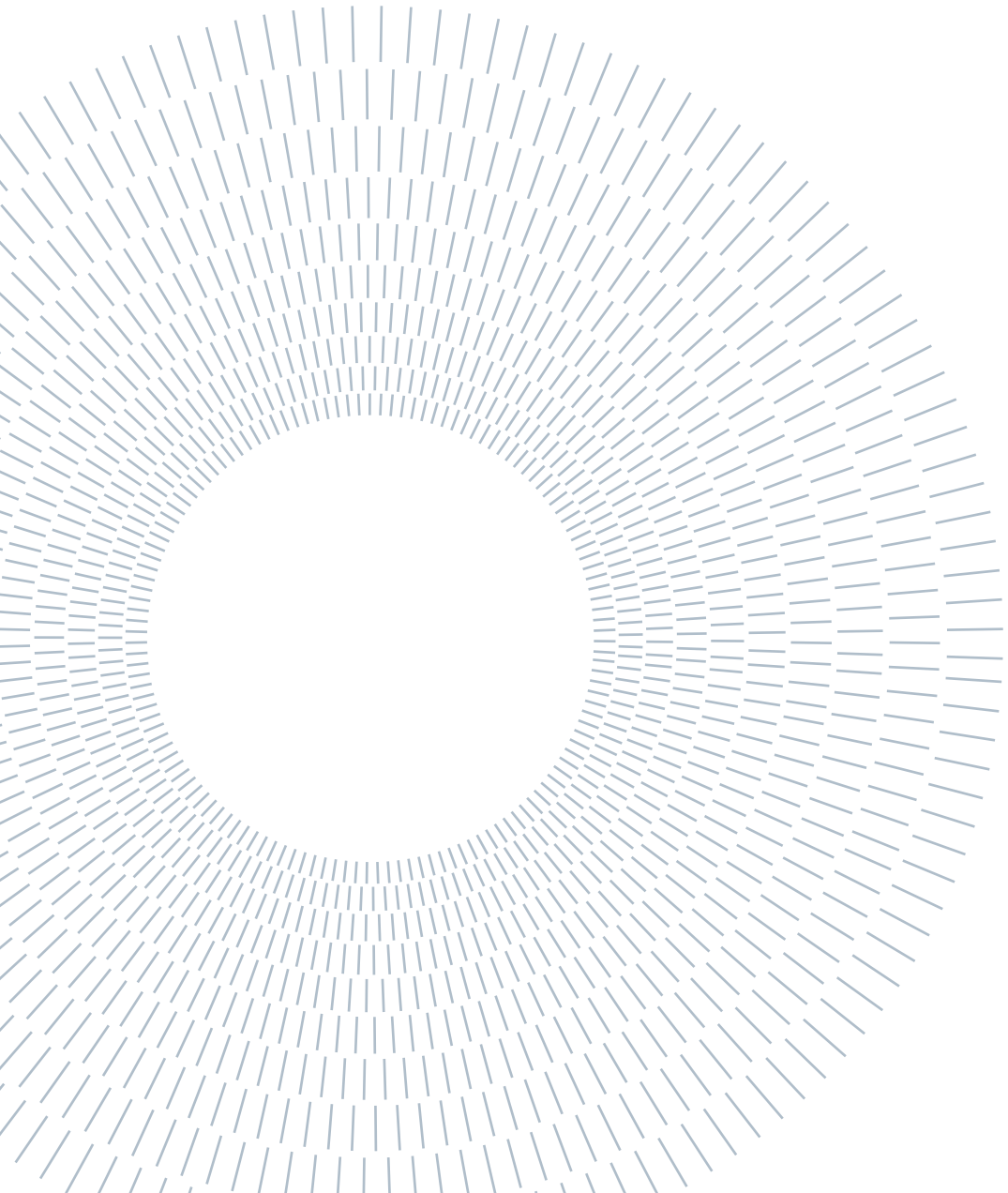
I would like to thank my co-supervisors Giovanni, Heiko and Thejasvi and all friends in Konstanz for welcoming me and being part of this amazing adventure. Thanks also to my supervisor Prof. Fabio Antonacci, who made it possible and to all the people that contributed in this thesis.

Thanks to my family who always supported me.

Thanks to all the amazing people and friends I met during this Master. I will always bring the best memories from this period with me.

Special thanks to *Cerbero* for being my second family in Cremona.

And finally thanks to Eleonora, always believing in me.



Contents

1 | Introduction

1.1. Objective

This thesis is the first contribution to the RO-BAT project, which aims at understanding how bats echolocate in large swarms when they are affected by the so-called "Cocktail party problem" [14]. This refers to a situation where an individual in a swarm, typically an animal or a robot, encounters numerous overlapping stimuli from other group members and must interpret them using various strategies. Understanding how animals solve this problem is relevant to both biology and robotics. On the one hand, biologists still do not know how bats can fly in large groups while echolocating themselves in relation to the environment and nearby conspecifics [40]. On the other hand, discovering the mechanisms that enable sound source localisation in noisy environments has the potential to inspire a new generation of robotic autonomous systems. Indeed, an efficient acoustic echolocation system for robot swarms has yet to be developed, and bats could provide an answer to this problem.

Furthermore, the study of swarms and, in particular, the collective behaviour of individuals within a swarm, has long been a fascinating problem in biology. Recently, however, the growing field of autonomous robotic systems has driven an interest in understanding collective behaviour, with the aim of utilizing these insights to robotic applications.

Many animals, for instance, use visual cues to navigate the environment and move as a group. Vision allows them to see their surroundings and detect multiple neighbours, even as the group size increases. However, in active-sensing animals like bats, the ability to locate other individuals rapidly declines as swarm size increases, due to sensory limitations or "bottlenecks" in their echolocation capabilities [10]. This challenge is similarly encountered in robotics, where active sensing strategies, such as RADAR or infrared (IR) systems, also scale poorly in densely populated environments.

Therefore, by implementing bats' group echolocation in a swarm of echolocating robots, this research seeks to understand and potentially replicate these natural navigation strategies in robotics.

1.2. Hosting institution and workplace

It is important to provide an overview of the lab where the project was developed. The project was hosted by the Centre for the Advanced Study of Collective Behaviour (CASCB) at the University of Konstanz, an excellence cluster connected to various departments of the university, but mainly studying swarms of animals and robots. This thesis was specifically supervised by Dr. Andreagiovanni Reina and his robotics group, Dr. Thejasvi Beleyur's, expert in biological echolocation, and Prof. Dr.-Ing. Heiko Hamann, leading the Cyber-Physical Systems group, which primarily focuses on studying swarm robotics inspired by biology.

Throughout this thesis, the university's workshop provided fantastic support. Workshop engineers have been involved in the prototyping phase and offered valuable consultancy regarding the electronic components to purchase. I led most of the interactions between the research team and the workshop engineers, which I coordinated with a high degree of autonomy. Unfortunately, the long lead times for component delivery at the university have impacted the project's development, sometimes slowing down the prototyping process.

1.3. Original contributions

In my thesis, I only developed the first part of the RO-BAT project, which will continue for several years. This thesis focuses on the passive aspect of sound-based localisation, i.e., the robot locates external sound sources without the robot emitting its own signals. This passive sound localisation capability is a critical first step, providing a robust foundation for future advancements in active echolocation. By refining algorithms for detecting and interpreting sound directions, I established baseline capabilities that will allow for the integration of active sound processing in subsequent phases. The use of passive echolocation presents a low-cost, low-energy solution suitable for lightweight, mobile robots, offering an effective way to enable real-time sound source localisation on resource-constrained platforms, typically used in swarm robotics.

In particular, in this thesis, I selected, tested, and assembled the necessary components to build a small robot (named "ro-bat") capable of passive sensing of the environment. In order to do this, I implemented affordable but advanced technologies such as Micro-Electro-Mechanical-System (MEMS) microphones connected by standard digital transmission protocols such as PDM and I^2S. I designed and created various prototypes of microphone arrays with different geometries, in order to find the best solution for our

problem. I selected and adapted some well-known Direction of Arrival (DOA) algorithms such as Generalised Cross Correlation with Phase Transform (GCC PHAT), Steered Response Power with Phase Transform (SRP PHAT) and Multiple Signal Classification (MUSIC). I compared the performance of the three DOA algorithms by running them on board the ro-bat in a series of experiments. I developed the navigation strategy to let the ro-bat move around the sound sources and avoid them only based on the DOA calculation.

The ro-bat, that I designed and built, has become one representative platform to showcase the type of research on bio-inspired robotics that is conducted at the CASCB. This led to my involvement in several scientific dissemination events, where at each of them I delivered a short presentation on the RO-BAT project and ran live demonstrations with my ro-bat. Among others, I presented and demonstrated the project at the CASCB's Konstanz School of Collective Behaviour and to various professors and organisations visiting our centre. I also won a travel grant from ICRA@40, the 40th anniversary of the IEEE International Conference on Robotics and Automation, where I successfully led the four-day demonstration of the ro-bat [1] and coordinated a team of three people comprising colleagues from the University of Konstanz.

1.4. Structure of the thesis

This thesis is organised into five chapters.

In Chapter 1, I provide an overview of the general problem and the context in which the thesis has been developed, introducing the necessary interdisciplinary topic involved.

In Chapter 2, I present the background theory and the essential tools for addressing the thesis problem, along with the state-of-the-art on the topic. The background section covers key components of the robot, including MEMS microphones, I^2S and PDM communication protocols, microphone array configurations, and Direction of Arrival (DOA) algorithms. The state-of-the-art section explores significant studies on sound source localisation and ultrasound-based echolocation systems.

In Chapter 3, I explain the hardware selection, design, and prototyping steps involved in creating the ro-bat. I present some key components including the Thymio II robot, its loudspeaker characterisation, the Raspberry Pi, the custom-developed microphone arrays, and the MCHStreamer Kit sound card. In addition to hardware, I present an overview of the software and libraries used, as well as the implementation of Direction of Arrival (DOA) algorithms on the ro-bat.

In Chapter 4, I explain the experimental results obtained from testing the ro-bat in the lab. I first outline the structure of the experiments, followed by a presentation of the quantitative results.

In Chapter 5, I draw the conclusion about the thesis, underlying its role in the bigger project where it is involved. I suggest future developments and the possible applications of this technology.

2 | Background and state of the art

In this chapter, I present essential information and tools relevant to addressing the problem proposed in this thesis, alongside an overview of the literature that offers valuable insights and solutions for related challenges.
This chapter is divided into two main sections: the background (Section 2.1) and the state of the art (Section 2.2). In the background section, I introduce the theory behind the functioning of some of the fundamental parts theat compose the robot used in this thesis. This includes the MEMS microphones in Subsection 2.1.1, which capture sound signals, and the I^2S and PDM communication protocols in Subsection 2.1.2, used for audio signal transmission. I then describe microphone array configurations in Subsection 2.1.3 and, finally, discuss the theory behind Direction of Arrival (DOA) algorithms in Subsection 2.1.4, which allow the robot to interpret its surrounding environment.
In the state of the art section, I first explore notable works in sound source localisation in Subsection 2.2.1, followed by a review of interesting studies related to echolocation systems using ultrasound in Subsection 2.2.2.

2.1. Background

2.1.1. MEMS microphones

MEMS (Micro-Electro-Mechanical Systems) microphones are miniature microphones fabricated using silicon-based structures. As depicted in Figure 2.1, they usually integrate a mechanical component (transducer) and the Application-Specific Integrated Circuit (ASIC), that transforms vibrations into digital or analog electrical signals. MEMS microphones technology, thanks to the really small form factor (only a few millimeters), is widely used in smartphones, laptops and smart-speakers, where performance, space and power consumption are key factors to consider.

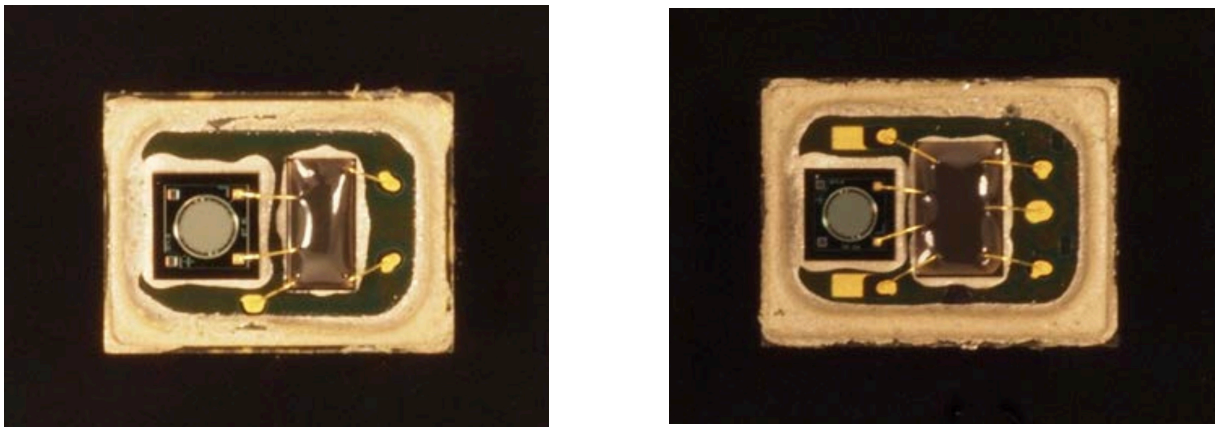


Figure 2.1: The analog (left) and the digital version (right) of MEMS microphones' internal components. In both configurations the left part is occupied by the Transducer, instead the ASIC is positioned in the right part. Photos taken from [8].

2.1.1.1. MEMS internal components

The transducer, or capsule, is the central component of a MEMS microphone, crucial for both analog and digital setups. It exploits movement of a top movable plate (known as the diaphragm) and a bottom fixed one to transform the acoustic field into electrical signals. As the diaphragm moves in response to sound pressure waves, the distance between the plates changes and so the capacitance accordingly. This change generates a voltage signal that conveys information about the acoustic field.
The second component in a MEMS microphone is the Application-Specific Integrated Circuit (ASIC), which determines if it is defined as digital and analog. In the analog case, the ASIC amplifies and adapts the analog signal from the transducer and allows it to be transferred to other electronic components, instead, in the digital version, it also converts the analog electrical signal into a digital one, allowing these microphones to be mounted onto small electronic devices without the need of an external Analog-Digital converter.
specifically, the transducer is constructed entirely from a silicon wafer, with additional layers added during manufacturing. The diaphragm is typically made of polysilicon to allow potential differences to be detected between the plates. MEMS transducers can be designed in two configurations: top-port and bottom-port. In a top-port design, sound waves reach the diaphragm and the components chamber first, while in a bottom-port design, the diaphragm is reached first and the components belong to the back chamber.

The geometry of the microphone affects its acoustic response. The presence of an internal chamber in the MEMS microphone determines a resonance peak at high frequencies due to the Helmholtz resonator effect. Additionally, a roll-off at low frequencies is present

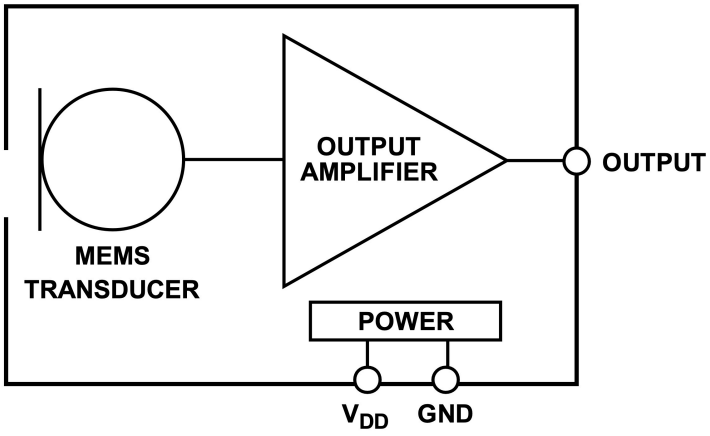


Figure 2.2: Analog MEMS microphone block diagram. Image taken from [8].

because of the small dimensions of the device compared to the large wavelength of low frequencies. The following quantities are defined: c is the sound speed in air, S is the cross-section of the enclosure aperture, V is the volume of the enclosure, and L_{eff} is the effective length of the neck aperture. The latter is defined as: $L_{eff} = L + 0.85r$, where L is the length of the neck and r the aperture radius. Then, the resonance frequency can be computed as:

$$f_R = \frac{c}{2\pi}\sqrt{\frac{S}{VL_{eff}}}\,. \tag{2.1}$$

2.1.1.2. Analog MEMS microphones

The analog MEMS microphone provides a continuous output which is the result of the transducer movement under the sound pressure waves action. However, as shown in Figure 2.2, the output voltage needs to be preamplified in order to increase the output signal level and avoid noise corruption by the other electronic components.

2.1.1.3. Digital MEMS microphones

Similarly to analog MEMS microphones the digital counterpart only differs by the fact that the analog to digital conversion (ADC) is performed by the microphone itself, so that the output is already digital. The MEMS output signal is amplified and then taken into an anti-aliasing filter. A typical digital output format in MEMS microphones is Pulse Density Modulation (PDM) or Inter-IC Sound (I^2S) format. Pulse Density Modulation (PDM) is a representation of an analog signal with a 1-bit binary signal. This signal makes use of zeros and ones which are sent at high-rate from the microphone to the sound

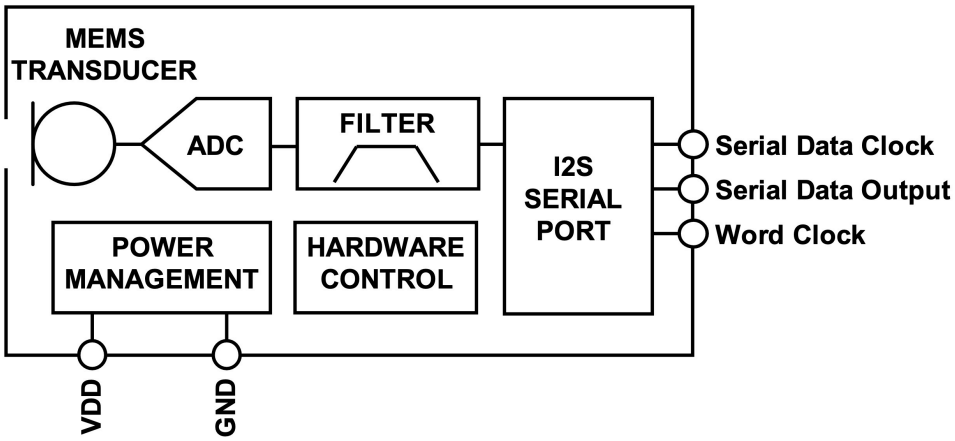


Figure 2.3: I²S MEMS microphone block diagram. Image taken from [8].

card. This means that the information is not encoded as continuous amplitude values, like normally happens with analog sensors, but instead the relative density of the pulses (ones) corresponds to the analog signal's amplitude. Inter-IC Sound (I²S) is an electrical serial bus interface standard used for audio signals. Differently from high sample rate PDM signal, I²S outputs digital data at a decimated audio sample rate and, thanks to integrated processing within the microphone, allows fewer external components in the audio processing chain.

2.1.2. Communication protocols for MEMS microphones

To connect and transfer digital data from MEMS microphones to other electronic components, a communication protocol is essential. Two of the most relevant and popular protocols for this purpose are I²S (Inter-IC Sound) and PDM (Pulse Density Modulation). This section presents both protocols, highlighting their advantages and disadvantages.

2.1.2.1. I²S (Inter-IC Sound)

The I²S (Inter-IC Sound) protocol [31], developed in 1986 by NXP (formerly Philips Semiconductors), is specifically designed for PCM (Pulse Code Modulated) audio data transfer between digital audio devices. It is crucial in applications where audio fidelity is important, such as in high-end audio equipment and represents the standard communication protocol between integrated circuits in an electronic device. I²S uses the three lines in Fig. 2.4: Serial Data (SD), Continuous Serial Clock (SCK) and Word Select (WS). These lines facilitate the synchronisation and transmission of audio data between devices, minimising the number of pins required. Specifically, two time-division multiplexed data

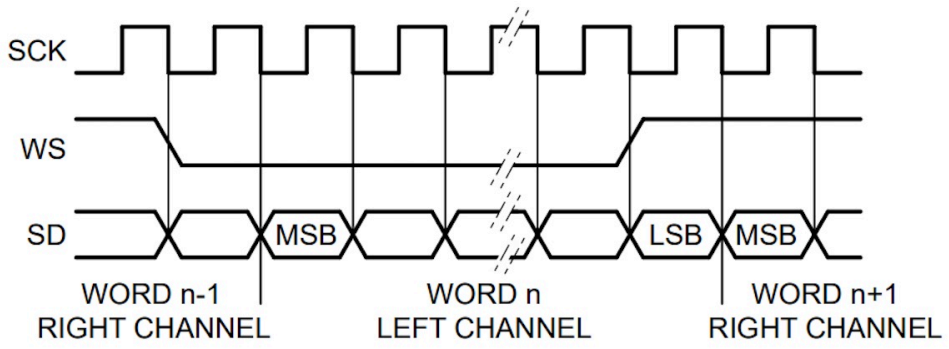


Figure 2.4: I²S channel interface showing the Serial Data (SD), Continuous Serial Clock (SCK) and Word Select (WS) channels. Image taken from [31].

channels run on the SD line, the WS line indicates which channel is being transmitted (0=left, 1=right) and the SCK line is used to share the clock signal. Both the WS and SCK signals are generated by the controller and received by the target.

As described in Fig. 2.5 the protocol supports multiple connection configurations:

- **Transmitter** (controller): Sends audio data.
- **Receiver** (target): Receives audio data.
- **Controller**: Manages the communication process between the transmitter and receiver.

The I²S protocol also supports the following Operation Modes:

- **Right Justified**: Data is right-aligned in the data word.
- **Left Justified**: Data is left-aligned in the data word.
- **Philips Standard**: The default and most widely used mode, aligning data with specific clock edges for timing accuracy.

2.1.2.2. PDM (Pulse Density Modulation)

PCM (Pulse Code Modulation) is largely used to represent a signal in digital audio systems, as it has the advantage of being easy to manipulate and allows signal processing operations such as mixing, filtering and equalisation to be performed on the stream. Pulse Density Modulation (PDM) [22] was introduced because it offers the benefits of digital signals, such as low noise and interference, using an even simpler method of data transmission with respect to PCM. The PDM uses only three main lines, similar to the I²S

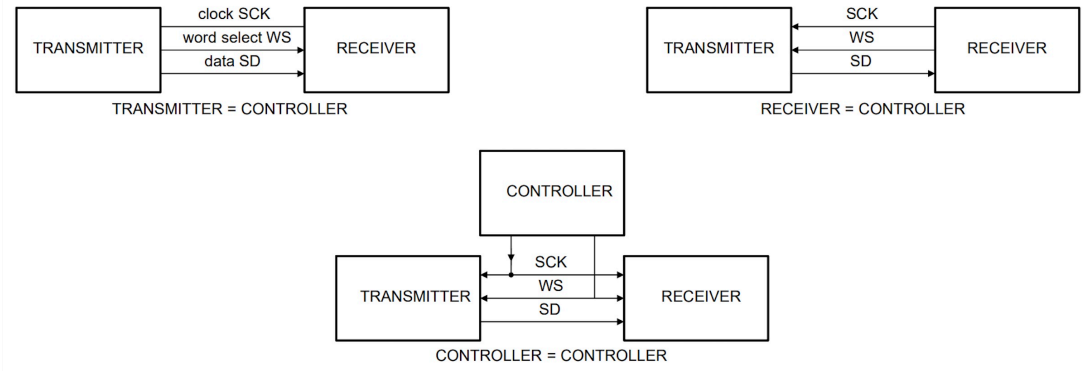


Figure 2.5: Standard I^{2}S possible connection configurations between transmitter, receiver and controller. Image taken from [31]

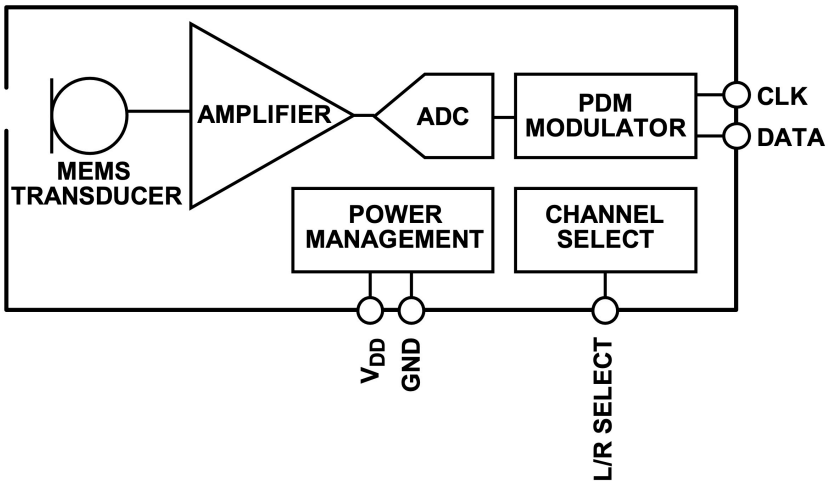


Figure 2.6: PDM MEMS microphone block diagram of the basic internal components. Image taken from [8]

protocol: Data line, Master Clock line and Select line. The Master Clock is provided by the external device and determines both the sampling rate of the system and the rate at which bits are sent. It is typically around 3 MHz. The single-bit data is transmitted on the Data line on either the rising or falling edge of the Master Clock, defined by the Select line. This allows each data line to carry two streams at the same time, minimising the required connection points.

2.1.3. Microphone arrays

A microphone array consists of multiple microphones distributed across different spatial locations, designed to simultaneously capture an acoustic field [11]. This setup is carefully engineered to accurately provide an acoustic image of the sound environment, enabling the estimation of the direction of arrival (DOA) of incoming sound sources. Additionally, through specific beamforming algorithms, the array can selectively enhance sounds from certain directions while reducing unwanted noise from others. Microphones can be arranged in various spatial configurations depending on the specific requirements and are generally classified into the following categories:

- Linear microphone arrays
- Planar microphone arrays
- 3-Dimensional microphone arrays

2.1.3.1. Linear microphone arrays

In a linear microphone array, the positions of the microphones (or capsules) are arranged along a single axis [39]. This is the simplest configuration for an array and the sampling positions are most frequently chosen between uniform spacing and logarithmic spacing. In the first case, the distance between each microphone is kept constant. In the second case, the distance changes in a logarithmic fashion [13]. Due to its symmetric shape, the array is not capable of discerning mirrored positions (front-back) with respect to its main longitudinal axis.

2.1.3.2. Planar Microphone Arrays

A planar array of microphones is distributed on a two-dimensional (2D) surface and can exploit different microphone patterns based on the specific application. Some example include Spiral arrays and Randomised array, which have the named distribution along the surface. Uniform Rectangular Arrays (URA), where microphone are positioned in

a regular rectangular pattern, can be used for example in applications such as three-dimensional (3D) sonar mapping or acoustic cameras, since they can detect the position of the source along two axes.

2.1.3.3. Three-Dimensional microphone arrays

In a three-dimensional microphone array, the microphone capsules are distributed over a 3D shape, typically equidistant from a common central point. A spherical array is a common example in this category, though other geometries like the tetrahedron, octahedron, or icosahedron can also be used. The main advantage, compared to linear or planar configurations, is the ability to capture signals coming from all directions. However, this comes at the cost of requiring more sophisticated algorithms to retrieve spatial information. This design is commonly used in applications such as advanced sound field representation, 3D sound capture, and spatial audio processing [39].

2.1.3.4. Spatial aliasing

The problem of aliasing occurs when sampling a signal in the time domain, particularly when the Nyquist-Shannon theorem is not respected. Similarly, spatial sampling must be considered when dealing with a representation of the sound field captured by microphone arrays. The anti-aliasing condition must be met in order to avoid the creation of artifacts such as grating lobes in the directivity pattern of the array and limits the frequency range recorded [25]. According to the temporal sampling theorem, a signal must be sampled at a rate f_s (with a period T_s) such that:

$$f_s = \frac{1}{T_s} \geq 2f_{\max} \tag{2.2}$$

where $f_{\max}$ is the highest frequency component in the signal's spectrum. An analogous relationship can be defined for spatial sampling. For the sake of simplicity, let's consider the specific case of a uniformly spaced linear array of omnidirectional microphones. The sound source produces a planar wavefront entering the array at a θ angle. The signal is considered to be narrow-band at frequency ω_c, so its wavelength is:

$$\lambda = \frac{c_0}{f_c} = \frac{2\pi c_0}{\omega_c} \tag{2.3}$$

where c_0 is the speed of sound in air, equal to 343 $\frac{\text{m}}{\text{s}}$. The spatial frequency ω_s can be

defined as follows:

$$\omega_s = \omega_c \frac{d \sin\theta}{c_0} = \frac{2\pi d \sin\theta}{\lambda} . \tag{2.4}$$

From the sampling theorem, the following condition must be satisfied:

$$|\omega_s| \leq \pi , \tag{2.5}$$

which can be written as:

$$\left| \frac{2\pi d \sin\theta}{\lambda} \right| \leq \pi . \tag{2.6}$$

This leads to a simple anti-aliasing condition, related to the microphone spacing d, where

$$d \leq \frac{\lambda}{2} \tag{2.7}$$

The same condition can be expressed in terms of maximum frequency as

$$f_{max} = \frac{c_0}{2d} . \tag{2.8}$$

The frequency f_{max} is the theoretical upper limit that the array can capture without aliasing [11]. However, a lower limit is also present and strongly depends on the total array dimensions. Since an incoming sound wave must be detected by at least two capsules, the maximum distance between two microphones L determines the array's low-frequency limit. Hence, the minimum frequency is

$$f_{min} = \frac{c_0}{2L} . \tag{2.9}$$

2.1.4. Direction Of Arrival (DOA) algorithms

Direction of Arrival (DOA) estimation refers to techniques used to determine the direction of the signal arriving at the microphone array. In the case of a linear microphone array, the position can be determined as an angle between ± 90 degrees, but can be extended to three-dimensional space with other geometries such as rectangular or spherical arrays. However, the distance from the array can only be calculated when the sound source is located in the near field, i.e. when the sound source is similar in size to the array [39]. In this thesis, the far-field condition is assumed due to the small dimensions of the microphone array, and consequently the incoming signals are expected to be planar waves. As illustrated in Fig. 2.7, the wavefront forms an angle θ with the line perpendicular to

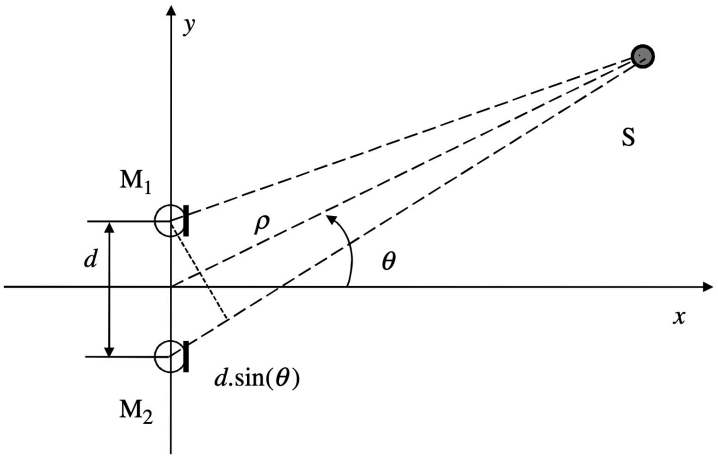


Figure 2.7: Two-microphone DOA estimation: the source S is located in the far-field, M_1 is the reference microphone, the incident angle is θ and the spacing between the two microphones is d.

the array, and the signal S received at microphone M_2 is a time-delayed version of the reference signal at M_1. Considering the spacing between M_1 and M_2 to be d, the time delay τ_{12} is given by

$$\tau_{12} = \frac{d \sin\theta}{c}, \tag{2.10}$$

where c is the speed of sound and $d \sin\theta$ is the time required for the plane wave to propagate from the first to the second microphone. Knowing the time delay τ_{12} the angle θ can be estimated as:

$$\theta = arcsin\left(\frac{\tau_{12}c}{d}\right). \tag{2.11}$$

2.1.4.1. Time Difference Of Arrival (TDOA)

A crucial point of DOA estimation in the far-field is the accurate detection of the time delay between each microphone along the array. This problem is also known as Time Difference Of Arrival (TDOA) estimation [11]. When dealing with M equidistant microphones, the kth delay τ_k can be computed simply by adding the multiplier $(k-1)$:

$$\tau_k = (k-1)\frac{d \sin\theta}{c} \qquad k = 1, ..., M \tag{2.12}$$

and further deriving the DOA θ as previously explained. Many approaches have been introduced in order to solve this problem, but here only the most relevant for this application are presented.

2.1.4.2. Generalised Cross Correlation with Phase Transform (GCC PHAT)

An accurate calculation of the TDOA is a really important step to correctly and reliably detect the angular position of the sound source. The first and most intuitive approach is cross-correlating signals from different microphones to find a maximum value [39]:

$$R_{12} = \text{iFFT}(G_{X_1X_2}), \tag{2.13}$$

where X_1, X_2 are the Fourier transforms of the microphone input signals and $G_{X_1X_2} = X_1X_2^H$ is the cross-power spectral density function. The cross correlation function R_{12} peaks at the time shift between the two signals, indicating the delay. However, a more robust implementation of the same concept is the so-called Generalised Cross Correlation (GCC), which introduces various weighting functions ψ to artificially whiten the signal spectrum:

$$R_{12} = \text{iFFT}(\psi \cdot G_{X_1X_2}). \tag{2.14}$$

In particular, a well-known weighting function is the phase transform function (PHAT), which eliminates the magnitudes, giving equal weight to the phase in each frequency bin:

$$\psi_{\text{PHAT}}(f) = \frac{1}{|G_{X_1X_2}(f)|}\,. \tag{2.15}$$

2.1.4.3. Beamformers

The simple approach of using the cross correlation to retrieve the DOA presented in 2.1.4.2 is further developed here with the broader category of beamformers, which include the more advanced methods named Steered Response Power (SRP) and Multiple Signal Classification (MUSIC) presented respectively in 2.1.4.4 and 2.1.4.5. Beamforming, in a conventional sense, involves a filter-and-sum process where temporal filters are applied to microphone signals before summing them into a single, focused output. These filters are often adopted to enhance the desired signal while suppressing others. The simplest filter performs time shifts to match the source signal's propagation delays, a method called Delay-And-Sum (DAS) beamforming. Unfortunately, reverberation consists of time-delayed copies of the same signal and beamformers cannot always suppress this interfering signal, and this can lead to errors in the estimation of the DOA. To overcome this problem, another filtering stage can be introduced to extend the the delay-and-sum concept to the so-called filter-and-sum approach [15].

2.1.4.4. Steered Response Power (SRP)

If the source location is unknown, beamformers can be used to scan a predefined spatial region by adjusting the steering delays (and possibly their filters). The output of the beamformer when used in this way is known as the steered response. The steered response power (SRP) can peak under different conditions, but it is maximised when the steering delays match the actual propagation delays. By modelling the characteristics of the propagating waves, steering delays can be mapped to a location indicating the source location [15]. The steered response power, considering an array of M microphones, is expressed as the output power of a filter-and-sum beamformer and can be written as:

$$P(\Delta_1 \cdots \Delta_M) \equiv \int_{-\infty}^{+\infty} Y(\omega, \delta_1, \ldots, \delta_M) Y'(\omega, \delta_1, \ldots, \delta_M)\, d\omega \tag{2.16}$$

where $\Delta_1 \cdots \Delta_M$ are the steering delays which are function of the terms τ_m, τ_0, which are the m-th and the reference propagation delay, respectively:

$$\Delta_m = \tau_0 - \tau_m \quad \text{for} \quad m = 1 \cdots M\,. \tag{2.17}$$

The term $Y(\omega, \Delta_1 \cdots \Delta_M)$ in Eq. (2.16) is the output of the filter-and-sum beamformer:

$$Y(\omega, \Delta_1 \cdots \Delta_M) \equiv \sum_{m=1}^{M} G_m(\omega) X_m(\omega) e^{-j\omega\Delta_m}. \tag{2.18}$$

The previous equation is defined in the frequency domain by the Fourier transforms of the microphone input signals $X_1(\omega) \cdots X_M(\omega)$ and the temporal filter transforms $G_1(\omega) \cdots G_M(\omega)$.

The steering delays, $\hat{\Delta}_1 \cdots \hat{\Delta}_M$, which maximise the equation (2.18), provide the estimates of the TDOA (Time Difference Of Arrival) between the microphones. This is analogous to the behaviour of generalised cross correlation for two microphones, since the peak in the correlation function corresponds to the TDOA of the sound waves at the microphone pair. In this case, the TDOA estimate between the l-th and q-th positions in the array is equal to the difference of their steering delays:

$$\hat{\tau}_{lq} \equiv \hat{\Delta}_l - \hat{\Delta}_q. \tag{2.19}$$

The beamformer exhibits a global maximum in output power when it is pointing towards on the source's spatial position, thus providing an estimate of the source's location. If $\vec{d}$

is the candidate location, then the SRP is a function of $\vec{d}$ defined as:

$$P(\vec{d}) = P(\Delta_1 \cdots \Delta_M) \quad \text{for} \quad \Delta_m = \frac{\tau_0 - (\vec{d}_m - \vec{d})}{c} \tag{2.20}$$

Similar to what has been defined in the equation (2.15) for Generalised Cross Correlation, there is a phase transform (PHAT) weighting function also for the SRP beamformer. It can be defined for all M microphones of the array as follows:

$$G_m(\omega) \equiv \frac{1}{|X_m(\omega)|} \quad \text{for} \quad m = 1, \ldots, M \tag{2.21}$$

Similarly to what was done in the GCC phase transform, these filters are designed to whiten the microphone signals, which effectively cause a sharpening of the peaks in the phase transform and an improved steered response power.

2.1.4.5. Multiple Signal Classification (MUSIC)

An alternative approach to TDOA estimation is provided by eigenvector-based techniques. Originally developed for DOA estimation in radar applications, these methods have recently been extended to the processing of broadband signals using microphone arrays. A well-known algorithm among these methods is the so-called Multiple Signal Classification (MUSIC) [11]. Consider the data matrix $\mathbf{X}$, which contains the signals received at each microphone for multiple signal sources. The signal model is given by:

$$\mathbf{X} = \mathbf{AS} + \mathbf{V} \tag{2.22}$$

where $\mathbf{A}$ is the steering matrix, $\mathbf{S}$ represents the source signals, and $\mathbf{V}$ is the noise matrix. The covariance matrix of the source signals is:

$$R_s = \mathbb{E}\{\mathbf{S}(t)\mathbf{S}^H(t)\} \tag{2.23}$$

The following hypotheses are made:

- The number of sources N is known in advance and it is lower then the number of microphone M ($N < M$).
- The covariance matrix of the source signals is non-singular.
- The sensor noise is spatially white, with independent and identically distributed

components, having equal variance:

$$\mathbb{E}\{\mathbf{e}(t)\mathbf{e}^H(t)\} = \sigma^2\mathbf{I}_M \tag{2.24}$$

- The noise is uncorrelated with the source signals.
- The Direction of Signals (DOAs) are different, leading to distinct spatial frequencies.

Under these assumptions, the input covariance matrix $\mathbf{R}$ is decomposed as:

$$\mathbf{R} = \mathbf{A}R_s\mathbf{A}^H + \sigma^2\mathbf{I}_M = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^H \tag{2.25}$$

Computing the eigenvalue decomposition and sorting to have a descending order of the eigenvalues the matrix can be written as:

$$\mathbf{R} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^H \tag{2.26}$$

where $\mathbf{U}$ contains orthonormal eigenvectors, and $\mathbf{\Lambda}$ is a diagonal matrix containing the real eigenvalues $(\lambda_1, \lambda_2, \ldots, \lambda_M)$. Since signals and noise are uncorrelated it can be reduced to:

$$\mathbf{\Lambda} = \mathbf{\Lambda_s} + \mathbf{\Lambda_e} = \mathbf{\Lambda_s} + \sigma^2\mathbf{I}_M \tag{2.27}$$

Any vector orthogonal to $\mathbf{A}$ is an eigenvector of $\mathbf{R}$ with eigenvalue σ^2, and there are $M - N$ such vectors. The remaining N eigenvalues are larger than σ^2, allowing us to distinguish between *signal eigenvectors* and *noise eigenvectors*. The covariance matrix can now be written as:

$$\mathbf{R} = \mathbf{U}_s\mathbf{\Lambda_s}\mathbf{U}_s^H + \mathbf{U}_n\mathbf{\Lambda_n}\mathbf{U}_n^H \tag{2.28}$$

where $\mathbf{U}_s$ are associated to source signal and $\mathbf{U}_n$ are instead associated to noise. To estimate the Direction of Arrival (DOA), the MUSIC algorithm exploits the orthogonality of the noise subspace to the steering matrix $\mathbf{A}$. The projection operators onto the signal and noise subspaces are:

$$\mathbf{P}_s = \mathbf{U}_s\mathbf{U}_s^H = \mathbf{A}(\mathbf{A}^H\mathbf{A})^{-1}\mathbf{A}^H \tag{2.29}$$

$$\mathbf{P}_n = \mathbf{U}_n\mathbf{U}_n^H = \mathbf{I} - \mathbf{A}(\mathbf{A}^H\mathbf{A})^{-1} \tag{2.30}$$

Since the signals are linearly independent, the matrix $\mathbf{A}^T\mathbf{A}$ is full rank. Additionally, since the eigenvectors in $\mathbf{U}_0$ are orthogonal to $\mathbf{A}$, this leads to the following relationship:

$$\mathbf{A}^H\mathbf{U}_n = \mathbf{0} \tag{2.31}$$

that can be written also as:

$$\mathbf{A}^H \mathbf{U}_n \mathbf{U}_n^H \mathbf{A} = 0 \tag{2.32}$$

In other words, the estimated signal covariance matrix will produce an estimated orthogonal projection onto the noise subspace:

$$\mathbf{P} = \hat{\mathbf{U}}_n \hat{\mathbf{U}}_n^H \tag{2.33}$$

Finally, the Directions of Arrival (DOAs) can be retrieved from the N highest peaks of the MUSIC spatial pseudo-spectrum function, which is defined as:

$$P(\theta) = \frac{1}{\mathbf{a}^H(\theta)\mathbf{P}\mathbf{a}(\theta)} \tag{2.34}$$

Essentially, the MUSIC algorithm estimates the distance between the signal and noise subspaces in the direction where a signal is present. Since these two subspaces are orthogonal, the distance between them at that angle will be zero or near zero. Conversely, if no signal is present in a particular direction, the subspaces are not orthogonal, and the result will be non-zero.

2.2. State of the art

In robotics, the problem of echolocation with ultrasonic signals has been relatively understudied compared with the vast literature on robot navigation with light-based sensors and cameras. While there are numerous studies on sound source localisation methods (SSL), to our knowledge, there are no works in the context of the cocktail party problem in groups of echolocating robots. In this section, I first present in Subsection 2.2.1 the most relevant work regarding sound source localisation and in Subsection 2.2.2 some other interesting works regarding echolocation systems with ultrasounds, in order to give an overview of the main research areas and problems connected to this thesis.

2.2.1. Sound source localisation overview

Starting with sound source localisation, the most relevant work in the literature [32] summarises well the different aspects to be considered when dealing with this topic. First of all, the notion of sound source localisation can be divided into two main components: estimating the *direction* from which the sound is coming and estimating its *distance*. In most cases, only direction is considered, since it was historically the first to be implemented and allows the computation to be simplified (it only requires two microphones and running

simpler algorithms).

Another important aspect to consider is the *propagation model*, as it affects how sound waves travel through space and, most importantly, their shape and characteristics. The most commonly used model is the far-field free-field model. In this model, the sound source is assumed to be in the far-field, meaning that the sound waves are planar by the time they reach the microphones. Additionally, the free-field assumption implies that there are no obstructions between the sound sources and the microphones, and no reflections from the environment are considered, so reverberation is not taken into account.

The most popular *acoustic characteristics* used in the audio analysis can be classified as:

- **Time Difference of Arrival** (TDOA): Measures the time delay between signals arriving at different microphones.
- **Inter-Microphone Intensity Differences** (IID): Measures the energy differences between two microphones.
- **Spectral notches, binaural/spectral cues**: Techniques that analyze the spectral information between microphones.
- **Beamforming techniques and subspace methods**: advanced techniques used with more than two microphones, e.g., delay-and-sum, MUSIC (Multiple Signal Classification).

Mapping methods are another crucial point to consider, as they help in understanding and extracting information from the environment being analysed. These methods are generally divided into the following categories:

- **1-Dimensional single DOA Estimation**: Methods focused on determining the direction of the sound source on a single plane. These commonly use TDOA-based methods, such as Pearson correlation or the generalised cross correlation with phase transform (GCC-PHAT), which is known for its robustness against noise and reverberation.
- **2-Dimensional Single DOA Estimation**: Methods for determining the sound source's direction in both azimuth and elevation angles. These methods often use more complex microphone arrays or machine learning models such as neural networks to map acoustic features into 2D locations. One popular technique involves using Head-Related Transfer Functions (HRTFs) to simulate human hearing.
- **Multiple DOA Estimation**: In scenarios with multiple sound sources, techniques like beamforming and MUSIC (Multiple Signal Classification) presented in 2.1.4.5

are used.

- **Distance Estimation**: Methods that estimate the distance between the sound source and the robot. These often involve multi-microphone arrays or integrate data from additional sensors like cameras.

The *characteristics of the sound source* are also important aspect to consider, when dealing with the SSL problem. In particular, the following characteristics should be taken into account:

- **Type of sound signal**: The majority of works in the literature deal with speech signals, as this is the most common way of communication and in a human-robot interaction. However, most of them are not limited to using only speech, but they can manage any kind of signal in the audible range. Ultrasonic signals, on the other hand, are not usually considered for localisation purposes because robots need special sensors and processing units to elaborate them and this is generally not of interest in this type of application. Ultrasound signals however suffer less from interference with ambient noise and are largely used for precise calculation of distances. They have been used extensively when echolocation mapping was performed, as in [17] [21] when the specific purpose was to mimic bat behaviour, as in [35] [42].
- The **number of sources** can range from one to around ten different sources. However, most studies focus on locating a single source [30][24], primarily because it allows for simpler and more computationally efficient algorithmic implementations.
- The **mobility of the sources** is an important consideration in many studies [41][36], as robotic platforms typically have some degree of movement in space. This means that the sound source will always have a relative velocity with respect to the array, which distinguishes these kind of implementations from generic audio systems designed to be static.
- The **distance of the sound sources from the microphones** typically reaches a total range of approximately five metres, as the area of interest for localisation is usually smaller. Within this range, the signals are less affected by noise, reverberation and other disturbances so this normally simplifies the computation [9][19].
- **Noise** is often considered to be uncorrelated with the input signal and is mainly generated by the electronic components in the data stream [20]. Many algorithms try to reduce its effect on the final localisation, but some are more robust than others, since some methods, such as MUSIC, consider noise as part of the model, resulting in a better handling of this aspect.

Hardware components are a key aspect of the SSL system as they can affect the performance of the whole system. Again, the best way to analyze this topic is to break it down into different areas:

- **Microphones**: In most cases, omnidirectional microphones are used, since they can scan the environment and locate the sound source from all directions. The main disadvantage is that they are also very sensitive to noise and interference, while other polar patterns, such as the cardioid, are used to limit the recording to a specific area and reduce interference. However, non-omnidirectional microphones require a special type of capsule, which is more difficult to obtain, especially in small microphones (e.g. MEMS), and adds a layer of complexity at the algorithmic level.
- **Audio interface**: The audio interfaces used are typically self made or have been adapted from generic signal processor to work on audio signals. Some commercial audio interfaces are also used, but they typically occupy more space and so they are more difficult to integrate in small robots.
- **Array geometry**: Two main categories can be identified: symmetric arrays and irregular arrays. The first category includes the 1-dimensional arrays previously described, such as binary arrays (the most common) and linear arrays with more than two microphones, along with 2-dimensional and 3-dimensional arrays, which are less commonly used. The second category, more rare, involves scattered positioning of the microphones, as seen in [21]. The number of microphones in the array can range from one to many, but typically two are used. This is mainly because the robot aims to mimic human hearing with a binaural approach, and also because audio data is easier to compute and transmit in real-time applications. However, eight microphones are often used, particularly when algorithms are designed to perform better with more sensors.
- **Robotic platform**: Although the specific platform does not have a major impact on audio performance, it can limit the electronic equipment and processing power that can be carried on board. The platform can vary greatly in shape and size, but is generally a mobile base for indoor or outdoor navigation and/or mapping. Arrays can be fixed to the platform or mounted on robotic heads or arms to allow rotation in specific directions, as in [17]. Finally, although less commonly used, flying mobile bases (unmanned aerial vehicles, or UAVs) can also be employed. A notable example of flying localisation is demonstrated in [42], using very small commercial drones.

2.2.2. Other relevant works

In this section, additional works specifically related to this thesis is presented, as several ideas have been inspired by these sources. These papers provide a broader perspective on the challenges inherent in this project and offer insights into the future developments in active ultrasonic echolocation.

The first interesting work is [21]. The Cosys-Lab of the Faculty of Applied Engineering of the University of Antwerp, in Belgium, created a fully embedded 3D Imaging Sonar Sensor for robotic application, which uses ultrasonic signals to create spatial maps in real-time and on-board. This sensor, unlike the previous works presented in Subsection 2.2.1, uses active echolocation to map the environment by emitting pulses of ultrasonic signals from a single transducer. The reflections are then captured by an array of 32 microphones, distributed in two different layouts: a flat ellipsoidal shape and a rectangular shape, both using Poisson disc sampling. Incoming reflections are pre-filtered and processed to create an "Energyscape": a delay-and-sum (DAS) beamformer is used in conjunction with multiple filtering stages to process the data. An STM32 microcontroller featuring an ARM Cortex-M4 processor, and a Cyclone V SoC, which includes an FPGA and an ARM Cortex-A9 dual-core processor, are used in the two different layouts so that their performance can be then compared. Another important electronic component to mention is the high-voltage amplifier, which is necessary to drive the transducer, but considerably increases the power consumption. MEMS microphones are used as sensors, allowing the array to be compact and the overall dimensions to remain quite small. The sensor is mounted on a mobile robot that moves at 0.3 $\frac{\text{m}}{\text{s}}$ to create a 2D map of the environment in real time, at a rate of 3 Hz. Some aspects of this work closely follow other sound source localisation (SSL) methods, such as the use of a Delay-and-Sum (DAS) beamformer, a 2D MEMS microphone array geometry, and a robotic mobile platform. However, specific components and processing techniques have been developed to address the unique challenges posed by ultrasonic signals.

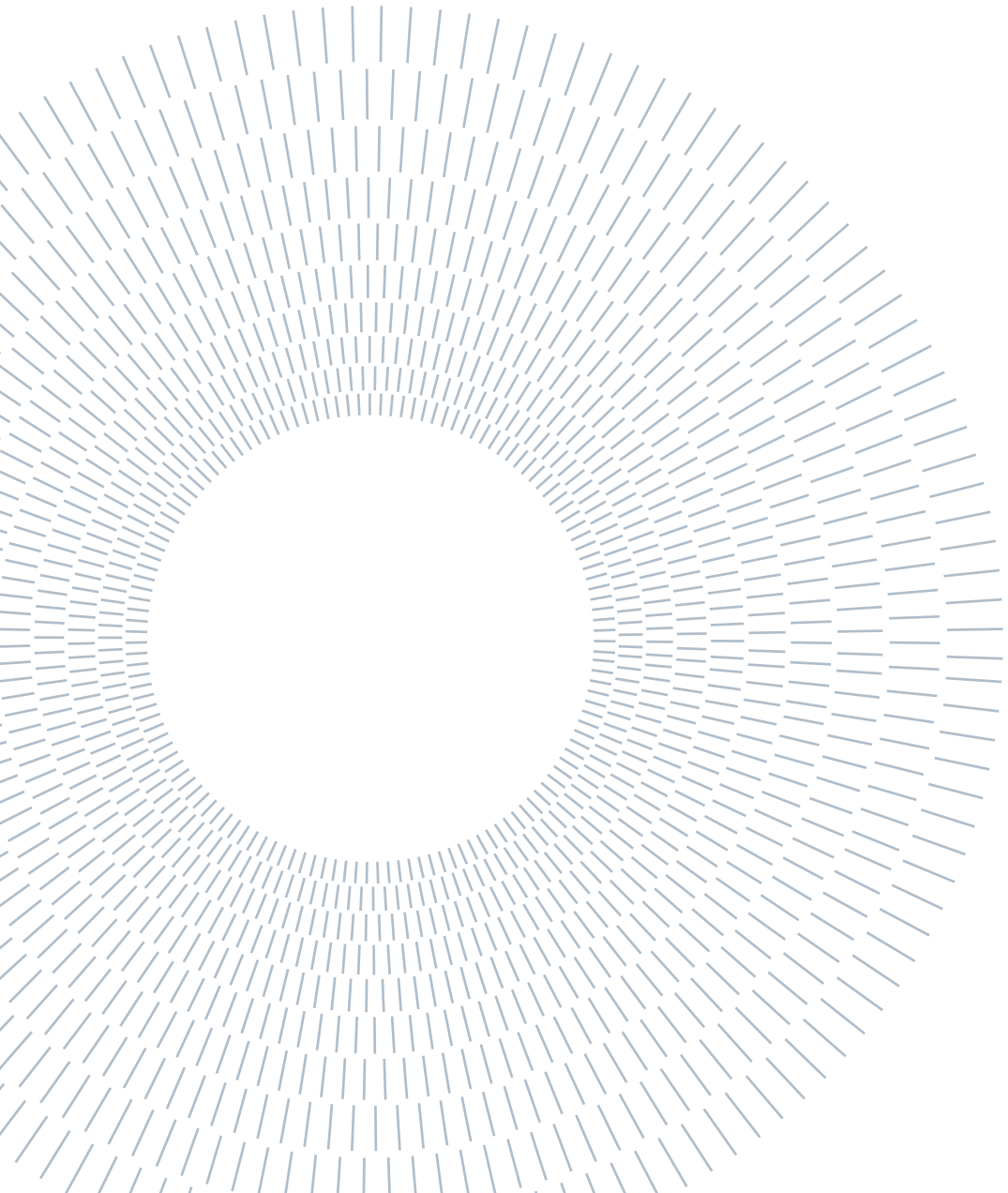
Although the work by Kerstens *et al.* [21] is very promising, its implementation is not straightforward. It requires specialised skills to develop the electronic boards and handle the large amount of data generated by the microphones. Additionally, the system cannot be miniaturised to fit on small robots and requires a large battery to operate. These factors make it difficult to use their method in (large) swarms of small echolocating robots. Furthermore, the system is likely unable to function effectively when multiple ultrasonic sources emit simultaneously, since the signal processing has been specifically designed to capture reflection instead of dealing with direct pulses arriving at the array, limiting its

use in studying, for example, the cocktail party problem.

Another interesting work [42] explores the miniaturisation of both the robotic and audio systems while still handling ultrasonic signals. As in previous studies, the inspiration comes from biology, specifically bat echolocation, with the goal of locating an ultrasound signal using a small flying drone. The most notable aspect of this work is the miniaturisation: the entire system is a modified version of a Crazyflie 2.0 drone [12], equipped with a pair of MEMS microphones for localisation, weighing only 30 grams in total. The onboard electronics were designed in-house to process the input signal in real-time. The drone is able to turn toward an ultrasonic pulse emitted from a moving position on the ground. The processing is simplified to be performed on-board the drone, primarily calculating the direction of arrival (DOA) using cross correlation, after retrieving the time difference of arrival (TDOA) of the sound source. The authors reduced the interference from rotor noise by employing a pre-filtering on the input data. While the drone could correctly turn in place toward the signal, the accuracy is limited, and localisation is only performed in 2D. Although this is a really good implementation of SSL with ultrasounds, there are two critical points. First, the testing conditions were limited to an anechoic room, eliminating disturbances and reverberation problems. Second, the need for custom electronics and a small battery to reduce weight and size restricts the system's adaptability to different conditions because it has limited processing power and time window of use.

The last relevant work to mention [17] is also inspired by bats, but differs significantly from the others. This work aims to mimic bats' ability to echolocate, scan new environments and navigate autonomously through them. To achieve this, the robotic platform is equipped with a robotic head that can rotate to scan a 180-degree field of view, subdivided into three 60-degree portions. Similar to [21], the system uses a single ultrasonic emitter that produces frequency-modulated (FM) chirps at a typical bat rate, and two condenser microphones that function like bat ears. The signals are fed to A/D and D/A converters using the USB-1608GX-2AO NI DAQ board, sampling at 250 KS/s. The mapping process involves moving the robot through the environment, stopping to perform a scan, and then moving forward. The environment is unfamiliar to the robot and contains various types of objects, including plants. Obstacles are identified by using an on-board camera that feeds images into a neural network, which classifies objects as either plants or non-plants. Unlike other works [35], this robot did not focus only on bat-inspired acoustic echolocation capabilities, but it also created a map of the environment and its obstacles. However, this comes with the drawback of the robot being large and slow. It moves 0.5 meters, stops for about 30 seconds to perform scanning, and continues, which limits its ability to echolocate itself in real time. This reason together with the large size of the equipment

and its high cost reduce the feasibility of deploying it on large swarms of robots.



3 | Realisation

In this chapter, considering that this project was primarily focused on ro-bat implementation rather than simulation, I discuss, in Section 3.1, the selection, design, and prototyping of all the robot components. Specifically, I present the Thymio II robot in Subsection 3.1.1, followed by its loudspeaker characterisation in Subsection 3.1.2. Next, I introduce the Raspberry Pi in Subsection 3.1.3 and provide a detailed description of the microphone arrays developed for this thesis in Subsection 3.1.4. Finally, I discuss the most crucial audio component, the MCHStreamer Kit sound card, in Subsection 3.1.5.

Following the hardware discussion, I present an overview of the most relevant software and libraries used in this project in Section 3.2, along with the actual implementation of the Direction of Arrival (DOA) algorithms on the ro-bat in Section 3.3.

3.1. Hardware

The first step in the project was the hardware selection, that took into consideration also the future development of the project. In particular, attention was given to selecting components, such as the sound card and MEMS microphones, that could operate within the ultrasonic frequency range. This was necessary to develop an array capable of receiving pulses generated by an ultrasonic transducer, and consequently study the echolocation problem.

A schematic overview of the ro-bat is showed, together with a photo of the real robot, in Figure 3.1. The individual parts and the connections between them will be explained in detail in the following Sections.

3.1.1. Thymio II robot

One fundamental part of the ro-bat is its mobile platform, since it enables the robot to move and run the echolocation algorithm while navigating around an environment. The chosen platform for this project is the Thymio II robot [29] from Mobsya [28], selected for two specific reasons. Firstly, these robots were already available in the lab and pre-

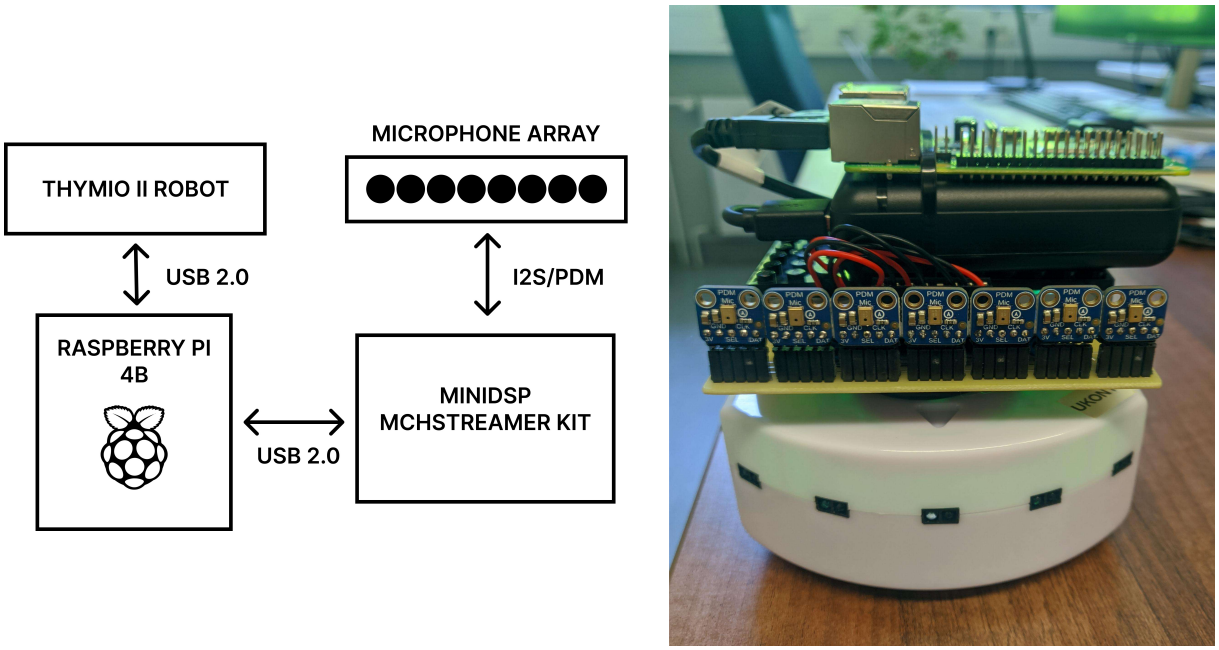


Figure 3.1: Overview of the main components, on the left, and the real ro-bat, on the right, with the array V1 described in Subsection 3.1.4

configured to function effectively in this context. Secondly, they are relatively small, when compared to other ground platforms, making the project's future goal to scale the number of robots simpler economically and logistically. The scalability factor is a key aspect of studying bats' echolocation, since the complexity of audio processing increases with each additional unit in the swarm. Lastly, this commercial solution allows the research to focus on audio components without the additional demands of developing and maintaining custom-built robots.

Thymio II is an educational robot designed to introduce students to programming and robotics. However, it can be used for professional applications thanks to its extensive sensors' capabilities and its compatibility with the Thymiodirect [7] Python library and ROS (Robot Operating System).

The Thymio II robot is 11 cm wide, 11.2 cm long, and 5.3 cm tall, and its most important features are:

- Five infrared proximity sensors on the front, two on the back, and two infrared ground sensors, enabling it to avoid collisions while detecting ground color.
- Two motors to move on the ground, that individually control the speed and direction of the wheels.
- USB connection for charging and data sharing.

- Five capacitive buttons to operate the robot.
- Speaker to emit sounds.
- Memory card slot for data uploading and recording.
- Programmable RGB LEDs.

3.1.2. Thymio II loudspeaker characterisation

As will be explained in the experimental setup Section 4.1.1 in Chapter 4, the Thymio's on-board speaker is used to generate a standardised sound source for the robot to follow or avoid. This choice is driven by the need to have a quick and easy solution for creating a sound source that can independently emit the same or different signals, depending on the experiment, while moving or standing still. However, in order to have a better idea of the frequency response and spatial distribution obtained from the loudspeaker, a directivity and SPL characterisation was performed.

3.1.2.1. Measurement setup

The measurements have been done in the SwarmLab room at the University of Konstanz. The room size is approx 5×5 m^2 and mostly composed of flat surfaces, except for the ceiling made of perforated panels and the window side covered by a thick curtain. The following setup was used:

- G.R.A.S preamplifier Type 26AC with 1/4" 40BF microphone capsule.
- G.R.A.S power module amplifier Type 12AA set at +40 dB Gain.
- Brüel & Kjær Type 4231 Sound Level Calibrator set at 1 kHz 94 dB SPL.
- RME Fireface UC audio interface [34] set at 0 dB Gain.
- Thymio II robot with loudspeaker set at maximum volume with LEGO bricks to raise powerbank and raspberry Pi from its top surface as in Figure 3.2.
- Acquisition PC connected via USB to the RME audio interface.

To best match the experimental conditions, the measurement microphone was positioned 10 cm from the ground, which is approximately the height of the array mounted on the Thymio. The distance was set to 0.5 m from the centre of the loudspeaker, which is positioned on top of the robot and is diffused by the power bank located above it.

The procedure used to characterize the loudspeaker involves recording the output of the

Figure 3.2: Thymio II loudspeaker measurement setup. The Thymio's loudspeaker under testing is position on the floor at 0.5 metres (displayed on the laser rangefinder) from the measurement microphone, which is attached to the ro-bat, in order to simulate the experimental conditions.

robot under test in seven positions, i.e. every 30° angle starting from 0°. Assuming a symmetrical response between the right and left parts, only half plot is analysed. The output signal used is a broadband white noise sampled at 8 bit 8 kHz for compatibility reasons with the robot, stored as a file on an external SD card mounted directly in the Thymio and played continuously for 30 minutes. The procedure has been repeated with the same testing conditions for six robots, in order to verify the consistency of the output signal across different sources during the experimental process.

A pure tone at 1 kHz and 94 dB SPL has been recorded by putting the measurement microphone capsule into the Brüel & Kjær Calibrator, and it has been used as the reference signal.

3.1.2.2. Results

The results from the measurements are summarised for the six robots in Figure 3.3 and 3.4. The overall emission pattern of all the robots in the tested configuration is nearly omnidirectional, indicating that the reflective surface above the robot effectively distributes sound in all directions. However, the polar plots reveal a peak in the emission pattern at 180 degrees for all tested units. This aligns with the fact that the loudspeaker is positioned towards the back edge of the robot's top surface, off-centred to the reflector above, resulting in a more prominent emission from the back.

Another important observation is the reduced emission between 30 and 60 degrees, and to a lesser extent at 120 degrees, due to the positioning of the loudspeaker on the Thymio and the presence of supporting LEGO blocks, which partially obstruct the emitted signal as Shown in Figure 3.2. However, in all the Thymios tested the variation moving around the robot is smaller than 5 dB RMS, indicating overall a fairly uniform emission. The directivity pattern also appears quite uniform and does not display significant differences across the various analysed bands. However, in the 3 kHz to 4 kHz band, there is a stronger emission at 0, 90 and 180 degrees, representing the positions where the emission is unobstructed by the top plate supports.

3.1.3. Raspberry Pi

The Thymio II platform is used in combination with a Raspberry Pi 4B [33], which significantly extends the ro-bat's processing power and enables wireless communication via Wi-Fi. For this project, the following Raspberry Pi 4B features are exploited:

- Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC @1.8GHz.

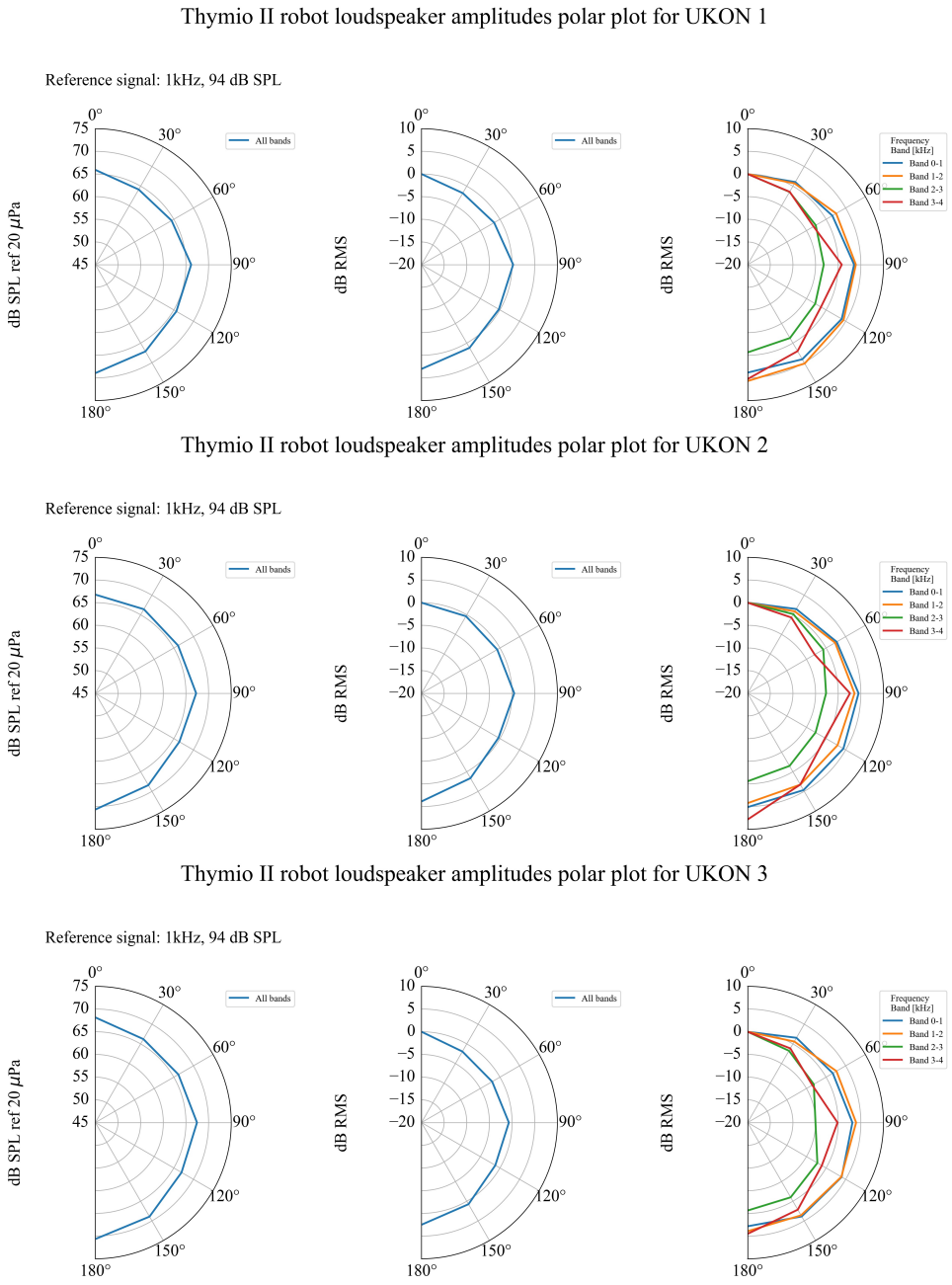


Figure 3.3: Directivity patterns of the Thymio II loudspeaker, tested on the first three robots. Starting from the left the polar plot shows the dB SPL level, calculated using the reference 1 kHz tone produced by the calibrator; the central plot shows the dB RMS level normalised at zero angle; the plot on the right represents the dB RMS levels divided into 4 sub-bands and normalised at zero angle.

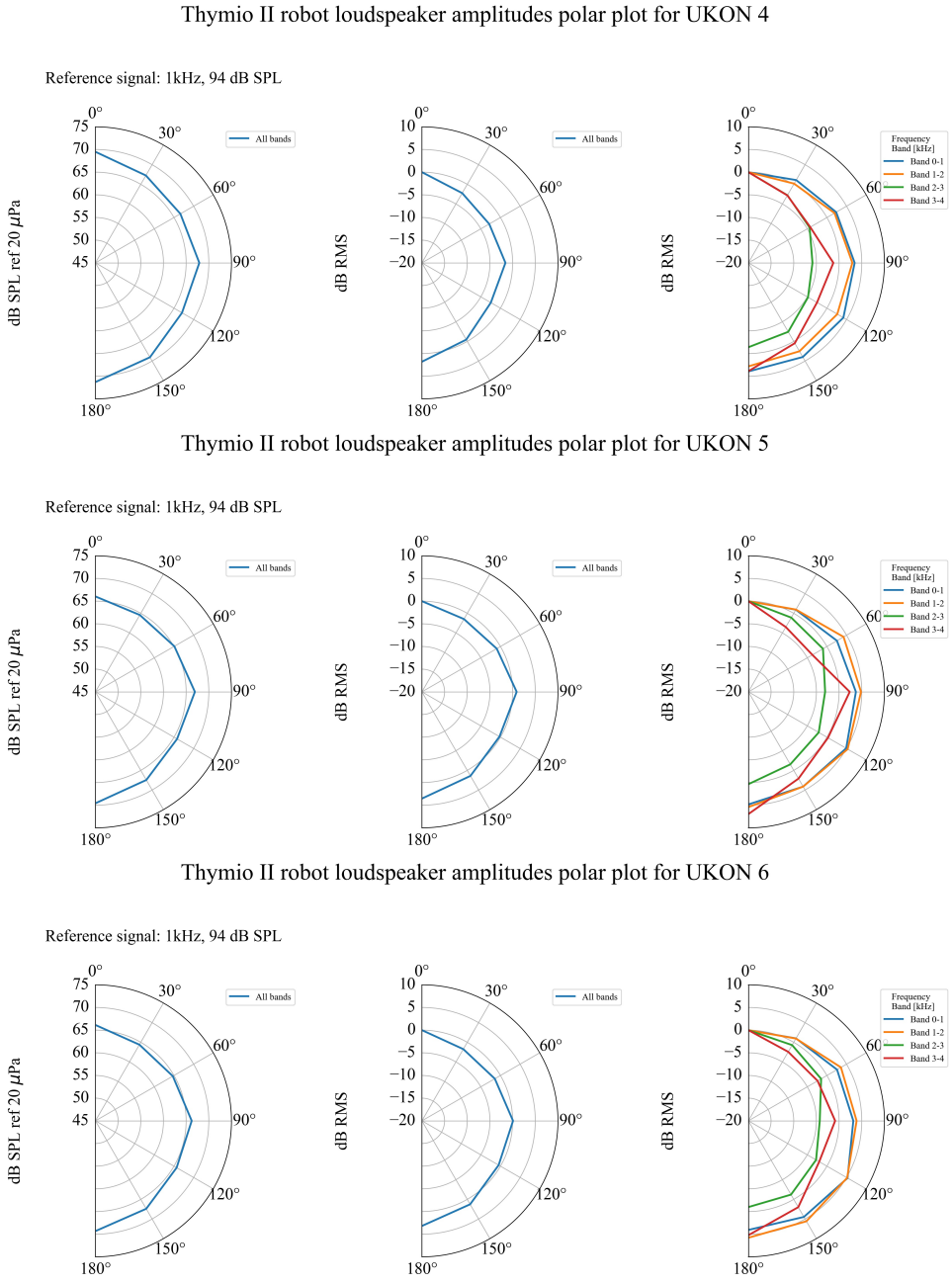


Figure 3.4: Directivity patterns of the Thymio II robot loudspeaker, tested on last three units. See description of the subplots in the caption of Figure 3.3.

- 2GB LPDDR4-3200 SDRAM.
- 2.4 GHz and 5.0 GHz IEEE 802.11ac Wi-Fi.
- Two USB 3.0 ports and two USB 2.0 ports.

The Thymio robot and the Raspberry Pi are connected via USB 2.0, and both are powered by an external battery. The battery and the Raspberry Pi are positioned on top of the robot, together with the sound card (described in Section 3.1.5), which is also connected and powered via USB 2.0, making the entire system compact and easy to set up.

3.1.4. Microphone array prototypes

The microphone array geometry was developed and refined through successive prototypes featuring different MEMS microphones and inter-element distances to achieve the optimal configuration. Since the selected MEMS microphones are already digital, no additional ADC (Analog to digital converter) is required between them and the sound card, facilitating easy modifications to the array geometry.

3.1.4.1. Array V0: Eight Adafruit I²S MEMS microphones

The initial idea was to test the sound card's limits by using all available channels with the I²S protocol. This approach aimed to evaluate the feasibility of using and comparing algorithms that enhance accuracy based on the number of sensors, while also introducing hardware-level redundancy to reduce errors in algorithms that theoretically do not require many microphones.

The first array prototype is shown in Figure 3.5 and consists of eight breakout boards from Adafruit, which facilitate the connection of the MEMS microphone SPH0645LM4H-B via six standard pins. The SPH0645LM4H-B [23] bottom port omnidirectional microphone from Knowles provides digital I²S output with a linear frequency response between 100 Hz and 10 kHz. This setup provides an effective trade-off between the microphone inter-distance (d) of 18 mm and the total array length (L) of 126 mm, resulting in low and high theoretical spatial aliasing frequencies of 1361 Hz and 9527 Hz, respectively, as discussed in Section 2.1.3.4. After some preliminary tests, conducted using single MEMS microphones connected to a breadboard, the possibility to build a custom array with this technology was confirmed. The custom connection board in Figure 3.6 was then designed and fabricated by the university workshop to provide a stable, unified connection between the sound card and the single microphones, in order to prevent failures during the tests on the ro-bat. This single attachment point uses the standard I²S protocol with the following

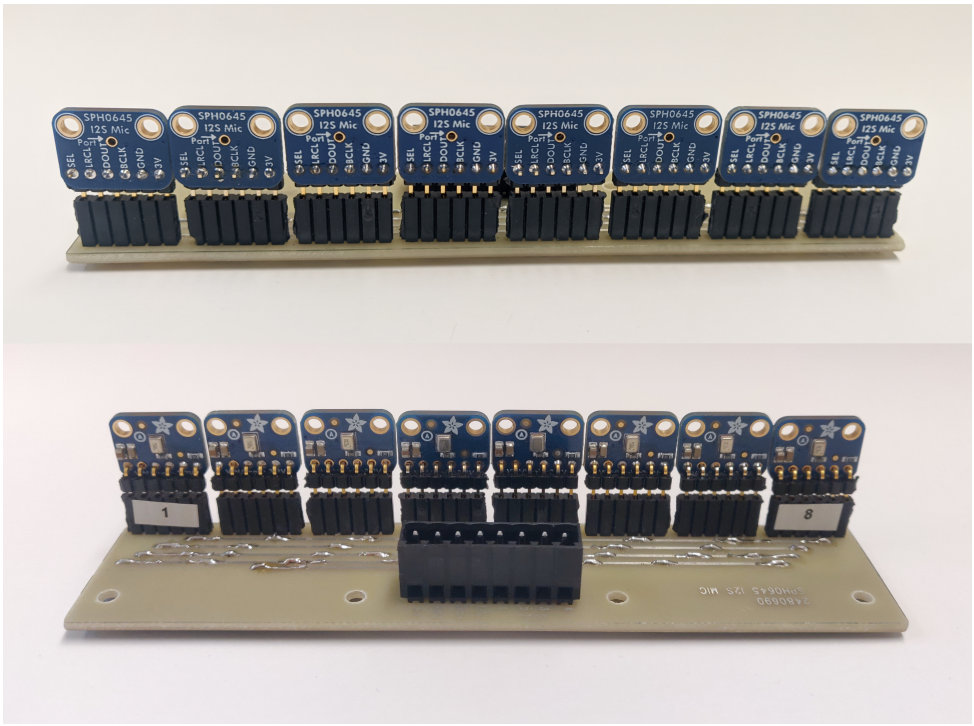


Figure 3.5: V0 array: I^2S 8 microphone array composed of SPH0645LM4H-B MEMS breakout boards from Adafruit and custom connection board. Front view (top) and back view(bottom).

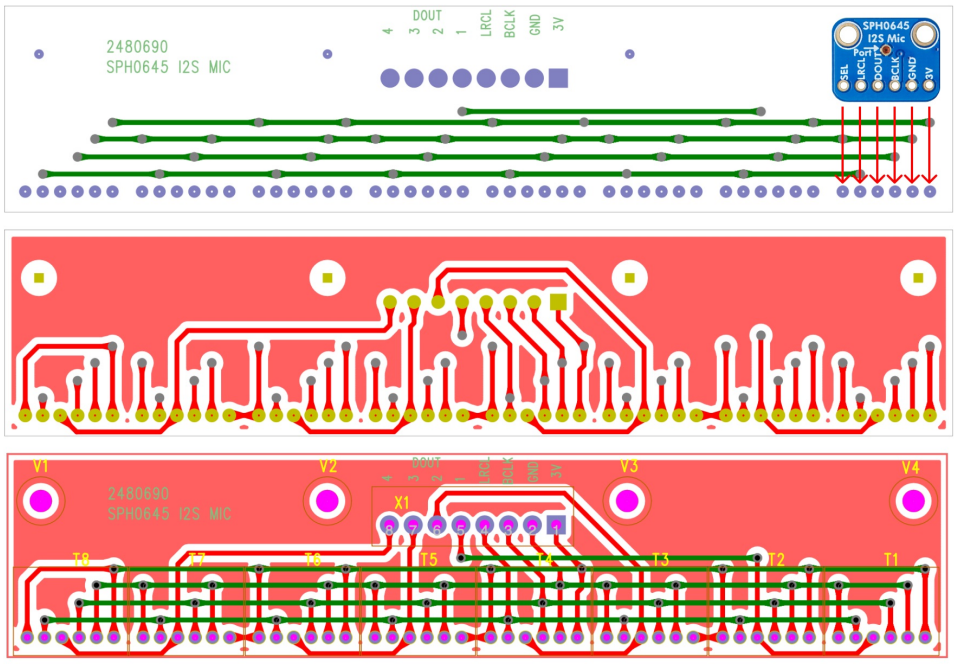


Figure 3.6: Schematics of the custom connecting plate for eight Adafruit SPH0645LM4H-B MEMS microphone breakout boards. The schematic layout is separated into three parts: top view (first image), bottom view (second image) and combined view (third image).

connections on the MCHStreamer:

- **3V**: voltage supply from pin 2 of J3 header.
- **GND**: ground from pin 11 of J1 header.
- **BCLK**: bit clock from pin 10 of J1 header.
- **LRCLK**: I^2S frame sync from pin 12 of J1 header.
- **DOUT**: I^2S data IN from channel 1 to 8 on Pins 2,4,6,8 on J1 header.

The same pins are also replicated on the Adafruit board, which includes an additional select pin (*SEL*) that can be connected to either *GND* or *3V*. As explained in Section 2.1.2.1, a single data line can carry signals from two microphones simultaneously. The *SEL* pin designates each microphone as either left (connected to *GND*) or right (connected to *3V*) in a stereo configuration, or as the first and second channels for our specific setup. Following this scheme, the eight signals are mapped to *DOUT* pins 1,2,3,4 on the array connector, corresponding to data input Pins 2, 4, 6, 8 of the J1 header on the sound card.

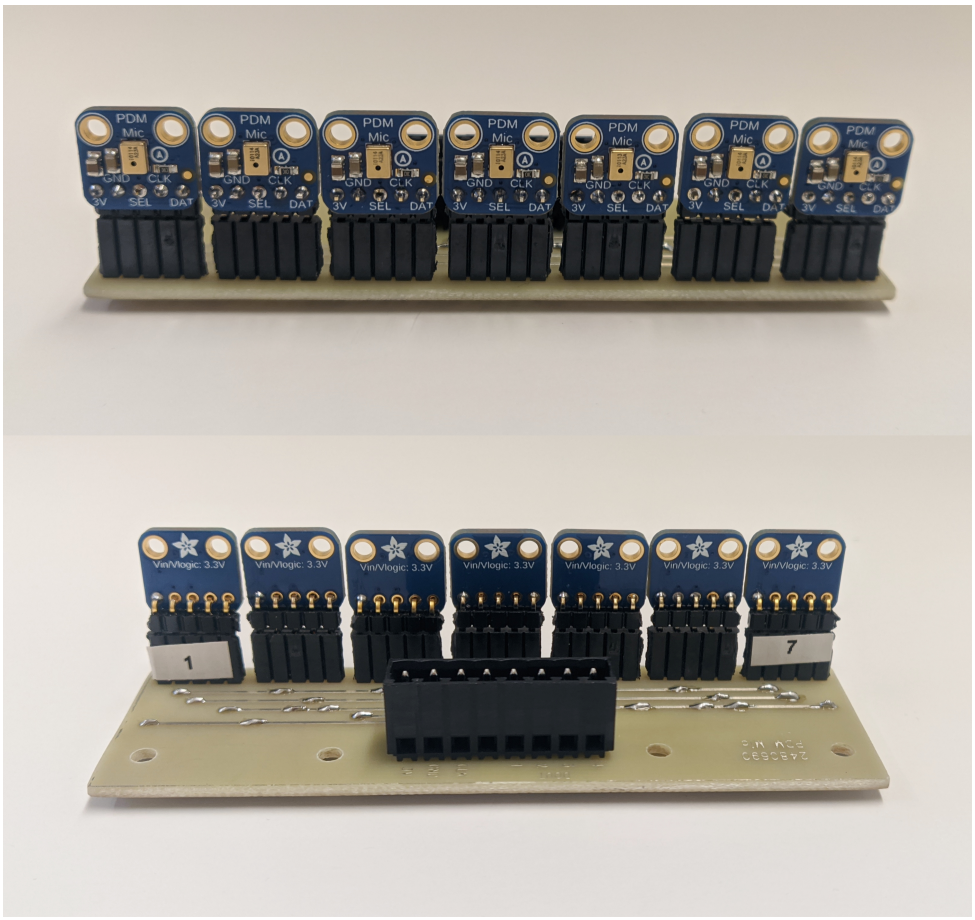


Figure 3.7: Array V1: seven microphone array composed of MP34DT01-M PDM MEMS breakout boards from Adafruit and custom connection board. Front view (top) and back view(bottom).

3.1.4.2. Array V1: Seven Adafruit PDM MEMS microphones

The array V1 (Figure 3.7) is conceptually similar to the array V0 (Sec. 3.1.4.1) as it remains a linear omnidirectional MEMS array, however, with a few distinctions. The design has the same priorities as the V0 design of an easy-to-swap configuration, using Adafruit breakout boards. Differently from the V0 design, the array now incorporates PDM MEMS digital microphones, specifically the MP34DT01-M [37], which are top-port omnidirectional microphones from STMicroelectronics. As with the previous array, these MEMS offer an approximately linear frequency response from 100 Hz to 10 kHz, with the advantage of PDM using only one clock signal. This allows for a smaller breakout board with only five pins, reducing the microphone inter-distance *(d)* to 15 mm and resulting in a higher theoretical spatial aliasing frequency of 11433 Hz.

To accommodate mounting on a Thymio II robot, the design was scaled down to seven

breakout boards, maintaining good redundancy. In this way, the array does not exceed robot's width and, thanks to a total width (L) of 90 mm, the theoretical spatial aliasing frequency is 1905 Hz, higher than V0 but suitable for our echolocation purposes. Additionally, the symmetrical design enables the use of a central reference microphone, which, when calculating cross correlation algorithms, allows DOA (Direction of Arrival) computations to be aligned with the ro-bat's movement, ensuring consistent movements along its natural axis. The sound card is interfaced to the array making use of a custom board shown in Figure 3.8. In this case, the connection is completely made through the J3 connector and is here explained in detail:

- **3V**: voltage supply from pin 2.
- **GND**: ground from pin 1.
- **CLK**: clock from pin 11.
- **DAT**: PDM data IN from channels 1 to 7 on Pins 3,4,5,6.

As already seen for the first array the pins are the same on the Adafruit board, but an additional select pin (*SEL*) is present. Similarly to I^2S, PDM transports two signals on a single data line, so the microphones can be connected to either *GND* or *3V* to indicate the left or right channel. Therefore, the data from the seven microphones are transmitted on four cables in pairs, except for the last one, which has only one signal.

3.1.4.3. Array V2: Eight I^2S custom MEMS array

To test the ultrasound capabilities of the I^2S protocol, a custom array (V2) of eight SPH0645LM4H-B MEMS microphones [23] was built by the university workshop, as shown in Figure 3.9. To sample high-frequency signals, the inter-distance between microphones was reduced to 3 mm, corresponding to a theoretical maximum frequency of 57133 Hz and a minimum frequency of 8166 Hz, since now the total length *(L)* is equal to 21 mm. This choice was primarily driven by the MEMS capsule width of 2.65 mm, which physically limited closer positioning.

However, given the fact that these MEMS are not optimized for such high frequencies, a significant amount of noise is generated when pushed beyond a 48 kHz sampling rate. Specifically, at the target sampling rate of 192 kHz, which would ideally take advantage of the 3 mm spacing, the noise level becomes quite high, preventing the array from being used effectively in this condition.

The connection configuration between the sound card and microphones, shown in Figure 3.11, is identical to that of the first array, enabling easy swapping between the two

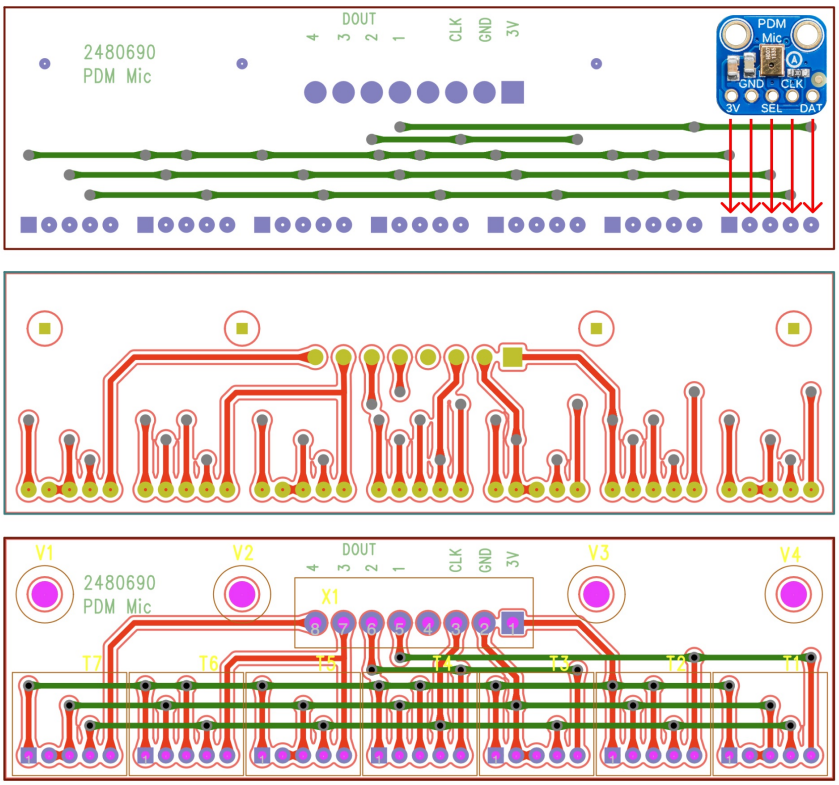


Figure 3.8: Schematics of the custom connecting plate for Adafruit MP34DT01-M PDM MEMS microphone breakout boards. The schematic layout is separated into three parts: top view (first image), bottom view (second image) and combined view (third image).

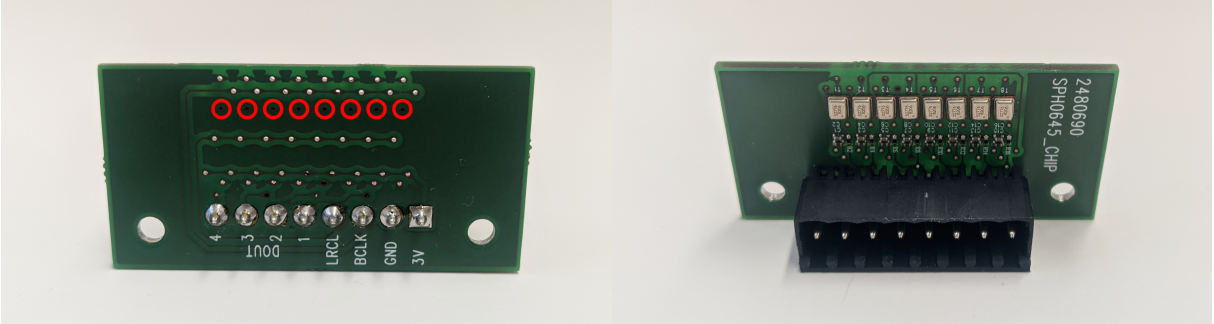


Figure 3.9: Array V2: Eight channels custom array layout for SPH0645LM4H-B MEMS microphones. Red dots indicate the microphone positioning on the board on the front view (left).

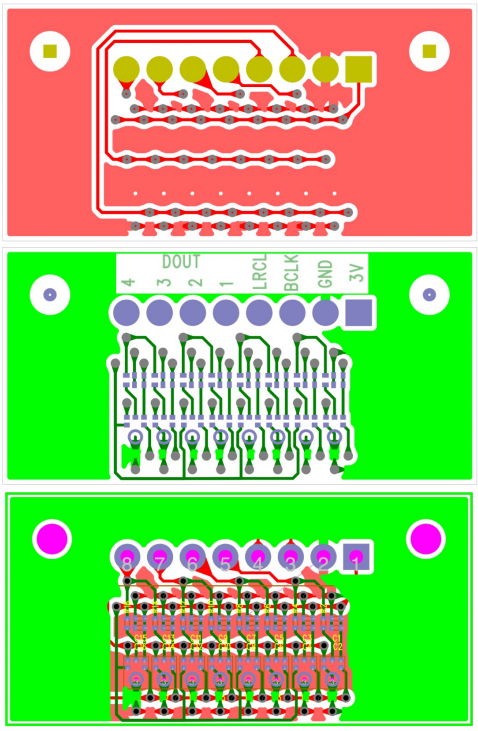


Figure 3.10: Eight channels custom connection layout for SPH0645LM4H-B MEMS microphones.

Figure 3.11: Schematics of the custom connecting plate for eight SPH0645LM4H-B MEMS microphones microphone breakout boards. The schematic layout is separated into three parts: front view (first image), back view (second image) and combined view (third image).

when attached to the same sound card.

Despite being subject to high noise levels at critical frequencies, this configuration serves as a valuable starting point for developing an ultrasonic array and can still complement the array V0 by sampling the high end of the spectrum, which V0 lacks.

Some final comments can be made about these prototypes: the linear shape for the array has been selected as the best compromise for this application, as it simplifies geometric calculations and keeps it compact when compared, for example, to circular arrays. This design also minimises environmental disturbances since the microphones face the front of the ro-bat, passively reducing reflective components of sound coming from above and helping focus on a 2D plane. However, thanks to the omnidirectional characteristic of MEMS capsules, it allows the ro-bat to detect sound sources behind it by exploiting the front-back spatial ambiguity of linear arrays (as described by Tashev in Section 5.2.3 *Spatial Aliasing and Ambiguity* in [38]), thus enhancing space awareness.

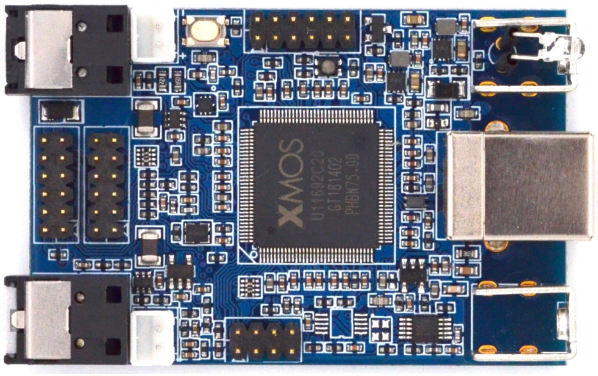


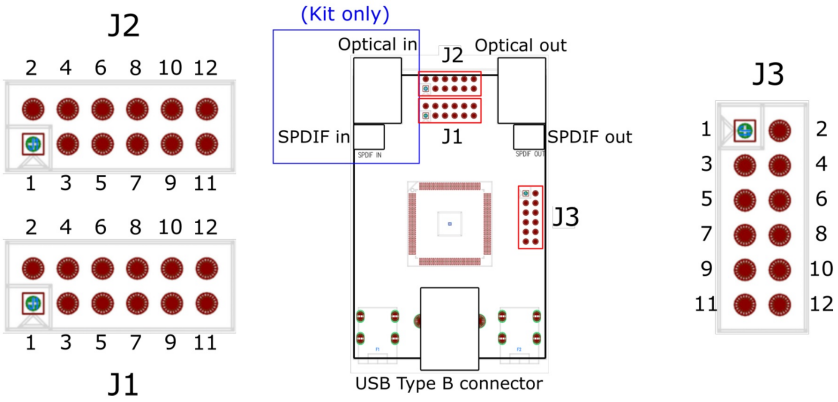
Figure 3.12: MiniDSP MCHStreamer Kit multichannel sound card.

3.1.5. MCHStreamer Kit

The core of the audio signal processing is the sound card, which manages all audio input and output for the ro-bat, reducing the load on the Raspberry Pi. To synchronise and manage the multiple MEMS microphones of the array, I selected the multichannel sound card MCH Streamer Kit [26] from MiniDSP (Figure 3.12), which was the ideal solution for this purpose. This audio interface supports multichannel audio in formats such as TOSLINK, ADAT, S/PDIF (coaxial), I2S, TDM, DSD, and PDM, accessible from pin headers *J1*, *J2*, *J3* and optical connectors, as shown in Figure 3.13. Data is managed by an XMOS XCore 200 processor via USB 2.0 full-speed connection, which is also powering it. The sound card is compliant with USB Audio Class 2.0 (UAC2) and allows for easy control via PC, Mac, Linux, iOS, and Android. It also generates power and provides clock synchronisation for the MEMS microphones during acquisition. The MCH Streamer Kit can be easily controlled through firmware provided by miniDSP, which varies depending on the data format. The default *AllRate* firmware is used for I^2S MEMS, while the *PDM* firmware is used for PDM MEMS. Moreover, it supports various sampling rates ranging from 8 kHz to 384 kHz, making it suitable for ultrasonic applications. Lastly, the selling price is very competitive and much cheaper than high-end sound cards with similar performance, making it perfect to be used in swarms, also thanks to its compact size of 13 x 40 x 62 mm.

3.2. Software

The programming language chosen for this project is Python. As previously mentioned, Python offers a high degree of flexibility and compatibility across various platforms, combined with ease of use. This is particularly beneficial for managing swarms, as it allows the use of multiple types of mobile platforms and potentially even aerial ones in the fu-



J1 (I2S mode)	
Pin	**Description**
1	I2S data OUT Ch 1&2
2	I2S data IN Ch 1&2
3	I2S data OUT Ch 3&4
4	I2S data IN Ch 3&4
5	I2S data OUT Ch 5&6
6	I2S data IN Ch 5&6
7	I2S data OUT Ch 7&8
8	I2S data IN Ch 7&8
9	Master clock (MCLK OUT)
10	Bit clock out (BCLK)
11	Ground (GND)
12	I2S frame sync (LRCLK)

J2	
Pin	**Description**
1	Ground (GND)
2	NC
3	Ground (GND)
4	NC
5	NC
6	GPIO (future)
7	GPIO (future)
8	RST (negative low)
9	GPIO (future)
10	GPIO (future)
11	Ground (GND)
12	5V external power

J3	
Pin	**Description**
1	GND
2	3.3V
3	PDM 1-2
4	PDM 3-4
5	PDM 5-6
6	PDM 7-8
7	PDM 9-10
8	PDM 11-12
9	PDM 13-14
10	PDM 15-16
11	PDM CLK
12	PDM CLK2

Figure 3.13: MiniDSP MCHStreamer Kit board layout and pinouts. Image taken from [27].

ture. Furthermore, Python provides a range of specialised libraries for managing and processing audio signals, including pre-implemented algorithms that can accelerate the testing process and facilitate comparisons between different algorithms. In particular, I used Python version 3.12.4, with a Conda environment set up on the Raspberry Pi. The Raspberry Pi's operating system is installed on a 32 GB SD card via the Raspberry Pi Imager software. The installed version is: `Linux raspberrypi 6.6.28+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.6.28-1+rpt1 (2024-04-22) aarch64 GNU/Linux`. This OS is configured to connect to the Cyber-Physical-Systems Wi-Fi network, with SSH (Secure Shell protocol) and VNC (Virtual Network Computing) options enabled for remote access. The most relevant libraries installed include:

- **Thymiodirect** [7]: It provides connection and control for the Thymio II robot via USB, allowing for reading sensor values and issuing commands to control the wheels and LEDs.
- **Sounddevice** [5](version 0.4.6): It manages sound card and audio streams for input and output, enabling full control of MiniDSP MCHStreamer card parameters, such as the sampling frequency for acquisition and playback, directly from Python.
- **Pyroomacoustics** [4] (version 0.3.1): It offers algorithms and tools for room impulse response estimation and acoustic signal processing in simulated environments. It can also be used for beamforming and source localisation algorithms in real-time applications, with some speed limitations. An older version of the library (0.3.1) was preferred instead of the latest release to ensure compatibility with the Raspberry Pi 4B.
- **Soundfile** [6] (version 0.12.1): It enables easy read and write of sound files in different sampled formats, compatible across platforms such as Windows, OS X, and Unix. It is used to save and retrieve audio recordings from array microphones for post-processing.

3.3. Implementation

While the long-term project goal is to have a swarm of echolocating robots capable of collision-free movement in cluttered environments, the short-term goal of my thesis project is to design and build a first ro-bat prototype capable of navigating a controlled environment while avoiding obstacles that emit sound. Hence, each ro-bat must be able to emit ultrasonic pulses to process the reflections and map the environment (active echolocation) as well as locate the source of other robots emitting their sound pulses (passive sound lo-

calisation). The ro-bat V1 that I designed and implemented in my thesis is only equipped with passive sound-based localisation, that means the ro-bat listens to external sounds rather than actively emitting signals and measuring echos. There are several reasons for this design choice. Primarily, the limited timeframe, combined with restricted access to the necessary components for testing, made it difficult to implement active echolocation. Additionally, the ro-bat required an initial setup period in the lab before it could be tested effectively. This passive approach, however, serves as a strong foundation for future development of the system. It also provides valuable insight into the potential challenges and solutions for working with ultrasonic signals in similar contexts. The later stage will focus on implementing a fully echolocating robot capable of both the active and the passive components.

The implementation of the algorithms for Direction of Arrival (DOA) estimation is now explained, along with an explanation of the underlying assumptions. The implementation is divided into three main steps:

- **Data Input Buffer**: This step involves collecting and storing the incoming audio data in a buffer, preparing it for analysis. Proper handling of the data buffer is crucial for ensuring real-time processing and accurate DOA estimation.
- **DOA Computation**: Here, the algorithm processes the buffered audio data to estimate the direction from which the sound originates. Various signal processing algorithms are employed and compared allowing to give a better idea of the system's capabilities.
- **Navigation and Avoidance**: Based on the computed DOA, the ro-bat uses navigation algorithms to make decisions on movement, adapting its path to avoid sound-emitting obstacles. This step involves real-time adjustments to the ro-bat's trajectory to ensure safe navigation through the environment.

3.3.1. Data input buffer

First, an explanation of the data management system is essential, as this is a crucial component for the system's functionality. Audio data is acquired from the MCHStreamer Kit sound card and transferred via USB to the Raspberry Pi, where it is processed by the algorithm. The Sounddevice Python library facilitates the transfer of audio buffers from the sound card to the Raspberry Pi, making it easy to manage these buffers directly in the Python code. Since the MCHStreamer is natively compatible with ALSA (Advanced Linux Sound Architecture) drivers, no additional installation of software is required for the Raspberry Pi to recognise the sound card and interface with the library.

3.3.1.1. Sampling rate choice

A brief explanation is needed regarding the choice of sampling rate. While the system can operate at much higher sampling rates, only up to 96 kHz has been tested due to equipment limitations, the rate was reduced to 8, 16, 32 kHz. This adjustment was made to lower the computational load on the Raspberry Pi, as higher sampling rates would involve processing and storing significantly more data. The selected rates represent a balance between computational efficiency and algorithm reliability, without compromising performance, considering that the output from the Thymio robot's speakers is limited at 8 kHz.

3.3.1.2. Buffer management and responsiveness

Additionally, the input buffer size has been limited in the tests to 1024 samples, which is the amount of data that must be processed before the next batch arrives. Both the sampling rate and buffer size were optimised in the lab to ensure the ro-bat's rapid response to external sound stimuli and to enhance obstacle avoidance effectiveness.

3.3.1.3. Data streaming and storage

The audio stream is initialised to continuously read input data buffers from the microphone using a callback function. This function also temporarily copies each buffer to facilitate later storage. Every 30 seconds, the buffered data is saved to the Raspberry Pi's local storage as multi-track WAV (Waveform Audio File Format) files, which can later be downloaded for analysis. The recording duration was limited to 30 seconds to reduce saving times; at the 16 kHz sample rate, this duration allows for a quick save time of approximately 0.5 seconds.
This data management setup ensures an efficient handling of audio data, real-time processing, and minimisation of delays in the system leading to smooth and responsive robot navigation.

3.3.2. DOA computation

Each buffer of data enters the Direction of Arrival (DOA) algorithm to provide an angular estimate of the sound source, which then guides the ro-bat's movement. Three different algorithms have been implemented in the code and can be selected while keeping other conditions constant. These algorithms are:

1. **GCC-PHAT**: As previously explained in Section 2.1.4.2, this is the most basic

algorithm implemented in the ro-bat, providing reliable real-time behavior. Its implementation is divided into two main steps: the first is calculating the Time Difference of Arrival (TDOA), and the second is performing PHAT weighting and cross correlation to retrieve an angular estimate.
Specifically, for an array of *N* microphones, one is selected as reference and its signal is cross-correlated with each of the other *N-1* microphones. Each pair calculates a time delay for the incident wave between the two, which is then converted into an angular value relative to the axis perpendicular to the array.
Some comments needs to be made here about the reference's choice. Among the prototypes described in Section 3.1, there are even and odd number of microphones. When the array is odd, the central microphone is chosen as the reference, instead with an even number, the reference is typically located furthest to the left or right. In the latter condition, this means that in a small mobile platform, like the Thymio, the direction of movement may not align perfectly with the array's central axis, which cause navigation and avoidance to be biased towards one side with respect to the other.
Another important assumption regards the distance of the sound source from the array. The implementation assumes a theoretical far-field condition as already mentioned in Section 2.1, which is generally valid for the experiment, but it also compensates for near-field cases by averaging the angular values computed from different microphone pairs. This approach helps reducing the inevitable errors that occur when the source is situated close to the array by compensating large and small contributions virtually reducing the array size and resulting in a more accurate angular estimation. However, rather than being a limitation, this technique also improves stability and reliability when the source is in the far-field, as it mitigates errors generated by the algorithm during cross correlation. The trade-off to accept is a reduced angular resolution for the array, as averaging tends to bias the angular estimates towards the frontal direction rather than to the right or left of the ro-bat.

2. **SRP-PHAT**: The implementation of this algorithm utilises the Pyroomacoustics library, specifically the `pyroomacoustics.doa.srp` class, which provides an effective structure for DOA estimation using Steered Response Power with Phase Transform (SRP-PHAT). Here, differently from the previous implementation of GCC-PHAT, the input buffer is processed by the algorithm directly yielding the location of the sound source as output, without any further step.
However, this approach results in a noticeable increase in computation time, rising from approximately 0.007 seconds in the previous method to 0.075 seconds, with

a sampling rate of 8 kHz. This increase is primarily due to the greater complexity of the computational operations involved and the non-optimal performance of the Pyroomacoustics library when used for real-time applications. Despite this, SRP-PHAT offers significant advantages: One major benefit is the general robustness of this algorithm compared to GCC-PHAT, as it can theoretically provide more reliable localisation even in challenging conditions. Additionally, SRP-PHAT offers an increased angular aperture, allowing for a wider range of sound source angles to be detected with higher accuracy.

3. **MUSIC**: As previously discussed for GCC-PHAT, a similar approach is used here to estimate the DOA. The `pyroomacoustics.doa.music` class from Pyroomacoustics library is employed to derive an estimate of the obstacle's location from the audio input buffer. However, in this case, the time required by the Raspberry Pi to compute the DOA increases further, from approximately 0.075 seconds in the SRP-PHAT method to around 0.2 seconds, with a sampling rate of 8 kHz. This increase in computation time is attributed to the algorithm's more complex structure and the Pyroomacoustics library's suboptimal performance in real-time applications. Nevertheless, several points set this algorithm apart from the previously analysed methods. The MUSIC algorithm offers higher theoretical accuracy due to its sharper and more focused detection of the angular position, making it highly effective for pinpointing the sound sources' direction with great precision, combined with the possiblity of being used also for detecting multiple sound sources at the same time.

3.3.3. Navigation and avoidance

The missing piece for testing the algorithms on the ro-bat is establishing data transfer with the Thymio for navigation. This step depends on the prior DOA computation, as it requires the estimated angle input from the algorithm to determine the ro-bat's next movement. The main strategies to control and move the ro-bat in the environment are explained by the following sections.

3.3.3.1. Decibel threshold and sensitivity control

Initially, a decibel threshold relative to the input signal is calculated to control when angle computation occurs, allowing it only if a -50 dB RMS value is exceeded. Although set specifically for lab conditions, this sound intensity effectively determines the distance at which the ro-bat can detect and avoid obstacles, reducing the processing load on the Raspberry Pi by limiting the computation to specific moments. Adjusting this threshold

modifies the ro-bat's sensitivity to its environment: a lower threshold for example increases distance perception, but may introduce noise from sources like the Thymio's wheels, which could interfere with accurate angle calculation. The minimum threshold is therefore constrained by the microphones' intrinsic noise level and any noise generated by the robot itself.

3.3.3.2. Angle-based navigation

Once the threshold conditions are met, the calculated angle feeds into the navigation algorithm, which divides the space into directional sectors to instruct the ro-bat to move opposite to the sound source. For instance, if an obstacle is detected 30 degrees to the right, the ro-bat will rotate left, increasing the right wheel's speed in proportion to the angle: the greater the angle, the sharper the turn. This operation is segmented into discrete steps for simplicity and responsiveness, with adjustments occurring in ranges of 1 to 5 degrees, 5 to 30 degrees, and 30 to 90 degrees for both left and right directions.
The zero-angle case represents a special scenario. When the threshold is passed and the zero angle is detected, the ro-bat slows down to approximately one fifth of its initial speed. This allows it to approach the sound source more carefully, giving it time to accurately determine the direction of arrival and which way to turn. If the sound source is positioned at exactly zero degrees, the ro-bat will move towards it until it bumps into the obstacle. To avoid this an escape strategy has to be used: if a critical threshold of -45 dB RMS of the input signal is surpassed, the ro-bat randomly decides to turn either right or left and move away from the sound source.

3.3.3.3. General movement and exploration pattern

The final point to address is the general movement of the ro-bat. To demonstrate sound localisation behaviour, we let the ro-bat move in the environment avoiding collisions. The environmental arena is delimited by white tape. In each experiment, the ro-bat is programmed to move in a straight line at a constant speed and continuously reads the ground sensor values. When the ro-bat detects a white line on the floor, it performs a reverse manoeuvre. In a small arena, this results in a random, well-distributed movement pattern, allowing the ro-bat to explore the entire area effectively while avoiding obstacles as they are encountered.

4 | Tests and results

In this chapter I present the experimental results performed with the ro-bat in the labs. First, I introduce how I decided to structure the experiments in Section 4.1, followed by the actual presentation of the quantitive results in Section 4.2.

4.1. Evaluation tests of the performance

In this Section I explain the testing procedures used to evaluate the performance of the ro-bat. In Subsection 4.1.1 I introduce the setup and conditions under which the testing was conducted, followed by the procedure used to test and compare the individual algorithms in Subsection 4.1.2. Lastly, I explain how I managed the data processing and analysis to obtain the results in Subsection 4.1.3.

4.1.1. Experimental setup

All tests were conducted at the University of Konstanz, specifically within two different labs dedicated to swarm robotics. The first, the SwarmLab, is a room of about $5 \times 5 \times 3$ metres, not acoustically treated, but quiet, except for the noise generated by the robots during the experiment. The second is a larger lab, approximately $7 \times 7 \times 4$ metres, featuring a $3 \times 3 \times 0.5$ metres deep arena in the centre, dedicated to robotic experiments. The second room was chosen to run the experiment presented Section 4.2 thanks to the larger dedicated space and the better filming possibilities, even though it is also not acoustically treated and has a higher level of background noise due to fans in electronic equipment and a ventilation system in the room. The setup shown in Figure 4.1 is equipped with the following materials:

- **Arena**: It is created by a dark carpet on the floor with white lines delimiting a rectangular shape on it. This solution is the best to perform testing since it provides a good grip for the ro-bat wheels and the dark floor helps to better identify the black and white markers during the post processing, eliminating possible reflections of the ambient light that can disturb the video recording.

Figure 4.1: Setup used for the experiments in the lab where the arena is created by using white tape on the dark floor. I positioned additional tape on each side to allow the ro-bat to occupy also the middle part of the arena instead of moving only around the external parts.

- **Ro-bat**: The components I described in 3.1, which form the ro-bat, are assembled for testing. Specifically, the configuration includes a microphone array (version V1) with seven PDM MEMS microphones, the MCHStreamer Kit sound card, and a Raspberry Pi 4B. These components are arranged on top of the mobile platform, with the array positioned to face the front of the ro-bat. The tests are done with only one ro-bat.

- **Sound obstacles**: I used a variable number (between 2 and 5) of Thymio robots as static sound obstacles, positioned randomly in the arena and continuously emitting the same 30-minutes white noise from their onboard loudspeakers.
  I now explain in more detail why I chose this setup, specifically the use of noise as the output signal. This decision was influenced by both the characteristics of the experimental lab setup and the particular requirements of calculating the direction of arrival (DOA). Noise provides the best solution in terms of uniformity of obstacles' emission, combined with a unique localisation signature. This setup ensures that each obstacle is perceived similarly in terms of loudness, maintaining experimental uniformity around the arena while enabling differentiation in direction of arrival (DOA) detection since each source is detected as distinct. This effect cannot be achieved with simple sine waves, which would create issues with cross correlation and introduce more errors due to reflection and overlapping signals. Additionally, using noise allows for flexibility in adjusting the sampling rate without requiring any changes to the signal type, which would otherwise be necessary if using fixed-frequency sine waves.
  The choice to use static obstacles instead of dynamic ones is related to the task to perform. Having moving obstacles means that the robots representing the obstacles (which play the noise sound) have to constantly move in the arena randomly. To do so I wrote a code in python to allow the Thymios emitting the noise to perform a continuous random walk. This however is only possible if the moving robots avoid each other by using proximity sensors, because these robots are not equipped with the acoustic system for obstacle avoidance. The major problem with this approach comes from the fact that such dynamic robots prevent testing the limit case in which the ro-bat moves very close to an obstacle because the moving obstacle also avoids collisions. Therefore, this setup does not allow me to analyse failures that can happen in the DOA algorithm, e.g., a ro-bat collision, because the proximity sensors of the other robots prevent this from happening.

- **Overhead camera**: I positioned a camera attached to the ceiling, as shown in Figure 4.1, directed at the arena to capture the ro-bat's movements. The videos are

stored and post-processed to obtain the position and orientation of the ro-bat and of the obstacles, and the distances and relative angular position between the ro-bat and the obstacles.

- **ArUco markers**: Each object in the arena is equipped with a unique marker called ArUco [18], which provides information on orientation and 3D position in space. Since the experiment was performed on a 2D surface, I was not interested in using the Z axis and consequentially all markers were positioned as parallel as possible to the floor.
  The arena's corners are also identified with markers to obtain its exact arena dimensions during the post-processing analyses.
- **External PC**: I used an external laptop to access and run the code on the ro-bat via Wi-Fi, without the need for a wired connection.

4.1.2. Testing procedure

The tests I conducted to evaluate the ro-bat's performance followed the standardised procedure, here explained, across four repetitions, aimed at producing a robust dataset that minimises accidental errors and external factors that could potentially compromise data integrity.

First, I positioned the obstacles in the arena in a random distribution while ensuring they were spread out to uniformly cover the arena. The main objective of this setup was to maximise the number of effective interactions between the ro-bat and the obstacles, allowing for a high rate of avoidance events within a relatively short testing period. I used the same number of sound sources for each repetition, set to five, within an arena measuring 1.4×2.2 metres. I varied the position of the sources between each repetition. To ensure optimal camera recording conditions, I also provided diffuse lighting over the arena, keeping it the same across all experiments.

Once the arena was set up, I began the experiment by first activating all the sound sources. Then, I started the ro-bat with the selected algorithm and the settings outlined in Tab. 4.1. After, I started the video recording and finally I synchronised the various data sources with a clap. I opted for synchronisation through a clap event rather than alternatives, such as using timestamps from the camera and ro-bat, as this was the simplest and most reliable way to ensure precise alignment of the data. Once the experiment was complete, I performed another synchronisation clap and then stopped recording.

Experimental parameter settings

Run number	Parameters	GCC-PHAT	SRP-PHAT	MUSIC
1	Total time [min:sec]	8:27	8:16	8:33
	Input buffer size [samples]	1024	1024	1024
	Sampling frequency [kHz]	16	16	8
2	Total time [min:sec]	8:33	8:19	7:45
	Input buffer size [samples]	1024	1024	1024
	Sampling frequency [kHz]	32	32	8
3	Total time [min:sec]	8:18	8:14	7:12
	Input buffer size [samples]	1024	1024	1024
	Sampling frequency [kHz]	32	32	8
4	Total time [min:sec]	10:26	15:29	4:48
	Input buffer size [samples]	1024	1024	1024
	Sampling frequency [kHz]	32	32	8

Table 4.1: Settings used in the four different runs of the experiment. Time duration is approximately constant around 8 minutes for the first three runs, while varies consistently only in the fourth. The buffer size of data going from the sound card into the DOA algorithm is kept constant at 1024 samples. The sampling frequency used for recording and processing of DOA varies from 8 to 32 kHz across different algorithms.

4.1.3. Post processing data elaboration

In this section, I explain how the raw data from the recordings were processed and modified to prepare them for analysis. Additionally, I outline the steps taken to derive the final results by making use of the computer vision library OpenCV in Section 4.1.3.2.

4.1.3.1. Data post processing

Before performing the actual analysis, I pre-processed the data coming from the camera and the audio recordings captured by the ro-bat. As mentioned in Section 4.1.1, the files were synchronised by manually aligning them at the moment of the clap. This manual alignment was applied across all repetitions in the experiment, allowing me to inspect the raw data for errors early in the process and prevent them from affecting subsequent steps.

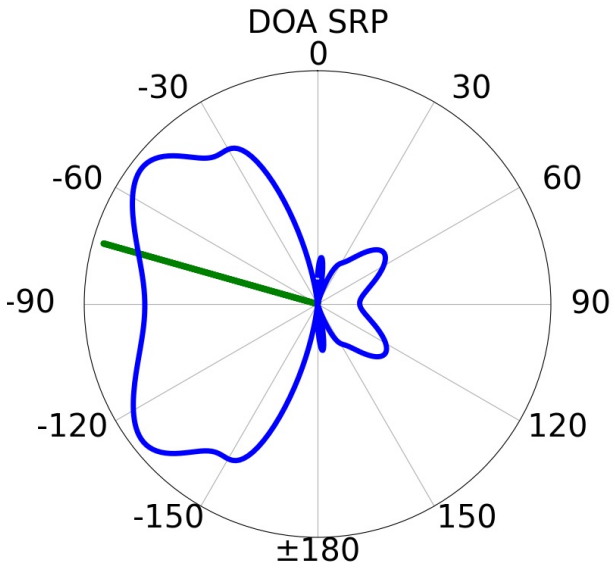


Figure 4.2: Polar plot of the estimated direction of arrival (in blue) and the ground truth angle (in green), obtained from a SRP-PHAT run.

The video footage was captured using a GoPro Hero 7 camera, recording at 24 frames per second with a resolution of 2704×2028 pixels. I specifically chose this quality setting to utilise the *linear* mode provided by this camera, which compensates for the fisheye effect typically caused by the wide field-of-view lens. This resolution was consistently high enough to enable successful recognition and tracking of the ArUco markers. After synchronizing the camera footage with the audio, given the slow movement of the ro-bat across the arena, I decided to export the files with a reduced frame rate of 5 frames per second. This reduction significantly sped up subsequent processing without impacting the analysis results, facilitating data management, storage, and tracking.

4.1.3.2. Data tracking

The data are then imported into a tracking code that analyses the video frames and combines them with the audio recordings from the microphones. To ensure accurate results, the best approach was to recompute the DOA algorithms with the same parameters as those used in the experiment. Specifically, I utilised the opencv-python library [3] (version 4.10.0.84) from OpenCV (Open Source Computer Vision Library) [2] to process each video frame along with an audio buffer. I then applied the same DOA algorithms on this buffer to obtain the estimated angle.

To evaluate the DOA algorithms' performance, I computed a ground truth angle by first calculating the distances between the ro-bat and each obstacle in the arena. I then selected

the closest obstacle as a reference, calculating a ground truth angular position relative to the ro-bat's front. This approach assumes that the closest obstacle generates the loudest sound, making it the primary source detected by the ro-bat. The assumption is valid in most of the cases, however it has some limitation related to certain disposition of the sound sources on the arena. For example, if the robat has a source nearby, but there are two sources positioned further away in another direction, the robat will detect those as they are louder the the closest one. Additionally, as explained in Section 3.1.2, the obstacle, represented by the Thymio II robot, has an asymmetric emission which is louder from its back. This causes another asymmetry with respect to the assumption I made.

This process was made possible by the integration of ArUco markers' detection into OpenCV, as they provide precise information on each object's position and orientation in space. I used selected black and white 6×6 grid markers, generated using OpenCV's *DICT_6X6_250* dictionary.

Simultaneously, I stored the values for the ro-bat's distance from each obstacle, the ground truth angle (i.e., the angle in the ro-bat's reference frame between the ro-bat and the closest obstacle), the estimated angle (by the DOA algorithm), and the difference angle between the estimated and the ground truth angle. This data is stored on a CSV (Comma-separated values) file for further analysis and plotting of the results of Section 4.2. Additionally, I used these data to create polar plots (see one example in Figure 4.2) overlaying the video frame, displaying both the ground truth angle (in green) and the estimated angle (in blue) as a visual reference for the experiment. To enhance clarity, I added a line on the video showing the ro-bat's path throughout the experiment as displayed in Figure 4.3. This overlay provided an easy-to-follow reference of the ro-bat's turns and navigation trajectories during the test, allowing viewers to observe the ro-bat's response to detected obstacles in real time.

4.2. Experimental results and comparison

I now introduce the results by presenting first the single runs in Subsection 4.2.1 and afterwards, the result obtained by combining them into a single dataset in Subsection 4.2.2.

These results have been obtained after numerous testing sessions conducted throughout the entire thesis period. Initially, I performed pre-tests using the sound card and microphone array in a static configuration with a single sound source to verify if the algorithms could operate in real-time conditions. The results were highly promising, as all tested algorithms performed effectively.

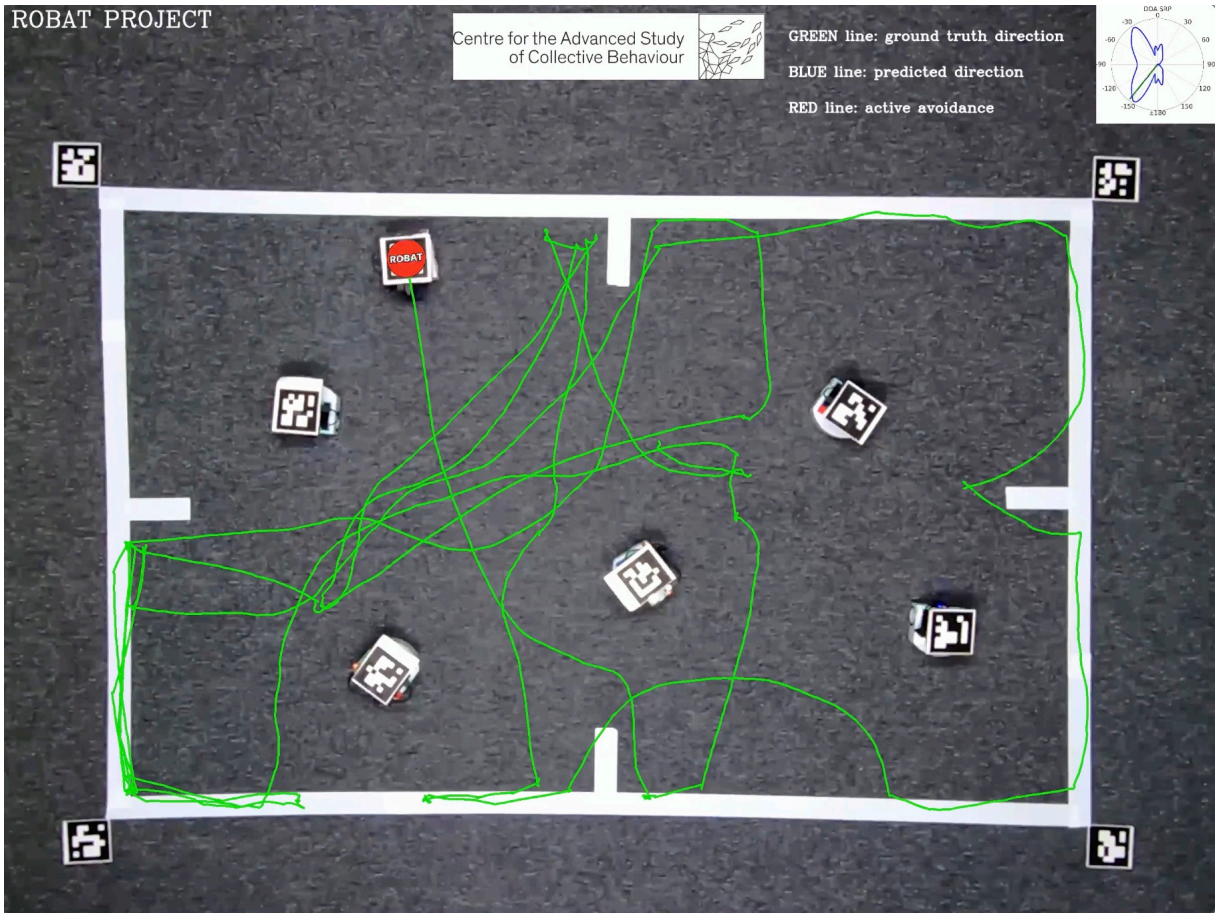


Figure 4.3: Example of one of the experiments being tracked. The green line on the arena shows the trajectories made by the ro-bat. The polar plot in the top-right corner shows in real time the ground truth angle and the direction of arrival computed over 360 degrees by the ro-bat. The video containing all the experimental runs is available at [16].

Next, I conducted preliminary tests with the microphone array mounted on the robot and used the Raspberry Pi as the processing unit. Here, I encountered some speed limitations, as the less powerful platform introduced latency in DOA computations, resulting in slower reaction times to environmental changes. However, the avoidance could still be performed by the ro-bat, so I decided to move along without major changes.

Afterwards, I began evaluating performance in a multi-source setup to advance toward the ultimate goal of simulating the cocktail party problem introduced in Chapter 1. Testing in a multi-source environment was particularly valuable, as it added complexities such as increased background noise levels perceived by the robot due to multiple simultaneous sound sources.

The experiments shown in the following results refer only to the multiple-source case with static obstacles, since, as I mentioned, this is the most relevant configuration in terms of results and, at the same time, the most challenging both for biology and robotics.

4.2.1. Single runs

In this section, I present the results from individual runs, each differing only in the positioning of obstacles inside the arena and the duration of the test. Each run includes tests of the three algorithms, following the procedure outlined in Section 4.1.2, for a total of 12 different conditions, shown in 4.4. The comparison between GCC-PHAT, SRP-PHAT, and MUSIC is valid, as these algorithms were limited to consider only one source, even though the last two are capable of detecting multiple sources. Table 4.1 summarises the settings used during the runs.

I now introduce some preliminary expectations for the results, explaining the thoughts behind the comparisons. I wanted to test several algorithm in the same setup to understand the trade-off required in this type of application. Specifically, as discussed in Section 2.2, many robotics applications in sound source localisation primarily employ basic cross correlation algorithms. These algorithms are often preferred in small robotic systems because they are fast and easier to implement on hardware with limited computational power. However, in this thesis, I try to determine if more advanced algorithms could enhance localisation performance without significantly impacting the robot's speed and responsiveness.

The inclusion of the MUSIC algorithm, in particular, was intended as more of a stress test than a necessary component for this specific application, as I already expected its performance to be lower in terms of processing speed.

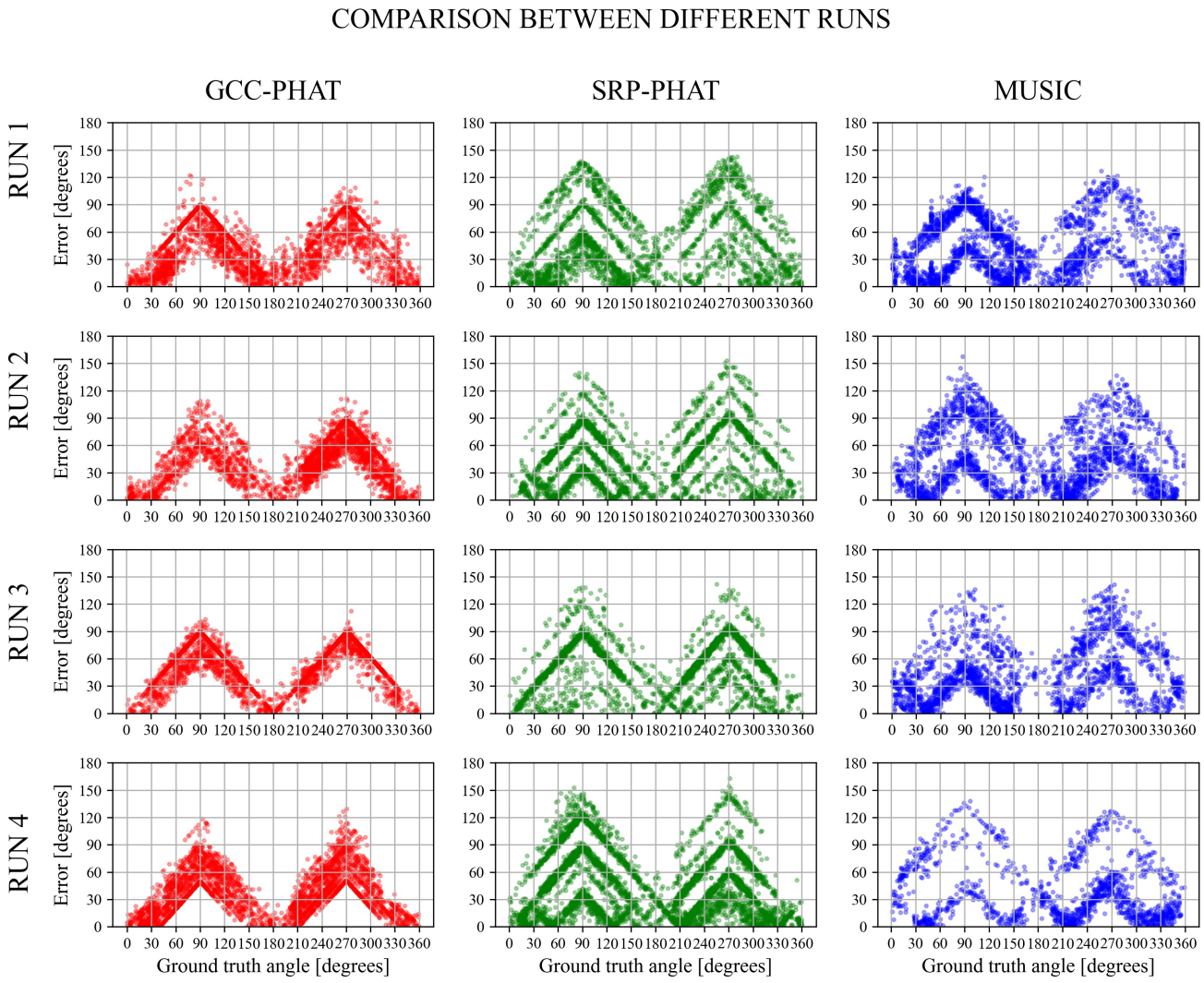


Figure 4.4: Experimental run results, compared across the different algorithms tested and following the settings in Table 4.1.

I now describe the structure of the results displayed in Figure 4.4 in detail. The error distributions from each run are displayed horizontally while the algorithms are ordered in the columns. All the plots presented here show the ground truth angle, tracked in post-processing, on the x-axis. An angle of zero degrees here corresponds to the direction perpendicular to the array or the front of the robot, with angles increasing counter-clockwise up to 360 degrees. This means that the first half of the plots (0 to 180 degrees) represents the left side of the robot, while the second half (180 to 360 degrees) represents the right. The front of the robot is limited by angles between 0 and 90 degrees and between 270 and 360 degrees, and it is situated on the outer edges of the plot. Instead, the rear side corresponds to the central part of the plot (angles between 90 and 270 degrees).

The y-axis represents the absolute value of the difference between the ground truth angular position of the obstacles and the angle detected by the various algorithms, defined here as the error. This error is limited to a maximum of 180 degrees, which would indicate the worst-case scenario for detection.

For the GCC-PHAT algorithm, I observe that the error reaches its maximum at 90 and 270 degrees, with values up to around 120 degrees. This outcome is anticipated because of the linear geometry of the array and the front-facing MEMS microphone arrangement, that are physically limiting the ability to pick up signals from side angles. I instead attribute errors in the upper part of the plots that exceed 90 degrees to the ground truth calculation in the tracking process. When tracking, I select the nearest object to calculate the ground truth of the ro-bat's movement around the arena. However, this position is based on the central position of the ArUco marker, which is not aligned with the actual speaker placement of the Thymio robot. The misalignment biases the loudspeaker's emission, making it more powerful towards the back compared to the front and sides. Additionally, the tracking assumes no interaction of the ro-bat with the obstacle positioned further way. These non-uniformities inside the arena causes significant errors in the calculation of the ground truth value (of the source of the loudest sound), as the asymmetries in the sound field within the arena are not taken into account. Now, imagine for example that the tracking detects an obstacle being geometrically closer to the ro-bat at 60 degrees on the right, while the sound field creates the polar distribution as the one shown in Figure 4.2. This would result in an error of 110 degrees for a short period of time. This particular condition can lead to error values above 90 degrees. Similar considerations, regarding the values over 90 degrees, apply to the second and third columns, which contain the SRP-PHAT and MUSIC algorithms respectively.

Despite the described anomalies and errors, GCC-PHAT has been shown to be able to effectively discriminate sources positioned to the right or left of the ro-bat, with an error

averaging around 15 degrees in the frontal 60 degree sector.

GCC-PHAT, while reliable and fast, performs well only within the frontal sector, with poor accuracy on the sides. SRP-PHAT algorithm achieves lower absolute errors at side angles, as seen in Figure 4.2 in the central column. For instance, at the 90 and 270-degree angular positions, the SRP-PHAT consistently shows a minimum error of approximately 30 degrees, compared to GCC-PHAT's minimum error of around 50 degrees. Similar to GCC-PHAT, SRP-PHAT also exhibits errors exceeding 90 degrees, which I attribute to the same factors discussed earlier. The error distribution, however, is broader, reaching up to 150 degrees, which noticeably impacts the overall algorithm's performance, even if the single error could be lower than GCC-PHAT. A unique feature of SRP-PHAT is the linear distribution pattern seen in the plot. I interpret this pattern as indicative of poor angular resolution, estimated to be approximately 30 degrees. This suggests that for each ground truth angle, the errors tend to cluster roughly 30 degrees apart from one another.

Similarly to SRP-PHAT, MUSIC consistently suffer from estimation errors distributed over 90 degrees and again up to around 150 degrees. Here, differently from the SRP-PHAT the distribution splits up in two main regions, being separated again around 30 degrees apart, but having error values that are more sparse distributed inside the single regions.

In both the SRP-PHAT and MUSIC results, I attribute the high variability of error values exceeding 90 degrees to the same factors mentioned for GCC-PHAT. This variability, combined with the discrete nature of the error distribution, results in an overall wider and less uniform error distribution, which consequently reduces the reliability of these methods. Additionally, in the case of MUSIC, the lower sampling rate of 8 kHz limits the array's ability to capture some directional information, further exacerbating this issue.

Another notable observation involves the distribution of data points in the left and right sections of the plots. The specific placement of obstacles and the path followed during different runs lead to a higher likelihood of encountering obstacles either on the left or right side of the ro-bat.

Additionally, run 4 displays some significant differences compared to the other runs, primarily due to the varied duration of this experiment. Specifically, SRP-PHAT operates for nearly twice as long as in the other runs, while MUSIC has a shorter duration, approximately half as long, as detailed in Table 4.1. This results in fewer data points on the MUSIC error plot and more on the SRP-PHAT plot. However, it is noteworthy that both algorithms continue to perform consistently with the other runs.

4.2.2. Combined results

In this section, I discuss the results presented in Figure 4.5, which combines data from the previous four runs into a single dataset, once again divided by the three algorithms tested. This comprehensive dataset provides a broader view of the ro-bat's performance without bias from specific configurations or experiment durations.

The observations from the individual runs apply here as well, without significant differences. The GCC-PHAT algorithm shows consistent performance with predictable behaviour across all directions, unaffected by environmental conditions or changes in sampling rate. SRP-PHAT and MUSIC instead have a larger and more inconsistent variability of errors, which lead to poorer performance.

Two more graphs are shown to further analyse the error distribution. The first one in Figure 4.6 presents the three algorithms compared in terms of Mean value and Standard deviation across ground truth angles. The second one in Figure 4.7 instead looks only at the error distribution across algorithms, without referring to the ground truth angles.

In Figure 4.6 the mean error follows an expected distribution, similar to the one seen in Figure 4.5. The mean has the largest peak at 90 and 270 degrees an the minimum values are positioned at the front and back of the robot, which correspond to 0 an 180 degrees. In these points, the GCC-PHAT shows the best performance among the three, being its error 15 degrees less than MUSIC and 5 to 10 degrees less than SRP-PHAT. The mean error is generally very similar between the algorithms, however the standard deviation varies much more. As discussed previously, the GCC-PHAT is very consistent an uniform, showing the narrowest deviation among the three. In contrast, SRP-PHAT and MUSIC exhibit a much higher standard deviation of about 33 degrees. This outcome aligns with my expectations, as mentioned in Section 4.2.1, where the error in both cases reaches up to 150 degrees.

The results I provided until now seem to show a slightly better overall performance of GCC-PHAT with respect to the other algorithms, also considering the fact that it is about 40 time faster than MUSIC and 16 times faster than SRP-PHAT. However, I can see a different trend looking at Figure 4.7. The mean error is similar across the three algorithms; however, GCC-PHAT shows a more uniform error distribution from 0 to 90 degrees, with only a few values exceeding this range. In contrast, SRP-PHAT and MUSIC have a denser error distribution below 30 degrees, with the highest concentration around 10 degrees. However, these latter algorithms also display a wider spread, with consistently high errors above 90 degrees. Based on these results, I can conclude that there is an improvement in error distribution from the first to the third algorithm, even

though their overall performance across the 0 to 360 degree range in Figure 4.6 appears similar.

Lastly, I describe the collision avoidance performance with the help of Figure 4.8. The objective of the experimental runs I did, was to test the ro-bat and its ability to avoid sound obstacles. The collisions have been manually counted considering only the physical contact between the ro-bat and the Thymios. The percentage is calculated with respect to the total number of avoidances. The GCC-PHAT and SRP-PHAT experienced no collisions during the whole duration of the experiment (respectively 35 and 40 minutes), while MUSIC obtains 45.5% in 27 minutes, which corresponds to 15 collisions over 33 total interactions. This result shows poor performance of the MUSIC algorithm in terms of avoidance, and indicates that its theoretical advantage in spatial estimation of direction of arrival is not beneficial in this context. The high variability in the error distribution, combined with a slow response time due to a long computation, suggests to avoid its use with this implementation and the experimental setup used.

In general, the trade-off between fast computation and angular precise estimation I expected is not clearly visible. However, despite GCC-PHAT being in general faster and more reliable also on the direction of arrival estimation, I can see its limitation in accuracy. SRP-PHAT and MUSIC instead, despite being slower and less reliable, have in general shown a slightly better estimation capability, providing a better starting point in terms of accuracy if better implemented and adapted to the specific conditions.

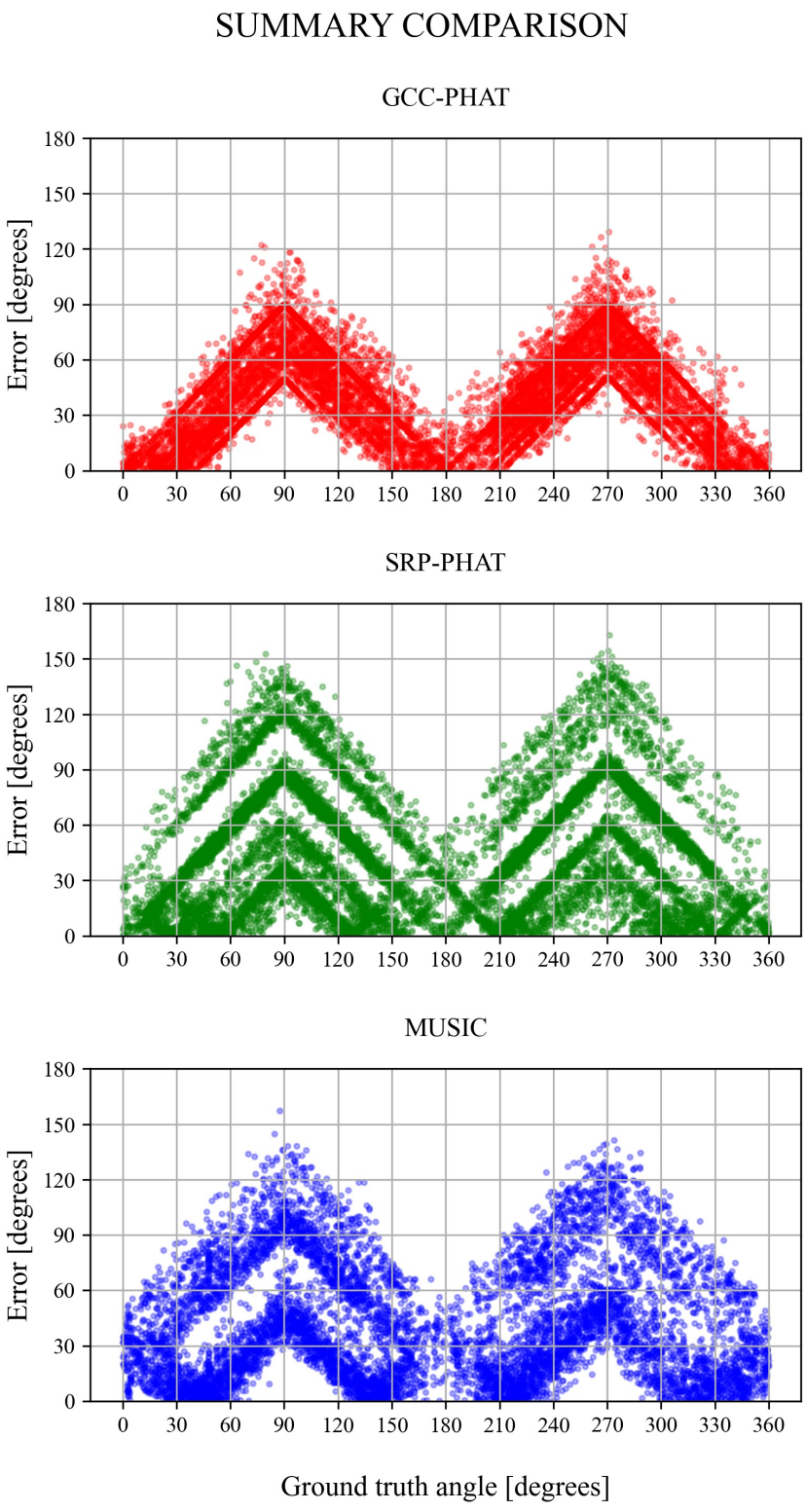


Figure 4.5: Summary and comparison of the experimental results taken from the four runs

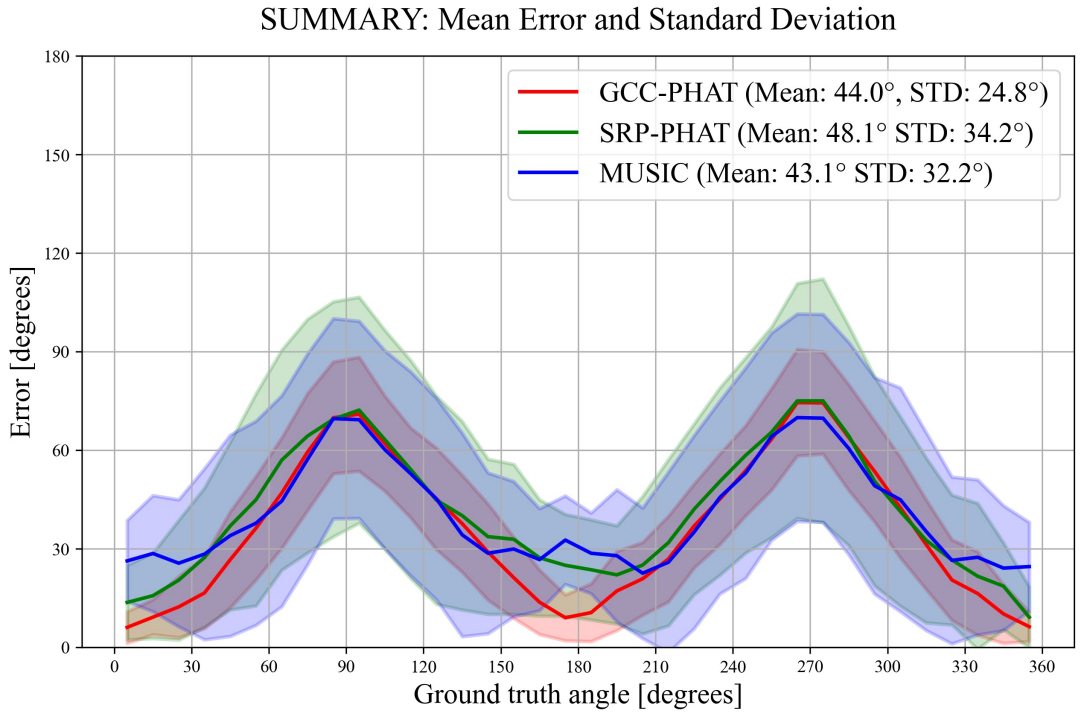


Figure 4.6: Comparison of the distribution of the mean error and standard deviation along the ground truth axis. All algorithms performed similarly on average, but with different spread of the data. GCC-PHAT performs better at 0 (front) and 180 (back) degrees and has a lower standard deviation with respect to SRP-PHAT and MUSIC.

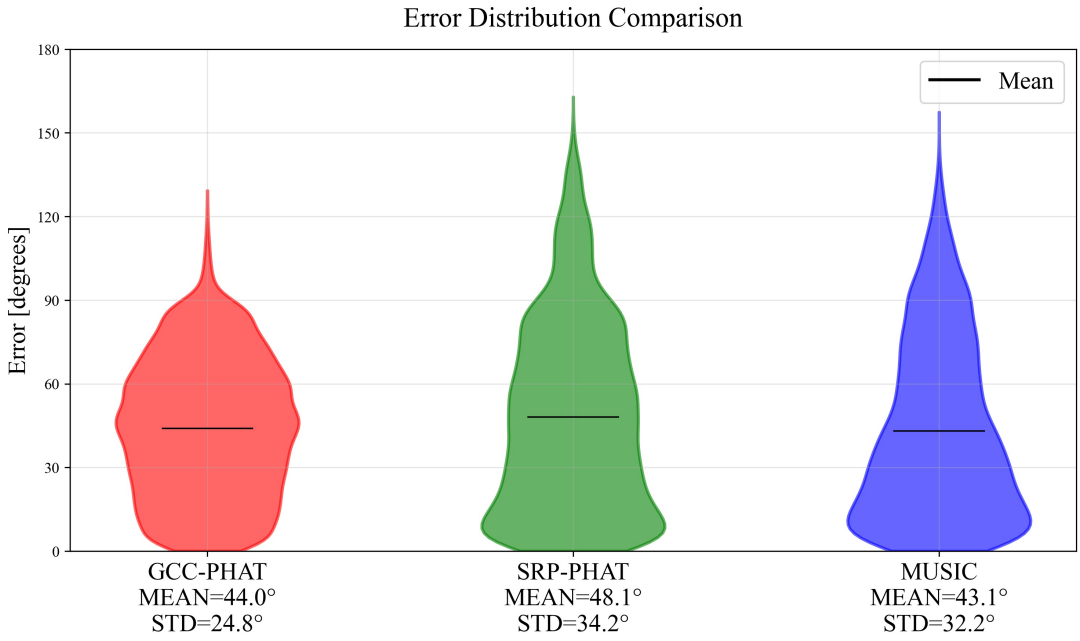


Figure 4.7: Distribution of the angular estimation error compared between different Direction of Arrival (DOA) algorithms. The mean error is almost the same for the three algorithms. The distribution is more uniform in GCC-PHAT (left) from 0 to 90 degrees, compared to SRP-PHAT (centre) and MUSIC (right), that show denser distributions below 30 degrees but larger tails.

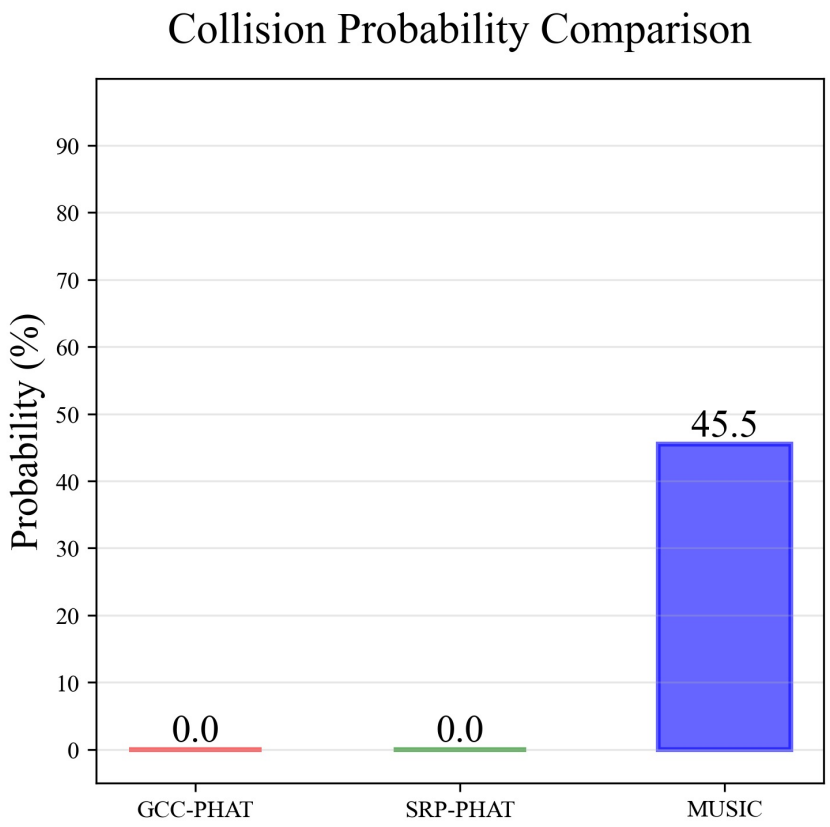


Figure 4.8: Probability of having a collision between the ro-bat and a sound source during the experimental runs.

5 | Conclusions and future developments

In this thesis I equipped a mobile robot with a microphone array to develop a biologically-inspired navigation system for small mobile robots using Direction of Arrival (DOA) estimation. I selected and tested of various hardware components, including MEMS microphones, the MCHStreamer sound card, as well as designed various microphone arrays. I created and tested the various prototypes of microphone arrays to obtain the best and cheapest solution for the application, taking into consideration also of the future possibility of scaling of the number of robots to be equipped with this technology.

I implemented three different algorithms for Direction of Arrival (DOA) estimation, adapting each to function in real-time on small robotic platforms. Additionally, I developed a navigation algorithm for the robot, enabling testing of localisation performance through controlled lab experiments, where I could directly compare each algorithm's effectiveness.

The tested algorithms (GCC-PHAT, SRP-PHAT, and MUSIC) exhibited distinct strengths and limitations. GCC-PHAT demonstrated high processing speed but was limited in directional accuracy, particularly at side angles. SRP-PHAT and MUSIC, while theoretically more accurate, showed greater spatial inconsistency and higher error values, even though their distribution of the error is denser towards low error values. MUSIC, in particular, resulted in the least effective collision avoidance performance, primarily due to its longer computation time compared to the other algorithms.

Overall, the experiment partially confirmed my expectation of a trade-off between speed and accuracy in algorithm selection. A clear trend in processing speed emerged, with GCC-PHAT performing 40 times faster than MUSIC and 16 times faster than SRP-PHAT. However, improvements in accuracy were only partially achieved and did not improve overall performance. Two out of the three algorithms, GCC-PHAT and SRP-PHAT, effectively enabled sound-based obstacle detection and avoidance in real time. Nevertheless, the slower speed of MUSIC prevented the robot's ability to complete the task effectively in this case.

The compact, cost-effective, and flexible design of the ro-bat, combined with the results presented in this thesis, successfully lay the groundwork for bio-inspired, acoustic-based navigation in robotics. The goal of developing an echolocating robot builds upon the existing passive robots, adding active signal emission to enhance navigation capabilities and understanding of the swarm where it is positioned. Specifically, exploring alternative array geometries and optimizing Direction of Arrival (DOA) algorithms for robustness and efficiency could significantly improve performance, potentially enabling the detection of multiple sound sources.

The long-term project in which this thesis is positioned opens up a variety of potential applications for this technology, primarily in robotics, but also in biological studies of collective behaviour.

Starting with robotics, the main advantage of using sound over vision-based sensing lies in its effectiveness in challenging environments with poor visibility, such as low light, direct sunlight, smoke, or fog. In these situations, audio-based navigation could be crucial for completing tasks. A particularly promising application is search and rescue in dangerous, low-visibility, or dark conditions. In such scenarios, the low cost and scalability of a large robotic swarm could justify the potential loss of some units in order to complete the task. Moreover, the passive sound localisation developed in this thesis, and future advancements in active echolocation, could be highly beneficial to swarm robotics research. Sound-based sensing offers a cost-effective, low-power alternative for small robots, enabling decentralised and scalable swarm systems that rely on local communication for coordination.

Biology studies represent another significant area for the application of this project and possible future developments. Sound-based robotic sensing can enable controlled simulations of animal behaviour, providing biologists with valuable insights into behaviour patterns without the need to tag and observe real animals. A key example is the study of the “cocktail party problem” in bats, but this technology could also aid research on understanding a wide range of animals that communicate through sound signals

Bibliography

[1] Icra 40 conference, 2024. URL: https://icra40.ieee.org/icra-2024/program/live-demos/.

[2] Opencv, 2024. URL: https://opencv.org/.

[3] opencv-python, 2024. URL: https://pypi.org/project/opencv-python/.

[4] pyroomacoustics, 2024. URL: https://pyroomacoustics.readthedocs.io/en/pypi-release/index.html.

[5] sounddevice, 2024. URL: https://python-sounddevice.readthedocs.io/en/0.5.1/.

[6] soundfile, 2024. URL: https://pypi.org/project/soundfile/.

[7] thymiodirect, 2024. URL: https://pypi.org/project/thymiodirect/.

[8] Analog-Devices. Analog and digital mems microphone design considerations, 2013. URL: https://www.analog.com/media/en/technical-documentation/technical-articles/analog-and-digital-mems-microphone-design-considerations-ms-2472.pdf.

[9] D. Bechler, M.S. Schlosser, and Kristian Kroschel. System for robust 3d speaker tracking using microphone array measurements. volume 3, pages 2117 – 2122 vol.3, 01 2004. doi:10.1109/IROS.2004.1389722.

[10] Thejasvi Beleyur and Holger Goerlitz. Modeling active sensing reveals echo detection even in large groups of bats. *Proceedings of the National Academy of Sciences*, 116:201821722, 12 2019. doi:10.1073/pnas.1821722116.

[11] Jacob Benesty, Jingdong Chen, and Yiteng Huang. *Microphone Array Signal Processing*. Springer Science & Business Media, March 2008.

[12] Bitcraze. Crazyflie 2.0, 2024. URL: https://www.bitcraze.io/products/old-products/crazyflie-2-0/.

[13] Michael S. Brandstein and Darren B. Ward. *Microphone Arrays - Signal Processing Techniques and Applications*. Digital Signal Processing. Springer, 2001. doi:10.1007/978-3-662-04619-7.

[14] E. C. Cherry. Some experiments on the recognition of speech, with one and two ears. *Journal of the Acoustical Society of America*, 25:975–979, 1953.

[15] Joseph Hector Dibiase. *A high-accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays*. PhD thesis, Brown University, Rhode Island, August 2000.

[16] Alberto Doimo. Ro-bat project full lab tests footage, 2024. URL: https://youtu.be/FPZS9TciNro.

[17] Itamar Eliakim, Zahi Cohen, Gábor Kósa, and Yossi Yovel. A fully autonomous terrestrial bat-like acoustic robot. *PLOS Computational Biology*, 14:e1006406, 09 2018. doi:10.1371/journal.pcbi.1006406.

[18] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco Madrid-Cuevas, and Manuel Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47:2280–2292, 06 2014. doi:10.1016/j.patcog.2014.01.005.

[19] Jie Huang, Tadawute Supaongprapa, Ikutaka Terakura, Fuming Wang, Noboru Ohnishi, and Noboru Sugie. A model-based sound localization system and its application to robot navigation. *Robotics and Autonomous Systems*, 27(4):199–209, 1999. doi:10.1016/S0921-8890(99)00002-0.

[20] Carlos Ishi, Olivier Chatot, Hiroshi Ishiguro, and Norihiro Hagita. Evaluation of a music-based real-time sound localization of multiple sound sources in real noisy environments. pages 2027 – 2032, 11 2009. doi:10.1109/IROS.2009.5354309.

[21] Robin Kerstens, Dennis Laurijssen, and Jan Steckel. ertis: A fully embedded real time 3d imaging sonar sensor for robotic applications. pages 1438–1443, 05 2019. doi:10.1109/ICRA.2019.8794419.

[22] Thomas Kite. Understanding pdm digital audio, 2012. URL: https://users.ece.utexas.edu/~bevans/courses/rtdsp/lectures/10_Data_Conversion/AP_Understanding_PDM_Digital_Audio.pdf.

[23] Knowles. Sph0645lm4h-b rev b datasheet. URL: https://cdn-shop.adafruit.com/product-files/3421/i2S+Datasheet.PDF.

[24] Hong Liu and Miao Shen. Continuous sound source localization based on microphone array for mobile robots. pages 4332 – 4339, 11 2010. doi:10.1109/IROS.2010.5650170.

[25] Iain Mccowan. Microphone arrays: A tutorial. *Technical Report*, 2001. Queensland University, Australia.

[26] miniDSP. Mchstreamer kit, 2024. URL: https://www.minidsp.com/products/usb-audio-interface/mchstreamer?gad_source=1&gclid=Cj0KCQjwpvK4BhDUARIsADHt9sTqD2SMuApyLqA60UqLwR0fxdXbLk1V_Ifa9ulB65o77h8rbgwWjzUaArHyEALw_wcB.

[27] miniDSP. Mchstreamer kit manual, 2024. URL: https://www.minidsp.com/images/documents/MCHStreamer%20User%20Manual.pdf.

[28] Mobsya, 2024. URL: https://www.thymio.org/it/.

[29] Francesco Mondada, Michael Bonani, Fanny Riedo, Manon Briod, Lea Pereyre, Philippe Retornaz, and Stephane Magnenat. Bringing robotics to formal education: The thymio open-source hardware robot. *IEEE Robotics Automation Magazine*, PP:1–1, 02 2017. doi:10.1109/MRA.2016.2636372.

[30] John C. Murray, Harry R. Erwin, and Stefan Wermter. Robotic sound-source localization and tracking using interaural time difference and cross-correlation. 2004. URL: https://api.semanticscholar.org/CorpusID:18482609.

[31] NXP. I2s bus specification, 2022. URL: https://www.nxp.com/docs/en/user-manual/UM11732.pdf.

[32] Caleb Rascon and Ivan Meza. Localization of sound sources in robotics: A review. *Robotics and Autonomous Systems*, 96:184–210, 08 2017. doi:10.1016/j.robot.2017.07.011.

[33] RaspberryPi, 2024. URL: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/.

[34] RME. Fireface uc. URL: https://rme-audio.de/fireface-uc.html.

[35] Jan Steckel and Herbert Peremans. Batslam: Simultaneous localization and mapping using biomimetic sonar. *PloS one*, 8:e54076, 01 2013. doi:10.1371/journal.pone.0054076.

[36] Rainer Stiefelhagen, Hazım Ekenel, Christian Fugen, Petra Gieselmann, Hartwig Holzapfel, Florian Kraft, Kai Nickel, Michael Voit, and Alex Waibel. Enabling mul-

timodal human–robot interaction for the karlsruhe humanoid robot. *Robotics, IEEE Transactions on*, 23:840 – 851, 11 2007. doi:10.1109/TRO.2007.907484.

[37] STMicroelectronics. Mp34dt01-m, 2024. URL: https://cdn-learn.adafruit.com/assets/assets/000/049/977/original/MP34DT01-M.pdf.

[38] Ivan Tashev. *Sound Capture and Processing: Practical Approaches*, chapter 5, pages 178–181. John Wiley Sons, 2009.

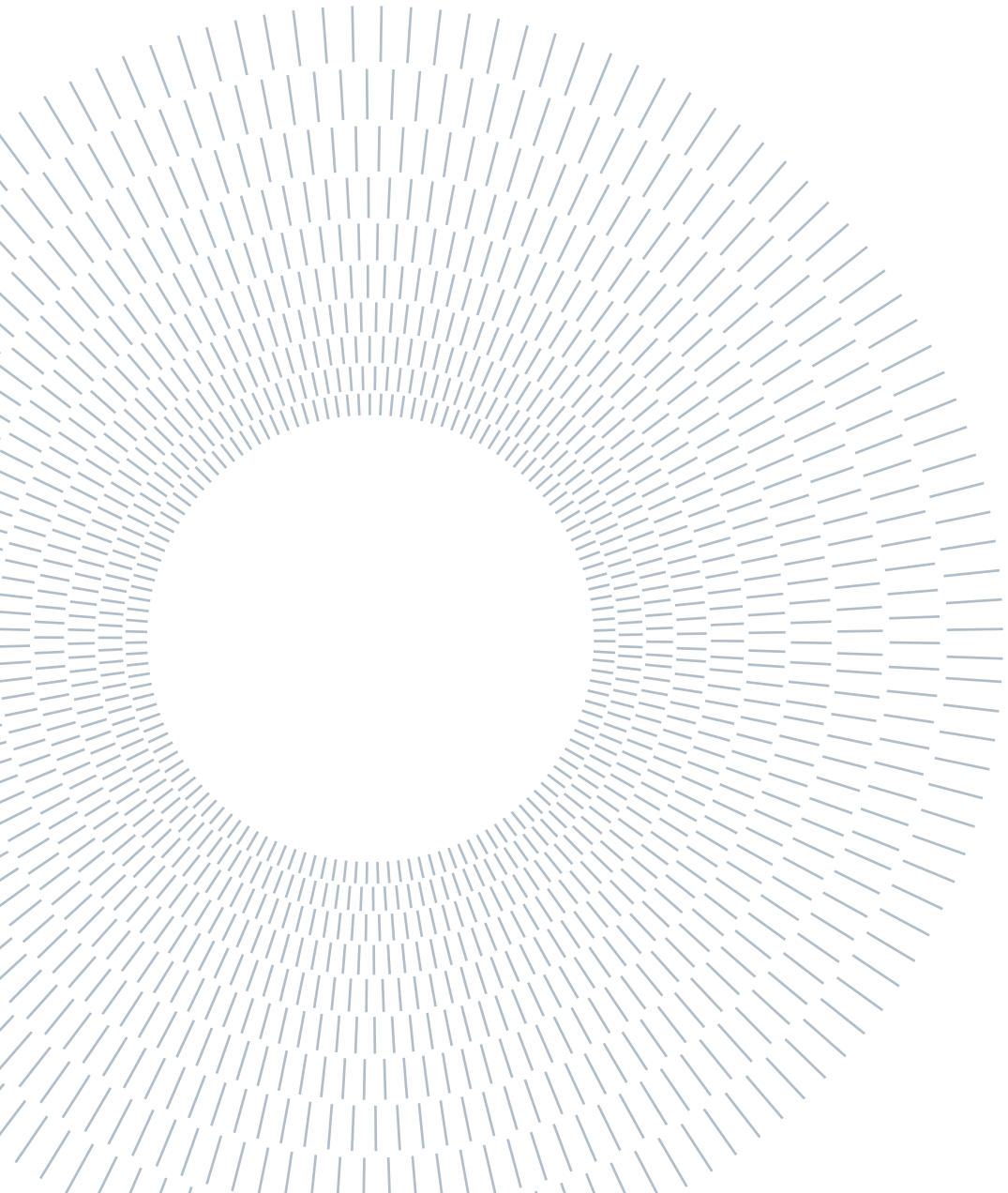
[39] Ivan Jelev Tashev. *Sound capture and processing: practical approaches*. Wiley, Chichester, 2009.

[40] Nachum Ulanovsky and Cynthia Moss. What the bat's voice tells the bat's brain. *Proceedings of the National Academy of Sciences of the United States of America*, 105:8491–8, 07 2008. doi:10.1073/pnas.0703550105.

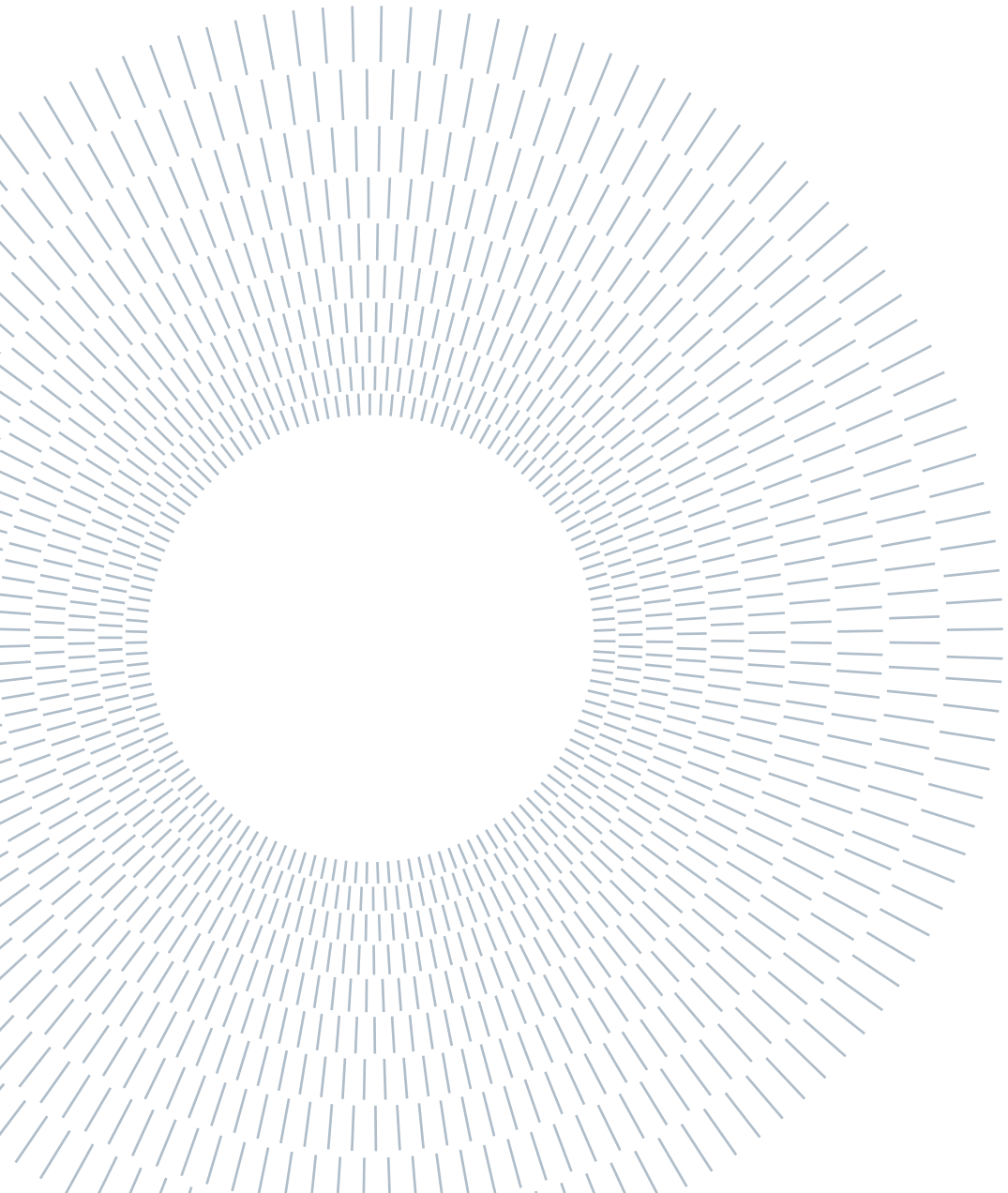
[41] Jean-Marc Valin, François Michaud, Brahim Hadjou, and Jean Rouat. Localization of simultaneous moving sound sources for mobile robot using a frequency-domain steered beamformer approach. volume 2004, pages 1033–1038, 01 2004. doi:10.1109/ROBOT.2004.1307286.

[42] Roei Zigelman, Ofri Eitan, Omer Mazar, Anthony Weiss, and Yossi Yovel. A biomimetic miniature drone for real-time audio based short-range tracking. *PLOS Computational Biology*, 18:e1009936, 03 2022. doi:10.1371/journal.pcbi.1009936.

List of Figures



List of Tables



List of Symbols

Variable	Description	SI unit
$\mathbf{A}$	Steering matrix	
c_0	Sound speed in air	m/s
d	Inter-microphone distance	m
Δ_m	m-th steering delay	
f	Frequency	Hz
f_{max}	Microphone array theoretical spatial aliasing high frequency	Hz
f_{min}	Microphone array theoretical spatial aliasing low frequency	Hz
f_s	Sampling frequency	Hz
ψ	Weighting function	
i	Imaginary unit	
λ	Wavelength	m
l	Microphone array capsules distance	m
L	Major microphone array capsules distance	m
Λ	Diagonal matrix of eigenvalues	
R	Cross correlation function	
$\mathbf{R}$	Covariance matrix	
ω	Angular frequency	rad/s
ω_c	Narrow-band angular frequency	rad/s
ω_s	Spatial angular frequency	rad/s
τ	Time delay	s
θ	Angle of impinging sound waves	rad
$\mathbf{V}$	Noise matrix	

Variable	Description	SI unit
I^2S	Inter-IC Sound	
ADC	Analog to Digital Conversion	
ASIC	Application Specific Integrated Circuit	
BCLK	Bit clock	
DAC	Digital to Analog Conversion	
DAS	Delay and Sum	
DOA	Direction of Arrival	
GCC	Generalised Cross Correlation	
LRCLK	Left right clock	
MCLK	Master Clock	
MEMS	Micro-Electro-Mechanical Systems	
MUSIC	Multiple Signal Classification	
PCM	Pulse Code Modulation	
PHAT	Phase Transform	
PDM	Pulse Density Modulation	
SRP	Steered Response Power	
TDOA	Time Difference Of Arrival	