TESIS DE LA UNIVERSIDAD DE ZARAGOZA

2023 114

David Díaz-Guerra Aparicio

A Geometric Deep Learning Approach to Sound Source Localization and Tracking

Director/es
Beltrán Blázquez, José Ramón
Miguel Artiaga, Antonio

ISSN 2254-7606

Prensas de la Universidad
Universidad Zaragoza

ISSN 2254-7606

Universidad Zaragoza
1542

Tesis Doctoral

A GEOMETRIC DEEP LEARNING APPROACH TO SOUND SOURCE LOCALIZATION AND TRACKING

Autor

David Díaz-Guerra Aparicio

Director/es

Beltrán Blázquez, José Ramón
Miguel Artiaga, Antonio

UNIVERSIDAD DE ZARAGOZA
Escuela de Doctorado

Programa de Doctorado en Ingeniería Electrónica

2023

Universidad Zaragoza
1542

A Geometric Deep Learning Approach to Sound Source Localization and Tracking

Author:
David Díaz-Guerra Aparicio

Thesis supervisors:
Dr. José Ramón Beltrán Blázquez
Dr. Antonio Miguel Artiaga

Department of Electronic Engineering and Communications
University of Zaragoza

A thesis submitted for the degree of
Doctor of Philosophy (PhD)

January 2023

Acknowledgements

First and foremost, I cannot begin to express my thanks to my thesis supervisors, Dr. José Ramón Beltrán and Dr. Antonio Miguel Artiaga, for always letting me conduct my research through the paths I chose but, at the same time, always being there when I needed any kind support or advice. This endeavor would not have been possible without them. I would also like to recognize all the assistance received over these years from the staff of the Department of Electronic Engineering and Communications.

I am also really grateful to all the people from the Audio Research Group at Tampere University, who made me feel at home during my stay there, and especially to Dr. Tuomas Virtanen and Dr. Archontis Politis, who supervised my work during it.

I should express my gratitude to the Regional Government of Aragon who has funded this work with a grant for postgraduate research contracts, but not to the government of the same institution that did not open a call for this kind of grant in 2015 originating a delay that caused that the funding of this work started one year later than it should have done. I should also express my gratitude to my family for funding the first year of this thesis.

Finally, I want to thank all the people who have fought, fight, and will fight for quality public education and for the labor rights and dignity of those who work in the research sector, especially to my mates from *Estudiantes en Movimiento* (previously known as *CEPA*) and *PIF Unizar*.

Abstract

The localization and tracking of sound sources using microphone arrays is a problem that, even if it has attracted attention from the signal processing research community for decades, remains open. In recent years, deep learning models have surpassed the state-of-the-art that had been established by classic signal processing techniques, but these models still struggle with handling rooms with strong reverberations or tracking multiple sources that dynamically appear and disappear, especially when we cannot apply any criteria to classify or order them. In this thesis, we follow the ideas of the Geometric Deep Learning framework to propose new models and techniques that mean an advance of the state-of-the-art in the aforementioned scenarios.

As the input of our models, we use acoustic power maps computed using the SRP-PHAT algorithm, a classic signal processing technique that allows us to estimate the acoustic energy received from any direction of the space and, therefore, compute arbitrary-shaped power maps. In addition, we also propose a new technique to analytically cancel a source from the generalized cross-correlations used to compute the SRP-PHAT maps. Based on previous narrowband cancellation techniques, we prove that we can project the cross-correlation functions of the signals captured by a microphone array into a space orthogonal to a given direction by just computing a linear combination of time-shifted versions of the original cross-correlations. The proposed cancellation technique can be used to design iterative multi-source localization systems where, after having found the strongest source in the generalized cross-correlation functions or

in the SRP-PHAT maps, we can cancel it and find new sources that were previously masked by the first source.

Before being able to train deep learning models we need data, which, in the case of following a supervised learning approach, means a dataset of multichannel recordings with the position of the sources accurately labeled. Although there exist some datasets like this, they are not large enough to train a neural network and the acoustic environments they include are not diverse enough. To overcome this lack of real data, we present a technique to simulate acoustic scenes with one or several moving sound sources and, to be able to perform these simulations as they are needed during the training, we present what is, to the best of our knowledge, the first free and open source room acoustics simulation library with GPU acceleration. As we prove in this thesis, the presented library is more than two orders of magnitude faster than other state-of-the-art CPU libraries.

The main idea of the Geometric Deep Learning philosophy is that the models should fit the symmetries (i.e. the invariances and equivariances) of the data and the problem we want to solve. For single-source direction of arrival estimation, the use of SRP-PHAT maps as inputs of our models makes the rotational equivariance of the problem undeniably clear and, after a first approach using 3D convolutional neural networks, we present a model using icosahedral convolutions that approximate the equivariance to the continuous group of spherical rotations by the discrete group of the 60 icosahedral symmetries. We prove that the SRP-PHAT maps are a much more robust input feature than the spectrograms typically used in many state-of-the-art models and that the use of the icosahedral convolutions, combined with a new soft-argmax function that obtains a regression output from the output of the convolutional neural network by interpreting it as a probability distribution and computing its expected value, allows us to dramatically reduce the number of trainable parameters of the models without losing accuracy in their estimations.

When we want to track multiple moving sources and we cannot use any criteria to order or classify them, the problem becomes invariant to the permutations of the estimates, so we cannot directly compare them with the ground truth labels since we cannot expect them to be in the same order. This kind of models has typically been trained using permutation invariant training strategies, but these strategies usually do not penalize the identity switches and the models trained with them do not keep the identity of every source consistent during the tracking. To solve this issue, we propose a new training strategy, which we call sliding permutation invariant training, that is able to optimize all the features that we could expect from a multi-source tracking system: the precision of the direction of arrival estimates, the accuracy of the source detections, and the consistency of the assigned identities.

Finally, we propose a new kind of recursive neural network that, instead of using vectors as their input and their state, uses sets of vectors and is invariant to the permutation of the elements of the input set and equivariant to the permutations of the elements of the state set. We show how this is the behavior that we should expect from a tracking model which takes as inputs the estimates of a multi-source localization model and compare these permutation-invariant recursive neural networks with the conventional gated recurrent units for sound source tracking applications.

Resumen

La localización y el *tracking* de fuentes sonoras mediante agrupaciones de micrófonos es un problema que, pese a llevar décadas siendo estudiado, permanece abierto. En los últimos años, modelos basados en *deep learning* han superado el estado del arte que había sido establecido por las técnicas clásicas de procesado de señal, pero estos modelos todavía presentan problemas para trabajar en espacios con alta reverberación o para realizar el *tracking* de varias fuentes sonoras, especialmente cuando no es posible aplicar ningún criterio para clasificarlas u ordenarlas. En esta tesis, se proponen nuevos modelos que, basados en las ideas del *Geometric Deep Learning*, suponen un avance en el estado del arte para las situaciones mencionadas previamente.

Los modelos propuestos utilizan como entrada mapas de potencia acústica calculados con el algoritmo SRP-PHAT, una técnica clásica de procesado de señal que permite estimar la energía acústica recibida desde cualquier dirección del espacio. Además, también proponemos una nueva técnica para suprimir analíticamente el efecto de una fuente en las funciones de correlación cruzada usadas para calcular los mapas SRP-PHAT. Basándonos en técnicas de banda estrecha, se demuestra que es posible proyectar las funciones de correlación cruzada de las señales capturadas por una agrupación de micrófonos a un espacio ortogonal a una dirección dada simplemente usando una combinación lineal de las funciones originales con retardos temporales. La técnica propuesta puede usarse para diseñar sistemas iterativos de localización de múltiples fuentes que, tras localizar la fuente con mayor energía en las funciones de correlación cruzada o en los mapas

SRP-PHAT, la cancelen para poder encontrar otras fuentes que estuvieran enmascaradas por ella.

Antes de poder entrenar modelos de *deep learning* necesitamos datos. Esto, en el caso de seguir un esquema de aprendizaje supervisado, supone un *dataset* de grabaciones de audio multicanal con la posición de las fuentes etiquetada con precisión. Pese a que existen algunos *datasets* con estas características, estos no son lo suficientemente extensos para entrenar una red neuronal y los entornos acústicos que incluyen no son suficientemente variados. Para solventar el problema de la falta de datos, presentamos una técnica para simular escenas acústicas con una o varias fuentes en movimiento y, para realizar estas simulaciones conforme son necesarias durante el entrenamiento de la red, presentamos la que es, que sepamos, la primera librería de *software* libre para la simulación de acústica de salas con aceleración por GPU. Tal y como queda demostrado en esta tesis, esta librería es más de dos ordenes de magnitud más rápida que otras librerías del estado del arte.

La idea principal del *Geometric Deep Learning* es que los modelos deberían compartir las simetrías (i.e. las invarianzas y equivarianzas) de los datos y el problema que se quiere resolver. Para la estimación de la dirección de llegada de una única fuente, el uso de mapas SRP-PHAT como entrada de nuestros modelos hace que la equivarianza a las rotaciones sea obvia y, tras presentar una primera aproximación usando redes convolucionales tridimensionales, presentamos un modelo basado en convoluciones icosaédricas que son capaces de aproximar la equivarianza al grupo continuo de rotaciones esféricas por la equivarianza al grupo discreto de las 60 simetrías del icosaedro. En la tesis se demuestra que los mapas SRP-PHAT son una característica de entrada mucho más robusta que los espectrogramas que se usan típicamente en muchos modelos del estado del arte y que el uso de las convoluciones icosaedricas, combinado con una nueva función *soft-argmax* que obtiene una salida de regresión a partir del resultado de una

red convolucional interpretándolo como una distribución de probabilidad y calculando su valor esperado, permite reducir enormemente el número de parámetros entrenables de los modelos sin reducir la precisión de sus estimaciones.

Cuando queremos realizar el *tracking* de varias fuentes en movimiento y no podemos aplicar ningún criterio para ordenarlas o clasificarlas, el problema se vuelve invariante a las permutaciones de las estimaciones, por lo que no podemos compararlas directamente con las etiquetas de referencia dado que no podemos esperar que sigan el mismo orden. Este tipo de modelos se han entrenado típicamente usando estrategias de entrenamiento invariantes a las permutaciones, pero estas normalmente no penalizan los cambios de identidad por lo que los modelos entrenados con ellas no mantienen la identidad de cada fuente de forma consistente. Para resolver este problema, en esta tesis proponemos una nueva estrategia de entrenamiento, a la que llamamos *sliding permutation invariant training* (sPIT), que es capaz de optimizar todas las características que podemos esperar de un sistema de *tracking* de múltiples fuentes: la precisión de sus estimaciones de dirección de llegada, la exactitud de sus detecciones y la consistencia de las identidades asignadas a cada fuente.

Finalmente, proponemos un nuevo tipo de red recursiva que usa conjuntos de vectores en lugar de vectores para representar su entrada y su estado y que es invariante a las permutaciones de los elementos del conjunto de entrada y equivariante a las del conjunto de estado. En esta tesis se muestra como este es el comportamiento que deberíamos esperar de un sistema de *tracking* que toma como entradas las estimaciones de un modelo de localización multifuente y se compara el rendimiento de estas redes recursivas invariantes a las permutaciones con redes recursivas GRU convencionales para aplicaciones de *tracking* de fuentes sonoras.

Contents

List of Publications

The following publications have been produced as result of the works in this thesis:

[I] D. Diaz-Guerra and J. R. Beltran, "Direction of Arrival Estimation with Microphone Arrays Using SRP-PHAT and Neural Networks," in 2018 IEEE 10th Sensor Array and Multichannel Signal Processing Workshop (SAM), Jul. 2018, pp. 617–621. doi: 10.1109/SAM.2018.8448492.

[II] D. Diaz-Guerra and J. R. Beltran, "Source cancellation in cross-correlation functions for broadband multisource DOA estimation," in Signal Processing, vol. 170, p. 107442, May 2020, doi: 10.1016/j.sigpro.2019.107442.
IF (JCR 2020): 4.662. Engineering, Electrical & Electronic rank: 44/273 (Q1)

[III] D. Diaz-Guerra, A. Miguel, and B. Jose R., "Acoustic Room Impulse Response Simulation with GPUs," presented at the Second International Conference on Advances in Signal Processing and Artificial Intelligence (ASPAI' 2020), Berlin, Germany, Jul. 2020.

[IV] D. Diaz-Guerra, A. Miguel, and J. R. Beltran, "gpuRIR: A python library for room impulse response simulation with GPU acceleration," in Multimedia Tools and Applications, Oct. 2020, doi: 10.1007/s11042-020-09905-3.
IF (JCR 2020): 2.757. Engineering, Electrical & Electronic rank: 120/273 (Q2)

[V] D. Diaz-Guerra, A. Miguel, and J. R. Beltran, "Robust Sound Source Tracking Using SRP-PHAT and 3D Convolutional Neural Networks," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 29, pp. 300–311,

2021, doi: 10.1109/TASLP.2020.3040031.
IF (JCR 2021): 4.364. Acoustics rank: 5/32 (Q1)

[VI] D. Diaz-Guerra, A. Miguel and J. R. Beltran, "Direction of Arrival Estimation of Sound Sources Using Icosahedral CNNs," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol 31, pp. 313-321, 2023, doi: 10.1109/TASLP.2022.3224282.
IF (JCR 2021): 4.364. Acoustics rank: 5/32 (Q1)

[VII] D. Diaz-Guerra, A. Politis, T. Virtanen, "Position tracking of a varying number of sound sources with sliding permutation invariant training", arXiv preprint arXiv:2210.14536.
Submitted to the 2023 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2023)

David Diaz-Guerra is the first author of all the publications in this thesis and he proposed the initial ideas for them and conducted the experimental works. Jose Ramon Beltran and Antonio Miguel supervised the works, discussed the ideas, and provided constant feedback during the experimentation and the writing of the manuscripts. The work presented in [VII] was conducted during a stay at the Audio Research Group at the Tampere University (Finland) under the supervision of Archontis Politis and Tuomas Virtanen.

The manuscript [VII] has been submitted to the 2023 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2023) and is currently under review.

This thesis includes the reproduction of some figures and text fragments from these papers with permission from the respective copyright holders.

List of Figures

List of Tables

1

Introduction

In this introduction chapter, we first make a general description of the context of the research developed along this thesis and the main ideas that guided it (section 1.1). Then, we make a brief review of the start-of-the-art of sound source localization and tracking with neural networks (section 1.2) and introduce the concept of Geometric Deep Learning (section 1.3). Finally, we summarize the main contributions and results of this thesis (section 1.4) and outline the structure of the following chapters of the thesis (section 1.5).

1.1 Context and motivation

Microphone array signal processing is a research field with a long trajectory within the signal processing community. Even if we can find earlier references ([1, 2] for example), it is typically considered that their study and implementation started in the late 70s and early 80s [3] and, since then, there has been a constant stream of both research studies and applications. However, in the last few years, after the revolution originated by the rebirth of neural networks in computer vision applications, there has been an explosion of new research studies that, using deep learning techniques, have clearly overcome the classic techniques that had constituted the state-of-the-art for years. This improvement in the capabilities of, for example, speech recognition systems, has led to the emergence of new applications of microphone arrays, such as

smart speakers or teleconference devices, that have increased even more the attention to multichannel signal processing research.

Focusing on the main topic of this dissertation, sound source localization and tracking have always been active fields of research since, apart from their own applications, they are a preliminary step for many other applications such as speech enhancement using beamforming techniques. Knowing this, it is not surprising that, in recent years, many deep learning-based techniques had been proposed. However, although these new techniques improve the state-of-the-art previously established by the classical methods, we are still far from being able to perfectly estimate and track the position of sound sources in highly reverberant spaces using compact arrays.

In this context of a growing number of proposals for new sound source localization and tracking systems based on deep learning, the main ideas that guided this thesis were:

1. *Exploiting the classic signal processing knowledge of the problem:* even if the performance of the classical methods quickly decreased in presence of reverberation, they were based on interesting analytical models that allow us to extract spatial information from the audio signals recorded with microphone arrays. Even if end-to-end approaches can also offer some advantages, carefully designed preprocessing stages can provide more refined input features to the neural networks and allow them to obtain better results.

2. *Using models that fit the nature of the problem:* many of the first deep learning-based proposals for sound source localization and tracking were straightforward translations of the techniques that had obtained good results in computer vision problems using the spectrograms of the microphone array signals as input images of convolutional neural networks (CNNs). However, audio spectrograms have a very different nature than images since, while both dimensions of an image represent the same magnitude (space), the two dimensions of an audio spectrogram represent two different magnitudes (time and frequency) with different properties and characteristics. What started more like an intuition at the beginning of this thesis has recently been properly formalized under the name of Geometric

Deep Learning (GDL), an approach with growing popularity in many fields and applications of deep learning.

3. *Avoiding the use of non-causal elements:* while most of the classic methods were strictly causal, and therefore they were feasible for real-time applications (although the computational complexity of some of them made impossible their real-time implementation at that time), many of the techniques based on deep learning that have been proposed in the recent literature include non-causal elements (such as bidirectional recurrent units) that make impossible any kind of real-time implementation independently of the available computational resources. However, although sound source localization can be performed offline in some applications (like audio forensics), most require real-time estimations.

1.2 Sound source localization and tracking using neural networks

In this section, we present a brief review of the state-of-the-art of sound source localization and tracking using deep learning techniques, especially focusing on those techniques that have influenced this work. For a more in-depth and extensive literature review, we recommend the remarkable survey by Grumiaux et al. [4].

In general, we can consider that, in order to design a deep learning system, we must choose: i) with which representation of the input of our problem we will feed our neural network, ii) with which neural architecture we are going to process that input, iii) how the neural network should represent the result of our problem and iv) how we are going to train it (including with which dataset).

1.2.1 Input representation

Although feeding a neural network directly with the raw audio samples obtained with a microphone array is also possible [5, 6, 7], most of the state of the state-of-the-art techniques perform some preprocessing over the audio samples to provide their neural networks with a higher level representation of the acoustic scene.

The simplest (and most general) input features typically used as input of neural networks for sound source localization are multichannel spectral representations. Some proposals use only magnitude spectrograms [8, 9, 10] but, since most of the localization information is usually on the inter-channel phase differences, it is more common to use phase spectrograms [11, 12, 13] or combinations of both, stacking two separate channels with the magnitude and phase spectrum [14, 15, 16, 17] or the real and imaginary part of the spectrum [18, 19, 20], or using a single channel with complex values [21].

Other proposals use higher-level representations more suited for the sound source localization task, such as generalized cross-correlation coefficients [22, 23, 24, 25] or Ambisonics intensity vectors [26, 27, 28]. We could consider as the highest level representations those based on classical signal processing techniques that allow computing spatial representation of an acoustic scene from the multichannel signals of a microphone array, which can be based on the eigendecomposition of the cross-correlation matrices of the signals, such as MUSIC [29] or ESPRIT [30], or in acoustic power maps, such as the SRP-PHAT method [31, 32]. For example, we can find the use of the eigenvectors of the cross-correlation matrices as inputs of a neural network in [33], the use of MUSIC-based spatial pseudo-spectrums in [34], or SRP-PHAT power maps in [35] and in the works presented in this thesis.

Finally, it is worth mentioning that combining several of the input features discussed above is also a popular approach [36, 37, 38]. A comparison between several of these combinations can be found in [39].

1.2.2 Network architecture

Although multi-layer perceptrons were the most popular approach among the first proposals for sound source localization using neural networks [22, 23], the attention soon moved towards the architecture that was revolutionizing several computer vision problems: convolutional neural networks (CNNs) [9, 12, 37, 40]. In order to improve the tracking capabilities of CNNs, many authors propose the use of one or several recurrent layers after them, creating recurrent convolutional neural networks (RCNNs) [14, 41, 42]; however, most of these RCNNs use bidirectional recurrent layers that are

non-causal and, therefore, are not suitable for real-time applications. More recently, other architectures have also been proposed for sound source localization and tracking, like the use of attention mechanisms [43, 44, 45] or autoencoder and variational autoencoder architectures [46, 47, 48, 49, 50].

In section 1.3, we provide a further analysis of the different approaches of using CNNs for sound source localization and their advantages and disadvantages.

1.2.3 Output representation

Among the first proposals for sound source localization based on deep learning, the most popular approach to encode the direction of arrival (DOA) at the network output was to define a set of possible DOAs and use a network output to represent the probability of each one of them containing a sound source; i.e., they solved the localization problem as a classification problem [22, 51, 52]. In the case of single-source localization, the estimated DOA simply corresponds to the maximum of the outputs, but in the case of multiple sound source localization, we need to use some peak-picking algorithm to determine the number of sources and which maxima actually correspond to real sources.

One of the main drawbacks of this classification approach is that the maximum accuracy that the system can reach is limited by the number of outputs of the network. This problem is not too serious if we only want to estimate the azimuth component of the DOAs, but the number of outputs needed to keep the desired resolution grows quadratically if we want to estimate both the azimuth and the elevation and even cubically if we also want to estimate the distance or if we want to estimate XYZ Cartesian coordinates in the case of distributed arrays. Increasing the number of outputs of the network does not only mean increasing its size (and therefore the computational complexity of the system) but also the size of the training dataset, which should contain enough examples of sources in all the possible positions (especially when using fully-connected layers at the end of the network, which does not have any geometrical structure and whose outputs are completely independent regardless of the position they represent).

On the other hand, most of the recent proposals seem to prefer regression approaches, where the DOAs are directly provided by the continuous value of the outputs of the network. Since DOAs are generally expressed in terms of spherical coordinates (azimuth and elevation or polar angle) we could expect them to be the most common representation at the output of the regression networks, but it has been proved that estimating unitary 3D Cartesian vectors pointing towards the DOA provides better results [53, 54] and this is the most popular approach nowadays [14, 38, 55].

However, the regression approach makes working with several sources more complex. First, we need to set a maximum number of sources the system can work with (which will determine the number of outputs of the network) and, after that, we need to establish a strategy to determine which outputs are representing active sources at every time frame. To do this, many proposals use an additional binary-classification output for every localization output to act as activity detector (which can also be used for sound event detection when every one of these outputs represents a different event class) [14, 38, 56, 57] but in the case of using 3D Cartesian vectors to represent the DOA, we can also encode the activity estimation into the norm of the DOA vector, leading to what it is called activity-coupled Cartesian direction of arrival (ACCDOA) in [58].

In sound event localization and detection (SELD) techniques, it is frequently supposed that only one source of each type of event can be simultaneously present and a localization output is employed for each one of them. However, systems designed to locate several sources of the same class (e.g., several unknown speakers) have to face the source permutation problem [13], which is further explained in section 1.3.

Finally, it is worth saying that there are also some proposals where the neural networks are not used to directly estimate the DOAs, but to help other classical methods through time-frequency mask estimation [8, 59], signal dereverberation [60], or time difference of arrival estimation [48] for example.

1.2.4 Training and datasets

Although some unsupervised, semi-supervised, or weakly supervised approaches have been recently proposed [16, 46, 50], most of the deep learning-based sound source

localization systems use supervised strategies where the train dataset includes the ground-truth DOA of the sound sources present in the recordings This leads us to another problem: the lack of large datasets of multichannel recordings with accurate spatial annotations.

To solve this issue, the image source method (ISM) [61] is typically used to simulate RIRs for different positions and in different acoustic environments and then signals taken from single-channel datasets are convolved with them to be used for training; speech corpora are typically used to obtain the source signals, but more diverse datasets can also be used [62]. Since we can simulate as many positions and scenarios as we want, this should allow us to have infinite-size training datasets, but, due to the time needed to execute the ISM, most proposals first generate a fixed-size dataset and then use it to train their models instead of generating the training signals on the fly, which has been proven to provide better results in other acoustic signal processing problems [63]. The use of synthetic signals for training usually generates a drop in the performance of the models when tested on actual real-world recordings and several techniques have been proposed in recent years to fix this issue [64, 65, 66].

1.3 Geometric deep learning

1.3.1 General concepts

Geometric Deep Learning (GDL) is an emerging concept in the deep learning community that attempts to propose a unifying framework for many well-known deep learning architectures and that also provides a new approach to design new architectures for new kinds of data [67]. Their main idea is that, although in multidimensional spaces the amount of data needed to learn a distribution grows exponentially with the number of dimensions (phenomena known as *the curse of dimensionality*), we can still work on them by exploiting their symmetries (i.e., those transformations that leave invariant the properties of interest).

A function f is said to be invariant to a group of transformations $\mathfrak{G}$ if applying them to the function input does not affect the function output (i.e., $f\left(\mathfrak{g}(x)\right) = f(x)\ \forall \mathfrak{g} \in \mathfrak{G}$) and it is said to be equivariant if applying the transformation to the input leads to

the same transformation in the output (i.e., $f(\mathfrak{g}(x)) = \mathfrak{g}(f(x)) \ \forall \mathfrak{g} \in \mathfrak{G}$). Probably, one of the clearest and best-known examples of invariant and equivariant models that have succeeded in deep learning are the CNNs [68]. Bidimiensional convolutions are equivariant to translations and, according to the GDL ideas, this would explain the success of convolutional architectures in many computer vision problems that are invariant or equivariant to translations, such as image classification [69, 70] and image segmentation [71, 72] respectively.

Conventional CNNs can be generalized to group-equivariant convolutional neural networks (G-CNNs) [73] to exploit the equivariance to different types of symmetry groups. These symmetry groups can be discrete, such as discrete sets of rotations in the plane [74], in the space [75] or in 3D surfaces [76], or continuous, such as the group SO(3) of spherical rotations [77].

Another family of models that can be analyzed under the perspective of GDL are the graph neural networks (GNNs). The convolutional [78, 79], attentional [80, 81] and message-passing [82] approaches are all invariant to the permutations of their nodes and many other permutation-invariant models, such as Deep Sets [83] or the Transformers [84], can be seen as special cases of GNNs for fully connected graphs. Continuing with this GDL approach, we can find even modifications of the GNNs to include spatial information in their nodes and make them invariant both to their permutation and to spatial transformations [85].

1.3.2 Geometric deep learning for sound source localization and tracking

Since the success of 2D CNNs on computer vision tasks led to their use over audio spectrograms for many audio analysis and processing tasks, it makes sense to analyze if the translational equivariance of 2D convolutions is also an interesting property when analyzing audio spectrograms. For the time dimension, translational equivariance means equivariance to time shifts, which might be an interesting property in most of the audio processing tasks, however, in the frequency domain, the translational equivariance means equivariance to frequency shifts, whose interest is much more difficult to determine.

In the case of sound source localization, the main source of information is usually considered to be in the phase difference between the signals, but the phase difference for a determined position depends on the frequency, so being equivariant to frequency shifts does not seem to be an interesting property for this task. In the case of multichannel signals, the spectrum of every channel is typically used as a different input channel for the CNNs, but we can find a different approach in [12]. In this work, the time is not taken into account and the convolutions are performed along the channel axis. Since they use uniform linear arrays, we can expect the phase differences for a given DOA to be the same between adjacent channels, and therefore being equivariant to shifts in this dimension can be positive according to the ideas of GDL, but this would not be true for other array geometries.

As commented in the previous section, several input features are usually combined in sound source localization systems by stacking them as additional input channels for the convolutional layers. This might make sense from the point of view of GDL when all of them represent spectral features with the same frequency scale, but this is not the case when combining features that do not share the same axis, such as spectrograms and generalized cross-correlation (GCC) coefficients, where the frequencies of the spectrograms that are analyzed with each GCC coefficients as the convolution kernels move along them are arbitrary. In any case, it is worth saying that architectures that do not follow the GDL ideas can still obtain good results; for example, good results were obtained in [39] by stacking spectrograms and GCC coefficients and applying 2D convolutions over them. This means that spectrograms and GCCs indeed contain important information for sound source localization, but GDL suggests that there are probably better architectures to analyze them.

As we have seen, most of the deep-learning models for sound source localization and tracking that we can find in the literature present equivariances that are not shared by their inputs and by the problem that they are intended to solve. However, there are at least two symmetries closely related to this problem that we can exploit to design sound source localization and tracking models: the rotational symmetry and the source permutation symmetry.

1.3.2.1 Rotational symmetry

Since, in compact arrays, DOAs are typically represented in spherical coordinates, we could expect the sound source localization problem to exhibit a strong equivariance to spherical rotations. However, this symmetry is not easily exploited in many input representations, especially when working with arbitrary array geometries. In chapter 4, we propose the use of the steered response power with phase transform (SRP-PHAT) algorithm to obtain spatial representations of the acoustic scene where this symmetry becomes more obvious and it is easier to design network architectures that are able to exploit it. It is worth saying that, even if we used the SRP-PHAT algorithm, other kinds of spatial pseudo-spectrums could also be used, such as those computed with the multiple signal classification (MUSIC) algorithm, or even a combination of several techniques stacking different spatial representations as different input channels of the networks. Finally, spherical harmonics or Ambisonics representations could also be interesting for designing neural networks equivariant to the spherical rotations of the sources; some works have been recently proposed in this line [86, 87] but we do not explore this possibility in this work.

1.3.2.2 Source permutation symmetry

While the rotational symmetry could be ignored and still obtain good localization results, when we want to locate multiple sound sources and we do not have any valid criteria to organize them (as it is done in SELD) we have to face the problem of the source permutation symmetry when training our models; i.e., we cannot expect our model to output the DOAs in the same arbitrary order that they have in the ground-truth data of our dataset [13]. The typical way to solve this is to compute the loss function using the permutation of the sources that generates a better match between the network output and the ground truth, but choosing a common permutation for the whole acoustic scene generates local minima in the loss function where the training usually get stuck and choosing a different permutation for each time instant does not encourage the model to keep the identities of every output stable (this issue is further explained in chapter 5).

The source permutation symmetry has typically been seen as a problem when trying to train neural networks, but it also opens the door to the design of new network architectures to exploit it. In chapter 5, we propose both a new loss function to train multi-source trackers and a new permutation invariant recurrent layer where the information of every source is encoded in independent embedding vectors whose order does not affect the network output.

1.4 Main contributions and results of the thesis

The main contributions and results of this thesis are:

1. *We present a new analytical method to cancel the effect of a source in the inter-channel cross-correlation functions:* We generalize previous narrowband cancellation techniques to the broadband case and present an efficient time-domain implementation that can eliminate the effect of a sound source from the inter-channel cross-correlation functions. This new technique can be used to design iterative multi-source localization systems using these modified cross-correlation functions or other representations derived from them, such as the SRP-PHAT maps. We published this technique in [II] and we explain it in chapter 2.

2. *We present the first free and open-source implementation of the image source with GPU acceleration:* our python library gpuRIR reduces in two orders of magnitude the time needed to simulate room impulse responses using the image source method, which allows us to perform the simulations on the fly as we train our models, obtaining infinite-size datasets. This GPU implementation is explained in chapter 3 and we published it in [IV] and also presented it in [III].

3. *We prove that SRP-PHAT power maps are robust input features that allow the models to obtain higher accuracy in highly reverberant scenarios:* Compared with models using spectrogram as inputs, we prove that models using higher-level features such as SRP-PHAT power maps (or even GCCs) are more robust against reverberation. This is explained in chapter 4, whose results were published in [V].

4. *We present the first free and open-source implementation of the icosahedral convolutional neural networks (icoCNNs):* The icosahedral CNNs were originally proposed by Cohen et al. in [76] but they did not publish their implementation. We have published a PyTorch implementation which we believe will be useful for many researchers working with spherical signals.

5. *We present a new output layer that combines classification and regression:* We explain in chapter 5 how, interpreting them as probability distributions, we can use the spatial output of a CNN to obtain a regression output without needing to add any extra learnable parameters [VI].

6. *We prove that the use of models with rotational symmetry can improve the localization accuracy even with coarser inputs:* the use of power maps as inputs allows us to use models that are invariant to rotations and we show in chapter 4 that, as published in [VI], these models have higher localization accuracy even using lower resolution power maps.

7. *We present a new permutation invariant training strategy which allows the training of tracking models while encouraging them to keep the identities of the inputs stable:* while many of the loss functions and training strategies used to train tracking networks do not prevent identity switches in the models' outputs in order avoid local minima that make the training impossible, our new training strategy explained in chapter 5 (and published in [VII]) makes the model converge to solutions with lower identity switches without losing localization accuracy.

8. *We present a new permutation invariant recurrent layer for sound source tracking:* opposed to conventional recurrent units where the information of each source is represented in a unique state vector, our permutation invariant gated recurrent unit (PI-GRU) explained in chapter 5 split the information of every source in independent state vectors whose order does not affect the result of the model.

1.5 Outline and organization of the thesis

The reminder of this thesis is organized as follows:

- *Chapter 2* introduces the SRP-PHAT algorithm since we use them to compute the acoustic power maps that we use as input for our models in the rest of the thesis. In addition, we present the analytical method that we published in [II] to cancel the effect of a sound source in the inter-channel correlation functions of a microphone array, and that can be used to remove the effect of a source from the SRP-PHAT maps and therefore design iterative multi-source localization algorithms.

- *Chapter 3* explains the technique that we utilized to generate our training datasets, including our GPU implementation of the image source method [IV] and the algorithm that we used to generate synthetic moving sound sources.

- *Chapter 4* introduces our works on single source localization with neural networks using SRP-PHAT power maps as inputs. In section 4.1, we first present the 3D CNN published in [V] and then, in section 4.2, we present our model based on an icosahedral CNN [VI].

- *Chapter 5* explains our works on multi-source tracking. This includes both a new permutation invariant training strategy [VII] and our permutation invariant recurrent units.

- *Chapter 6* presents the conclusion of the thesis and discusses some future directions for sound source localization and tracking research.

2

The SRP-PHAT power maps

In this chapter, we present an introduction to the classic signal processing techniques for direction of arrival (DOA) estimation, focusing on the SRP-PHAT power maps since we will use them as input representation for the deep learning models presented in the following chapters. Finally, we also present a new technique to cancel the effect of a sound source on the inter-microphone generalized cross-correlation (GCC) functions that allows us to design iterative algorithms for multi-source DOA estimation systems.

This chapter includes the reproduction of some figures and text fragments from [II] and [V] with the permission of the copyright holders.

2.1 Classic signal processing techniques for broadband DOA estimation

As proposed in [32] for the sound source localization (SSL) problem, the broadband DOA estimation strategies may be divided into three categories: techniques employing time difference of arrival (TDOA) information, strategies based on maximizing the steered response power (SRP) of a beamformer, and approaches adopting high-resolution spectral estimation concepts. Some techniques have been proposed for specific geometric configurations, such as those based on the Spherical Harmonic Domain

[88, 89] that can only be used in spherical arrays, but in this chapter we will focus on geometry-independent techniques.

TDOA-based locators follow two-step strategies where, in the first step, the TDOA between pairs of microphones is estimated and, in the second step, the most reliable DOA for those TDOAs is computed. The most common approach to the estimation of the TDOA between two microphones is finding the maxima of the generalized cross-correlation (GCC) function [90]. In audio applications, especially for speaker localization, the phase transform (PHAT) is typically used [91, 92, 93] due to its good performance in reverberant environments [94]. The two-step strategy greatly reduces the complexity of these algorithms, but the loss of information between the two steps (only the maximums of the GCCs are used for the DOA estimation) dramatically degrades their performance when noise or reverberation levels increase [95].

The SRP methods are based on the calculation of the output power of a beamformer steered towards different directions and the search for its maximum. Computing the filter-and-sum beamformer output for each direction is computationally exhaustive, but fortunately, as proved in [31, 32], the SRP can be computed in terms of GCCs. In this case, most of the computational cost of each beamformer is shared and the computational complexity of the algorithms increases slower with the search space. Another advantage of computing the SRP in terms of GCCs is that including the phase transform becomes trivial, which results in the steered response power with phase transform (SRP-PHAT) algorithm [31, 32]. Despite this, its complexity can also become excessive if the search space increases considerably (as in 3D SSL problems) and several efficient search algorithms [96, 97, 98], functional modifications to allow coarser grids [98, 99, 100] and GPU [101, 102], FPGA or ASIC [103] implementations have been proposed.

Finally, the high-resolution spectral-estimation-based locators use beamforming methods based on techniques like autoregressive (AR) modeling, minimum variance (MV), spectral estimation, etc. Some of the most commonly used narrowband algorithms, such as multiple signal classification (MUSIC) [29] or estimation of signal parameters via rotational invariant techniques (ESPRIT) [30, 104], are based on the

signal and noise subspaces. After applying these narrowband techniques to each frequency bin of a broadband signal it is necessary to coherently combine the results of each bin [105] and many of these techniques are more sensitive to noise and reverberation [106] than the SRP-PHAT. In [107], an extension of the MUSIC algorithm for broadband signals based on the parameterized spatial correlation matrix (PSCM) or steered covariance matrix [108, 109] claims to outperform the SRP-PHAT algorithm even in high reverberant scenarios. This approach has a higher computational cost since, after computing the GCCs, the eigenvalue decomposition of the PSCM for each search direction must be computed.

Multi-source localization is another challenge in addition to robustness and computational efficiency, especially in wide dynamic range scenarios, where a more powerful source can mask a weaker one. Various techniques have been recently proposed to find multiple sources [110, 111, 112, 113, 114, 115], most assuming different degrees of time-frequency sparsity on the source signals. These sparsity constraints can be applied, for example, when speech sources need to be found [116] but cannot be applied with other sources like music or noise. Furthermore, these techniques are usually based on the incoherent application of subspace DOA estimation techniques in the time-frequency bins where just one source (or only a few sources) is present. This has two disadvantages: we lost the spatial information present in the bins where multiple sources are active and, on the other hand, subspace methods usually have high computational complexity and some of them may not work properly if correlation between sources exists. Finally, there are no studies about the performance of these techniques when the power level difference between sources increases.

Due to the non-stationary nature of most of the signals of interest, such as the speech or the music, a tracking stage is needed after the DOA estimation to exploit the temporal correlation between the source positions and to avoid inaccurate estimations in frames where the power of the signal is low or its auto-correlation makes the maximum of the power maps become too wide. The algorithms for one source tracking are typically based on the Kalman filter [117, 118] although more advanced techniques have been proposed to deal with multiple sources, such as those based on particle filtering [119, 120]. However, in these approaches, they use two-step strategies which make

them sensitive to potential information loss when only the DOA estimations are used for the tracking; e.g. the absolute maximum of the SRP-PHAT maps is always selected even if another local-maximum was much closer to the previous estimations and we assign the same likelihood to all the DOA estimations while some of them correspond to wider maximums from frames where the source was weaker. Including some of this information in the tracking algorithms may be possible, but it would increase both its complexity and the number of parameters that need to be fine-tuned. For example, in [121] a technique to share information between an iterative DOA estimator based on Expectation-Maximization [122] and a tracking system based on particle filtering is proposed.

2.2 The SRP-PHAT algorithm

The signal received at the n^{th} sensor of a microphone array in a room with N_s sources can be modeled as

$$x_n(t) = \sum_{i=1}^{N_s} a_i(t) * h_n(\boldsymbol{\theta}_i, t) + v_n(t), \tag{2.1}$$

where $a_i(t)$ is the signal generated by the source in the position $\boldsymbol{\theta}_i$, $h_n(\boldsymbol{\theta}_i, t)$ is the impulse response from $\boldsymbol{\theta}_i$ to the n^{th} sensor, and $v_n(t)$ is the noise of the sensor, which is typically supposed to be white, Gaussian, and uncorrelated with the source signal and with the noises of other sensors. It is worth mentioning that $\boldsymbol{\theta}_s$ is written in bold because it can represent an angle, two spherical coordinates, or even a point in 3D Cartesian coordinates depending on the geometry of the array.

One of the most classic and popular approaches to DOA estimation is finding the peaks of the steered response power (SRP) maps that we would obtain using a filter-and-sum beamformer:

$$P(\boldsymbol{\theta}) = \int_{-\infty}^{+\infty} \left| \sum_{n=0}^{N-1} G_n(\omega) X_n(\omega) e^{-j\omega\tau_n(\boldsymbol{\theta})} \right|^2 d\omega, \tag{2.2}$$

where N is the number of sensors of the array, $X_n(\omega)$ is the Fourier transform of $x_n(t)$, $G_n(\omega)$ is the frequency response of the filter for the channel n, and $\tau_n(\boldsymbol{\theta})$ is the time delay occurring from the position or direction $\boldsymbol{\theta}$ to the n^{th} sensor.

Although directly implementing (2.2) would be computationally expensive, it can be computed in terms of the GCC functions as

$$P(\boldsymbol{\theta}) = \int_{-\infty}^{+\infty} \left| \sum_{n=1}^{N} G_n(\omega) X_n(\omega) e^{-j\omega\tau_n(\boldsymbol{\theta})} \right|^2 d\omega = \sum_{n=1}^{N}\sum_{m=1}^{N} \int_{-\infty}^{+\infty} (G_n(\omega)X_n(\omega))(G_m(\omega)X_m(\omega))^* e^{-j\omega\tau_n(\boldsymbol{\theta})} d\omega = 2\pi \sum_{n=1}^{N}\sum_{m=1}^{N} R_{nm}(\Delta\tau_{nm}(\boldsymbol{\theta})), \quad (2.3)$$

where $\Delta\tau_{nm}(\boldsymbol{\theta}) = \tau_n(\boldsymbol{\theta}) - \tau_m(\boldsymbol{\theta})$ and R_{nm} is the GCC between the signals of the n^{th} and the m^{th} sensor:

$$R_{nm}(\tau) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \Psi_{nm}(\omega) X_n(\omega) X_m^*(\omega) e^{j\omega t} d\omega, \quad (2.4)$$

where * denotes the complex conjugate and $\Psi_{nm}(\omega) = G_n(\omega)G_m^*(\omega)$ is a weighting function.

Equation (2.3), combined with the use of the phase transform (PHAT) $G_n(\omega) = 1/|X_n(\omega)|$, is commonly known as the SRP-PHAT algorithm [31, 32], and allows us to obtain an acoustic power map of the environment whose peaks should correspond with the position of the sources.

Although the SRP-PHAT algorithm is a good trade-off between robustness and computational efficiency, obtaining more accurate results than two-step TDOA based techniques with a lower computational cost than most of the broadband subspace techniques, it still presents several issues. The main advantage of (2.3) is that most of its computational cost is in computing the GCCs and does not increase with the search space. However, the computation of its sums for each direction, especially if it is needed to interpolate $R_{nm}(\Delta\tau_{nm}(\boldsymbol{\theta}))$ from its adjacent samples, may not be negligible; this problem becomes more challenging when the search space is two-dimensional, e.g. the two angles of the spherical coordinates, or even three-dimensional, e.g. XYZ coordinates. Some search strategies have been proposed to reduce the number of evaluations of (2.3) that need to be computed to accurately find the maximum of $P(\boldsymbol{\theta})$ [97, 98, 123] but, due to the non-convexity of the SRP-PHAT power maps, the

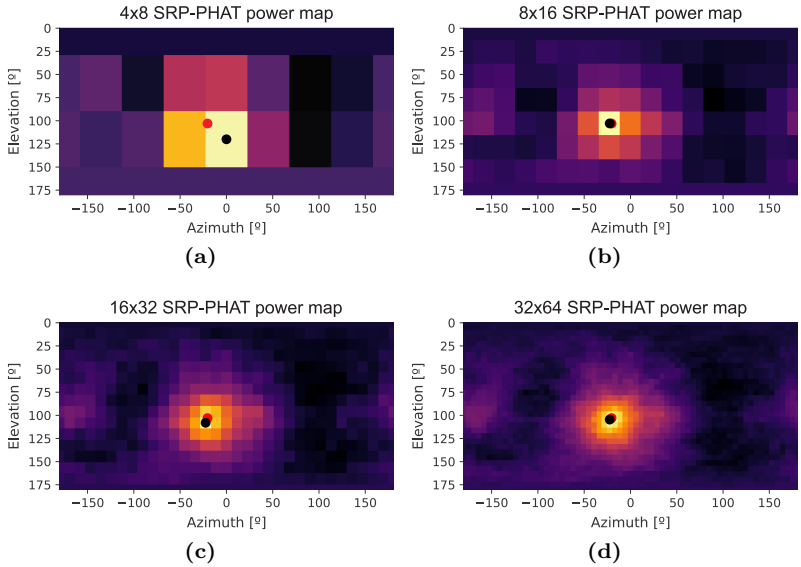


Figure 2.1: Example of SRP-PHAT power maps with different resolutions in a favorable scenario: SNR=30 dB and T_{60}=0.3 s. The red dot indicates the actual DOA of the sound source and the black dot is at the maximum of the map.

number of SRP-PHAT evaluations needed might still be an issue in some scenarios. In [98, 124], it is proposed to modify (2.3) to compute the power received from a space region instead of from a point, so they can use hierarchical search strategies over maps with lower resolution.

As we can see in Fig. 2.1, in favorable scenarios with high signal-to-noise ratio (SNR) and low reverberation, the SRP-PHAT power maps have a clear maximum in the DOA of the sound source that can be used to obtain a good estimation even with low-resolution maps but, when SNR decreases and the reverberation increases, such as in the scenario of Fig. 2.2, the maps present several local maxima that may be incorrectly interpreted as the DOA of the sound, especially when using low-resolution maps. However, in those maps, in addition to the maxima, we can also observe several patterns that are also related to the DOA of the sound and the geometry of the array and that may be exploited to obtain a more accurate DOA estimation.

Another limitation of the SRP-PHAT power maps is that, even if the phase transform is usually able to reduce the width of the maxima generated by the sources, weak

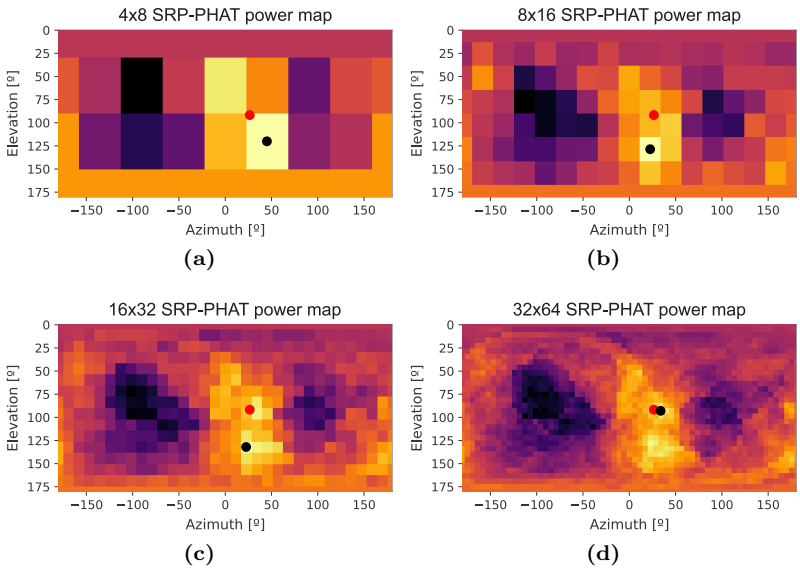


Figure 2.2: Example of SRP-PHAT power maps with different resolutions in an adverse scenario: SNR=5 dB and T_{60}=0.9 s. The red dot indicates the actual DOA of the sound source and the black dot is at the maximum of the map.

sources are still easily masked by stronger sources. Due to this reason, SRP-PHAT power maps are rarely used for multi-source SSL.

2.3 Source cancellation in Cross-Correlation functions

In this section, we present a new technique to eliminate the effect of a source from the inter-microphone cross-correlation (CC) functions or the GCCs in a broadband sensor array [II]. It can be applied to any array geometry and it does not need any time-frequency sparsity constraint, so the technique can work with white noise sources. In addition, this technique does not only eliminate the source in the desired direction but also the spurious maxima that appear when the sources are correlated. As it is a modification of the GCCs functions, most of the techniques and methods developed using the cross-correlation functions can be combined with it.

Our technique is based on a spatial correlation matrix (SCM) projection technique

widely employed in narrowband arrays [125, 126, 127], but we extend it to broadband sources. An efficient time-domain implementation is also presented and, if a lower computational cost is needed, we also propose a coarser approximation that can offer adequate results. Our technique is especially interesting in scenarios with correlation between sources and strong power differences between them.

In order to deal with multiple sources, the proposed technique can be used to iteratively find the strongest source, eliminate it, and find the following source. A similar idea has been recently published in [128] for DOA estimation of acoustic echoes, but it only deals with narrowband sources. A broadband technique is proposed in [129] to remove the effect of a source from the GCCs functions to locate multiple audio sources. However, it only suggests multiplying the GCCs by a de-emphasis function; they propose using a notch function with two parameters to control its sharpness. This approach lacks theoretical foundations and strongly depends on the choice of the proposed parameters, but [129] does not provide any technique to choose them.

2.3.1 Narrowband source elimination

The complex snapshot received by an array of N narrowband sensors in a scenario with N_s sources can be written as

$$\mathbf{x} = \mathbf{D}\mathbf{a} + \mathbf{v}, \tag{2.5}$$

where $\mathbf{a}$ is a vector of length N_s with the amplitudes of each signal, $\mathbf{v}$ is a zero-mean white noise vector of length N uncorrelated with $\mathbf{a}$, and $\mathbf{D}$ is the steering matrix composed of the steering vectors of each source:

$$\mathbf{D} = [\mathbf{d}(\boldsymbol{\theta}_1), \mathbf{d}(\boldsymbol{\theta}_2), ..., \mathbf{d}(\boldsymbol{\theta}_{N_s})] \tag{2.6}$$

$$\mathbf{d}(\boldsymbol{\theta}_i) = [e^{-j\omega\tau_1(\boldsymbol{\theta}_i)}, e^{-j\omega\tau_2(\boldsymbol{\theta}_i)}, ..., e^{-j\omega\tau_N(\boldsymbol{\theta}_i)}]^T, \tag{2.7}$$

With this model, the complex SCM is

$$\mathbf{R} = E\{\mathbf{x}\mathbf{x}^H\} = \mathbf{D}E\{\mathbf{a}\mathbf{a}^H\}\mathbf{D}^H + \sigma_v^2\mathbf{I}, \tag{2.8}$$

where $E\{\cdot\}$ is the expectation operator, and the superscript H denotes Hermitian transpose, σ_v^2 is the noise power and, if there is no correlation between the sources, the correlation matrix of signals $E\{\mathbf{a}\mathbf{a}^H\}$ is just a diagonal matrix with the power of each signal.

Several techniques have been proposed to prevent a strong source from hiding a weaker one in narrowband DOA estimation [125, 126, 127]. They are based on the projection of the correlation matrix onto a space orthogonal to the position $\boldsymbol{\theta}_0$ of the stronger source:

$$\mathbf{R'}(\boldsymbol{\theta}_0) = \mathbf{U}(\boldsymbol{\theta}_0)\mathbf{R}\mathbf{U}(\boldsymbol{\theta}_0), \tag{2.9}$$

where $\mathbf{U}(\boldsymbol{\theta}_0)$ is the projection matrix

$$\mathbf{U}(\boldsymbol{\theta}_0) = \mathbf{I} - \frac{\mathbf{d}(\boldsymbol{\theta}_0)\mathbf{d}^H(\boldsymbol{\theta}_0)}{\mathbf{d}^H(\boldsymbol{\theta}_0)\mathbf{d}(\boldsymbol{\theta}_0)}. \tag{2.10}$$

After obtaining $\mathbf{R'}(\boldsymbol{\theta}_0)$, several narrowband localization techniques can be used to find the second source and, if there are still more sources to be found, apply (2.9) with $\boldsymbol{\theta}_0$ equal to the position of the second source and then repeat the localization process.

2.3.2 Broadband source elimination

In broadband arrays, it is necessary to take into account the temporal characteristics of the signals and the effect of the reverberation as in the model followed in (2.1). The CC between the signals received in the n^{th} and the m^{th} sensors is defined in the frequency domain as

$$R_{nm}(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} X_n(\omega)X_m^*(\omega)e^{j\omega t}d\omega \tag{2.11}$$

and the Fourier transform of the CC, $\mathcal{F}\{R_{nm}(t)\} = \mathcal{S}_{nm}(\omega) = X_n(\omega)X_m^*(\omega)$, is known as the cross-power spectral density (CPSD).

In order to get the CC functions with the source in the position $\boldsymbol{\theta}_0$ canceled, we propose to apply (2.9) to each frequency bin of the CPSDs. The matrix multiplications

in (2.9) can be rewritten as:

$$\mathcal{S}_{nm}^{SC}(\boldsymbol{\theta}_0,\omega)=\sum_{k=1}^{N}\left[\sum_{l=1}^{N}\left[U_{nl}(\boldsymbol{\theta}_0,\omega)\mathcal{S}_{lk}(\omega)\right]U_{km}(\boldsymbol{\theta}_0,\omega)\right], \tag{2.12}$$

where $U_{nm}(\boldsymbol{\theta}_0,\omega)$ is the (n,m) entry of the matrix $\mathbf{U}(\boldsymbol{\theta}_0)$ for the frequency ω, obtained expanding (2.10):

$$U_{nm}(\boldsymbol{\theta}_0,\omega)=\begin{cases}e^{-j\omega\Delta\tau_{nm}(\boldsymbol{\theta}_0)}/N, & \text{if } n\neq m\\ (N-1)/N, & \text{if } n=m\end{cases} \tag{2.13}$$

From (2.12) we can expand the summations into three main terms:

$$\mathcal{S}_{nm}^{SC}(\boldsymbol{\theta}_0,\omega)=\frac{(N-1)^2}{N^2}\mathcal{S}_{nm}(\omega)-\frac{N-1}{N^2}\mathcal{P}_{nm}(\boldsymbol{\theta}_0,\omega)+\frac{1}{N^2}\mathcal{Q}_{nm}(\boldsymbol{\theta}_0,\omega), \tag{2.14}$$

where

$$\mathcal{P}_{nm}(\boldsymbol{\theta}_0,\omega)=\sum_{k=1,\ldots,N}^{k\neq n}\mathcal{S}_{km}(\omega)e^{-j\omega\Delta\tau_{nk}(\boldsymbol{\theta}_0)}+\sum_{l=1,\ldots,N}^{l\neq m}\mathcal{S}_{nl}(\omega)e^{-j\omega\Delta\tau_{lm}(\boldsymbol{\theta}_0)} \tag{2.15}$$

and

$$\mathcal{Q}_{nm}(\boldsymbol{\theta}_0,\omega)=\sum_{k=1,\ldots,N}^{k\neq n}\sum_{l=1,\ldots,N}^{l\neq m}\mathcal{S}_{kl}(\omega)e^{-j\omega(\Delta\tau_{nk}(\boldsymbol{\theta}_0)+\Delta\tau_{lm}(\boldsymbol{\theta}_0))}. \tag{2.16}$$

Finally, through the inverse Fourier transform, we conclude that the CC between the n^{th} and the m^{th} sensor with the source in the position $\boldsymbol{\theta}_0$ removed is just the original CC and a linear combination of the CCs between each pair of sensors with different delays:

$$R_{nm}^{SC}(\boldsymbol{\theta}_0,t)=\frac{(N-1)^2}{N^2}R_{nm}(t)-\frac{N-1}{N^2}P_{nm}(\boldsymbol{\theta}_0,t)+\frac{1}{N^2}Q_{nm}(\boldsymbol{\theta}_0,t) \tag{2.17}$$

$$P_{nm}(\boldsymbol{\theta}_0,t)=\sum_{k=1,\ldots,N}^{k\neq n}R_{km}\left(t-\Delta\tau_{nk}(\boldsymbol{\theta}_0)\right)+\sum_{l=1,\ldots,N}^{l\neq m}R_{nl}\left(t-\Delta\tau_{lm}(\boldsymbol{\theta}_0)\right) \tag{2.18}$$

$$Q_{nm}(\boldsymbol{\theta}_0,t)=\sum_{k=1,\ldots,N}^{k\neq n}\sum_{l=1,\ldots,N}^{l\neq m}R_{kl}\left(t-\Delta\tau_{nk}(\boldsymbol{\theta}_0)-\Delta\tau_{lm}(\boldsymbol{\theta}_0)\right). \tag{2.19}$$

With the term involving $P_{nm}(\boldsymbol{\theta}_0,t)$ in (2.17), we are subtracting to each original CCs, i.e. R_{nm} (black star in Fig. 2.3), a delayed version of the CCs that have a sensor

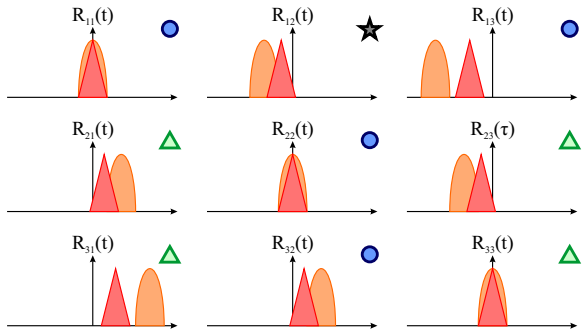


Figure 2.3: A representation of the auto-correlation functions of 2 sources and 3 sensors. In order to remove the effect of the first source (whose autocorrelation is represented with a red triangle) from $R_{12}(t)$ (indicated with a black star) we must use the CCs that have a sensor in common (indicated in blue circles) to compute $P_{12}(t)$ and the CCs without any sensor in common (indicated with a green triangle) to compute $Q_{12}(t)$. See equations (2.17), (2.18) and (2.19) in the text.

in common, i.e. $R_{n'm'}$ with $n' = n$ or $m' = m$ (blue circles in Fig. 2.3). The applied delay places the peak corresponding to the source we want to remove in the position $\Delta\tau_{nm}(\boldsymbol{\theta}_0)$. With the term involving $Q_{nm}(\boldsymbol{\theta}_0, t)$, we are adding the CCs without any sensor in common, i.e. $R_{n'm'}$ with $n' \neq n$ and $m' \neq m$ (green triangles in Fig. 2.3), also placing in $\Delta\tau_{nm}(\boldsymbol{\theta}_0)$ the peaks generated by the source in $\boldsymbol{\theta}_0$. This process is represented in Fig. 2.4.

In the presence of reverberation, the array does not only receive the source signals from its direct paths, but also from the DOA of the early reflections. Applying (2.17) with $\boldsymbol{\theta}_0$ equal to the position (or the direction) of a source, our technique does not remove the signals received due to its reflections; equivalently, with $\boldsymbol{\theta}_0$ equal to the DOA of a reflection, only that reflection is canceled. Not suppressing the reflections of the canceled source can be a disadvantage for source localization applications (although most tracking algorithms are designed to deal with the erroneous DOAs generated by the reverberation), but it is indispensable for the acoustic characterization of a room, where the goal is to locate the DOAs of its early reflections [130, 131].

2.3.3 Coarse approximation

Most of the computational complexity of (2.17) correspond to compute $Q_{nm}(\boldsymbol{\theta}_0, t)$, however, the elimination of the effect of the desired source happens in the subtraction

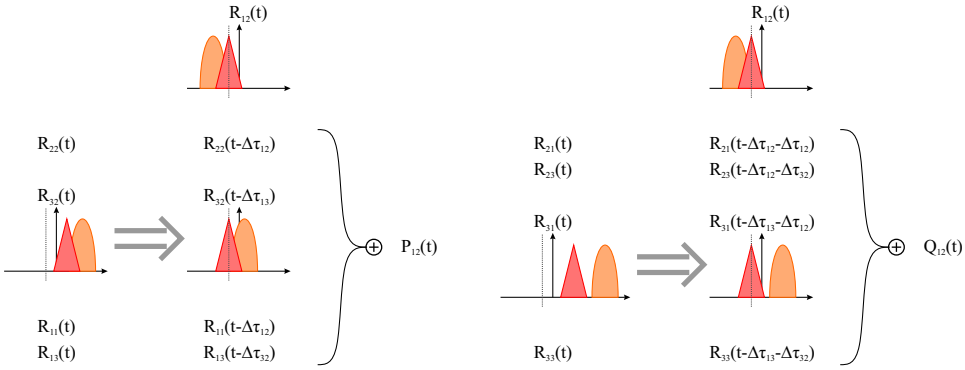


Figure 2.4: A representation of the terms $P_{12}(t)$ and $Q_{12}(t)$ in equations (2.17), (2.18) and (2.19). Following the picture in Fig. 2.3, in order to compute both $P_{12}(t)$ and $Q_{12}(t)$, the delays applied to each CC function place the peak generated by the source we want to remove (the red triangle) in the position it was in $R_{12}(t)$.

of $P_{nm}(\boldsymbol{\theta}_0, t)$ (although it is true that the CCs can have both positive and negative values, there will always be positive peaks at the positions of the sources). Therefore, we could get a strong reduction of the computational cost, i.e. the number of operations reduces from $\mathcal{O}((N-1)^2)$ to $\mathcal{O}(2(N-1))$, by approximating (2.17) by:

$$R_{nm}^{SC}(\boldsymbol{\theta}_0, t) \approx R_{nm}^{SC-T}(\boldsymbol{\theta}_0, t) = \frac{(N-1)^2}{N^2} R_{nm}(t) - \frac{N-1}{N^2} P_{nm}(\boldsymbol{\theta}_0, t). \tag{2.20}$$

The error introduced by this approximation is further analyzed in section 2.4.2.

2.3.4 Phase Transform (PHAT)

As previously explained, the CC function (2.11) can be generalized to the GCC (2.4) by introducing a weighting function $\Psi_{nm}(\omega)$ and, for SSL, the phase transform is typically used:

$$\Psi_{nm}^{PHAT}(\omega) = \frac{1}{|X_n(\omega) X_m^*(\omega)|}\ , \tag{2.21}$$

which equally emphasizes all the frequencies for the DOA estimation and has been proved to approximate the Maximum Likelihood solution in low-noise reverberant environments [132] and is especially useful to deal with speech sources.

In order to correctly obtain the GCC with the source in the position $\boldsymbol{\theta}_0$ removed, we should compute (2.14), apply the desired weighting function $\Psi_{nm}(\omega)$, and compute

the inverse Fourier transform. To compute $\Psi_{nm}(\omega)$ it would be necessary to get the signal $X_n(\omega)$ without the source in the position $\boldsymbol{\theta}_0$ but, in a real situation, we only have the signal with all the sources present. Therefore, it is impossible to obtain $\Psi_{nm}(\omega)$. To solve this issue and to exploit the benefits of the PHAT in reverberant scenarios, we propose to apply directly (2.17) by replacing the original CCs with the GCCs.

2.3.5 Examples

In order to better understand the effect of equations (2.17) and (2.20), in this section we present several examples with a linear array of 3 sensors.

2.3.5.1 Two uncorrelated Gaussian sources

If the signals generated by 2 uncorrelated white Gaussian sources arrive at the sensors with delays $\tau(\boldsymbol{\theta}_1) = [0, 10, 20]$ and $\tau(\boldsymbol{\theta}_2) = [-5, 6, 17]$ samples in an anechoic environment, the CCs between sensors have 2 peaks, some of them in adjacent samples, as shown in Fig. 2.5 (a). If we apply the coarse approximation proposed in (2.20) to eliminate the source $\boldsymbol{\theta}_0 = \boldsymbol{\theta}_1$, the only remaining positive peaks are in $\Delta\tau_{nm}(\boldsymbol{\theta}_2)$ and belong only to $\boldsymbol{\theta}_2$ (Fig. 2.5 (b)), but they have strong negative peaks and break some properties of the CC functions, e.g. the maximum of $R_{nn}(t)$ is not at $\tau = 0$. This could generate some artifacts in the power estimation, as negative values in the SRP maps, but if we only need to find the position of the main positive peak, the approximation may be good enough.

Finally, if we apply the complete expression, i.e. equation (2.17), the CCs still have only the desired positive peak, but they follow the properties of the Correlation functions and the negative peaks are reduced, as shown in Fig. 2.5 (c).

2.3.5.2 Two impulsive sources

If we replace the Gaussian sources of the previous example with impulsive sources, we get 2 strongly correlated sources, that may be seen as an original source and a non-attenuated echo. In Fig. 2.6 (a), we can see how the correlation between sources generates additional peaks in the CCs that could lead to the estimation of spurious sources. However, our elimination technique, as shown in Fig. 2.6 (b) and Fig. 2.6

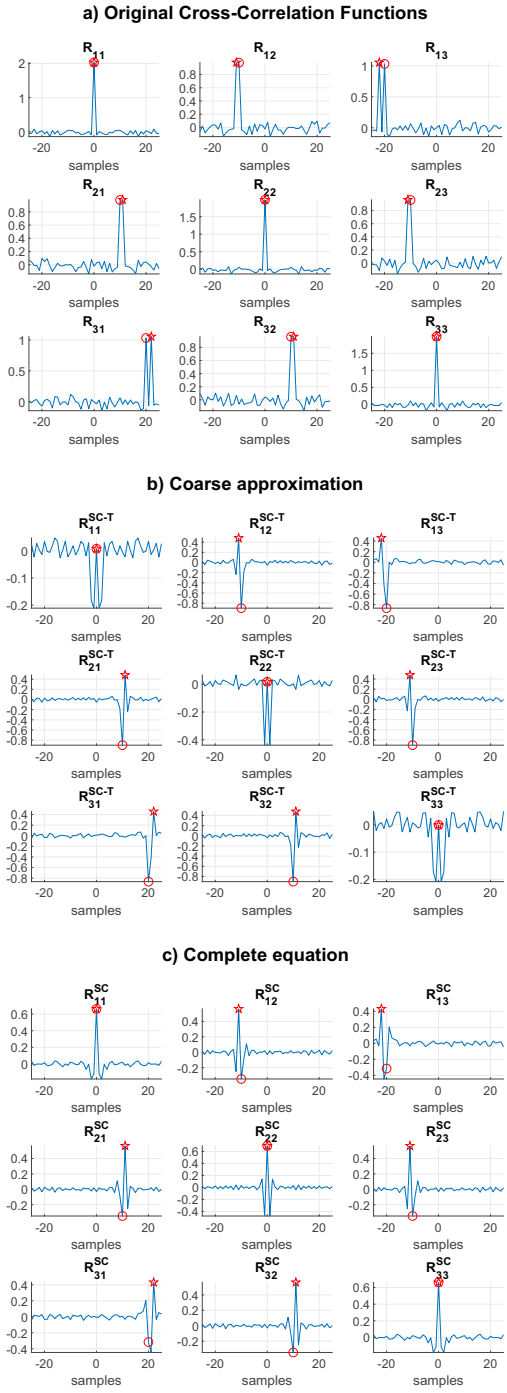


Figure 2.5: CCs between sensors in the scenario described in the example 2.3.5.1 (a), CCs after applying the coarse approximation to cancel the first source (b) and CCs after applying the complete equation of the proposed cancellation technique (c). The red circles are at $\Delta\tau_{nm}(\boldsymbol{\theta}_1)$ and the red stars at $\Delta\tau_{nm}(\boldsymbol{\theta}_2)$.

(c) for the coarse approximation and the complete expression, removes both the peaks corresponding to the source in $\theta_0 = \theta_1$ and the peaks generated by the correlation between sources, so correlated sources can be handled as if they were uncorrelated. This is especially useful for echo localization, where several correlated sources have to be found.

It is worth mentioning that, in both examples, we supposed that the first source was perfectly located and, since the auto-correlations of the sources are really narrow, small errors on it would have dramatically degraded its cancellation. However, the estimation of the sources is more accurate as the autocorrelation of the sources becomes narrower and, for wider autocorrelation, the accuracy of the estimation is not so critical since both the original source and the terms we are subtracting in (2.17) and (2.20) are wider. This effect is further studied in section 2.4.2.

2.3.5.3 Two speech sources with reverberation

Finally, we used Lehmann's implementation [133] of the image source method (ISM) algorithm [61] to simulate a linear array with 5 microphones and 10 cm of inter-microphone distance in a room with reverberation time $T_{60} = 0.1$ s; although this is a quite low reverberation, it allows us to check that our technique still works when the delay introduced by the propagation does not have a perfectly linear phase, i.e. each frequency component of the signal suffers a slightly different delay. Fig 2.7 (a) shows the CCs between the signals received by the microphones 1 and 5 when a speech source is placed in front of the array and another one at 45°; we can see how the source in front of the array (whose peak is indicated with a red circle) mask the effect of the second source (whose peak should be at the red star). As we can see in Fig. 2.7 (b), the truncated version of our cancellation technique removes too much energy from CCs and we need to apply the complete equation, Fig. 2.7 (c), to properly identify the peak corresponding to the second source.

We analyze higher reverberant rooms in section 2.4.3, but only locating peaks in the CCs is not enough to deal with them, so we need to employ more sophisticated DOA estimation algorithms.

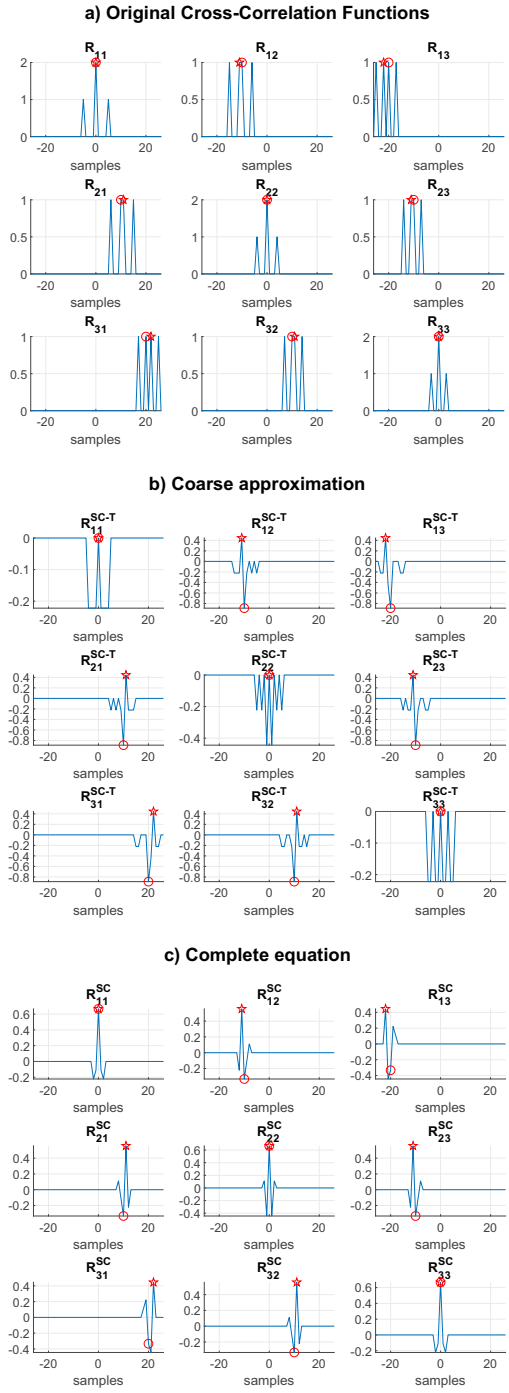


Figure 2.6: CCs between sensors in the scenario described in the example 2.3.5.2 (a), CCs after applying the coarse approximation to cancel the first source (b) and CCs after applying the complete equation of the proposed cancellation technique (c). The red circles are at $\Delta\tau_{nm}(\boldsymbol{\theta}_1)$ and the red stars at $\Delta\tau_{nm}(\boldsymbol{\theta}_2)$.

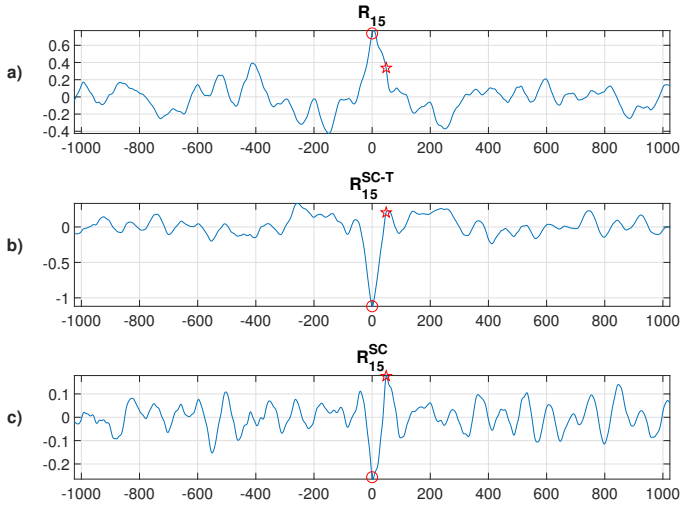


Figure 2.7: CC between sensors 1 and 5 in the scenario described in the example 2.3.5.3 (a) and the result of applying the coarse approximation (b) and the complete equation (c) over it. The theoretical positions of the peaks generated by the first and the second source are indicated with a circle and with a star, respectively.

2.4 An iterative multi-source localization algorithm based on source cancellation and SRP-PHAT

2.4.1 Algorithm

Several multiple-source localization algorithms may be proposed by combining the SRP-PHAT power maps with the source cancellation technique presented in the previous section. Here, we propose a simple iterative algorithm similar to the one used for narrowband arrays that, despite its simplicity, is able to prove the power of the proposed source cancellation technique.

The algorithm, detailed in Fig. 2.8, first computes the CCs (2.11) or GCCs (2.4) and uses them to compute the SRP or SRP-PHAT power map (2.3). Then, it iteratively searches for the strongest source and cancels it to compute a new power map. It also allows to iteratively apply the source cancellation in the same position if a new maximum is not found; however, as proved in sections 2.4.2 and 2.4.3, only a low

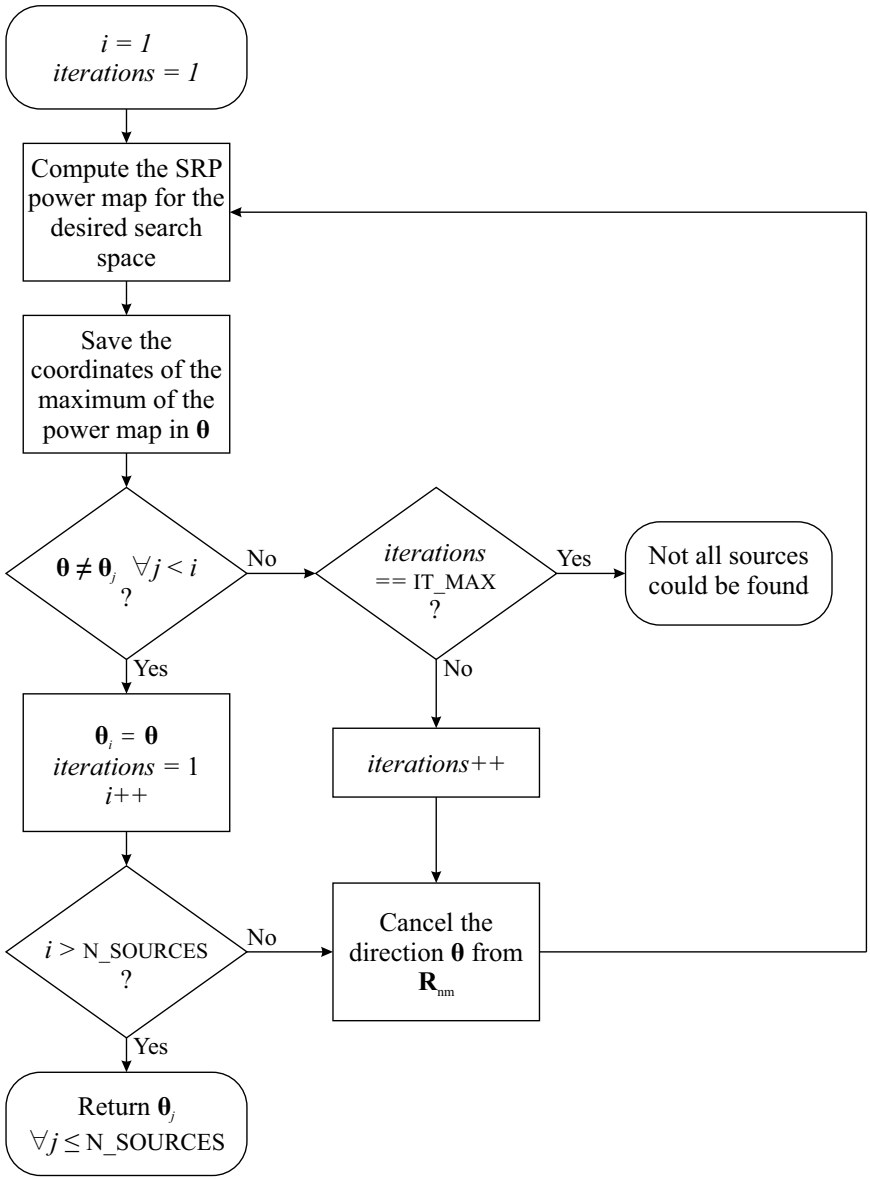


Figure 2.8: Iterative algorithm to find several sources using source cancellation and SRP techniques.

number of iterations, controlled by the IT_MAX parameter in Fig. 2.8, is needed.

For the sake of simplicity, it is assumed that the number of sources (N_SOURCES) is known, but it would be easy to modify the algorithm to dynamically obtain the number of sources. To do so, a threshold could be applied to the power of each new maximum to determine if it is a new source or if there are no more sources to be removed.

Implementing both (2.17) and (2.3) in a discrete-time system has the problem that $\Delta\tau_{nm}(\boldsymbol{\theta}_0)$ may not be multiple of the sampling period. In order to deal with that, we found that using a linear interpolation between adjacent samples in (2.17) and just taking the closest one in (2.3) provides good enough results.

2.4.2 Simulations

2.4.2.1 Uncorrelated white sources

Simulations were conducted to compare the performance of the conventional SRP method with our proposal. We carried out 1000 simulations with 1024 samples of 2 Gaussian sources (without correlation between them) in random positions, capturing the signals with a circular array with 16 sensors and a diameter of 0.5 meters in an anechoic environment. We repeated the 1000 simulations with 31 different power levels of the second source from $-30\,\text{dB}$ to $0\,\text{dB}$ relative to the first source. The signal-to-noise ratio (SNR) was always 10 dB with reference to the first source, i.e. the SNR was from $-20\,\text{dB}$ to $10\,\text{dB}$ with reference to the second source.

The SRP equation (2.3) was computed over a grid of spherical coordinates with 129 different values of polar angle θ and 129 of azimuthal angle ϕ, where the reference system $(\hat{x}, \hat{y}, \hat{z})$ has its origin in the center of the array with $\hat{z}$ normal to the array plane and $\hat{x}$ pointing to the first sensor. After that, the two principal local maxima were located assuming a minimum separation of 2°.

To prove the performance of our source cancellation technique, the algorithm previously proposed (SRP SC) was used with IT_MAX = 2. In order to compute the probability of finding the second source, for both methods we assumed that the second source had been found if the localization error was fewer than 5°. In addition, we

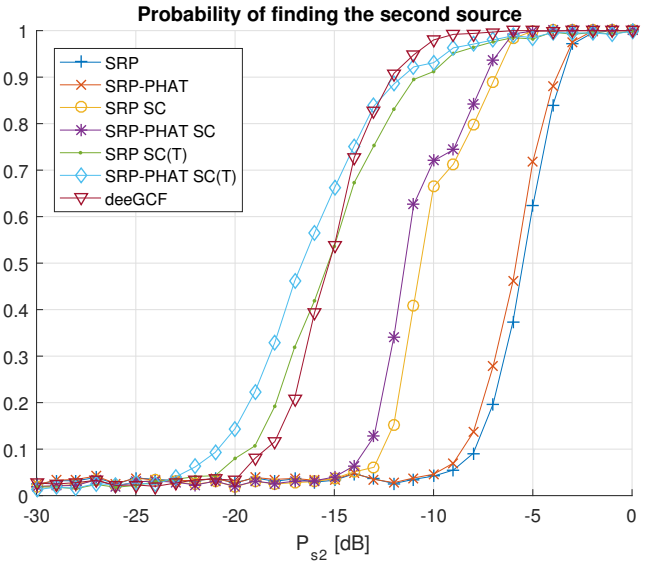


Figure 2.9: Probability of finding the second source in a 2 random sources scenario with $P_{s1} = 0\,\text{dB}$, $\text{SNR}_{s1} = -10\,\text{dB}$ and different values of P_{s2}.

repeated the simulation using the phase transform in the original GCCs (SRP-PHAT and SRP-PHAT SC) and with the truncation of (2.17) proposed in section 2.3.3: SRP SC(T) and SRP-PHAT SC(T). Finally, we implemented the source cancellation technique proposed in [129] (deeGCF). The authors did not provide any method to find the optimum parameters for their de-emphasis function, but for our array geometry we found that the best results were obtained for $p \in [0.5, 1]$ and that b has a low impact on the performance of the algorithm; we finally used $p = 0.75$ and $b = 2$.

As shown in Fig. 2.9, our algorithm suffers less degradation when the power of the second source decreases and, surprisingly, truncating (2.17) makes our proposal even more robust. When combining the proposed source cancellation technique with the SRP algorithm, we can locate sources about 5 dB weaker than when using only the SRP algorithm, and the coarse approximation proposed in section 2.3.3 achieves an additional gain of 5 dB. Finally, the results show that, as could be expected, the phase transform does not generate great improvements for uncorrelated white sources since it does not modify white signals.

The truncated source cancellation technique (SC(T)) seems to perform better than

the original (SC) because it removes more energy from the maps and sometimes applying the complete equation leaves residual energy around the source that may be identified as a new source. To solve this issue, it would be possible to modify the algorithm to not only repeat the source cancellation when the new maximum is in the same position as the original source, but also when it is closer than a threshold. Doing this would impose a trade-off between finding the weakest sources and the minimum necessary separation between them.

Multiplying the CC functions by a de-emphasis function as proposed in [129] gives a similar performance to the truncated version of our technique, but with our technique we do not need to fine-tune any parameters.

2.4.2.2 Uncorrelated speech sources

In order to analyze the performance of the proposed technique with more complex sources, we replaced the white Gaussian sources with segments of 1024 samples randomly taken from a database of speech signals without silences sampled at 44 100 Hz created by us from audio-book recordings in Spanish. We reduced IT_MAX to 1 as increasing it did not improve the results.

As shown in Fig. 2.10, the algorithm with a better performance in this scenario is our source cancellation technique applied over GCCs with PHAT, followed by its coarse approximation. As expected for speech sources, the phase transform improves the performance of all algorithms.

Contrary to other cases, there is a non-negligible probability that the SRP algorithm without PHAT cannot locate any of the sources (about 25% for $P_{s2} = P_{s1}$). This is due to the wide auto-correlation that some voice segments have, which leads to oversized peaks in the SRP maps. However, even in these cases where the estimation of the position of the first source is quite inaccurate, the proposed technique can eliminate the first source. Interestingly, the same issue that is responsible for the inaccurate localization of the first source, i.e. the wideness of the auto-correlation of the source, causes that the inaccuracy in the estimation of $\boldsymbol{\theta}_0$ does not lead to dramatic changes in the result of (2.17). Fig. 2.11 shows an example taken from a simulation where the second source was hidden by the first, but our technique was able to find it even after

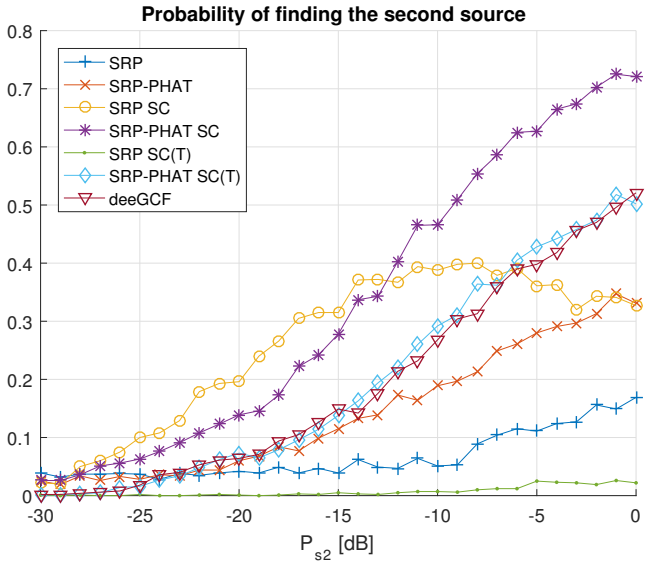


Figure 2.10: Probability of finding the second source in a 2 speech sources scenario with $P_{s1} = 0\,\text{dB}$, $\text{SNR}_{s1} = -10\,\text{dB}$ and different values of P_{s2}.

an inaccurate estimation of $\boldsymbol{\theta}_0$. In this situation, the cancellation technique proposed in [129] fails, since a notch function wide enough to eliminate the first source would also eliminate the second one.

The degradation SRP SC technique when the power of the second source approach the power of the first one is due to the fact that, when the two sources are too close, the second one is removed when the first one is canceled. This does not occur with SRP-PHAT SC since the phase transform narrows the wide of the peaks of the GCCs.

Although our cancellation technique is quite robust to errors in the estimation of the position of the first source, there are some cases where the two sources are too close and, if they have similar power, it removes both if used without PHAT. Applying the phase transform, it is possible to separate the two sources and, when our cancellation technique is applied, only the desired one is removed. However, we can see in Fig. 2.10 how SRP SC outperforms SRP-PHAT SC when the power of the second source decreases. Analyzing these cases, we found that in the original power maps (both with and without PHAT) the second source was completely hidden even when it was quite far from the first one. We also found that when applying the proposed technique, the

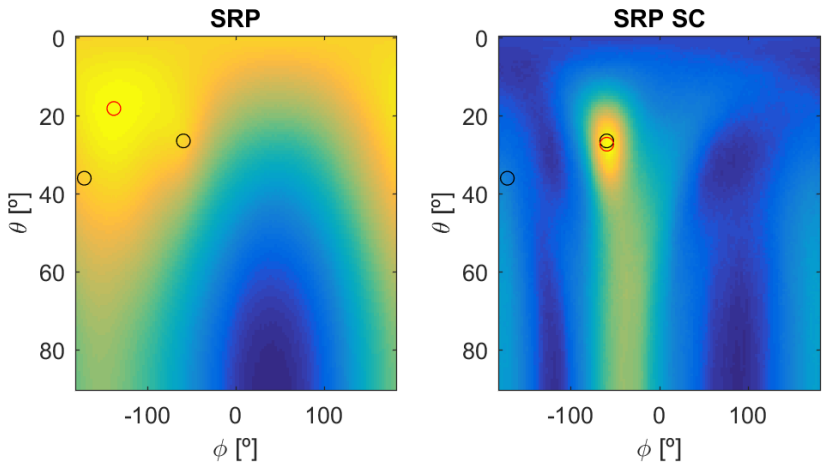


Figure 2.11: SRP power map resulting from the simulation of two speech sources with equal power and the SRP power map after applying our source cancellation technique at its maximum. The black circles indicate the actual position of the sources and the red circle the maximum of the map. The first source was successfully removed even after an inaccurate estimation of its position.

second source emerged from the SRP map but not from the SRP-PHAT map. This could be because the phase transform has been proven to approach the maximum likelihood solution in low-noise reverberant environments [132], but here the power of the second source is under the noise level ($P_N = P_{s1} - 10\,\text{dB}$). An example of this situation is shown in Fig. 2.12.

With speech signals, it is not feasible to use the coarse approximation over GCCs without PHAT. This is due, again, to the wideness of the auto-correlation of the speech segments, causing the coarse approximation to remove too much information from the GCCs which needs to be recovered with the last term of (2.17). As expected, the de-emphasis function also has this problem; we tried to modify its parameters but we could not improve its results.

2.4.2.3 Correlated speech sources

Finally, to test the effect of the correlation between sources, we repeated the previous simulation using the same signal (but with different power levels) for both sources, as if the second source was an echo of the first one. In this situation, the SRP without

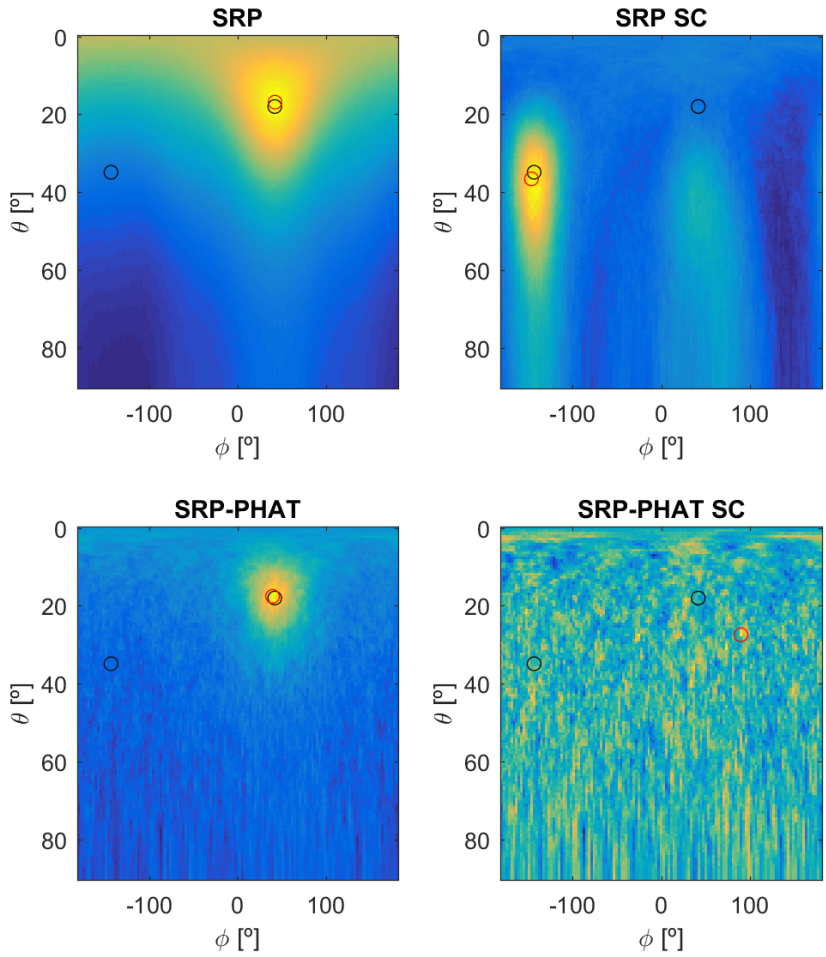


Figure 2.12: SRP and SRP-PHAT power maps resulting from the simulation of two speech sources with $P_{s2} = P_{s1} - 13\,\text{dB}$ and the SRP and SRP-PHAT power maps after applying our source cancellation technique. The black circles indicate the actual position of the sources and the red circle the maximum of the map. The first source was successfully removed both with and without PHAT, but the second source only appeared in the second case.

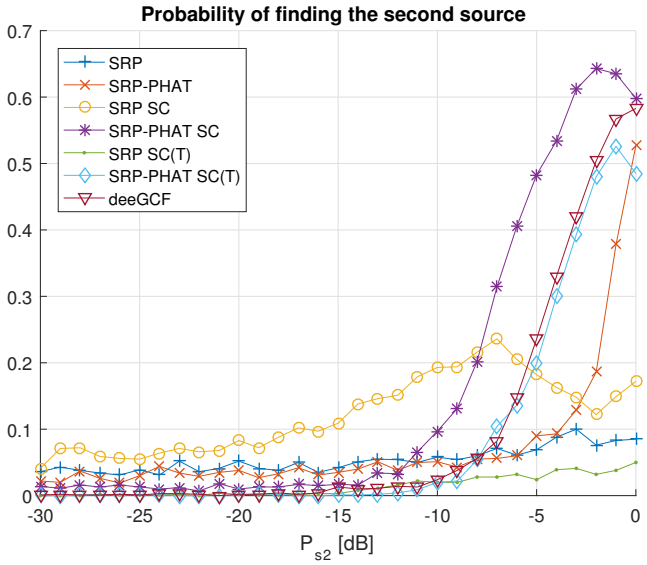


Figure 2.13: Probability of finding the second source in a 2 correlated speech sources scenario with $P_{s1} = 0\,\text{dB}$, $\text{SNR}_{s1} = -10\,\text{dB}$ and different values of P_{s2}.

PHAT is unable to find the first source and, even with PHAT, when the sources are too close and have similar power the maximum of the map is between them and it is impossible to locate them (this happens in about 30% of cases for $P_{s2} = P_{s1}$).

As shown in Fig. 2.13, the SRP-PHAT algorithm can find the second source only if both sources have similar power, but with the proposed source cancellation technique we significantly improve the dynamic range of the algorithm. Due to the strong correlation between the sources, the SRP maps present a third maximum in the middle, which leads to the poor performance of SRP SC when both sources have similar power. Fortunately, if we use our cancellation technique on this spurious maximum it disappears without affecting the maxima that correspond to the sources, so it would be possible to find the two sources if our technique is applied twice. If we apply the cancellation technique to a maximum that corresponds to a source, the spurious maximum is also removed, so this could be exploited to differentiate the spurious maxima from the real ones. When using the phase transformation, the spurious maximum is attenuated, making it easier to find the second source simply by canceling the maximum of the original SRP-PHAT map. As with previous scenarios,

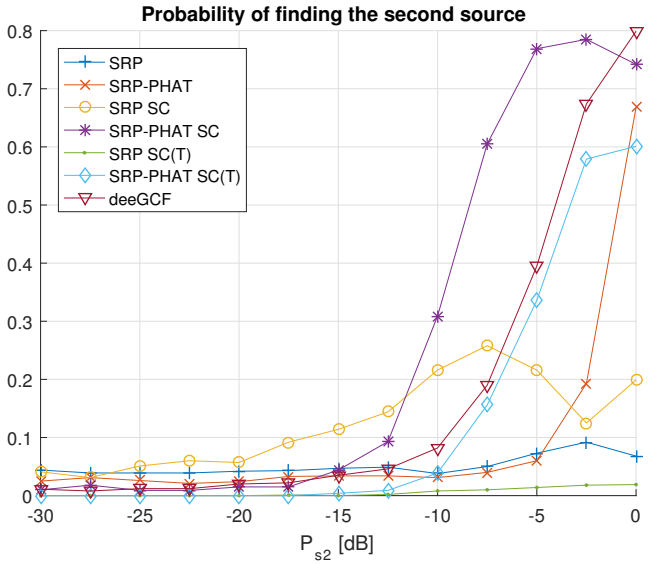


Figure 2.14: Probability of finding the second source in a 2 correlated speech sources scenario with $P_{s1} = 0\,\text{dB}$, $\text{SNR}_{s1} = -10\,\text{dB}$ and different values of P_{s2} using a longer window size ($K = 4096$).

the technique proposed in [129] performs similarly to the truncated version of our algorithm.

To test the effect of increasing the window size, we repeated the simulation with speech segments of 4096 samples and achieved the results shown in Fig. 2.14. As expected, using longer window sizes improves the performance of all the algorithms, but the relations between them remain constant.

2.4.3 Recordings

Finally, some recordings were made in a highly reverberant room (see Fig. 2.15) in order to validate the effectiveness of the algorithm in real environments. The dimensions of the room were $6\,\text{m} \times 8.5\,\text{m} \times 3\,\text{m}$ and we measured a reverberation time T_{60} of 1 second and a noise level of $55\,\text{dB}_{\text{SPL}}$. There was no acoustic foam in the walls or any other system to reduce reflections. Four loudspeakers were placed 80 cm apart, 4 meters in front of the array. The height of the array and the speakers were 1.05 m and 0.64 m, respectively.

Figure 2.15: Set-up of the recordings.

2.4.3.1 White sources

The power map in Fig. 2.16 (a) was obtained by playing white noise in only 2 loudspeakers, with 70 $\mathrm{dB_{SPL}}$ and 65 $\mathrm{dB_{SPL}}$ at the array position, and applying the conventional SRP-PHAT algorithm. After identifying the global maximum (which had an error of 2.04° regarding the first source position), the proposed technique (without truncation) was applied and we obtained the power map shown in Fig. 2.16 (b). The new global maximum was not the second loudspeaker, but the floor reflection of the first one. Applying our algorithm again to cancel this new direction, we obtained Fig. 2.16 (c), where the maximum is the reflection at the right wall. Finally, after canceling this direction, we found the second loudspeaker in Fig. 2.16 (d) with an error of only 1.97°. In order to locate the second source directly from the original SRP-PHAT map, we would have had to take its third local maxima as the second source (the second maxima was the floor reflection) which was at the same position as the maximum of Fig. 2.16 (d).

Fig. 2.17 and Fig. 2.18 show the same process for 3 and 4 loudspeakers, respectively. In all cases, we can see the ability of our algorithm to cancel the stronger sources, but sometimes some reflections are found before the weakest sources are located. In the first scenario (see Fig. 2.17), the localization error of each source were 1.63°, 2.01° and 0.81°, while the second maximum in the original map was 2.10° apart from the second source and the peak closest to the third source was the 9^{th} most energetic peak (so it would hardly have been considered a source if we did not know there had to be one in that direction) and its error was 2.11°. In the second scenario

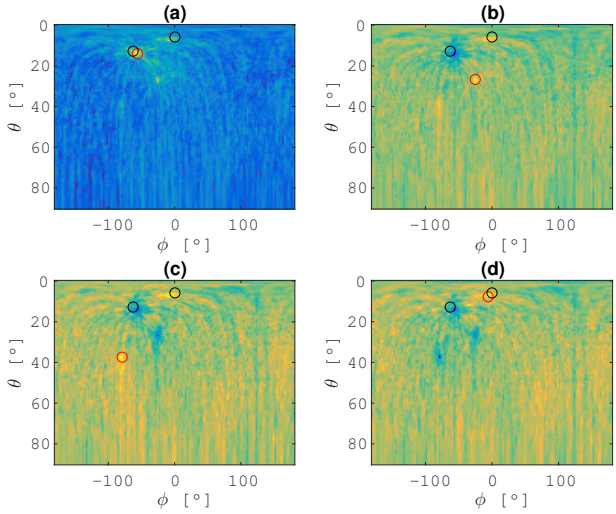


Figure 2.16: Power maps obtained with SRP-PHAT (a) and applying our algorithm at the position of the maximum of the previous map (b, c, d), for 2 loudspeakers with 70 $\mathrm{dB_{SPL}}$ and 65 $\mathrm{dB_{SPL}}$ of white noise at the array position. The black circles are the actual positions of the loudspeakers and the red circle the maximum of each power map. After removing two reflections of the first source, we found the second source in (d).

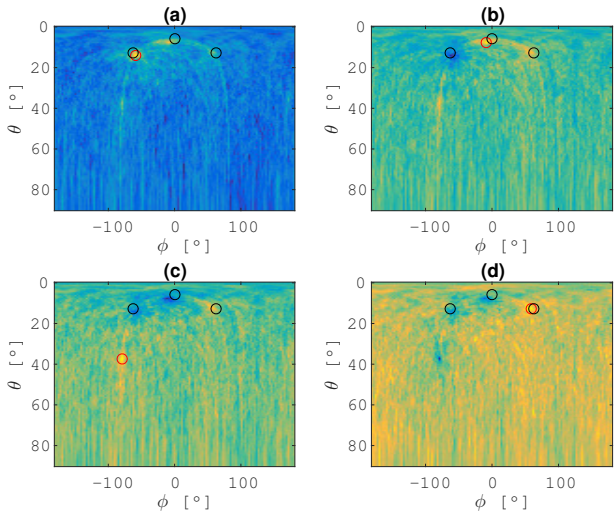


Figure 2.17: Power maps obtained with SRP-PHAT (a) and applying our algorithm at the position of the maximum of the previous map (b, c, d), for 3 loudspeakers with 75 $\mathrm{dB_{SPL}}$, 70 $\mathrm{dB_{SPL}}$ and 65 $\mathrm{dB_{SPL}}$ of white noise at the array position. The black circles are the actual positions of the loudspeakers and the red circle the maximum of each power map.

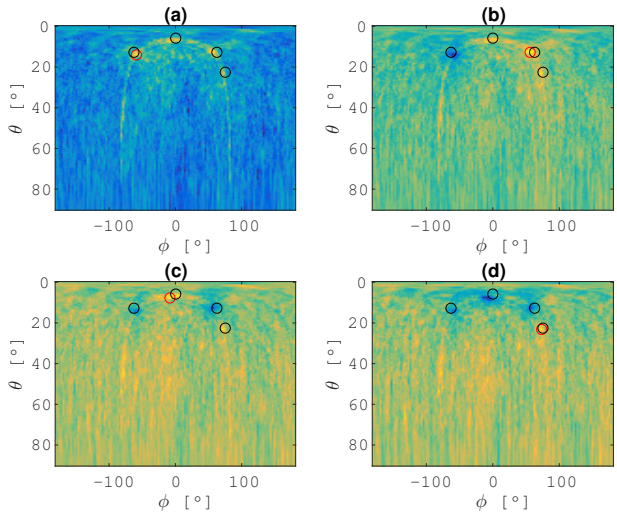


Figure 2.18: Power maps obtained with SRP-PHAT (a) and applying our algorithm at the position of the maximum of the previous map (b, c, d), for 4 loudspeakers with 65 $\mathrm{dB_{SPL}}$ of white noise at the array position. The black circles are the actual positions of the loudspeakers and the red circle the maximum of each power map.

(see Fig. 2.18), the errors were 1.63°, 2.10°, 1.43° and 1.21° while the maximum of the original SRP-PHAT map closest to the sources was at the same positions, but to find the fourth source we would have needed to take into account the 6 strongest maxima.

2.4.3.2 Speech sources

Finally, we replaced the white noise recordings with speech recordings. Due to the non-stationary nature of the speech signal, the power generated by each loudspeaker may be strongly different in each 1024-sample frame, even after removing the silences from the recordings.

Fig. 2.19 and Fig. 2.20 had been obtained using 2 and 3 loudspeakers respectively and selecting a frame when the sources had power enough to locate all of them. In order to implement a complete speaker localization system, some tracking algorithm would be needed so it can deal with the fluctuations of the power of the sources and with the presence of estimated DOAs corresponding to reflections instead of sources (this issue is common to most of the DOA estimation algorithms when working with

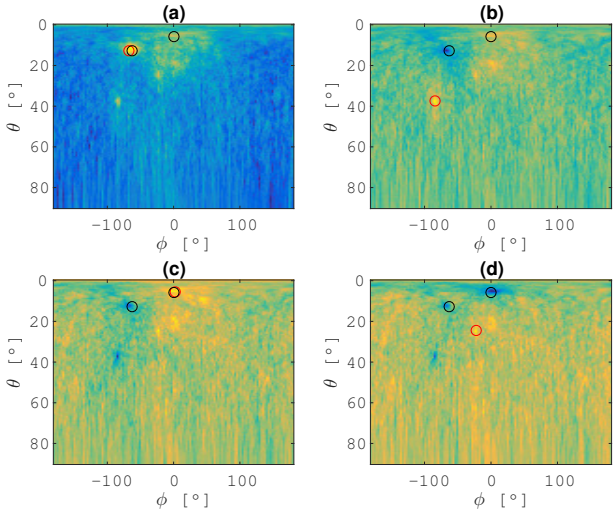


Figure 2.19: Power maps obtained with SRP-PHAT (a) and applying our algorithm at the position of the maximum of the previous map (b, c, d), for 2 loudspeakers playing speech signals. The black circles are the position of the loudspeakers and the red circle the maximum of each power map.

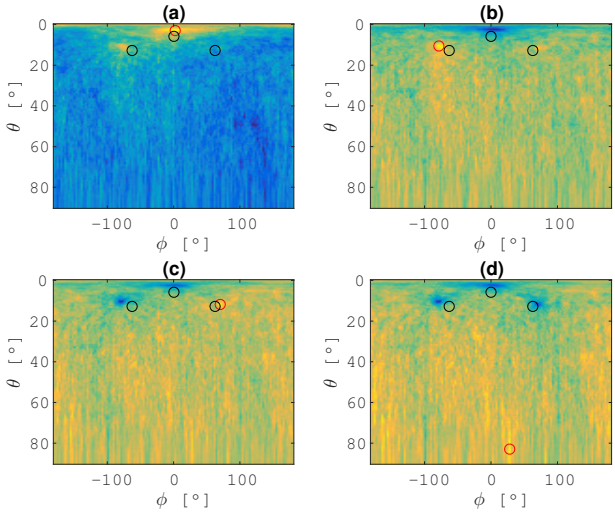


Figure 2.20: Power maps obtained with SRP-PHAT (a) and applying our algorithm at the position of the maximum of the previous map (b, c, d), for 3 loudspeakers playing speech signals. The black circles are the actual positions of the loudspeakers and the red circle the maximum of each power map.

speech signals and this is the reason why tracking algorithms are implemented).

The localization error in Fig. 2.19 were 1.04° and 0.38°. The distance between the second source and the 4^{th} peak of the original SRP-PHAT map (the closest one) was 0.99°, but it was surrounded by multiple maxima of similar power, so it would have been difficult to choose which of them are sources and which are not if we had not known the number of sources, while with our technique most of these maxima were cleaned. In Fig. 2.20, the localization error was 3.84°, 3.07° and 1.77° while using only the SRP-PHAT algorithm we would have only been able to find two sources (with errors 2.58° and 3.07°) since none of the 20 strongest peaks of the original map were close to the 3^{rd} source.

We can see with these examples how our technique does not improve the accuracy of the location of the sources that could be found with the SRP-PHAT algorithm but makes it possible to find sources that were hidden in the original maps. The main issue in using our cancellation technique (or the proposed in [129]) for speaker localization applications would be dealing with the estimated DOAs which correspond to reflections of an already found source rather than the direct path of a new one. An approach to classify the reflections could be using a beamformer steered to each estimated DOA and studying the correlation between the sources received from each direction, but the computational cost would be prohibitive for most applications. A setup that could facilitate the classification of the reflections would be arranging multiple arrays in the room, so only the DOAs of the direct path from each source to each array would intersect at one point since each array would receive different reflections; another approach could be using near-field arrays whose spherical wave propagation model is more robust against wall reflections. Finally, a survey on the multi-source tracking algorithms should be performed to find those that better fit with the DOA estimates that our cancellation technique provides.

2.5 Conclusions

Sound source localization (SSL) is a problem that has been extensively studied using classical signal processing techniques and a broad range of techniques have been pro-

posed to solve it, each one with different advantages and disadvantages. Among them, SRP-PHAT provides an efficient algorithm to compute acoustic power maps whose peaks allow us to estimate the DOAs of the sound sources recorded with a microphone array. Despite having many advantages, like being more robust to reverberation than other techniques, SRP-PHAT still has some drawbacks, like the need of high resolution maps to ensure that their peaks really correspond to the DOAs, which increases its computational cost, and the issue of strong sources masking the weakest ones, which complicates its use for multi-source applications.

In order to solve this second issue, a new technique has been presented that allows broadband sensor arrays to cancel the effect of a source at an affordable computational cost. This technique is an extension of previous narrowband techniques with an efficient implementation that can be used iteratively to cancel several sources. It is compatible with other broadband techniques, like the SRP-PHAT, or with most of the multi-source tracking algorithms already proposed. It can be used in arrays with any geometry, no assumptions about the sources need to be made and it can deal with correlated sources.

The proposed algorithm is capable of locating multiple sources even when the second source is much weaker than the first one. The simulations made with a microphone array obtained an increase between 5 dB and 10 dB in the dynamic range in comparison with the conventional SRP-PHAT algorithm and the recordings demonstrate that the algorithm also works in real environments. This is especially important when the sources are not sparse and the weakest source will always be masked, both in time and frequency, by the strongest one. We have proved through simulations that the coarse approximation of our algorithm has a similar performance to the technique proposed in [129] but without needing to fine-tune any parameter and that, working with the complete equation, we clearly outperform it. This coarse approximation, and the technique proposed in [129], has a good performance for white sources but suffers a strong degradation when dealing with more complex sources like speech. It may be useful in applications where we have control over the sound sources, such as in the acoustic characterization of rooms, or when we want to locate noise sources that we already

know are white, but for speaker localization applications the complete equation should be employed.

It should be noted that the new technique cancels the desired direction, its diffraction lobes, and the artifacts generated by the correlation between sources. But, when used in a reverberant real acoustic environment, it does not eliminate the room reflections generated by the same source. In cases where the aim is only to detect sound sources, further stages will be required in order to classify and detect the reflections. However, if the objective is to study the acoustics of a room, the reflections need to be located and, therefore, must not be eliminated.

3

An infinite-size synthetic dataset for sound source localization and tracking

Due to the difficulty of obtaining an accurately hand-labeled dataset of moving sources recorded with microphone arrays, we opted to train our models with simulated signals. Another approach might have been using measured room impulse responses (RIRs) convolved with speech signals, but this would have reduced the amount of different acoustic conditions seen by the model during training, increasing the possibility of overfitting to those conditions and not generalizing.

Since we simulate our training signals from completely random parameters, we can simulate them as they are needed during the training obtaining an infinite-size dataset. This makes the training slower, but has two important advantages: i) we increase the different acoustic conditions that the models see during the training, which has proven to increase the model accuracy [63], and ii) we have higher flexibility to modify the probability distribution of the parameters of the simulation, such as the signal-to-noise ratio or the reverberation time, during training so we can perform curriculum learning strategies [134].

In this chapter, we first describe the procedure that we follow to simulate moving sources in section 3.1 and then we present the GPU implementation of the image

source method (ISM) that we used to accelerate these simulations.

This chapter includes the reproduction of figures and text fragments from [V] and [IV] with the permission of the copyright holders.

3.1 Moving sources simulation

3.1.1 Trajectory generation

In order to generate trajectories that can be used to train tracking systems, we need to randomly generate continuous trajectory points that can be tracked, but we also need to ensure that they have enough diversity to avoid the network to learn how they are generated and overfit to them.

In order to do so, after randomly choosing the room dimensions, we randomly select two points within the room boundaries to be the starting ($\mathbf{p}_0 = [p_{x0}, p_{y0}, p_{z0}]^T$) and ending ($\mathbf{p}'_L = [p'_{xL}, p'_{yL}, p'_{zL}]^T$) points of the trajectory and add to the straight line that connects them a sinusoidal function in each axis with random frequencies ($\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T$) and amplitudes ($\mathbf{A} = [A_x, A_y, A_z]^T$) ensuring that no more than 2 oscillations are performed during the trajectory in each axis and that the amplitude is low enough to avoid the source to exit the room:

$$\mathbf{p}_i = \mathbf{p}_0 + \frac{i}{L-1}(\mathbf{p}'_L - \mathbf{p}_0) + \mathbf{A} \circ \sin(\boldsymbol{\omega} i), \tag{3.1}$$

where L is the number of points of the trajectory, $\circ$ stands for the pointwise product and the sin function also operates pointwise. Although the generation model is quite simple, it generates quite diverse trajectories (some examples are shown in Fig. 3.1) and, since the network only sees the azimuth and elevation coordinates and has a limited temporal perceptive field, the model should not overfit to it. In order to confirm this, we always test our models with the recordings of the LOCATA dataset (see section 4.1.4.3).

With this generation algorithm, we do not have direct control over the speed of the sources and it depends on the size of the rooms and the duration of the trajectories.

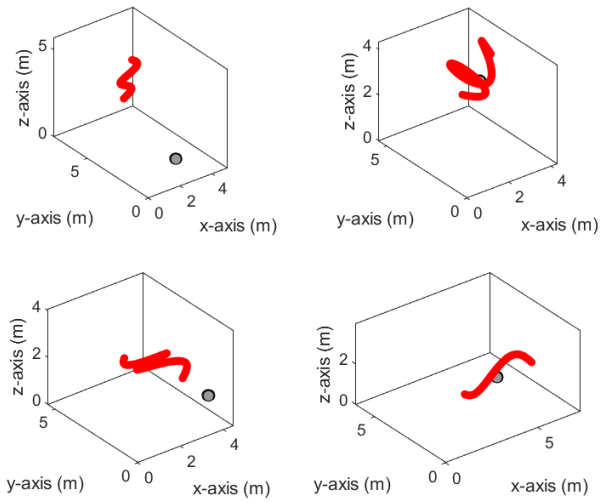


Figure 3.1: Examples of source trajectories used to train the models. The red dots are the trajectory points and the gray points represent the microphones.

The histogram in Fig. 3.2 shows the distribution of the source speed generated by the parameters employed in all the experiments presented in this thesis.

3.1.2 Trajectory simulation

As shown in Fig. 3.3, we use LibriSpeech utterances as sound sources. The LibriSpeech corpus [135] contains 960 hours of speech sampled at $f_s = 16\,\text{kHz}$ extracted from audiobooks. Although audiobooks could be expected to contain quite clean speech signals, we found that some of them have a strong background noise that, after filtered by the RIRs, would be located in the same position as the source and would facilitate its localization and tracking in silent segments. To prevent our network learning to exploit this fact, which will not be present in actual recordings, we use the WebRTC voice activity detector (VAD) [136] to detect silent segments and clean them by completely removing the signal in those frames.

The sizes of the rooms are randomly selected from the range $3\,\text{m} \times 3\,\text{m} \times 2.5\,\text{m}$ to $10\,\text{m} \times 8\,\text{m} \times 6\,\text{m}$ and the array is randomly placed inside the room, being restricted to have a separation from the walls of a 10% of the room size in each dimension and

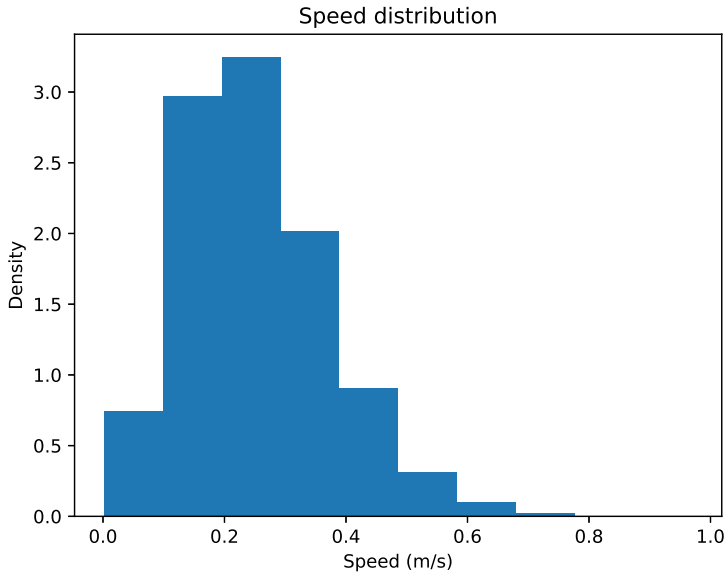


Figure 3.2: Distribution of the source speed in the synthetic dataset.

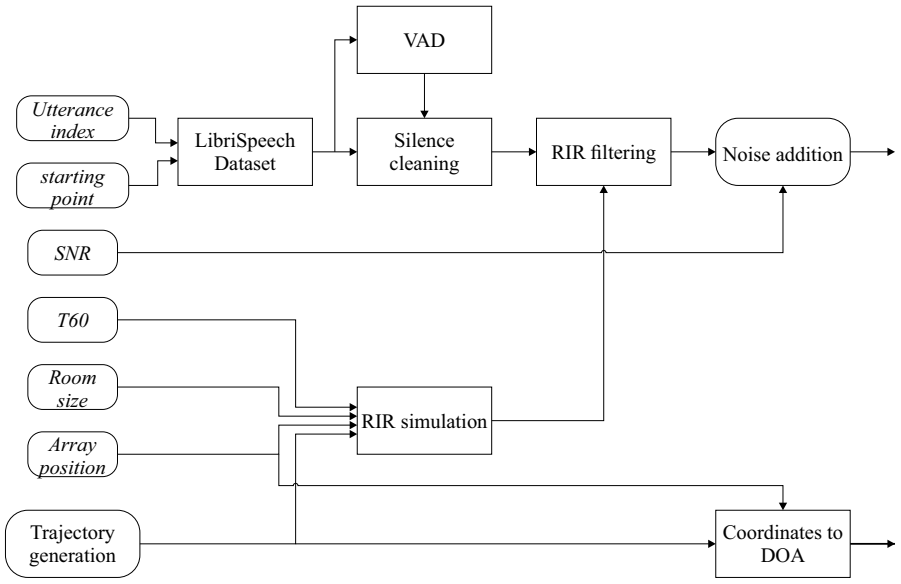


Figure 3.3: Dataset generation process. Italic letters represent variables and regular letters represent processes. Right-angled boxes represent deterministic processes and round boxes represent stochastic variables or processes.

be in the lower half of the room for the vertical axis. The signal-to-noise ratio (SNR) and reverberation time (T_{60}) are also randomly selected from the ranges 5 dB to 30 dB and 0.2 s to 1.3 s respectively. Uniform distributions over the specified ranges are used for all the random parameters of the dataset.

In order to be able to evaluate the models trained with this synthetic dataset with actual recordings, we use the same microphone geometry as the microphone array with 12 sensors designed to be mounted over an NAO robot head employed in the LOCATA dataset [137]. The minimum and maximum inter-microphone distance of the array are 1.3 cm and 12.1 cm and the actual position of each microphone can be found in [138].

Once we have the position of the trajectory points and the microphones and the properties of the room, we can use the image source method (ISM) to obtain the RIRs for every trajectory position and convolve the source signal with them using the overlap-add method to obtain the sound signals received at the microphone positions. After that, we finally add an omnidirectional Gaussian noise to obtain the desired SNR. In order to compute the noise power needed to obtain the desired SNR, we compute the average energy of the sound signal during the whole trajectory excluding the silent segments (including the silent segments in the average would have led to lower noise levels for the same SNR).

3.1.3 Multi-source scenes

For the acoustic scenes with a varying number of sources employed to train the models presented in chapter 5, we need to model how new sources appear (born) and how existing sources disappear (die). In order to allow several sources to appear in the same time frame, we model the number of sources that are born in a time frame using a Poisson distribution:

$$P\{k \textit{ new sources born in the time frame } t\} = \frac{e^{-\lambda}\lambda^k}{k!}, \tag{3.2}$$

where λ is the rate of the distribution, which is equal to its mean and its variance. This is similar to the birth assumption typically used in multi-object trackers based on probabilistic models [139, 140] but, in order to control the maximum number of

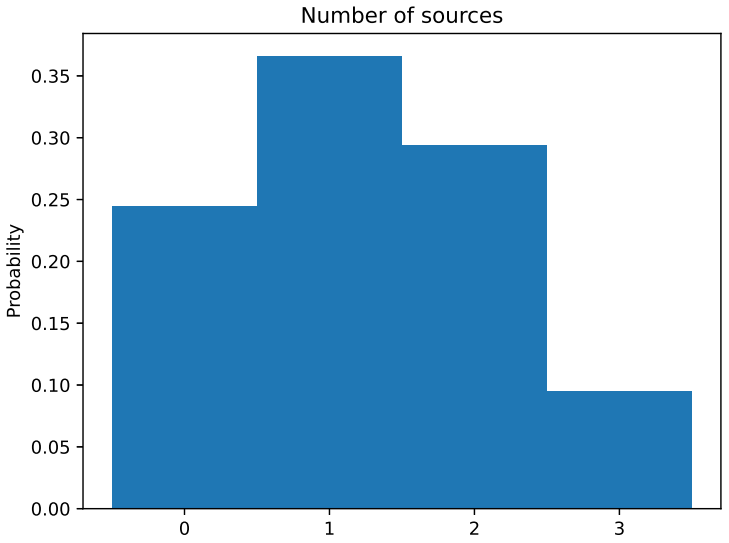


Figure 3.4: Distribution of the number of sources in every time frame.

sources simultaneously active, we made the birth rate λ dependent on the number of sources active in the previous frame and truncated the probability of (3.2) to avoid exceeding the chosen maximum of sources. In the experiments shown in chapter 5, we allowed up to 3 sources simultaneously active and employed a birthrate of 0.06, 0.04, or 0.02 depending if the number of active sources in the previous frame was 0, 1, or 2.

Following again the typical assumptions used in multi-object trackers, we model the death probability of every source in every time frame with a Bernoulli distribution, but in order to avoid sources that lived too short, we added a minimum life length:

$$P\{\textit{the active source } i \textit{ dies in the time frame } t\} = \begin{cases} p & t - t_i^b > T_{min} \\ 0 & else \end{cases}, \tag{3.3}$$

where p is the death probability, T_i^b the time frame when the source i was born and T_{min} the minimum life of the sources. In the experiments shown in chapter 5 we used $p = 0.02$ and $T_{min} = 200\,\text{ms}$.

This model, with the parameters used in chapter 5, generates the distribution shown in Fig. 3.4 for the number of sources in every time frame.

3.2 Room impulse response simulation with GPU acceleration

The technique presented in the previous section allowed us to generate infinite acoustic scenes to train our models, but the simulation of the room acoustics was too slow to simulate them as they were needed during the training. This issue has typically been overcome by pre-generating enough acoustic scenes to train the models, but that would have meant fixing the size of the dataset. Oppositely to this approach, we opted by exploiting the power of the graphics processing units (GPUs) to implement a faster version of the image source method (ISM).

3.2.1 Introduction

The simulation of the acoustics of a room is needed in many fields and applications of audio engineering and acoustic signal processing, such as training robust speech recognition systems [141] or training and evaluating sound source localization [142] or speech enhancement [143] algorithms. Although there are many low-complexity techniques to simulate the reverberation effect of a room in real time, such as the classic Schroeder Reverberator [144], some applications require an accurate simulation of the reflections causing the reverberation. The information of all those reflections is gathered in the room impulse response (RIR) between the source and the receiver positions, which allows the simulation of the reverberation process by filtering the source signal with it. The image source method (ISM) is probably the most used technique for RIR simulation due to its conceptual simplicity and its flexibility to modify parameters such as the room size, the absorption coefficients of the walls, and the source and receiver positions. We can simulate any level of reverberation by modifying the room size and the absorption coefficients, but the computational complexity of the algorithm grows fast as the number of reflections to simulate increases.

Initially developed to support the graphics computations of video games, graphics processing units (GPUs) are today one of the best and cheapest ways to increase the speed of many algorithms that can be expressed in a parallel form. Despite parallelizing most of the stages of the ISM is quite straightforward, to the best of our knowledge,

Table 3.1: Comparison of some state-of-the-art ISM implementation.

	RIR generator [146]	pyroomacoustics [147]	[133]	[145]	gpuRIR
Open source library (language)	✓(Matlab and Python)	✓(Python)	✓(Matlab)	✗	✓(Python)
Implementation language	C++	Python and C++	Matlab	CUDA	CUDA
Fractional delays	✓	✓	✓	✓	✓
Negative reflection coefficients	✗	✗	✓	✗	✓
Diffuse reverberation model	✗	✗	✓	✗	✓
GPU acceleration	✗	✗	✗	✓	✓
Lookup table implementation	✗	✓	✗	✗	✓
Mixed precision implementation	✗	✗	✗	✗	✓

only [145] proposed to implement it in GPUs. Although they showed that using GPUs it was possible to speed up the RIR simulations, they did not provide the code of their implementation and the acoustic signal processing and audio engineering communities have not embraced their approach. In addition, they used an overlap-add strategy with atomic operations to combine the contributions of each image source, which strongly reduces the level of parallelism. In this chapter, we present a new GPU implementation with a higher degree of parallelism, which allows us to achieve higher speed-ups with cheaper GPUs. Motivated by the performance boost obtained with the use of lookup tables (LUTs) in the CPU implementations, we also study its use in our GPUs implementation. Finally, we propose a 16-bit precision implementation which can increase even more the simulation speed in the newer GPUs with mixed precision support.

Table 3.1 shows some state-of-the-art implementations of the ISM and compares some of their main characteristics. We can see how our implementation is the only one with GPU acceleration that is available as a free and open source library and how it includes some features (further explained in section 3.2.2.2) that are not included in other Python libraries. Using our library does not require any knowledge about GPU programming, but just having a CUDA compatible GPU and the CUDA Toolkit, and it can be installed and used as any CPU RIR simulation library.

3.2.2 The Image Source Method (ISM)

The method of images has been widely used in many fields of physics to solve differential equations with boundary conditions, but its application for RIR estimations was originally proposed by Allen and Berkley [61]. In this section, we first review

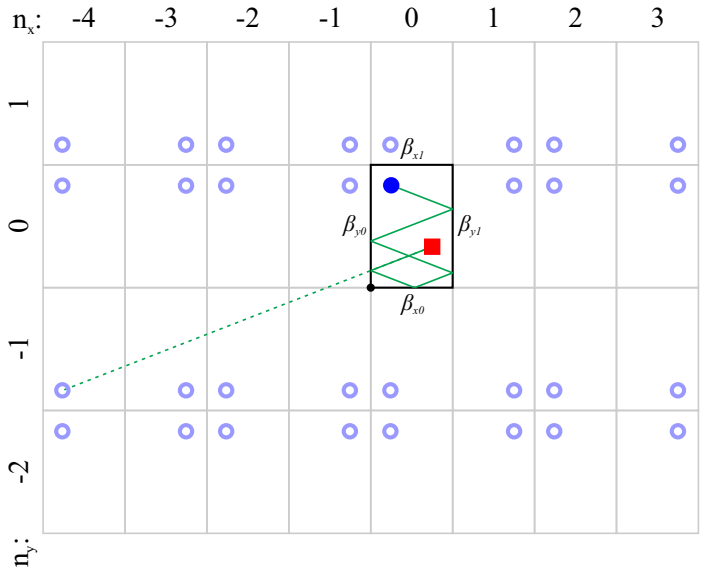


Figure 3.5: Image sources for a two-dimensional room. The red square and the blue dot represent the receiver and the source and the blue circumferences represent the image sources. The solid green line represents one of the multiple reflection paths and the dashed green line is the direct path of the equivalent image source. The black dot is the origin of the coordinates system.

their original algorithm and then explain some of the improvements that have been proposed to improve both its accuracy and computational performance.

3.2.2.1 Original Allen and Berkley algorithm

The main idea behind the ISM is to compute each wave-front that arrives at the receiver from each reflection off the walls as the direct path received from an equivalent (or image) source. In order to get the positions of these image sources, we need to create a 3D grid of mirrored rooms with the reflections of the room in each dimension; as shown in Fig. 3.5 simplified to 2D for an example.

If the number of images we want to compute for each dimension is N_x, N_y and N_z, then we define a grid $\mathcal{N}$ of image sources $\mathbf{n} = (n_x, n_y, n_z) : \lceil -N_x/2 \rceil \leq n_x < \lceil N_x/2 \rceil, \lceil -N_y/2 \rceil \leq n_y < \lceil N_y/2 \rceil$ and $\lceil -N_z/2 \rceil \leq n_z < \lceil N_z/2 \rceil$ (where $\lceil \cdot \rceil$ stands for the round toward positive infinity operator). The coordinates of the position of each image $\mathbf{p_n} = (x_\mathbf{n}, y_\mathbf{n}, z_\mathbf{n})$ are calculated using its grid indices, the position of the source

and the dimensions of the room; as an example, the component x would be calculated as

$$x_{\mathbf{n}} = \begin{cases} n_x L_x + x_s & \text{if } n_x \text{ is even} \\ (n_x + 1)L_x - x_s & \text{if } n_x \text{ is odd} \end{cases}, \tag{3.4}$$

where $\mathbf{L} = (L_x, L_y, L_z)$ is the size of the room and $\mathbf{p}_s = (x_s, y_s, z_s)$ is the position of the original source. The y and the z coordinates can be obtained similarly.

The distance $d_{\mathbf{n}}$ from the image source $\mathbf{n}$ to a receiver in the position $\mathbf{p}_r = (x_r, y_r, z_r)$, and therefore the delay of arrival $\tau_{\mathbf{n}}$, is trivial if we know the image source position:

$$d_{\mathbf{n}} = ||\mathbf{p}_r - \mathbf{p}_s||, \tag{3.5}$$

$$\tau_{\mathbf{n}} = \frac{d_{\mathbf{n}}}{c}, \tag{3.6}$$

where $||\cdot||$ denotes the Euclidean norm and c is the speed of sound.

In order to calculate the amplitude with which the signals from each image source arrive at the receiver, we need to take into account the reflection coefficients of the walls of the room. We define β_{x0} as the reflection coefficient of the wall parallel to the x axis closest to the origin of the coordinates system and β_{x1} as the farthest; β_{y0}, β_{y1}, β_{z0} and β_{z1} are defined equivalently. Finally, if we define $\beta_{\mathbf{n}}$ as the product of the reflection coefficients of each wall crossed by the path from the image source $\mathbf{n}$ to the receiver, its amplitude factor will be

$$A_{\mathbf{n}} = \frac{\beta_{\mathbf{n}}}{4\pi \cdot d_{\mathbf{n}}}. \tag{3.7}$$

Knowing the amplitude and the delay for each image, we can easily obtain the RIR as the sum of the contribution of each image source:

$$h(t) = \sum_{\mathbf{n} \in \mathcal{N}} A_{\mathbf{n}} \cdot \delta(t - \tau_{\mathbf{n}}), \tag{3.8}$$

where $\delta(t)$ is the Dirac impulse function.

3.2.2.2 Improvements to the original algorithm

Fractional delays

In order to implement (3.8) in the digital domain, we need to deal with the fact that the values of $\tau_{\mathbf{n}}$ may not be multiples of the sampling period. The original algorithm proposed to just approximate the fractional delays by the closest sample, however, the error introduced by this approximation is too high for some applications, such as sound source localization (SSL) with microphone arrays. In [148], Paterson proposed to replace the Dirac impulse function with a sinc windowed by a Hanning function:

$$\delta'(t) = \begin{cases} \frac{1}{2}\left(1 + \cos\frac{2\pi t}{T_\omega}\right)\mathrm{sinc}(2\pi f_c t) & \text{if } -\frac{T_\omega}{2} < t < \frac{T_\omega}{2} \\ 0 & \text{otherwise} \end{cases}, \tag{3.9}$$

where f_c is the cut-off frequency, T_ω is the window length, and the sinc function is defined as $\mathrm{sinc}(x) = \sin(x)/x$. This is motivated by the low pass anti-aliasing filter that would be used if the RIR was recorded with a microphone in the real room. A window duration of $T_\omega = 4\,\mathrm{ms}$ and a cut-off frequency equal to the Nyquist frequency, i.e. $f_s/2$, are typically used.

Using the Paterson approach with $T_\omega = \infty$ is equivalent to computing (3.8) in the frequency domain as the sum of complex exponential functions as proposed in [149] [150], but using shorter window lengths reduces the computational complexity of the algorithm.

Negative reflection coefficients

Using positive reflection coefficients as proposed in [61] generates a low-frequency artifact that must be removed using a high-pass filter. In addition, while a RIR recorded in a real room has both positive and negative peaks, all peaks generated by the ISM are positive. Using negative reflection coefficients as proposed in [150] solves both problems without the need for adding any posterior filter to the ISM algorithm.

Diffuse reverberation

In order to properly simulate a RIR, we need to use values of N_x, N_y, and N_z high enough to get all the reflections that arrive in the desired reverberation time. Since

the delays of the signals of each image source are proportional to their distance to the receiver, and the distance is to the image index, the number of images to calculate for each dimension grows linearly with the reverberation time, and, therefore, the number of operations in (3.8) grows in a cubic way.

A popular solution to allow the simulation of long reverberation times in a reasonable time is decomposing the RIR in two parts: the early reflections and the late, or diffuse, reverberation. While the early reflections need to be correctly simulated with the ISM method to avoid losing spatial information, the diffuse reverberation can be modeled as a noise tail with the correct power envelope. In [133], Lehmann and Johansson propose using noise with logistic distribution and the technique introduced in [151] to predict the power envelope.

Although the technique presented in [151] generates better predictions of the power envelope obtained in real rooms, its computational complexity is quite high. Therefore, for the sake of computational efficiency, we decided to use a simple exponential envelope following the popular Sabine formula [152]. According to this model, the reverberation time T_{60} that takes for a sound to decay by 60 dB in a room, is

$$T_{60} = \frac{0.161V}{\sum S_i \alpha_i}, \tag{3.10}$$

where V is the volume of the room and S_i and $\alpha_i = 1 - \beta_i^2$ are the surface area and the absorption coefficient of each wall[1]; and the power envelope of the RIR is

$$P(t) = \begin{cases} A \exp\left(\log_{10}\left(\frac{T_{60}}{20}\right)(t - t_0)\right) & \text{if } t > t_0 \\ 0 & \text{otherwise} \end{cases}. \tag{3.11}$$

Therefore, knowing T_{60}, we can easily estimate A from the early reflections simulated with the ISM and then multiply the logistic-distributed noise by $\sqrt{P(t)}$ to simulate the diffuse reverberation.

[1]It should be noted that, as done in [61], we are defining the absorption ratio α as a quotient of sound intensities (energies) while the reflection coefficient β is defined as a quotient of pressures (amplitudes).

Table 3.2: Kernels and functions of the CUDA implementation.

CUDA functions	Description	Time (%)
calcAmpTau_kernel	Equations (3.6) and (3.7)	0.68
generateRIR_kernel	Sincs computation and initial sum (3.8)	90.34
reduceRIR_kernel	Parallel sum (3.8)	1.07
envPred_kernel	Power envelope prediction	0.03
generate_seed_pseudo	cuRAND function (diffuse reverberation)	7.78
gen_sequenced	cuRAND function (diffuse reverberation)	0.01
diffRev_kernel	Diffuse reverberation computation	0.01
CUDA memcpy	[CPU to GPU]	0.00
CUDA memcpy	[GPU to CPU]	0.06

3.2.3 Parallel implementation

As shown in Fig. 3.6, the parallel computation of the delays and the amplitudes of arrival for the signals from each image source and their sinc functions is straightforward since there are not any dependencies between each image source, and computing RIRs for different source or receiver positions in parallel is also trivial. However, the parallelization of (3.8) involves more problems, as the contributions of all the image sources need to be added into the same RIR.

It is worth mentioning that, though it would be possible to compute RIRs from different rooms in parallel, we choose to implement only the parallelization of RIRs corresponding to the same room. This was because the number of image sources to be computed depends on the room dimensions and the reverberation time and, to compute different rooms in parallel, we would have needed to use the worst-case scenario (i.e. the smallest room and higher reverberation time) for all of them, which would have decreased the average performance.

In order to implement the ISM in GPUs, we decided to use the Nvidia™'s Compute Unified Device Architecture (CUDA) [153] and divide our code into the kernels[1] listed in Table 3.2. For illustrative purposes, we show in Table 3.2 the average proportion of time employed by each kernel to compute a standard case of 6 RIRs with $T_{60} = 1\,\mathrm{s}$ using the ISM method for the 250 first milliseconds and the diffuse model for the

[1] A CUDA kernel is a function that, when is called, is executed N times in parallel by N different CUDA threads in the GPU. For more details, see the CUDA programming guide: `https://docs.nvidia.com/cuda/cuda-c-programming-guide/`

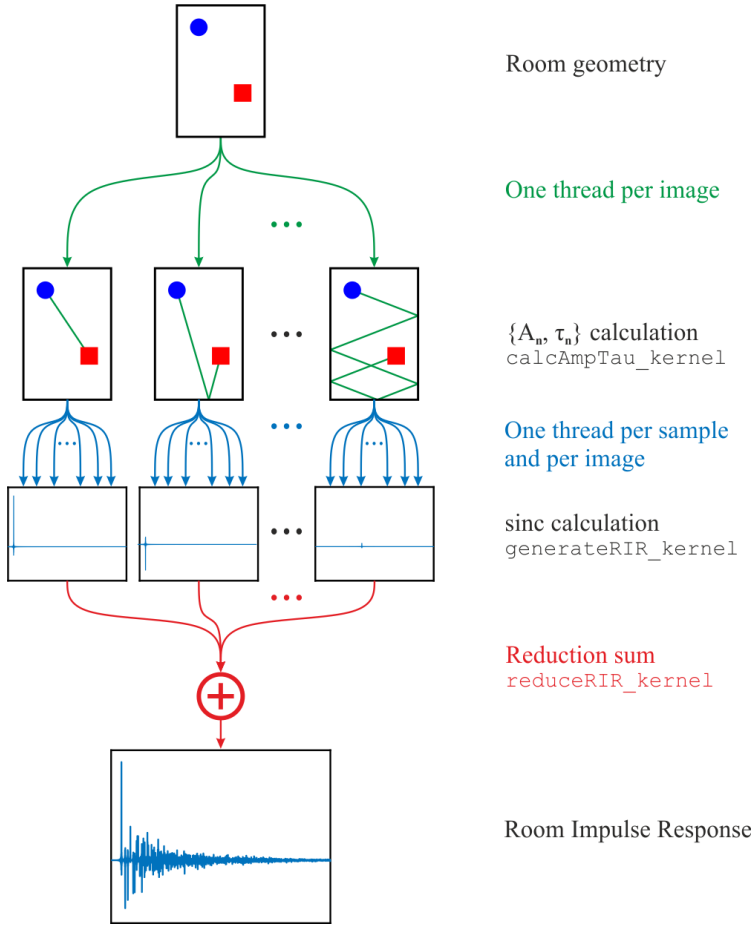


Figure 3.6: ISM parallel implementation. Our library actually computes some of the sincs sequentially, which leads to more efficient memory use. The reduction sum is detailed in Fig. 3.7

following 750ms using a Nvidia™ GTX 980Ti. It can be seen how the bottleneck is located at the beginning of the computation of (3.8), which is due to the high amount of sinc functions that are needed to be computed. The following sections provide further details about the implementation of the different parts of the algorithm.

3.2.3.1 Amplitudes and delays computation

For computing (3.6) and (3.7), we use `calcAmpTau_kernel`, which computes sequentially each RIR but parallelizes the computation for each image source. Although parallelizing the computations for each RIR would have been possible, since $N_x \cdot N_y \cdot N_z$ is generally greater than the number of RIRs to compute, the level of parallelization is already quite high and, as shown in Table 3.2, further optimizations of this kernel would have had a slight impact on the final performance of the simulation.

3.2.3.2 Computation and sum of the contribution of each image source

The computation of (3.8) is the most complex part of the implementation as it implies a reduction operation (the sum of the contributions of each image source into the final RIR), which is hard to parallelize since it would imply several threads writing in the same memory address, and the calculation of a high number of trigonometric functions. We can see it as creating a tensor with 3 axes (each RIR, each image source, and each time sample) and summing it along the image sources axis. However, the size of this tensor would be huge and it would not fit in the memory of most GPUs.

To solve this problem, we first compute and sum a fraction of the sources contributions sequentially, so the size of the tensor we need to allocate in the GPU memory is reduced; we do that through `generateRIR_kernel`. Specifically, each parallel thread of this kernel performs sequentially the sum of 512 images for a time sample of a RIR. This sequential sum reduces the degree of parallelism of the implementation but, since the number of threads is already high enough to keep the GPU always busy, it does not decrease the performance. It should be noted that, although all the threads can potentially run in parallel, the number of threads which actually run in parallel is limited by the number of CUDA cores of the GPU and, if we have more threads than CUDA cores, many threads will be queued and will run sequentially.

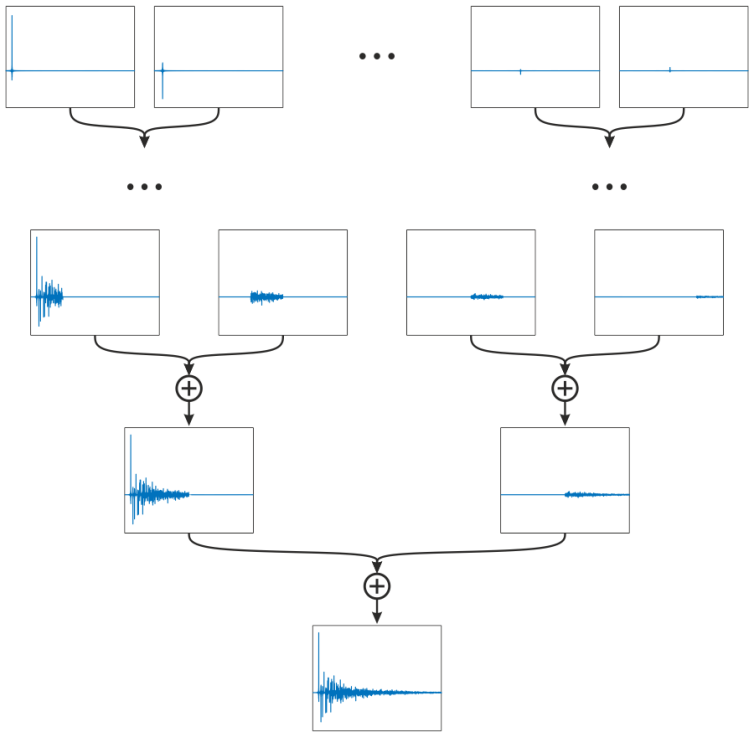


Figure 3.7: Parallel reduction sum of the sincs (each level is performed by a call to `reduceRIR_kernel`). The sum must be performed pairwise to prevent several threads from concurrently writing in the same variable. The sums of each time sample are also performed in parallel.

After that, we use `reduceRIR_kernel` recursively to perform the reduction in parallel by pairwise summing the contribution of each group of images as shown in Fig. 3.7. Performing the whole sum in parallel would lead to all the threads concurrently writing in the same memory positions, which would corrupt the result.

It can be seen in Table 3.2 how most of the simulation time is expended in `generateRIR_kernel`, this is due to the high amount of sinc functions that need to be computed and it also happens in the sequential implementations. However, thanks to the computing power of modern GPUs, we can compute many sinc functions in parallel and therefore reduce the time we would have needed to sequentially compute them in a CPU. We analyze the implementation of these sinc functions using lookup tables (LUTs) in section 3.2.3.5 and its performance in section 3.2.5.2.

3.2.3.3 Diffuse reverberation computation

For the diffuse reverberation, we first use `envPred_kernel` to predict in parallel the amplitude and the time constant of each RIR. After that, we use the cuRAND library included in the CUDA Toolkit to generate a uniformly distributed noise (the functions `generate_seed_pseudo` and `gen_sequenced` in Table 3.2 belong to this library) and we finally transform it to a logistic distributed noise and apply the power envelope through `diffRev_kernel`, which parallelizes the computations of each sample of each RIR. The function `generate_seed_pseudo` generates the seed for the cuRAND random number generator and it is only called when the library is imported, not every time a new RIR is calculated.

3.2.3.4 Simulating moving sources

In order to simulate a moving source recorded by a microphone array, we need to compute the RIR between each point of the trajectory and each microphone of the array and filter the sound source by them using the overlap-add method. In sequential libraries, the complexity of the filtering is negligible compared to the RIR simulation; however, in our library, thanks to the performance of the GPUs, we found that we also needed to parallelize the filtering process if we did not want to be limited by it (especially for short reverberation times). To solve this problem, our library is able to compute multiple convolutions in parallel using the cuFFT library (included in the CUDA Toolkit) and a custom CUDA kernel to perform the pointwise complex multiplication of the FFTs.

3.2.3.5 Lookup tables (LUTs)

Motivated by the performance increase that the CPU implementations achieve by using lookup tables (LUTs) to calculate the sinc functions (see section 3.2.5), we also implemented it in our GPU library.

Our LUT stores the values of a sinc oversampled in a factor $Q = 16$ multiplied by a Hanning window:

$$LUT[n] = \frac{1}{2}\left(1 + \cos\frac{2\pi n}{QT_\omega}\right)\operatorname{sinc}\left(\pi\frac{n}{Q}\right) \quad \text{for } n \in \left\{\frac{-T_\omega}{2}Qf_s, ..., \frac{T_\omega}{2}Qf_s\right\} \tag{3.12}$$

and then we use linear interpolation between the closest entries of the table to compute each sample of the sinc functions of each image source.

The main design choice we must make is to define the type of memory that will be used to place the LUT. CUDA GPUs have, in addition to the registers of each thread, 4 different memories: shared, global, constant, and texture memory. On the one hand, shared memory is shared only between threads of the same block and it has the fastest access, however, it is generally lower than 100 kB. On the other hand, global memory is shared by all the threads and usually has several gigabytes, but it has the lowest bandwidth and the highest latency. Finally, constant and texture memories are read-only cached memories, constant memory being optimized for several threads accessing the same address and texture memory being optimized for memory access with spatial locality. Although constant memory has a lower latency than texture memory, texture memory implements some features like several accessing modes and hardware interpolation, which are extremely useful for the implementation of LUTs. We implemented the windowed sinc LUT both in shared memory and texture memory and obtained better performance with the texture memory thanks to the hardware interpolation.

3.2.3.6 Mixed precision

Since the Pascal architecture, the Nvidia™ GPUs include support for 16-bit precision floats and are able to perform two 16-bit operations at a time. To exploit this feature, we developed the kernels `generateRIR_mp_kernel` and `reduceRIR_mp_kernel`, which compute two consecutive time samples at a time so we can halve the number of threads needed. We focused on these kernels and did not optimize the others because, as shown in Table 3.2, most of the simulation time is spent on them.

CUDA provides the data type `half2`, which contains 2 floating point numbers of 16 bits, and several intrinsics to operate with it. These intrinsics allow us to double the number of arithmetic operations that we can perform per second; however, we found that the functions provided to compute two 16-bit trigonometric functions were not as fast as computing one 32-bit function. To increase the simulation speed, we developed our own `sinpi(half2)` and `cospi(half2)` functions. For the sine function,

we first reduce the argument to the range [-0.5, 0.5], then we approximate the sine function in this range by

$$\sin(\pi x) \approx 2.326171875x^5 - 5.14453125x^3 + 3.140625x \tag{3.13}$$

and finally, multiply the result by -1 if the angle was in the second or the third quadrant. The coefficients of the polynomial are the closest numbers that can be represented with half-precision floats to those of the optimal polynomial in a least-squares sense. Equivalently, for the cosine function, we used the polynomial:

$$\cos(\pi x) \approx -1.2294921875x^6 + 4.04296875x^4 - 4.93359375x^2 + 1 \tag{3.14}$$

with the advantage that, since we only used it for computing the Hanning window in (3.8), we do not need to perform argument reduction or sign correction.

The polynomial evaluation can be efficiently performed with Horner's method:

$$\begin{aligned} b_n &= a_n \\ b_{n-1} &= a_{n-1} + b_n x \\ &... \\ p(x) &= b_0 = a_0 + b_1 x \end{aligned} \tag{3.15}$$

where a_i is the coefficient of the n degree polynomial $p(x)$ we want to evaluate and the computation of b_i can be done in parallel for two different values of x using the CUDA intrinsic `__hfma2(half2)` that performs the fused multiply-add operation of the two elements of three `half2` variables at a time. More information about the polynomial approximation of transcendental functions can be found in [154].

Obviously, working with half-precision representation reduces the accuracy of the results. We found that the most critical part was in subtracting $t - \tau_n$. Working with 16-bit precision floats, we can only represent 3 significant figures accurately, so, when t grows, we lose precision in the argument of the sinc function which leads to an error that increases with time; when t grows we expend the precision in the integer part and we do not represent accurately the fractional part. To solve this issue, we perform the subtraction with 32 bits arithmetic and then we transform the result to 16-bit

precision. Working this way, we have always maximum precision in the center of the sinc and the lower accuracy is outside the Hanning window.

Unfortunately, the hardware interpolation of the texture memory does not support 16-bit arithmetic, so the mixed precision implementation is not compatible with the LUT.

3.2.4 Python library

We have included the previous implementation in a Python library[1] that can be easily compiled and installed using the Python packet manager (pip) and be used as any CPU library. The library provides a function that takes as parameters the room dimensions, the reflections coefficients of the walls, the position of the source and the receivers, the number of images to simulate for each dimension, the duration of the RIR in seconds, the time to switch from the ISM method to the diffuse reverberation model, and the sampling frequency and it returns a 3D tensor with the RIR for each pair of source and receiver positions. Information about the polar pattern of the receivers and their orientation can be also included in the simulation.

We also provide some python functions to predict the time when some level of attenuation will be reached, to get the reflections coefficients needed to get the desired reverberation time (expressed in terms of T_{60}, i.e. the time needed to get an attenuation of 60 dB), and to get the number of image sources to simulate in each dimension to get the desired simulation time without loss reflections. Finally, we include a function to filter a sound signal by several RIRs in order to simulate a moving source recorded by a microphone array. In the repository of the library, some examples can be found about how to simulate both isolated RIRs and moving sources.

Since the use of the LUT to compute the sinc function improves the performance in most of the cases and the precision loss is negligible (see section 3.2.5.2), its use is activated by default, but the library provides a function to deactivate it and use the CUDA trigonometric functions instead. In order to exploit the mixed precision capabilities of the newer GPUs, it has a function to activate it and use the 16-bit

[1] The code, the documentation, the installation instructions, and examples can be found in `https://github.com/DavidDiazGuerra/gpuRIR`

precision kernels instead of the 32-bit; activating it automatically deactivates the use of the LUT.

3.2.5 Results

3.2.5.1 Base implementation

In order to show the benefits of using GPUs for RIR simulation, we have compared our library against three of the most employed libraries for this purpose: the Python version of the RIR Generator library presented in [146], whose code is freely available in [155] and has been used, for example, in [143, 156, 157]; the Python package pyroomacoustics presented in [147] that has been employed in [158, 159, 160] among others; and the Matlab™ library presented in [133], whose code is freely available in [161], and that has been used, for example, in [142, 162, 163]. Since all the libraries are based on the ISM, whose acoustical accuracy is well known, we focus on the computation time of each library.

Neither RIR Generator nor pyroomacoustics implement any kind of diffuse reverberation model, so they are expected to have worse performance than the Matlab™ library and our GPU library if we use it. The Matlab™ library uses the formula presented in [151] to model the power envelope of the diffuse reverberation, which is more complex than our exponential envelope model, so, for the sake of a fairer comparison, we modified the Matlab™ implementation to use an exponential model. The simulations with the sequential libraries and the ones with the Nvidia™ GTX 980Ti and the RTX 3090 were performed in a computer with an Intel™ Core i7-6700 CPU and 16 GB of RAM, while the simulations with the Nvidia™ Tesla P100 and V100 were performed in an n1-highmem-4 instance in the Google Cloud Platform™ with 4 virtual CPUs cores and 26 GB of CPU memory; more details about the GPUs employed for the simulations can be found in Table 3.3.

Fig. 3.8 represents the runtime of the different libraries for computing different numbers of RIRs in a room with size $3\,\mathrm{m} \times 4\,\mathrm{m} \times 2.5\,\mathrm{m}$ and $T_{60} = 0.7\,\mathrm{s}$. It can be seen how our library can simulate a hundred times more RIRs in a second than the Matlab™ library even with a GPU designed for gaming (the Nvidia™ GTX 980

Table 3.3: GPUs employed for the performance analysis.

GPU model	Release year	Architecture	Memory	Single Precision FLOP/s	Memory Bandwidth
GTX 980 Ti	2015	Maxwell	6GB	5.6 TeraFLOP/s	337 GB/s
Tesla P100	2016	Pascal	16GB	9.5 TeraFLOP/s	732 GB/s
Tesla V100	2017	Volta	16GB	14.9 TeraFLOP/s	900 GB/s
RTX 3090	2020	Ampere	24GB	29.3 TeraFLOP/s	936 GB/s

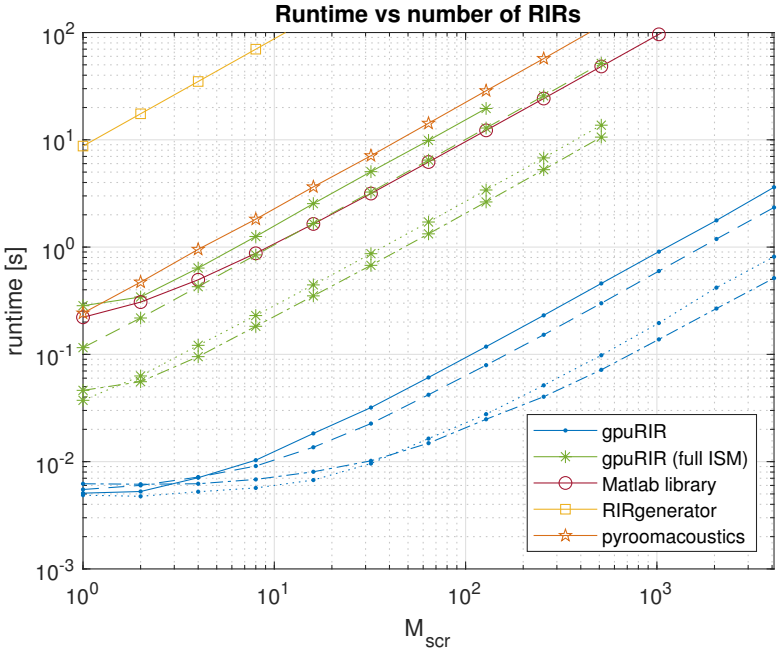


Figure 3.8: Runtime of each library for computing different numbers of RIRs (M_{src}) in a room with size $3\,m \times 4\,m \times 2.5\,m$ and $T_{60} = 0.7\,s$. For the gpuRIR library, the solid line times were obtained with the GTX 980 Ti GPU, the dashed lines with the Tesla P100, the dotted lines with the Tesla V100, and the dash-dot lines with the RTX 3090.

Ti). Using our library without any kind of diffuse reverberation modeling, we have a similar execution time to the Matlab™ library, which only computes the ISM until the RIR has an attenuation of 13 dB, and we are also about a hundred times faster than the RIR Generator library. Finally, it is worth noting how pyroomacoustics performs quite similarly to our library when we use a GTX 980 Ti and compute the whole RIR with the ISM without using any diffuse reverberation model; this is due to the use of LUTs to compute the sinc functions by pyroomacoustics (to confirm this hypothesis we modified the code of pyroomacoustics to avoid the use of LUTs and its performance degraded to the same results than RIR Generator). However, using a faster GPU, i.e. the Tesla V100 or the RTX 3090, our library can compute ten times more RIRs in a second than pyroomacoustics even without using LUTs, since we can set at full performance all the parallelization mechanisms presented in section 3.2.3.

Comparing the performance of our library using different GPUs, we can see how the lower results are obtained using the GTX 980 Ti and the Tesla P100 and the best results are obtained with the Tesla V100 and the RTX 3090 (being between 5 and 10 times faster than the GTX 980 Ti). It is worth noting how, in less than 5 years, a gaming-oriented GPU as the RTX 3090 has matched and ever surpassed the performance of the professional GPU Tesla V100. We can expect the number of RIRs that we can compute per second with our parallel implementation to continue increasing at a good peace, something that we cannot expect from most of the sequential implementations designed to run in CPUs.

In Fig. 3.9 we show the runtime of the different libraries for computing 128 RIRs in a room with size $3\,\mathrm{m} \times 4\,\mathrm{m} \times 2.5\,\mathrm{m}$ and different reverberation times. We can see again how our library is about two orders of magnitude faster than the sequential alternatives which do not use LUTs. It must be said that our library has some limitations because calculating a large number of RIRs with high reverberation times may require more memory than what is available in the GPU; however, using the diffuse reverberation model, this limitation appears only for really high number of RIRs and reverberation times. Furthermore, it would be always possible to batch the RIRs in several function calls to circumvent this problem.

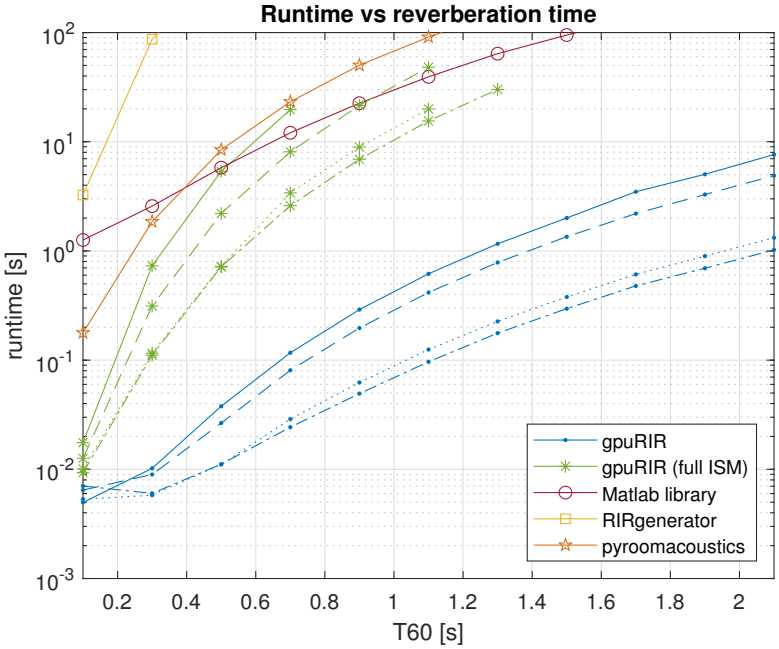


Figure 3.9: Runtime of each library for computing 128 RIRs in a room with size $3\,\mathrm{m} \times 4\,\mathrm{m} \times 2.5\,\mathrm{m}$ and different reverberation times. For the gpuRIR library, the solid line times were obtained with the GTX 980 Ti GPU, the dashed lines with the Tesla P100, the dotted lines with the Tesla V100, and the dash-dot lines with the RTX 3090.

Table 3.4: Lookup table (LUT) and mixed precision (MP) simulation times and speedups for computing different numbers of RIR with $T_{60} = 0.7\,\text{s}$.

		Number of RIRs	Diffuse reverberation model				Full ISM		
			1	16	128	1024	1	16	128
Matlab Library [ms]			221,52	1,643.20	12,252.67	96,208.58	-	-	-
pyroomacoustics [ms]			-	-	-	-	242.35	3,6409.16	28,646.86
gpuRIR	GTX 980 Ti	Base [ms]	4.98	17.43	117.60	898.54	283.88	2,601.82	19,630.60
		LUT [ms]	5.19	16.64	109.38	834.38	279.28	2,434.33	18,547.03
		speedup	x0.96	x1.05	x1.08	x1.08	x1.02	x1.07	x1.06
		MP [ms]	-	-	-	-	-	-	-
		speedup							
	Tesla P100	Base [ms]	5.81	13.86	79.28	596.02	115.57	1,661.35	12,879.31
		LUT [ms]	5.97	12.14	63.90	471.16	86.86	1,235.64	9,397.40
		speedup	x0.97	x1.14	x1.24	x1.27	x1.33	x1.35	x1.37
		MP [ms]	5.52	9.45	45.49	324.12	59.46	847.74	6,493.92
		speedup	x1.05	x1.47	x1.74	x1.84	x1.94	x1.96	x1.98
	Tesla V100	Base [ms]	4.76	7.13	28.14	195.69	37.62	447.04	3,403.60
		LUT [ms]	5.01	6.79	23.66	156.91	30.66	394.54	2,595.97
		speedup	x0.95	x1.05	x1.19	x1.25	x1.23	x1.13	x1.31
		MP [ms]	4.55	6.29	19.57	128.72	21.76	253.03	1,900.52
		speedup	x1.05	x1.13	x1.44	x1.52	x1.73	x1.77	x1.79
	RTX 3090	Base [ms]	5.95	8.04	23.27	139.41	46.75	351.10	2,638.73
		LUT [ms]	6.02	8.12	21.42	127.14	35.02	331.02	2,437.89
		speedup	x0.99	x0.99	x1.09	x1.10	x1.33	x1.06	x1.08
		MP [ms]	5.69	7.06	16.22	84.33	27.79	200.90	1,481.32
		speedup	x1.04	x1.14	x1.43	x1.65	x1.68	x1.74	x1.78

3.2.5.2 Lookup tables

Motivated by the huge speedup generated by the use of LUTs in the CPU implementations (a factor 5 in Fig. 3.8) we replaced the trigonometric computations with a LUT as described in section 3.2.3.5. Tables 3.4 and 3.5 show the speedup (defined as the runtime without using the LUT divided by the runtime using it) for several numbers of RIRs and reverberation times using different GPUs.

We can see how our library obtains a speedup much lower than the one obtained by pyroomacoustics over CPUs. This is due to the high computation power of the GPUs, which makes the computation of trigonometric functions quite efficient and therefore they are not so benefited by replacing computation tasks with memory calls. Despite that, we can see how using LUTs is faster than computing the trigonometric functions, i.e. the speedup is higher than 1.0, in most of the cases, especially when the number of RIRs or the reverberation time increases.

Among the studied GPUs, the Tesla P100 obtains the higher speedups since it has

Table 3.5: Lookup table (LUT) and mixed precision (MP) simulation times and speedups for computing 128 RIRs with different reverberation times.

		T_{60} [s]	Diffuse reverberation model					Full ISM		
			0.3	0.7	1.1	1.5	1.9	0.3	0.7	1.1
		Matlab Library [ms]	2,573.67	12,078.52	39,330.40	94,946.73	136,522.39	-	-	-
		pyroomacoustics [ms]	-	-	-	-	-	1,854.08	23,253.22	90,960.54
gpuRIR	GTX 980 Ti	Base [ms]	8.90	118.00	627.15	2,016.40	5,073.48	731.59	19,657.05	-
		LUT [ms]	8.62	109.89	588.90	1,896.48	4,769.68	694.10	18,466.15	-
		speedup	x1.03	x1.07	x1.06	x1.06	x1.06	x1.05	x1.06	
		MP [ms]	-	-	-	-		-	-	-
		speedup								
	Tesla P100	Base [ms]	8.97	80.78	416.57	1,349.47	3,289.13	494.33	12,875.14	76,383.87
		LUT [ms]	7.39	64.90	321.97	1,023.39	2,452.18	391.39	9,406.31	55,402.08
		speedup	x1.21	x1.25	x1.29	x1.32	x1.34	x1.26	x1.37	x1.38
		MP [ms]	6.64	45.18	218.18	699.95	1,698.17	258.96	6,484.46	38,393.03
		speedup	x1.35	x1.79	x1.91	x1.93	x1.94	x1.91	x1.99	x1.99
	Tesla V100	Base [ms]	5.80	28.81	125.02	379.13	896.97	141.86	3,400.95	19,935.71
		LUT [ms]	5.95	24.43	101.85	332.35	690.02	117.55	2,594.15	15,363.05
		speedup	x0.97	x1.18	x1.23	x1.14	x1.30	x1.22	x1.31	x1.30
		MP [ms]	5.08	19.80	76.71	220.66	519.46	87.48	1,901.21	11,052.52
		speedup	x1.14	x1.46	x1.63	x1.72	x1.73	x1.62	x1.79	x1.80
	RTX 3090	Base [ms]	6.02	24.31	96.60	295.95	694.41	112.20	2,585.85	15,498.97
		LUT [ms]	6.09	22.38	88.22	273.19	639.95	103.85	2,380.35	14,225.71
		speedup	x0.99	x1.09	x1.09	x1.08	x1.09	x1.08	x1.09	x1.09
		MP [ms]	5.80	17.21	57.32	170.21	401.21	63.81	1,477.57	8,544.19
		speedup	x1.04	x1.41	x1.69	x1.74	x1.73	x1.73	x1.76	x1.81

a higher memory bandwidth compared with its computing power. The GTX 980 Ti gets really humble speedups due to its low memory bandwidth and the Tesla V100 and the RTX 3090, though they have the higher bandwidth, do not reach the speedups obtained by the Tesla P100 due to their huge computing power.

Fig. 3.10 shows the first 0.5 seconds of the RIR of a room with $T_{60} = 1\,\text{s}$ computed with our GPU implementation working with single (32-bit) precision trigonometric functions and the error introduced by replacing them with our LUT. We can see how, as could be expected, the error introduced by the use of the LUT is negligible: three orders of magnitude lower than the amplitude of the RIR.

3.2.5.3 Mixed precision

In the case of using the 16-bit precision kernels, we are reducing the accuracy of the simulation, so we need to analyze its impact. Fig. 3.10 also shows the error introduced by computing the same RIR using our half (16-bit) precision kernels. We can see how the error is 3 orders of magnitude lower than the amplitude of the RIR at

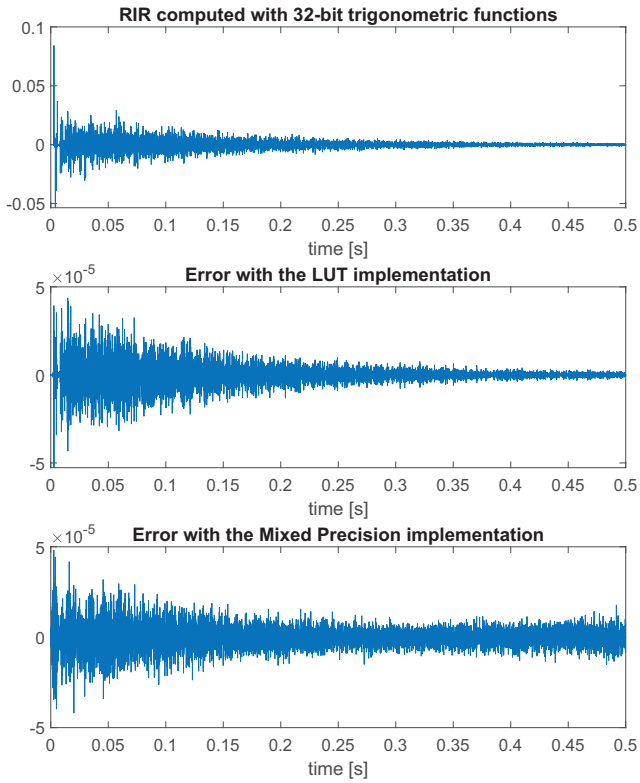


Figure 3.10: RIR computed with single (32-bit) precision trigonometric functions and the errors introduced due to computing it using a lookup table (LUT) and half (16-bit) precision functions (Mixed Precision).

the beginning, which should be acceptable for most of the applications; however, since the error does not decrease with time as much as the RIR does, the signal-to-error ratio deteriorates with the time. Fortunately, this higher error corresponds with the diffuse reverberation, where its perceptual importance is lower.

Theoretically, a twofold speedup could be expected from working with 16-bit precision floats instead of 32-bit floats, however, this speedup is generally not reachable as the number of operations is not the only limiting factor of many GPU kernels and some `half2` functions are not as fast as their equivalent `single` functions. Tables 3.4 and 3.5 show the speedup that our mixed precision implementation achieves for several numbers of RIRs computed in parallel and several reverberation times. We can see how the speedup is higher when the workload increases, especially for long reverberation times where the operations per second are the main limiting factor of its performance and how the speedup achieved with the mixed precision implementation is always higher than the achieved with the LUTs.

The mixed precision support was introduced with the Pascal architecture, so it is not available in older models like the GTX 980 Ti. The Tesla P100 achieves speedups really close to 2 for high workloads and the Tesla V100 and the RTX 3090 also achieve better speedups than with the use of LUTs although not as high as the Tesla P100.

3.3 Conclusions

Due to the difficulty of recording a hand-labeled dataset of moving sources large enough to train a neural network, we have introduced a new procedure for generating random trajectories and simulating them as they are needed for training. With it, we have an infinite-size dataset whose parameters can be easily modified during training to accelerate the convergence or during test to analyze the performance of the model in specific scenarios. To reduce the simulation time and allow the generation of the trajectories as they are needed during the training, we have implemented the image source method (ISM) using CUDA so we can reduce the simulation time by two orders of magnitude using GPUs.

We have published our GPU implementation of the ISM as a free and open-source library that has proven to be about one hundred times faster than other state-of-the-art CPU libraries. To the best of our knowledge, it is the first library with these features freely available on the Internet, and it could allow the acoustic signal processing community, for example, to generate huge datasets of moving speaker speech signals in a reasonable computation time or to compute the acoustics of a virtual reality scene in real time.

We have studied different methods to increase the speed of our GPU implementation, concluding that the best strategy is using 16-bit arithmetic, but this is only compatible with the newer GPUs. On the other hand, using LUTs stored in the GPU's texture memory, though it generates lower speedups, is compatible with most of the CUDA GPUs, so we have chosen to use this implementation as our library default.

4

Robust single source localization

In this chapter we present two sound source localization (SSL) models for scenarios with a single source, both using steered response power with phase transform (SRP-PHAT) power maps as inputs and being completely causal; i.e. they could be implemented in real-time and provide a new direction of arrival (DOA) estimate for every new input map. Both are fully convolutional models, the first one is based on a 3D convolutional neural network (CNN) and the second one combines icosahedral convolutions with 1D temporal convolutions.

Using both synthetic scenarios and real recordings, we prove that the use of SRP-PHAT power maps as input features of our models provides more robust estimations under adverse conditions and that the use of network architectures that fit the symmetries of the problem allows us to improve the localization accuracy even reducing the number of trainable parameters and the resolution of the input maps.

This chapter includes the reproduction of figures and text fragments from [V] and [VI] with the permission of the copyright holders.

4.1 Robust sound source localization using SRP-PHAT and 3D convolutional neural networks

As explained in chapter 2, we can use the SRP-PHAT algorithm to compute acoustic power maps from the signals received by a microphone array. These maps have traditionally been used for SSL by just choosing the position of their maximum as the estimated DOA but, even being more robust than other classic techniques, they still have some limitations especially when reverberation and noise increase. In this section, we propose the use of CNNs over SRP-PHAT power maps, performing the convolutions over the dimensions of the maps and the temporal dimension.

Any kind of SRP-PHAT power maps could be employed with this approach depending on the geometry of the array but, as we focus on compact arrays, we use 2D spherical power maps and therefore, since we include the temporal dimension, 3D CNNs. The extension to 4D CNNs over 3D SRP-PHAT maps to perform 3D SSL with distributed arrays would be straightforward.

Using conventional CNNs over equiangular projections of spherical maps is not the optimal solution since these projections generate deformations around the poles that need to be learned by the network instead of just being part of the equivariances of its architecture. However, this is still a good first approach to process this kind of maps with neural networks due to its simplicity and efficient implementation. In section 4.2 we study the use of icosahedral convolutions as an extension of this work.

Many of the state-of-the-art CNN architectures include bidirectional recurrent units at the last layers of the model. Recurrent neural networks (RNNs), as recurrent linear filters, make the output at any time instant dependent on the values of the input at every previous time instant and, therefore, applying them in the backward direction is extremely non-causal. Obviously, any tracking system can greatly benefit from the information on the future positions of the source but, in order to make our system feasible for real-time applications, we opt for using only causal convolutional layers. By using 3D convolutions, the localization is done including the information of the previous time frames, which does not happen in most of the classical systems where the localization is done using only the information of the current time frame

and then the localization result is filtered in a posterior tracking stage considering the localization results of the previous frames. Integrating the information in this way should lead to more robust estimates since we do not lose any information between the localization and tracking stages.

4.1.1 Preprocessing

The proposed model takes as input a 4-dimensional tensor ($\mathbf{M}$) with size $C \times T \times N_\theta \times N_\varphi$, whose first channel $\mathbf{M}_{1,t,i,j}$ contains SRP-PHAT power maps. In order to compute them, we first window the signals of a microphone array sampled at 16 kHz using Hanning windows of length $K = 4096$ samples (i.e. 256 ms) with a hop size of $3K/4$. After that, for the T resulting frames, we apply (2.3) at a rectangular grid with N_θ equispaced polar angles in the range $\theta \in [0, \pi]$ and N_φ equispaced azimuth angles in $\varphi \in [-\pi, \pi)$ to obtain the power map of each window; for planar arrays, the same model could be used sampling the polar angle only in $\theta \in [0, \frac{\pi}{2}]$. Finally, we normalize the maps by subtracting their mean and dividing them between their maximum to approximately fit them to the range [-1,1]. For the sake of computational efficiency, we do not perform any kind of interpolation in the computation of (2.3) and just approximate the fractional delays to the nearest sample.

Although the model must learn more complex patterns in order to exploit all the information available in the SRP-PHAT, it is obvious that one of the main sources of information about the DOA of the source is the position of the maximum of each map. Since it did not cause a significant increase in the computational complexity of the system, we decided to explicitly indicate to the network the position of the maximum of each map. After trying to introduce this information in different layers, we found that the best results were obtained including it in the input of the network, using $C = 3$ with $\mathbf{M}_{2,t,i,j} = \hat{\theta}_t^{SRP}$ and $\mathbf{M}_{3,t,i,j} = \hat{\varphi}_t^{SRP}$ for any $t \in \{1, ..., T\}$, $i \in \{1, ..., N_\theta\}$, and $j \in \{1, ..., N_\varphi\}$, where $\hat{\theta}_t^{SRP}$ and $\hat{\varphi}_t^{SRP}$ are the polar and azimuth angles corresponding to the position of the maximum of the map t normalized to be in the range [0,1].

Finally, we found that, since the synthetic dataset presented in chapter 3 does not include any directional noise, the models trained with it are very sensitive to directional noise sources. For example, in some of the recordings of the LOCATA dataset [137],

the noise of a fan is present and, although its power is very low, the models tracked it when it was the only active sound source. In order to avoid this issue, we use the WebRTC voice activity detector (VAD) to determine in which frames the speech source is active. We first tried to include the VAD information as an additional input channel to the network. However, during the training, the VAD sometimes failed and classified frames with speech information as silent so the network learned that even frames classified as silent could contain useful tracking information as long as they contained a directional source and, therefore, it ignored the VAD input. In order to prevent the network from tracking directional noise sources, we finally opted to turn to zero the maps corresponding to frames classified as silent by the VAD so no directional information is seen by the network when no speech source is active.

4.1.2 Model architecture

The first layer of our model is a 3D convolutional layer with 32 kernels of size $5 \times 5 \times 5$ and parametric rectified linear unit (PReLU) activations [164]. For the temporal axis, we always use causal convolutions, so this model could be used in real-time applications generating a new DOA estimation for each new power map available without introducing any delay.

Pooling layers are typically used in CNNs to progressively reduce the size of the input and make the model generalize; not using them, means that the fully connected layers used at the end of most of the convolutional models would have a huge number of trainable parameters which would surely overfit. When the desired output of the CNN is a summary of the information of the whole input, e.g. in image classification tasks, increasing the number of channels with convolutional layers and reducing their size with pooling layers progressively reduces the spatial information and gets higher-level representations of the input. However, since our desired output is not only related to the presence of some patterns but especially to their position, we must be careful when using them.

In order to get the benefits of pooling layers but allowing the spatial information to reach the last layers of the model, we opted to, as shown in Fig. 4.1, split the model into two branches and apply max pooling in a different dimension in each one.

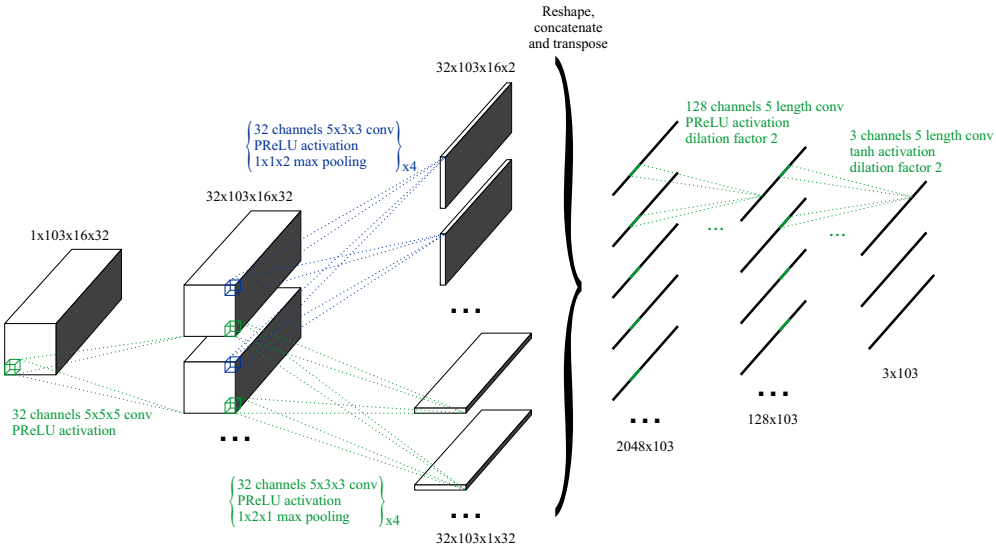


Figure 4.1: Model architecture. The noted sizes correspond to a model for 16x32 maps and an input sequence of length 103. For the sake of simplicity, we represented it with only 1 input channel, although it actually has 3.

Working this way, the branch which pools the φ axis can retain positional information about the θ coordinate of the maps and vice versa. Specifically, each branch has 4 layers with a convolution with 32 kernels of size $5 \times 3 \times 3$, PReLU activations, and a max-pooling with a kernel size of $1 \times 1 \times 2$ and $1 \times 2 \times 1$ respectively. If the input power maps have less than 16 points in the θ or the φ axes, it would not be possible to perform so many pooling layers; in those cases, we reduce the 4 layers to the maximum number possible: $\log_2(\min(N_\theta, N_\phi))$. Due to the use of 3D convolutional layers and these perpendicular branches, we named this model Cross3D.

After the 3D convolutional layers, we concatenate the results of each branch and reshape them so we have a temporal sequence of length T for each one of the elements of each channel and spherical coordinates. Each one of these temporal sequences is used as the input channels of a 1D causal convolutional layer with 128 kernels of length 5 and PReLU activations. Finally, the resultant 128 time sequences are passed through another 1D causal convolutional layer with only 3 kernels of length 5 and tanh activations. These layers are similar to the fully connected layers that most of the CNN architectures have, but we include a temporal convolution so they can still

exploit the tracking information. We use a dilation factor of 2 in order to allow the tracking to take into account a longer context without increasing the complexity of the network. With all the temporal convolutions included in the model, each DOA estimation is computed from the last 37 SRP-PHAT maps, i.e. the temporal receptive field and the tracking memory is 7.17 s.

The result of this process are 3 time sequences of length T whose elements are in the range $(-1, 1)$, which are considered to be the XYZ coordinates of a unitary vector pointing in the direction of the source in each time frame.

Tables detailing the network architecture of Cross3D for several SRP-PHAT map resolutions can be found in appendix A.

4.1.3 Training

We trained our model to minimize the Euclidean distance between the output of the network and the 3 time sequences obtained from the coordinates of the unitary vectors steering at the direction where the sound source was simulated in each time window. Similarly to the results reported in [14, 53, 54], we obtained better results using this approach than trying to directly obtain the spherical coordinates from the network even when using the great-circle distance between the output and the ground-truth DOA angles as cost function.

Although using an infinite-size dataset the term "epoch" does not have the same meaning as in most of the machine learning systems, we define an epoch as 585 trajectories (the number of book chapters in the LibriSpeech train-clean-100 subset). We employed 80 epochs with trajectories of 20 s, i.e. 103 SRP-PHAT maps, to train the model with the Adam algorithm [165] using Pytorch [166].

As explained in chapter 3, the training dataset contains scenes with reverberation times and signal-to-noise ratios (SNRs) uniformly distributed from 0.2 s to 1.3 s and from 5 dB to 30 dB respectively. However, we found that the training converged faster with higher SNRs. Therefore, we followed a curriculum learning strategy [134] using batches of 5 trajectories with SNR=30 dB for the first 20 epochs and for the following epochs we employed the full range of SNRs, increased the batch size to 10 trajectories, and reduced the learning rate from 1×10^{-4} to 1×10^{-5}.

4.1.4 Evaluation

4.1.4.1 Baseline methods

In order to analyze the convenience of using SRP-PHAT maps as input features of CNNs for DOA estimation, we developed some alternative CNNs to use them as baselines. We designed them to be as similar as possible to our proposed model and to have the same temporal perceptive field so they have the same tracking information.

Since we are including the position of the maximum of each map into the input of the network, we should verify if our model is actually exploiting the additional information that is within the SRP-PHAT maps or if it is only using the position of its maximums. To do that, we designed a 1D CNN which takes as input 2 time sequences with the coordinates of the maximum of each map normalized to the range [0,1] and applies to them 7 layers of 1D causal convolutions with PReLU activations and without any pooling. All the layers had a kernel size of 5 and the last two layers used a dilation factor of 2, so its temporal receptive field is 37 frames as in Cross3D. The number of channels of each layer was $\{1024, 512, 512, 512, 512, 128, 3\}$. The results shown in the following sections were obtained by training this network with the same process described in section 4.1.3 and using the coordinates of maps with resolution 64×128.

One of the most common input features employed by the first DOA estimation techniques based on neural networks were the generalized cross-correlations (GCCs). They typically employed fully connected perceptrons with not too many hidden layers and, since they only used the GCCs computed in a temporal window, did not perform any kind of tracking. Following this idea, but with the aim of including tracking information into the network, we used the same 1D causal CNN that we used over the map maximums but using as input sequences the temporal evolution of each element of the GCCs which represented an inter-microphone delay lower than the maximum inter-microphone distance divided by the speed of sound.

Although, as explained in chapter 1, the use of 2D CNNs over spectrograms may not be optimal, we also implemented a model following this approach since it is quite popular in the literature. For a fair comparison, we used causal convolutions with a

Table 4.1: Models employed for the evaluation.

Model	Input		Trainable parameters	Temporal perceptive field	Window length	Causal
Cross3D	Power maps	4×8	526 372	5.63 s	4096	Yes
		8×16	946 340	6.40 s		
		16×32	1 693 988			
		32×64	5 626 148	7.17 s		
		64×128	21 354 788			
1D CNN	GCCs		11 282 436	7.17 s	4096	Yes
	Maximums (64x128)		6 899 716			
2D CNN	Spectrograms		1 882 372	7.17 s	4096	Yes
SELDnet [14]			104 643	∞	512	No

similar architecture to Cross3D: one convolution with 256 5×5 kernels, four convolutions with 256 5×5 kernels with 1×4 pooling, a reshape to transform the remaining features into temporal sequences and two 1D causal convolutions with kernel size 5 and dilation factor 2, the first one with 128 channels and the last one with 3. For computing the spectrogram, we used the same windows as for computing the SRP-PHAT maps, extracted the magnitude and phase of each frequency of the Fourier transform (FT), and finally normalized the magnitude of each window to its maximum and the phase to the range [-1,1].

Finally, we also trained with our simulation procedure a replica of SELDnet [14] but without including the sound event detection (SED) output and with only a DOA output since we were only interested in tracking one source. This model takes as inputs the magnitude and phase of the spectrograms and has three 2D convolutional layers followed by two bidirectional gated recurrent units (GRUs) [167] and two fully connected layers. It is worth saying that this model, due to the bidirectional GRUs, is non-causal and that it uses shorter analysis windows than the other analyzed methods.

For the models that use spectrograms as input features, we found that they did not train properly with the full range of reverberations described in section 3.1, and we got the best results training them with values of T_{60} randomly selected from the range 0 s to 0.3 s.

All the models employed for the evaluation are summarized in Table 4.1 and tables detailing their architectures can be found in appendix A.

4.1.4.2 Simulated dataset

We trained different models for several power map resolutions with the whole range of reverberation times and SNRs, and then we tested their performance for several specific values of T_{60} and SNR in order to analyze the robustness of the proposed tracking system.

Since we are using SRP-PHAT power maps as the input of our algorithm, we started our evaluation by comparing our model with the classic SRP-PHAT algorithm. SRP-PHAT does not perform any kind of tracking, so, for a fairer comparison, we did not take into account the silent frames when computing the root mean squared angular errors (RMSAEs) shown in Fig. 4.2. As we can see in this figure, when working with high-resolution power maps in almost anechoic rooms with high SNR, using our 3D CNN over the SRP-PHAT maps does not improve the results compared to just taking the maximum of each map; actually, our system seems to slightly degrade the DOA estimation, probably due to the effect of applying an unneeded tracking. However, when the room conditions deteriorate, we can see how Cross3D is robust enough to get its performance degraded in only 5° when the T_{60} increases to 1.5 s (which is higher than any reverberation seen during the training) while the SRP-PHAT algorithm is just unable to perform a proper estimation. They are also surprising the results obtained with maps of only 4x8 resolution, which only perform a SRP-PHAT measurement each 45° in the azimuth and 60° in the elevation and, since $P(\theta, \varphi) = P(\theta, 0)\ \forall \varphi \in [0, 2\pi)$ if $\theta = 0$ or π, only needs to perform 18 computations of (2.3).

Fig. 4.3 shows a couple of examples of simulated trajectories and their estimated DOA. In Fig. 4.3a we can see how, for scenarios with high reverberation and low SNRs, the maximum of the SRP-PHAT maps becomes really noisy but our proposed system is able to maintain the estimated DOA quite close to the actual one. In linear systems, robust tracking with noisy estimations usually comes with the cost of being slow to track fast changes, at least with casual systems, but we can see how our model was able to follow the sudden change in the azimuth of the source at the fifth second of the trajectory. In Fig. 4.3b we can see how, when working with low-resolution power maps, our system can predict the DOA with much higher precision than the

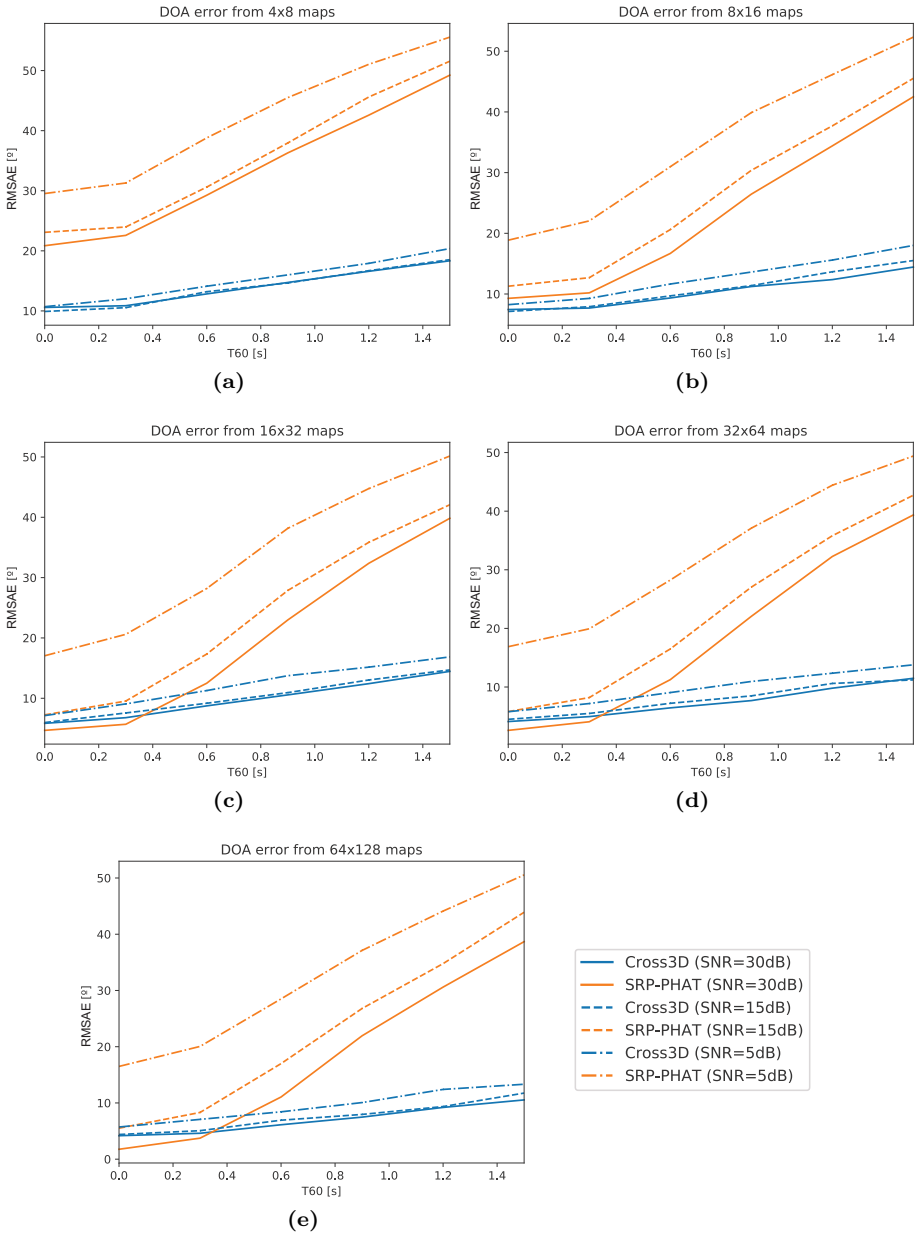
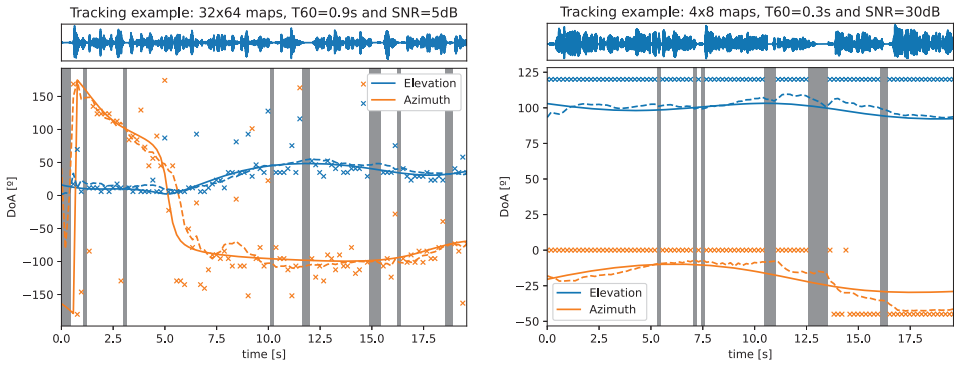


Figure 4.2: Localization root mean squared angular error (RMSAE) for several power map resolutions, SNR and reverberation times. The silent frames were not included in the computation of the RMSAE

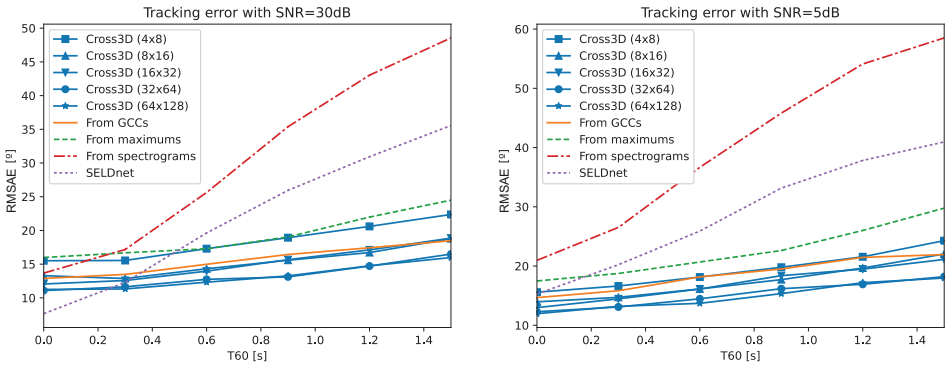


(a) **(b)**

Figure 4.3: Examples of the DOA estimated in a scenario with T_{60}=0.9 s and SNR=5 dB using maps with 32 × 64 resolution (a) and in a scenario with T_{60}=0.3 s and SNR=30 dB using maps with 4 × 8 resolution (b). The solid line represents the actual DOA of the source, the dashed line the estimated DOA and the crosses represent the maximum of each SRP-PHAT power map. Grey segments indicate silent frames.

maximums of the maps. This could not be done with a two-step DOA estimation and tracking algorithm that performed the tracking based only on the maximum of the maps. Our system is able to analyze the whole maps and it was able to learn to exploit the patterns in the SRP-PHAT maps to achieve higher resolution than the grid used to compute the maps.

Finally, we also tested the baseline methods under different reverberations and SNRs to compare the robustness of each model. In this case, since all the methods include tracking capabilities, we did not exclude the silent frames when we computed the RMSAEs shown in Fig. 4.4. We can see how the best results are obtained using our method with high-resolution power maps but, even reducing the resolution, its performance is still competitive. Using 1D CNN over the coordinates of the maximums of 64×128 SRP-PHATs maps performs worse than using our 3D CNN over 4×8 maps, so we can conclude that our model is exploiting the patterns present in the SRP-PHAT maps and not only using the information of the position of its maximums (this was also suggested by Fig. 4.3b). Using a 1D CNN over the GCCs —which is an approach, to the best of the authors' knowledge, unpublished— have a performance between using



(a) **(b)**

Figure 4.4: Tracking root mean squared angular error (RMSAE) of Cross3D with several power map resolutions and the baseline methods for SNR=30 dB (a) and SNR=5 dB (b) and several reverberation times. The silent frames were also included in the computation of the RMSAE.

3D CNNs over 4×8 and 8×16 maps and may be an interesting approach when a lower computational cost is needed. Finally, the models that use spectrograms as inputs perform well in favorable scenarios (SELDnet even outperforms our proposal in low-noise anechoic chambers) but they are not very robust against noise and reverberation.

4.1.4.3 LOCATA dataset

In order to confirm that, although it was trained with a simulated dataset, our system is general enough to track sound sources recorded in real rooms, we tested it with the LOCATA challenge dataset [137], which contains several recordings with the same array that we had simulated to train the models. We used the development dataset and we focused on tasks 1, 3, and 5 of the challenge: a static loudspeaker recorded with a static array, a moving talker recorded with a static array, and a moving talker recorded with a moving array; it is worth mentioning that the array was static in all the simulations employed to train the model. For the robot head microphone array that we simulated in the training dataset, the development dataset contains 3 recordings for each task and its ground-truth positions.

Table 4.2: RMSAE [°] of the DOA estimated for the LOCATA dataset with Cross3D using several map resolutions and the baseline tracking methods. The silent frames were included in the computation of the RMSAE.

	Model:	Cross3D					1D CNN		2D CNN	SELDnet
	Input:	SRP-PHAT maps					GCCs	Maximums	Spectrograms	
		4x8	8x16	16x32	32x64	64x128				
Task 1	Recording 1	17.93	11.92	8.30	4.62	5.16	16.18	7.54	93.76	29.70
	Recording 2	18.90	7.68	6.68	4.90	3.91	12.60	5.19	64.18	38.44
	Recording 3	10.35	6.34	2.98	3.25	2.24	11.57	5.09	140.21	54.81
	Average	15.72	8.65	5.99	4.26	3.77	13.45	5.94	99.38	40.98
Task 3	Recording 1	23.06	18.11	13.79	12.43	9.92	13.59	14.04	70.86	50.57
	Recording 2	20.97	13.71	10.01	8.36	9.22	14.17	12.02	83.42	48.71
	Recording 3	21.05	12.74	9.83	7.69	6.60	15.21	13.29	82.48	57.29
	Average	21.69	14.85	11.21	9.49	8.58	14.32	13.12	78.92	52.86
Task 5	Recording 1	11.93	10.83	7.25	5.74	5.49	10.93	10.53	58.33	37.24
	Recording 2	20.92	16.16	16.08	12.18	13.59	17.33	17.42	41.98	73.17
	Recording 3	23.57	18.25	13.58	15.64	15.49	20.14	23.58	66.91	66.50
	Average	18.81	15.08	12.31	11.19	11.52	16.13	17.18	55.74	58.97
Average		18.74	12.86	9.83	8.31	7.96	14.64	12.08	78.01	50.94

The only modification to the proposed technique that we made after seeing its performance with the LOCATA dataset was the use of a VAD. All the hyperparameters of the model and the acoustical properties of the training dataset were selected according only to the results obtained with simulated datasets. In other words, we used simulated signals for training and validation and the LOCATA recordings only for testing.

Table 4.2 shows the RMSAE of estimating the DOA of the source of each recording using our technique and with the baseline methods. Although it is difficult to draw conclusions from such a low number of recordings, we can see how the proposed tracking system clearly outperforms the baseline methods that use spectrograms as inputs and that it also outperforms the 1D CNN methods when we use maps with at least 16×32 resolution.

According to [137], the reverberation time of the room where the recordings were performed was $T_{60} \approx 0.5\,\mathrm{s}$, so we can observe some degradation in the performance of Cross3D when it is used over low-resolution power maps compared with the results obtained with the simulated dataset (see Fig. 4.4), but it disappears when the res-

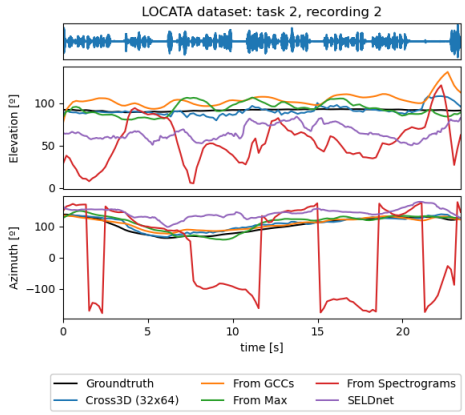


Figure 4.5: DOA estimated for the second recording of the second task of the LOCATA challenge using maps with Cross3D over 32×64 maps and the baseline methods.

olution of the maps increases; actually we even reach lower errors in the LOCATA dataset than with the simulated test dataset. Using a 1D CNN also suffers a similar degradation, but its most dramatic impact is on the methods which use spectrograms as inputs. In contrast, the use of a 1D CNN over the coordinates of the maximums of high-resolution SRP-PHAT maps does not suffer almost any degradation; but it may not be an interesting approach since, having computed the 64×128 resolution maps, we can obtain far better results using the whole maps as inputs of Cross3D.

As an example, Fig. 4.5 shows the DOA estimation of the second recording of the third task of the LOCATA dataset, where all the methods obtained an RMSAE quite close to their average. We can see how Cross3D performs the best estimation of the analyzed methods both for the elevation and for the azimuth. We can also see that the LOCATA dataset has longer silences than the ones present in the simulated dataset, which could also explain why some of the methods obtained lower results with this dataset. In order to make the methods based on CNNs more robust against longer silences, we should include them in the simulation of the training dataset and, probably, increase the temporal receptive field of the models, which could be done by increasing the number of layers, the temporal size of its kernels, or including longer temporal dilations in the convolutions.

4.2 A sound source localization model equivariant to the rotations of the source and the array

In the previous section, we have presented a new model based on 3D CNNs applied over SRP-PHAT power maps that has been proved to be more robust than other state-of-the-art models for SSL. However, this model is not really equivariant to the rotations of the source or the array, since the projection of the spherical SRP-PHAT maps into the 2D equiangular grid employed as input of the convolutions generates position-depending deformations that are not intrinsically compensated by the 3D convolutions (though the model could learn to do it during the training). In addition, this equiangular sampling is not the optimal way to sample the SRP-PHAT maps since it oversamples the regions close to the poles.

Many new kinds of CNNs have been proposed in recent years to obtain equivariance to spherical rotations. In [77] and in [168], it was proposed to perform the convolutions in the spherical harmonic domain and then transform their result back to the spatial domain to pointwise apply the nonlinear activations. Since then, several modifications of this approach have been published, some of them proposing nonlinear activations in the harmonic domain [169, 170]. Another approach is to analyze the spherical signal as a graph where each point is connected to its neighbors [171, 172, 173]. By working this way, they avoid the need to work in the harmonic domain, but, in most cases, their kernels are restricted to being isotropic, i.e. to having circular symmetry.

In this section, we propose the use of a third approach: the icosahedral CNNs presented in [76]. These networks are only strictly equivariant to the 60 rotational symmetries of the icosahedron instead of the continuous space of spherical rotations, but they have a much more efficient implementation based on standard 2D convolutions. They have been proven to smoothly generalize to the continuous space of spherical rotations when these rotations are shown during the training of the model and their hexagonal kernels are not restricted to be isotropic thanks to saving the results of every possible orientation as separate channels.

In addition, in order to preserve the equivariance of the whole model, we present a new layer, which we call soft-argmax, to replace the fully connected layers that

are usually employed at the end of the convolutional models. This layer interprets the output of the last convolutional layer as the probability distribution of the source position and computes its expected value, so allows us to convert a classification output into a regression output in a differentiable and interpretable way without adding any trainable parameters to the model.

4.2.1 Icosahedral CNNs

Several techniques have been recently proposed to extend the translation equivariance of conventional CNNs to spherical rotations, most of them based on the spherical harmonics domain. Although they are only equivariant to the 60 icosahedral rotations instead of the continuous space of spherical rotations SO(3), in this work we use the icosahedral CNNs proposed in [76] due to its efficient implementation based on conventional bi-dimensional CNNs. The icosahedron is the platonic solid with the highest number of faces and it allows us to approximate a sphere with lower error than other geometric shapes with a lower number of faces while being able to define a convolution operation on a hexagonal grid without needing any kind of interpolation and implement it using a conventional 2D convolution. The details of this implementation and some experiments proving its good performance approximating spherical signals (and its rotations) can be found in [76], but we summarize some of the main ideas in this section.

The icosahedral grid used to sample the spherical signals is built recursively starting from the vertices of the icosahedron (which we define as having a vertex in the south and another in the north pole as shown in Fig. 4.7a) and then subdividing each triangular face into 4 smaller triangles by introducing a new point in the center of each edge as shown in Fig. 4.6. Repeating this process r times, we obtain a grid with $5 \cdot 2^{2r+1} + 2$ points. As we can see in Figs. 4.7a and 4.7c, apart from the vertices of the icosahedron, which only have 5, every one of these inner sampling points has 6 neighbors, so we can see them as hexagonal pixels. In [76], due to their pentagonal shape, it is proposed to keep the corners values to 0 to preserve the equivariance of the model but, in order to avoid artifacts around them after applying the convolutions,

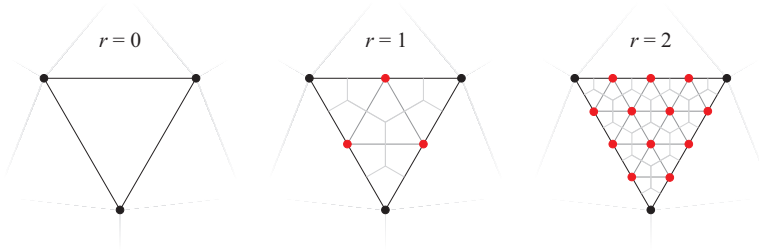


Figure 4.6: Sampling points of a face of the icosahedron for different resolutions.

we replace that 0 with the average value of their 5 neighbors (which also preserves the equivariance of the model).

Due to the hexagonal shape of the pixels defined this way, the icosahedral CNNs use hexagonal kernels which can be stored in 3×3 bi-dimensional kernels. If we want the model to be equivariant to the icosahedral rotations without restricting the kernels to be isotropic, we have to consider their 6 possible rotations so, for each kernel in the convolution, we have to work with 6 channels instead of with just one as it is done in conventional convolutions. Finally, in order to implement the icosahedral convolutions using standard 2D convolutions, a projection of the whole icosahedral grid into a $5 \cdot 2^r \times 2^{r+1}$ rectangular grid (Fig. 4.7b) is presented in [76] along with a way to circularly pad it into a $5 \cdot (2^r + 2) \times 2^{r+1} + 2$ extended grid to preserve the equivariance of the model.

To sum up, we can apply a convolution with C kernels over an icosahedral grid of resolution r using a conventional 2D convolutional layer with 3×3 kernels and $6C$ channels over a $5 \cdot (2^r + 2) \times 2^{r+1} + 2$ image.

In [76], it is not detailed how the pooling layers are implemented. In our implementation, an icosahedral pooling layer reduces an icosahedral grid of resolution r to a new one of resolution $r-1$ where each new hexagonal pixel is computed as the average of the pixel which has its same center and its 6 neighbors. This can be seen as the icosahedral equivalent of a 2D pooling with kernel size 3×3 and stride 2×2.

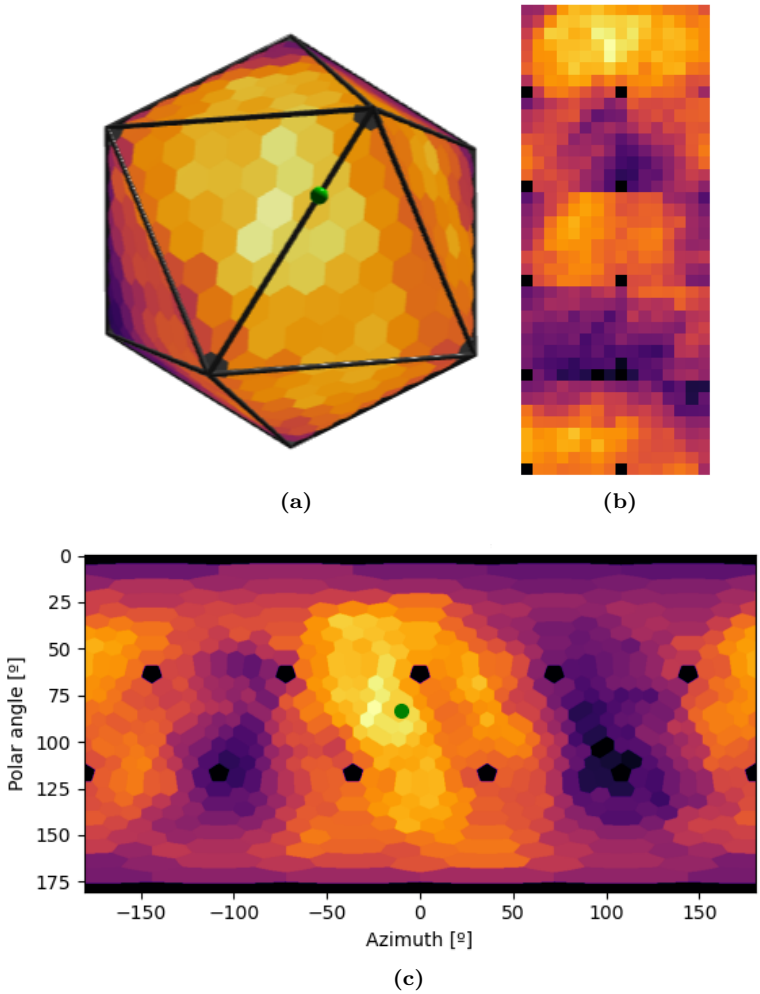


Figure 4.7: Example of an icosahedral SRP-PHAT power map with resolution $r = 3$ in a high reverberation low noise scenario: T_{60}=1.0 s and SNR=30 dB (a), the 2D representation employed for the convolution implementation (b) and a spherical projection for visualization purposes (c). The green sphere/circle indicates the actual DOA of the sound source.

4.2.2 Soft-argmax regression

In the model presented in the previous section using 3D convolutions, in order to perform the regression of the source coordinates after the convolutional layers, we flattened the activation maps of the last layer and fed it to several fully connected layers (actually, we used 1D convolutions in order to allow the model to also take into account the previous frames). Even when employing pooling layers along with the convolutional layers to reduce the number of activations that reach the fully connected layers, this approach has a high computational cost and highly increases the number of trainable parameters of the model, which increases its memory consumption and its risk of overfitting.

For our model based on icosahedral convolutions, we replace those fully connected layers with a new soft-argmax function, where we use a soft-max layer to ensure that the sum of the whole activation map is 1.0 and then we just sum the results of multiplying each hexagonal pixel by the coordinates that they represent in the icosahedral grid:

$$\text{soft-max}\left(P\left(\boldsymbol{x}\right)\right) = \frac{e^{P(\boldsymbol{x})}}{\sum_{\boldsymbol{x}\in\mathcal{X}} e^{P(\boldsymbol{x})}} = \frac{e^{P(\boldsymbol{x})-\max(P(\boldsymbol{x}))}}{\sum_{\boldsymbol{x}\in\mathcal{X}} e^{P(\boldsymbol{x})-\max(P(\boldsymbol{x}))}} \tag{4.1}$$

$$\text{soft-argmax}\left(P\left(\boldsymbol{x}\right)\right) = \sum_{\boldsymbol{x}\in\mathcal{X}} \boldsymbol{x}\,\text{soft-max}\left(P\left(\boldsymbol{x}\right)\right), \tag{4.2}$$

where $P\left(\boldsymbol{x}\right)$ is the output of the last convolutional layer of the model and $\boldsymbol{x} \in \mathcal{X}$ are the coordinates of the points of the icosahedral grid $\mathcal{X}$ where it is sampled. The subtractions of $\max\left(P\left(\boldsymbol{x}\right)\right)$ inside the exponential functions are done for numerical stability reasons without affecting the analytical result.

This way, the output of the icosahedral convolutions, after being normalized with the soft-max function, can be seen as the probability distribution of the coordinates of the source and the output of the soft-argmax function as its expected value. Another advantage of this approach, apart from the model interpretability and the reduction in the computational and memory costs, is that we avoid introducing any non-equivariant layer to the model.

Although we could directly estimate the spherical coordinates of the source defining $\boldsymbol{x}$ in spherical coordinates, we estimate the 3D coordinates of the unitary vector pointing at the direction of the source by defining $\boldsymbol{x}$ in 3D Cartesian coordinates. As previously explained, it has been proven that this brings better results than directly inferring the elevation and azimuth angles [53, 54] and, in addition, continuing with the interpretation of the output of the CNN as a probability distribution function, minimizing the mean squared error (MSE) of this vector does not only imply reducing the distance between its expected value and the source position but also reduces its variance, since a completely unitary vector would only be possible if only one pixel is activated. Therefore, we can see the norm of the output vector as a measurement of the confidence in the DOA estimation, being closer to 1 when the confidence is higher as done with the activity-coupled Cartesian direction of arrival (ACCDOA) representation [58].

In Fig. 4.8 we can see an example of the output of the last convolutional layer of the trained model after being normalized with the soft-max function and how it represents a probability distribution whose expected value accurately estimates the actual DOA of the sound. It is worth noting how the precision of the estimated DOA exceeds the grid used to sample the probability distribution, overcoming the main issues of the classification approaches to SSL.

4.2.3 Proposed technique

4.2.3.1 Model architecture

Since the regression of the estimated DOA is performed with our soft-argmax layer, which already integrates the coordinates of every map point, the input of the model has only one channel with the SRP-PHAT map and it does not include the additional channels that we used in section 4.1 to indicating the coordinates of the maximum of every map. Including these channels would have broken the rotational equivariance of the model and would have made it more difficult its extension to multi-source scenarios.

As shown in Fig. 4.9, we combine icosahedral convolutions with one-dimensional convolutions operating in the time dimension in order to take the temporal context

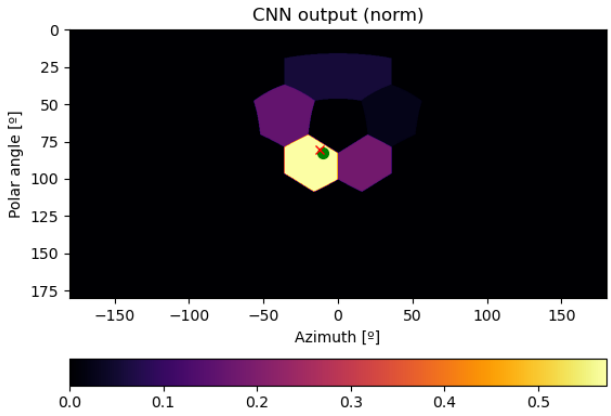


Figure 4.8: The output of the last convolutional layer after passing through the softmax function, corresponding to the input maps shown in Fig 4.7, and the result of the soft-argmax function converted into spherical coordinates (red cross). The green circle indicates the actual DOA of the source.

into account when performing the DOA estimation while being equivariant to both icosahedral rotations and temporal translations. Each convolutional unit is composed of an icosahedral and a temporal convolution followed by layer normalization [174] and rectified linear unit (ReLU) activation. The temporal convolution has a kernel of size 5 operating causally, i.e. its receptive field only includes past maps, and both the icosahedral and the temporal convolution have 32 kernels. To preserve the equivariance of the model, the 6 kernel-orientation channels of every icosahedral kernel are seen as 6 independent signals by the temporal convolutions. The impact of the number of convolutional kernels in the performance of the model is analyzed in section 4.2.4.2.

In [174], it was hypothesized that layer normalization did not provide relevant improvements in convolutional neural networks since the hidden units close to the boundaries of the images did not follow the same distribution as the rest of the hidden units. However, our inputs do not have boundaries and we found that, adding layer normalization to our model, it converged faster and more robustly during the training. To keep the model equivariant to the icosahedral rotations, we have implemented a layer normalization that normalizes the inputs along the 32 channels and its 6 kernel orientations but the scale weights are tied for the 6 kernel orientations in the affine transformation.

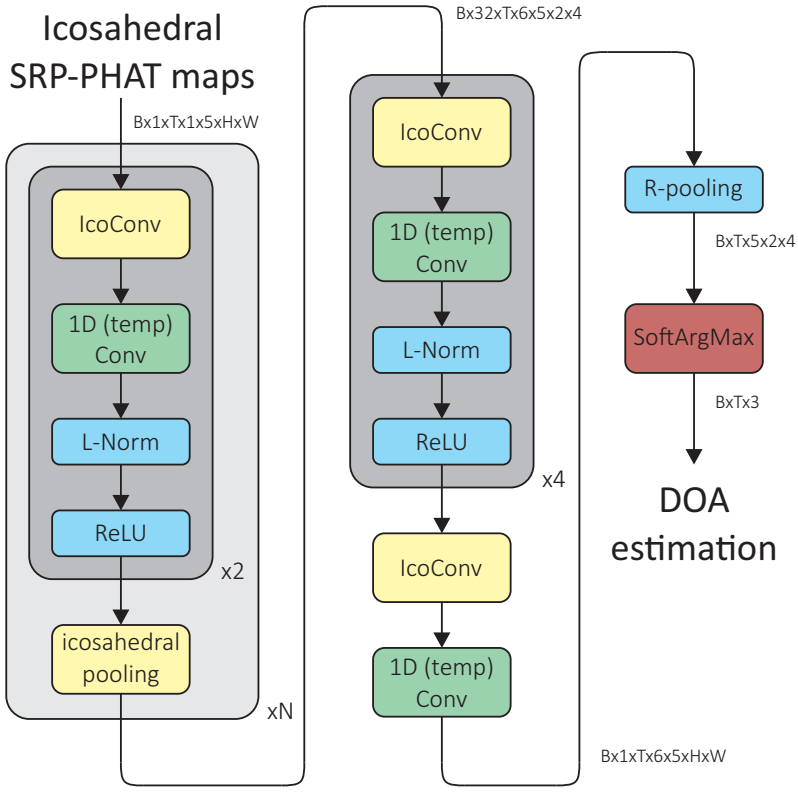


Figure 4.9: Architecture of the proposed model. B is the batch size, T is the number of temporal frames of the trajectory, $H = 2^r$ and $W = 2^{r+1}$ are the height and the width of the projections of the icosahedral grid, and $N = r - 1$ is the number of down-sampling units used for that input resolution.

We concatenate two of these convolutional units to an icosahedral pooling to build a down-sampling unit and stack as many of these down-sampling units as needed to get an $r = 1$ icosahedral map (in the case of using maps with $r = 1$ as input we do not use any down-sampling unit). When we have a minimal-size icosahedral map, we concatenate 5 convolutional units (the last one with only an output channel for the 1D convolution and without layer normalization and the ReLU activation). This way, we ensure that the receptive field of all the output cells of the icosahedral CNN includes all the cells of the input map independently of its resolution. Finally, we use a max-pooling layer over the 6 kernel-orientation channels and feed the resulting icosahedral maps to the soft-argmax layer explained in section 4.2.2.

4.2.3.2 Training

For the training dataset, we used again the technique described in chapter 3 to generate random source trajectories and simulated them using our GPU implementation of the Image Source Method [61] at a sample rate of 16 kHz using utterances from the LibriSpeech train-clean-100 dataset [135] as source signals. We simulated the same 12-microphones array included in the LOCATA dataset [137, 138] as we did for Cross3D, which has a minimum and maximum inter-microphone distances of 1.3 cm and 12.1 cm respectively.

As done for the Cross3D models, we computed the SRP-PHAT maps using frames of length $K = 4096$ samples (i.e. 256 ms) with a hop size of $3K/4$. We normalized the maps by subtracting their mean and dividing them between their maximum and used the Voice Activity Detector of the Web Real-Time Communication (WebRTC) project [136] to turn to 0 the maps corresponding to silent frames.

As explained in the previous section, using this approach we have an infinite-size dataset, but we define an epoch as 585 random trajectories of 20 s, each one with an utterance randomly taken from one of the 585 chapters present in the LibriSpeech train-clean-100 subset. We used Pytorch [166] to train the model using the Adam algorithm [165] over 50 epochs. Similar to the curriculum learning [134] strategy employed to train the Cross3D models, we keep fixed the SNR of the simulations to

Table 4.3: Models employed for the evaluation.

Model	Input		SRP-PHAT computations	Trainable parameters	Temporal receptive field
IcoCNN	Icosahedral SRP-PHAT maps	r=1	30	193 441	4.10 s
		r=2	150	289 953	5.63 s
		r=3	630	386 465	7.17 s
		r=4	2550	482 977	8.70 s
Cross3D	Equiangular SRP-PHAT maps	4×8	18	526 372	5.63 s
		8×16	98	946 340	6.40 s
		16×32	450	1 693 988	
		32×64	1922	5 626 148	7.17 s
		64×128	7938	21 354 788	
1D CNN	GCCs		0	11 282 436	7.17 s

30 dB during the first 25 epochs and then we employed uniformly distributed random values from 5 dB to 30 dB in the following epochs.

4.2.4 Evaluation

4.2.4.1 Baseline techniques

Since, in the previous section, they were the techniques that proved to be the most robust against reverberation, we compared the proposed technique with Cross3D and the 1D CNN operating over the GCC coefficients.

As we can see in Table 4.3, the proposed model has a much lower number of trainable parameters than the other models. This is because the final regression is performed with the soft-argmax function (without trainable parameters) instead of with fully connected layers.

4.2.4.2 Simulated dataset

In order to analyze the performance of the proposed model under different acoustic conditions, we first evaluated our model using synthetic signals simulated following the same procedure employed for the training dataset. We found that some utterances included a short period of silence at the beginning which artificially biased the results, so we have not taken into account the localization error of the first 5 frames, i.e. the

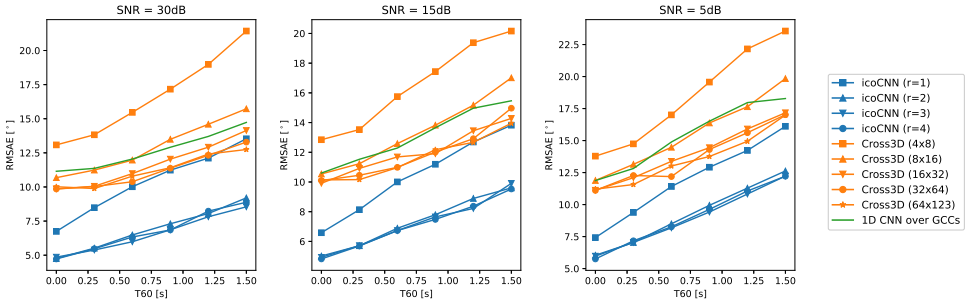


Figure 4.10: Localization root mean squared angular error (RMSAE) under several simulated conditions for the proposed and the baseline techniques.

first second, of each trajectory even if they were used to train the models. All the root mean localization errors described in this section include the frames where the source was silent, but the models are able to continue estimating its position thanks to the temporal context included in their receptive fields through the temporal convolutions.

As we can see in Fig. 4.10, the proposed models clearly outperform the baselines even using maps of lower resolution and having far less trainable parameters. To make this even clearer, Fig. 4.11 plots the localization error represented as a function of the number of computations of the SRP-PHAT functional (2.3), needed to compute the input maps of each model. It is worth noting that the reverberation times $T_{60} = 0.0\,\mathrm{s}$ and $T_{60} = 1.5\,\mathrm{s}$ are out of the range of reverberation times used during the training, but the model generalizes well to them continuing with the same tendency shown with the rest of reverberation times.

We can also see how using icosahedral maps with a resolution higher than $r = 2$ does not seem to improve the accuracy of the DOA estimations. Considering that those models had a higher number of down-sampling units, and therefore more trainable parameters and longer temporal receptive fields, we can conclude that the maps with $r = 2$ already contain all the information useful for tracking. This limit is probably determined by the size of the array we are using to compute the maps and that limits their spatial bandwidth; using bigger arrays would probably allow us to obtain even better results from higher-resolution maps.

In Fig. 4.12, we can see an example of a random trajectory in a scenario with

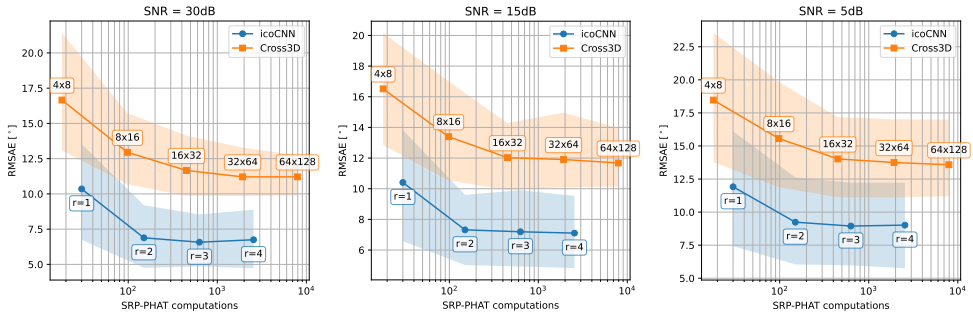


Figure 4.11: Localization root mean squared angular error (RMSAE) vs the number of computations of equation (2.3) needed to compute each input map. The semitransparent area indicates the whole reverberation interval from $T_{60} = 0.2\,\text{s}$ to $1.3\,\text{s}$ and the solid line indicates its average value.

high reverberation time and low noise using maps with resolution $r = 2$. We can see that the maximums of the SRP-PHAT maps are in spurious positions in many frames and, even in those where they are in the grid position closest to the ground truth, they are quite far due to their low resolution. However, we can see that the estimation of the proposed model stays always closer to the actual DOA of the source even during the silent frames since the icosahedral convolutions allow the model to analyze the whole maps instead of just taking only the information of the position of their absolute maximums and thanks to the temporal context provided by the temporal convolutions.

In Fig. 4.8 we have seen the $r = 1$ probability distribution inferred by the model for the $r = 3$ maps depicted in Fig. 4.7. We can see how the model is able to accurately adjust the probability assigned to every hexagonal pixel around the pentagonal vertex so the result of the soft-argmax function is precisely displaced from the center of the closest hexagonal pixel to the actual DOA of the simulated source. It is worth saying that, even if the vertex values were replaced by the average value of their neighbors during the convolutional layers, they are turned to 0 before the soft-argmax function.

Since that was the only difference that we had introduced in our implementation of the icosahedral convolutions with respect to the original implementation described in [76], we also studied the effect of replacing the vertices of the icosahedron by zeros

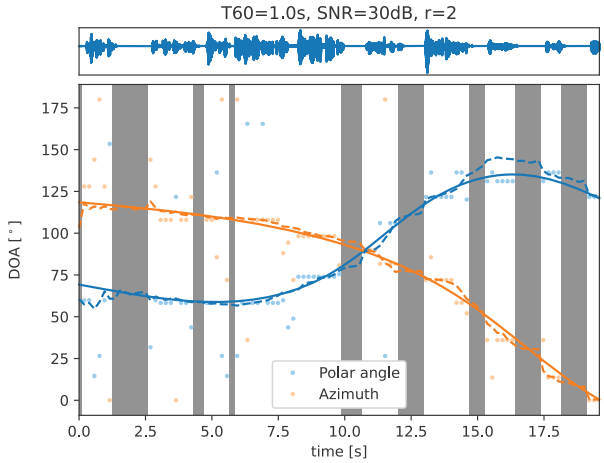


Figure 4.12: DOA estimation for a random trajectory simulated with T_{60}=1.0 s and SNR=30 dB. The solid lines represent the ground-truth DOA and the dashed lines the estimation performed by the proposed method using input maps of resolution $r = 2$. The dots indicate the position of the maximum of those maps.

Table 4.4: Mean RMSE [°] in the simulated dataset with the different strategies to handle the vertices in the icosahedral convolutions.

Vertices values	Map resolution		
	r=1	r=2	r=3
zeros	11.02	7.81	7.31
neighbors' average	10.89	7.81	7.57

instead of by the average of their neighbors. As we can see in Table 4.4, the difference between both strategies is negligible but, considering that the impact of the averaging in the computational cost of the model is also negligible, we still think that our strategy is preferable since it reduces the artifacts around the vertices in the hidden layers of the model.

In Table 4.5, we can see the mean RMSAE for all the reverberation times and SNRs of the evaluation dataset for several numbers of convolutional kernels. It should be noted that, if we need to reduce the computational complexity and memory consumption of the model, we can reduce the number of convolutional kernels without having a too large impact on the localization accuracy. This is possible thanks to the equivariances of the model since, contrary to conventional 2D CNNs, the same kernel

Table 4.5: Mean RMSAE [°] in the simulated dataset with different numbers of convolutional kernels.

Convolutional kernels	Map resolution		
	r=1	r=2	r=3
4	14.16	12.78	10.19
8	12.63	9.82	8.84
16	11.89	8.61	7.69
32	10.89	7.81	7.57
64	10.21	7.38	6.71

can model the different rotations of its pattern. This fact, combined with avoiding the distortions that the 2D projection of the maps generated in Cross3D, allows the proposed model to extract even more information than the previous model using less convolutional kernels.

4.2.4.3 LOCATA dataset

To confirm that the models trained with simulated signals are general enough to work with signals recorded in real environments, we have tested them using the recordings of tasks 1, 3, and 5 (i.e. the tasks with only one source) of the evaluation partition of the LOCATA dataset [137]. Table 4.6 and Fig. 4.13 show the average results for every task and for the whole evaluation partition and we have included the root mean squared error (RMSE) of every recording in appendix B.

As we can see in Fig. 4.14, some of the recordings of the LOCATA dataset include long periods of silence when the sound source was moving. During these silent periods, the models obviously cannot keep tracking the movements and if they constitute a high percentage of a recording, its RMSAE will be strongly biased by them; therefore, Table 4.6 also includes the RMSAEs without taking into account these silent frames. In addition, it is worth mentioning that the size of the dataset is quite small, with the 23 recordings used for this test adding less than 7 minutes after having removed the initial silences, which makes it very sensitive to any anomalous circumstance that could appear even if it is only present for a short period of time.

We can see that the difference between the model using icosahedral convolutions

Table 4.6: Mean RMSAE [°] of the DOA estimated for every task of the evaluation partition of the LOCATA dataset with the icosahedral CNNs using several map resolutions and the baseline tracking methods. The second (gray) numbers indicate the RMSAE without taking into account the frames when the sound source was silent.

Model:	IcoCNN				Cross3D					1D CNN
Input:	Icosahedral SRP-PHAT maps				Equiangular SRP-PHAT maps					GCCs
	r=1	r=2	r=3	r=4	4x8	8x16	16x32	32x64	64x128	
Task 1	8.04 8.34	5.88 6.08	5.57 5.78	5.25 5.23	28.62 27.09	22.47 22.16	13.89 14.32	9.84 8.51	5.28 5.48	12.54 12.80
Task 3	10.53 8.97	8.94 7.29	10.07 6.93	9.57 7.87	18.51 17.98	17.56 17.50	12.76 12.16	11.18 10.33	9.92 8.86	12.09 11.62
Task 5	15.48 13.03	11.13 8.87	11.54 9.66	10.95 8.49	19.13 15.82	17.49 13.38	13.03 10.75	12.20 10.63	12.49 10.73	16.47 13.31
Average	10.20 9.50	7.69 6.97	7.85 6.87	7.43 6.51	24.90 23.47	20.32 19.24	13.45 13.03	9.84 9.52	7.86 7.36	13.30 12.65
Median	9.91 9.94	7.17 6.79	6.66 6.53	6.60 6.45	18.51 18.25	15.32 14.43	9.90 7.65	7.58 7.09	5.97 5.74	12.72 12.00
Standard deviation	5.00 4.17	3.53 2.40	3.82 2.69	3.51 2.26	15.65 14.39	13.43 13.24	12.87 13.26	7.71 8.07	5.71 5.37	5.67 5.44

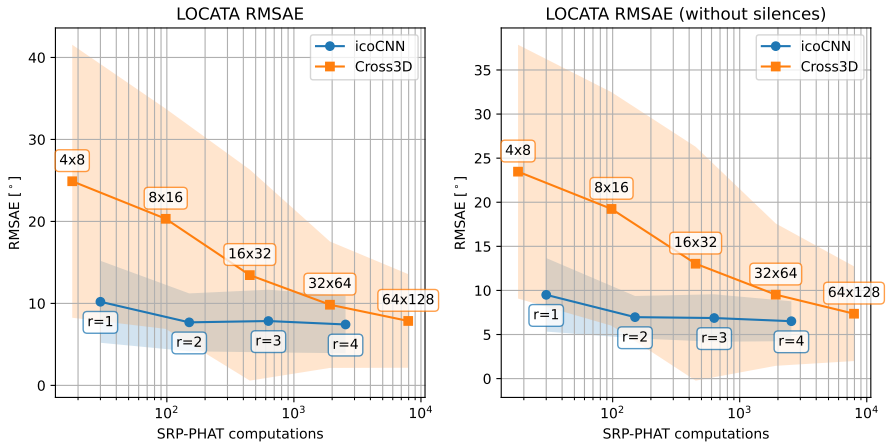


Figure 4.13: Average localization root mean squared angular error (RMSAE) in the LOCATA dataset vs the number of computations of equation (2.3) needed to compute each input map, along the whole signal (a) and without taking into account the silent frames (b). The semitransparent area indicates the average ± the standard deviation.

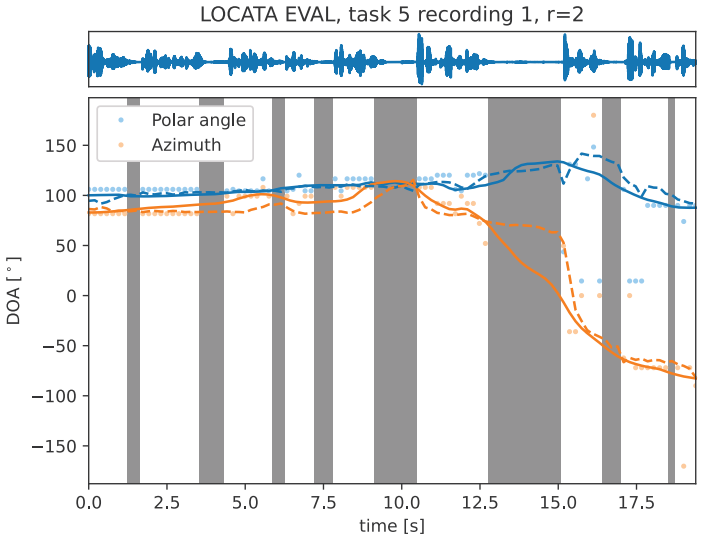
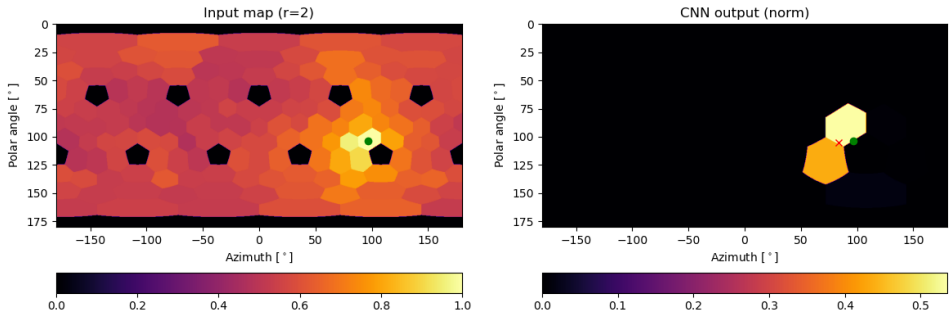


Figure 4.14: DOA estimation for the third recording of task 5 of the LOCATA dataset. The solid lines represent the ground-truth DOA and the dashed lines the estimation performed by the proposed method using input maps of resolution $r = 2$. The dots indicate the position of the maximum of those maps.

and Cross3D is not as great in this dataset as in the synthetic one, though the model using icosahedral convolutions still clearly outperforms the baseline model, especially when using low-resolution power maps. This could be due to the differences between the signals simulated with the image source method (ISM) method used to train the models and the signals recorded in a real room used for this test. It seems that we might be approaching the accuracy limit imposed by this dataset difference, so the accuracy with real recordings can not improve even when we improve the models. In recent years, several domain adaptation techniques have been proposed to improve the accuracy of models trained with simulated signals [46, 64, 65, 66] and it would be interesting to conduct further studies along these lines.

In any case, as can be seen in Fig. 4.13, the proposed model working with maps of resolution $r = 2$ still has in the LOCATA dataset an accuracy comparable with Cross3D using maps of much higher resolution, reducing in almost two orders of magnitude both the number of SRP-PHAT computations and the number of trainable parameters, which might be crucial in applications where the sound source localization must be done in low-cost devices in real time. We can also see how the proposed



(a) **(b)**

Figure 4.15: Example of an input map from the 1st recording of the 5th task of the LOCATA evaluation dataset (a) and the output of the last convolutional layer after passing through the soft-max function and the result of the soft-argmax function converted into spherical coordinates (red cross) (b). The green circle indicates the actual DOA of the source.

model provides far more consistent results than the baseline (whose results have a much higher variance), which also suggests a better generalization from the simulated dataset used for training to the actual recordings of the LOCATA dataset.

Finally, Fig. 4.15 shows an example of an input SRP-PHAT map extracted from the first recording of the fifth task of the evaluation partition of the LOCATA dataset and its corresponding model output. As in Fig. 4.8, we can see how the energy of output of the CNN is distributed in a way that, after normalized with the soft-max function and interpreted as a probability distribution, its expected value approaches the actual DOA of the sound.

4.3 Conclusions

In this chapter, we have presented two new models for sound source localization, both of them using fully convolutional architectures over SRP-PHAT power maps. The first one uses a 3D CNN and the second one uses an icosahedral CNN. Since they do not use any non-causal element, such as bidirectional recurrent layers, they are feasible for real-time applications and provide a new DOA estimate with every new input map; i.e., every 192 ms.

The experiments performed show that the SRP-PHAT maps are a good input feature to be used in tracking systems based on deep learning, being much more robust to reverberation and noise than the use of spectrograms as proposed in most of the recent literature. They also prove that it is possible to obtain a good tracking performance using only causal convolutional layers and that non-causal recurrent layers are not needed.

In addition, our second model also shows how using architectures that fit the equivariances of the problem that we want to solve allows us to increase the accuracy of the models while reducing the number of trainable parameters. The model is completely equivariant to time shifts and to the 60 rotational symmetries of the icosahedron, which is a good approximation of the continuous space of spherical rotations. It can be implemented using conventional 2D convolutional layers and has a low number of trainable parameters thanks to replacing the fully connected layers that are typically employed after the convolutional layers with a differentiable version of the argmax function, which, in addition, allows us to interpret the output of the convolutional layers as a probability distribution whose expected value is the DOA estimation.

Finally, it is worth mentioning that, even if we have focused on using SRP-PHAT maps as inputs for our models, any other spatial pseudo-spectrum could also be used (such as those computed using the MUSIC algorithm), so further studies comparing different input features and even combining them as different input channels would also be interesting.

5

Permutation invariant multi-source tracking

In this chapter we present our works on multi-source tracking. We first introduce the tracking problem and its invariance to the permutations of the sources in section 5.1 and our attempts to develop an iterative multi-source localization system based on source cancellation in section 5.2. Then we present our main proposals for sound source tracking (SST): the permutation invariant gated recurrent units (PI-GRUs) and the sliding permutation invariant training (sPIT) in sections 5.3 and 5.4 respectively. Finally, we evaluate these proposals in section 5.5 and we end the chapter with some conclusions in section 5.6.

This chapter includes the reproduction of figures and text fragments from [VII] with permission of the copyright holders.

5.1 Sound source tracking (SST) and the permutation invariance

In the previous chapter, we focused on providing direction of arrival (DOA) estimates as accurate as possible in scenarios where we knew that one and only one source was always present. To train deep-learning models to do this in a supervised manner, we could easily define loss functions by comparing the estimates of the model with

the ground-truth DOAs. However, when moving from single to multiple sound source localization (SSL), we need to assign every estimate to one of the ground-truth DOAs. If there are no possible criteria to classify or order the sources, we cannot expect the estimates to follow the same order as they have in the ground-truth labels and, therefore, any permutation of the estimates is equally valid. If we do not take into account this permutation invariance when defining our loss function, our models will converge to useless solutions as estimating all the sources in the middle of the actual DOAs since, not being able to match the order of the sources in the ground-truth labels, this is the feasible solution that minimizes the average localization error of the sources.

In addition, when moving from SSL to sound source tracking (SST), apart from providing DOA estimates, we must also detect when a new source appears or disappears and be able to associate every new estimation to the correct tracked trajectory avoiding identity switches (IDSs). In classical tracking systems, this is typically solved in two steps following what is usually called tracking-by-detection: they first try to detect every source present in a time frame and then they filter these estimated DOAs associating them to the tracked trajectories and determine if new sources have appeared or some of the tracked sources have disappeared. Although the deep-learning models for SST usually do not split the tracking into two completely separated steps, many of them use recurrent convolutional neural network (RCNN) where, according to the temporal receptive field of every layer and the information that they receive, we can expect the convolutional layers of the model to specialize in the detection and the recurrent layers in the tracking.

As shown in Fig. 5.1, we can see a tracking-by-detection system as a recurrent process that, at every time frame, combines the new DOA estimates provided by the detector with the last tracking output to update the tracked trajectories. Under this model, we would expect the tracking system to be invariant to the permutations of the DOA estimates generated by the detector (since they are an unordered set) and equivariant to the permutations of the tracked trajectories (since they are also an unordered set but the new estimates must be assigned to the correct trajectory).

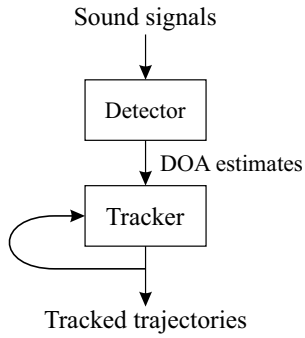


Figure 5.1: Diagram of a tracking-by-detection system seen as a recurrent process.

Table 5.1: Mean RMSAE [°] simulating just one source and simulating two sources and analytically canceling one of them.

Number of sources	Map resolution		
	r=1	r=2	r=3
One	11.02	7.81	7.31
Two (one analytically canceled)	31.49	26.96	26.03

5.2 Iterative multi-source localization through source cancellation using deep learning

In order to build a tracking system following a tracking-by-detection approach, we first need to have a detector that outputs a set of DOAs for every time frame. Having developed a technique that is able to cancel a source from the generalized cross-correlation (GCC) functions of a microphone array and a deep-learning model for single-source localization, we tried to combine both elements to design an iterative SSL system with the structure shown in Fig. 5.2a.

We first tried to simply use the analytical cancellation technique presented in chapter 2 to obtain the canceled maps. However, as can be seen in figures from Fig. 2.16 to Fig. 2.20, this cancellation has a too aggressive effect on the steered response power with phase transform (SRP-PHAT) maps when used in reverberant environments, which leads to the very poor performance of the model presented in the previous chapter even when trained using canceled maps (see Table. 5.1). Therefore, we tried to use deep learning techniques to improve the cancellation technique.

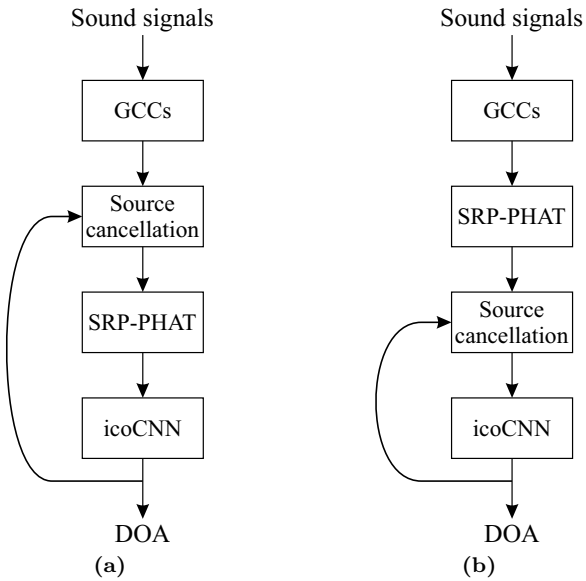


Figure 5.2: Structure of iterative multi-source localization systems using source cancellation over the GCCs (a) and over the SRP-PHAT maps (b).

The cancellation technique presented in chapter 2 computes the canceled GCCs as a linear combination of time-shifted versions of the original GCCs, where the scale coefficients applied to every GCC in the linear combination are constant and the time shifts are calculated based on the array geometry and the DOA of the source we want to cancel. Our first attempt to include neural networks into this process was letting a multilayer perceptron (MLP) compute these coefficients and time shifts with the idea of being able to apply multiple iterations of the cancellation technique using different sets of coefficients and time shifts generated with a neural network for the same canceled DOA.

First, in order to confirm that an MLP was able to, at least, replicate the analytical solution, we trained an MLP to replicate it as shown in Fig. 5.3a and we found that a small MLP with just one hidden layer was indeed able to easily replicate it. After that, we implemented the cancellation technique in the frequency domain so it was easier to track gradients through it but, when we tried to train the same MLP to minimize the error of the canceled GCCs (Fig. 5.3b) or the SRP-PHAT maps (Fig.

5.3c) generated using it, we found that the training did not converge to any useful solution. We also tried to train the MLP to obtain the GCCs and the maps obtained without including the source that we want to cancel into the acoustic simulation but, as could be expected after having seen the previous results, this did not converge either. We think this is due to the peaks of the GCCs being too narrow, especially when using the phase transform (PHAT), which led to the GCCs with the random time shifts generated by the MLP at the beginning of the training not having any overlap with the GCCs needed for the cancellation and therefore not generating any useful gradients to train the model.

After concluding that the previous approach was not feasible, we tried to use a residual U-net [71] architecture over the SRP-PHAT maps to cancel the effect of a source in order to develop an iterative system as the one represented in Fig. 5.2b. We first tried to train it to replicate the maps obtained without including the canceled source in the acoustic simulation, but then we found that we could obtain better results by training it to generate maps whose maximum, using the soft-argmax layer presented in section 4.2.2, was at the DOA of the remaining source. This led us to believe that, especially if we use 3D convolutions in the U-Net to increase the temporal receptive field of the model, finding the second source is easier for the models than canceling the first one since the maps generated by this model did not have any similarity with the original maps and looked as if the model had located the second source and created a completely new map for it.

All these results dissuaded us from keep trying this iterative approach for multi-source localization and we decided to move to models that could locate several sources at the same time. Using the icosahedral convolutional neural network (icoCNN) presented in the previous chapter, this is easily done by just increasing the number of output channels of the last convolutional layer from 1 to the maximum number of sources that we want our model to be able to locate and then applying the soft-argmax layer independently to every one of these channels so each one can provide a different DOA for a different source.

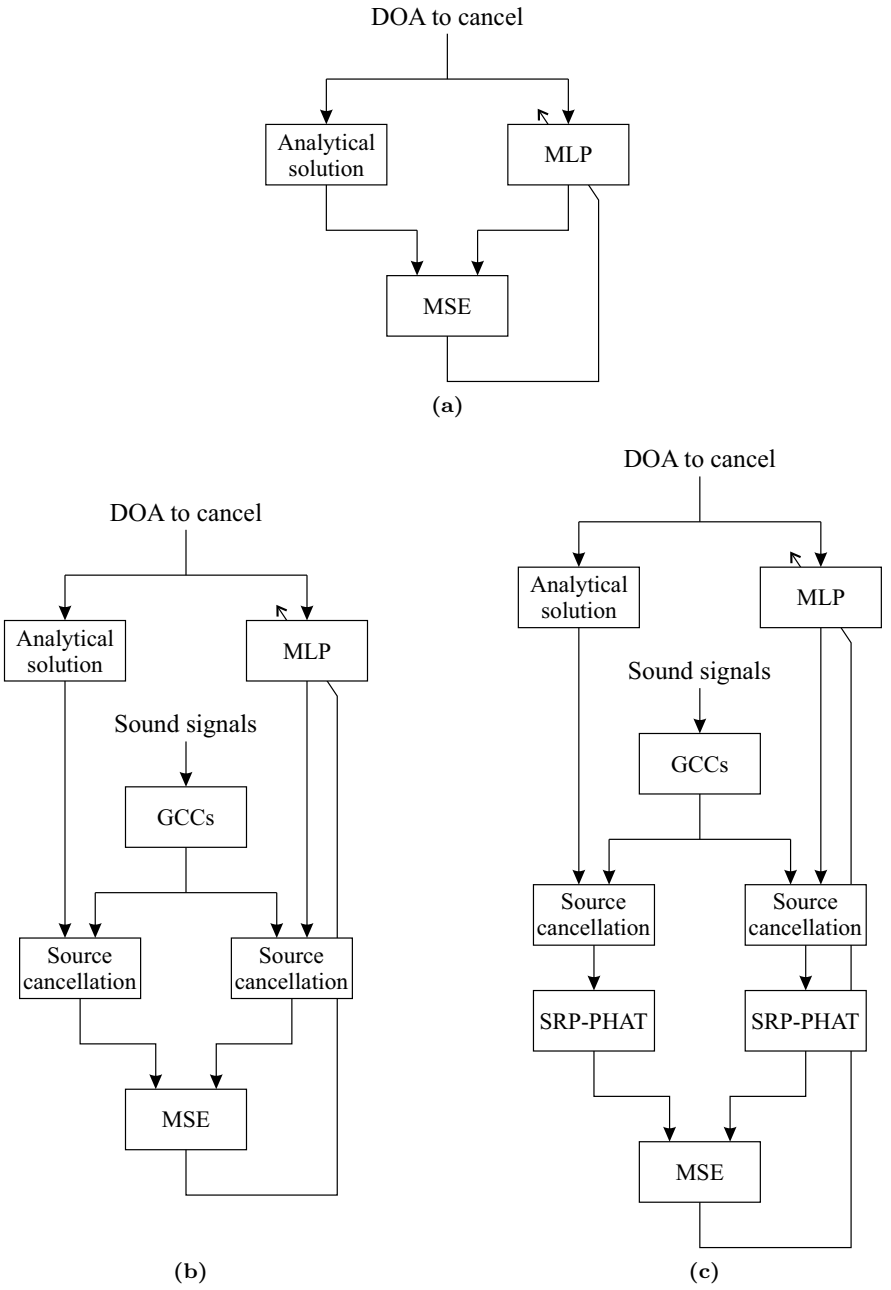


Figure 5.3: Experiments training an MLP to replicate the analytical coefficients and time shifts of the source cancellation technique presented in chapter 2.

5.3 Permutation invariant gated recurrent units (PI-GRUs)

Since tracking can be seen as a recursive process, using recurrent neural networks (RNNs) at the end of our models seems to be a natural option, and this is indeed what most of the state-of-the-art models do [14, 26, 38, 41, 42, 58, 62]. Conventional RNNs, such as the long short-term memory (LSTM) layers [175] or the gated recurrent units (GRUs) [176], use a $\mathbf{h}(t) \in \mathbb{R}^{d_h}$ vector to store the tracking state, which is updated at every time frame based on an input vector $\mathbf{x}(t) \in \mathbb{R}^{d_x}$ and, since these updates are usually computed using fully connected perceptrons, their computational complexity and their number of trainable parameters grow with the square of the length of $\mathbf{h}(t)$. When applied to multi-object tracking (MOT), the information corresponding to all the tracked objects is stored in $\mathbf{h}(t)$ without any predefined order or structure, which makes it hard to interpret and impose a compromise between the number of tracked objects, the amount of information that is stored for every one of them, and the computation complexity and the number of trainable parameters of the model. Furthermore, in addition to this compromise, if we stack the information of every object provided by the detector into a single vector, any change in the order of these objects can dramatically change the network output, so the network needs to learn the equivariance of the problem during the training.

In order to overcome the aforementioned issues of the conventional RNN for tracking applications, we propose, following the Geometric Deep Learning philosophy, the design of a new kind of recurrent architecture that shares the permutation invariances and equivariances of the tracking problem. As analyzed in section 5.1, if a tracking system is taking as inputs the estimates of a SSL system, we can see these estimates as an unordered set and therefore we should expect the tracking output to be the same regardless the order in which they are presented; i.e., the tracking system should be invariant to the permutation of its input estimates. On the other hand, if we want the identities assigned to every trajectory to be consistent during the time, once a source is randomly assigned to one of the tracking outputs, it should always be assigned to the same output but, in the case of having being assigned to a different output,

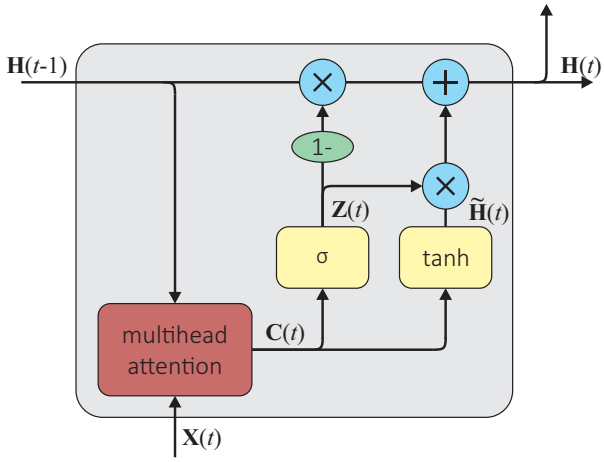


Figure 5.4: Architecture of the proposed permutation invariant gated recurrent unit (PI-GRU).

the result should be exactly the same except for the output where that trajectory is presented; i.e., the tracking system should be equivariant to the permutation of the tracked trajectories.

In contrast to these conventional RNNs, we propose replacing the input and state vectors $\mathbf{x}(t)$ and $\mathbf{h}(t)$ with sets of vectors $\mathbf{X}(t) = \{\mathbf{x}_1(t), \mathbf{x}_2(t), ..., \mathbf{x}_{M_x}(t)\}$ and $\mathbf{H}(t) = \{\mathbf{h}_1(t), \mathbf{h}_2(t), ..., \mathbf{h}_{M_h}(t)\}$ where every element $\mathbf{x}_i(t) \in \mathbb{R}^{d_x}$ and $\mathbf{h}_i(t) \in \mathbb{R}^{d_h}$ contains the information corresponding to only one input detection or one tracked trajectory respectively. Since the most efficient way to computationally represent data for deep-learning applications is in form of matrices, we can represent the input and state sets as two $M_x \times d_x$ and $M_h \times d_h$ matrices as far as we ensure that every row of $\mathbf{H}(t)$ is updated in a way that is invariant to the permutations of the rows of $\mathbf{X}(t)$ and equivariant to the permutations of the rows of $\mathbf{H}(t-1)$. For the sake of notation simplicity, we will consider $d_x = d_h = d$ in the remainder of this thesis, but the proposed architecture could easily be adapted to use different embedding sizes for the input and the state elements with just some minor modifications.

Interestingly, one of the most popular deep-learning modules nowadays already has these desired symmetries: the multi-head attention employed in transformer architectures [84]. The multi-head attention operation takes as input three matrices, typically

called queries, keys and values, and its output, in the case of using the same matrix for the key and the value inputs, is invariant to the permutations of the rows of the key and value matrix and equivariant to the permutations of the rows of the query matrix. In order to combine these properties with the gated structures that have been proved to offer good results in conventional RNNs, we propose the architecture presented in Fig. 5.4. This architecture is based on the conventional gated recurrent units (more precisely, on a simplified version of the Minimal GRU presented in [177]), so we call it permutation invariant gated recurrent unit (PI-GRU).

Instead of concatenating the input and the previous state as done in conventional GRUs, the PI-GRU uses multi-head attention [84] to obtain a new set $\mathbf{C}(t) = \{\mathbf{c}_1(t), \mathbf{c}_2(t), ..., \mathbf{c}_{N_h}(t)\}$ based on the input set and the previous state set:

$$\mathbf{C}(t) = \text{MultiHead}(\mathbf{H}(t-1),\ \mathbf{X}(t) \cup \mathbf{H}(t-1), \mathbf{X}(t) \cup \mathbf{H}(t-1)) \tag{5.1}$$

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\mathbf{head}_1, ..., \mathbf{head}_{N_{\text{heads}}}) \tag{5.2}$$

$$\mathbf{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V) \tag{5.3}$$

$$\text{Attention}(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i) = \text{soft-max}\left(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{d_k}}\right)\mathbf{V}_i \tag{5.4}$$

where $\mathbf{W}_i^Q \in \mathbb{R}^{d \times d_k}$, $\mathbf{W}_i^K \in \mathbb{R}^{d \times d_k}$, and $\mathbf{W}_i^Q \in \mathbb{R}^{d \times d_c}$ are learnable projection matrices and the sizes of the keys and values d_k and d_c are hyperparameters of the model. It is worth mentioning that (5.2) is typically defined including a linear output projection [84] but we do not include it here since we are using the output of the multi-head attention to directly feed fully connected perceptrons instead of a residual sum as is done in transformer architectures.

Finally, after obtaining the set $\mathbf{C}(t)$, we use two fully connected perceptrons (or gates) to update every element $\mathbf{h}_i(t)$ of $\mathbf{H}(t)$ depending only on $\mathbf{c}_i(t)$ and $\mathbf{h}_i(t-1)$:

$$\mathbf{h}_i(t) = [1 - \mathbf{z}_i(t)] \odot \mathbf{h}_i(t-1) + \tilde{\mathbf{h}}_i(t) \tag{5.5}$$

$$\mathbf{z}_i(t) = \sigma(\mathbf{c}_i(t)\mathbf{W}^z) \tag{5.6}$$

$$\tilde{\mathbf{h}}_i(t) = \tanh(\mathbf{c}_i(t)\mathbf{W}^h), \tag{5.7}$$

where $\mathbf{W}^z, \mathbf{W}^h \in \mathbb{R}^{N_{\text{heads}} d_c \times d}$ are learnable projection matrices, $\odot$ stands for point-wise multiplication, and $\sigma(\cdot)$ is the logistic sigmoid function.

We can interpret the role of the multi-head attention in the PI-GRU as reordering the information of the input set to make it match the order of the tracking trajectories in the state set. Actually, $\mathbf{Q}_i \mathbf{K}_i^T$ in (5.4) is the cross-correlation matrix between linear projections of the elements of $\mathbf{H}(t-1)$ (i.e., the information of every tracked trajectory) and linear projections of the elements of $\mathbf{X}(t) \cup \mathbf{H}(t-1)$ (i.e., the information of every new detection and every tracked trajectory) and therefore we can expect it to, after passing through the soft-max function, act as some kind of soft permutation matrix that, when applied over $\mathbf{V}_i$ (i.e., linear projections of $\mathbf{X}(t) \cup \mathbf{H}(t-1)$ again), generates a new set whose i-th element is built with the information of the elements of $\mathbf{X}(t) \cup \mathbf{H}(t-1)$ most related to the i-th element of $\mathbf{H}(t-1)$.

With this architecture, the computational complexity of the PI-GRU has a linear dependency with the number of tracked objects $M = M_h$ and new detections $M_{det} = M_x$ and the number of trainable parameters does not depend on them and, since every $\mathbb{R}^d$ vector in $\mathbf{X}(t)$ and $\mathbf{H}(t)$ only have to store information about one detection or one tracked trajectory, we would expect to need lower values of d_x and d_h than in conventional GRUs.

To the best of our knowledge, the PI-GRU is the first recurrent layer that works with unordered sets instead of with ordered vectors. The closest proposal in the literature is probably the TrackFormer [178], a model for multi-object tracking (MOT) on video signals that is based on the DETR transformer [179, 180], a model for object detection on images. DETR uses a set of learnable *object queries* as input for their transformer decoder that, through the multi-head attention layers of the decoder, become the object detections. In the TrackFormer, a recurrent loop is built around the decoder by using as *object queries* of every video frame the outputs of the decoder in the previous frame, so it can track the objects that have already been detected. Compared with the TrackFormer, the PI-GRU is not a model but a layer that can easily be integrated into many different models and that is based on an architecture, the conventional GRU, that, unlike the transformer, was designed to be used in recurrent loops. In addition, several PI-GRU layers can easily be stacked to build deeper

models with more recurrent loops, while the natural way of increasing the depth of the TrackFormer is increasing the number of layers of its transformer encoder and decoder, which does not increase the number of recurrences of the model.

5.4 Permutation invariant training (PIT)

Regardless we are taking into account the permutation invariance of the problem in the network architecture or not, we cannot ignore it during its training. When we want to train a multi-source localization or tracking model in a supervised manner and we cannot apply any criteria to classify and order the sources, we have to face the permutation invariance of the sources; i.e., we cannot directly compare the m-th trajectory estimated by the model $\hat{\mathbf{y}}_m(t)$ with the m-th trajectory of our ground-truth dataset $\mathbf{y}_m(t)$ since the model cannot infer the ground-truth order of the trajectories.

With the exception of [38], where a training strategy for MOT in computer vision tasks [181] is adapted to SSL, most multi-source localization models are trained using permutation invariant training (PIT). All the PIT strategies propose finding a permutation $\sigma : m \to \sigma_m,\ \forall m \in \{0, ..., M-1\}$ according to certain optimization criteria to reorder the outputs of the neural network and then use it to compare the estimated and ground-truth trajectories. When using activity-coupled Cartesian direction of arrival (ACCDOA) vectors to represent the DOA and the activity of the sources, we can use the mean squared error (MSE) as the loss function to train our models:

$$L_{PIT} = \frac{1}{TM} \sum_{t=0}^{T-1} \sum_{m=0}^{M-1} \|\mathbf{y}_m(t) - \hat{\mathbf{y}}_{\sigma_m(t)}(t)\|^2, \tag{5.8}$$

where M is the maximum number of trajectories that the model can estimate, T is the number of time frames in the scene, and $\|\cdot\|$ is the Euclidean norm operator. In the case of having a number of ground-truth trajectories lower than M, we can just add as many 0-norm padding trajectories as needed.

5.4.1 Frame-level Permutation Invariant Training (fPIT)

The original PIT was first proposed for training speech separation models [182] but, applied to SSL [13, 62], it proposes to find the permutation of the estimated sources

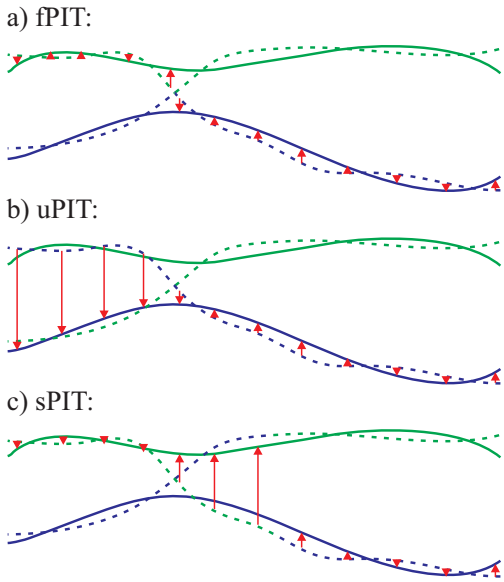


Figure 5.5: Examples of 1D trajectories and the result of applying the different PIT strategies. The dashed lines represent the estimates and their color how every PIT pairs them with the ground-truth trajectories (solid lines). The red arrows represent the gradients of the MSE of every pairing w.r.t. the first estimated trajectory.

that minimizes the matching error between the estimated and the ground-truth DOAs for every time frame:

$$\sigma^f(t) = \underset{\sigma \in \Pi_M}{\arg\min} \sum_{m=0}^{M-1} \|\mathbf{y}_m(t) - \hat{\mathbf{y}}_{\sigma_m}(t)\|, \tag{5.9}$$

where Π_M is the set of all the permutations $\sigma : i \rightarrow \sigma_i$ of M elements. To solve this optimization problem, we can compute the $M \times M$ distance matrices $\mathbf{D}(t)$ with elements $d_{ij}(t) = \|\mathbf{y}_i(t) - \hat{\mathbf{y}}_j(t)\|$ and apply the Hungarian algorithm [183] over them to find the optimal permutation at every time frame.

We call this approach frame-level permutation invariant training (fPIT) and, although it allows us to solve the permutation invariance problem, it does not penalize at all the IDSs. Instead, as can be seen in Fig. 5.5a, its gradients push the model to do these switches as fast as possible so the estimates are close to a ground-truth trajectory at all time frames. Therefore, if we want to keep the identity of every output stable during tracking, we need to add post-processing stages to fix the IDSs.

5.4.2 Utterance-level Permutation Invariant Training (uPIT)

In order to penalize the IDSs, utterance-level permutation invariant training (uPIT) [184] proposes finding the permutation that minimizes the error for a whole speech utterance or some other longer recording unit of interest, instead of a different one for every time frame:

$$\sigma^u = \underset{\sigma \in \Pi_M}{\arg\min} \sum_{t=0}^{T-1} \sum_{m=0}^{M-1} \|\mathbf{y}_m(t) - \hat{\mathbf{y}}_{\sigma_m}(t)\|, \tag{5.10}$$

where T is the number of time frames of the acoustic scene. In this case, we only need to apply the Hungarian algorithm once per acoustic scene after computing the time average of the matrices $\mathbf{D}(t)$.

Replacing $\sigma^f(t)$ by σ^u in (5.8) indeed penalizes the presence of IDSs since all the frames where the output ACCDOAs do not follow the main identity assignation compute as completely wrong estimates. However, as we can see in Fig. 5.5b, this penalization is excessive, being able to penalize situations that can not be solved by causal systems or generating gradients too much time after the IDS when, in most situations, it would be preferred to keep tracking the new identities rather than switching them again. Our experiments in using uPIT for multi-source tracking showed that it can easily generate a wide local minimum in the loss function that corresponds to estimating all DOAs in the middle of the active sources. That effect makes effective model training impossible, especially for long scenes of variable multiple sources.

5.4.3 Sliding Permutation Invariant Training (sPIT)

To overcome the limitations of both fPIT and uPIT, we propose a new PIT strategy that we call sliding permutation invariant training (sPIT) which consists in choosing, for every time frame, the optimal permutation for the last T_{avg} frames, i.e., for a causal sliding window of length T_{avg}:

$$\sigma^s(t) = \underset{\sigma \in \Pi_M}{\arg\min} \sum_{k=0}^{T_{avg}-1} \sum_{m=0}^{M-1} \|\mathbf{y}_m(t-k) - \hat{\mathbf{y}}_{\sigma_m}(t-k)\|. \tag{5.11}$$

In order to obtain $\sigma^s(t)$ for every time frame, we can follow the same procedure as in the fPIT but applying a causal moving average of length T_{avg} over the elements

of $\mathbf{D}(t)$ before computing the Hungarian algorithm, so the computational complexity is virtually the same. It is worth mentioning that, in the case of training non-causal trackers, we could replace the causal moving window in (5.11) with a centered window.

As shown in Fig. 5.5c, the sPIT penalizes an estimation if it does not follow the main source assignation of the last T_{avg} time frames, while it stops penalizing an IDS after a maximum of T_{avg} frames and focuses on maintaining the new identities. Hence, the global minimum of the loss function corresponds to a solution without any IDSs but it also prevents the training from converging to the useless local minima generated by uPIT, estimating all DOAs in the middle point of the active sources.

In addition, when used over ACCDOA vectors, if the number of estimated sources is lower than the actual number, one of the estimated ACCDOA vectors whose norm is lower than the detection threshold will be paired with the ground-truth ACCDOA vector of the missed source and the gradients of (5.8) will pull that estimated ACCDOA towards it. Similarly, in the case of a false positive, the gradients will pull the false-positive ACCDOA towards 0. Hence, sPIT is able to optimize both the source detections and the consistent source assignments that we expect from a competent SST system.

5.5 Evaluation

5.5.1 Experiment design

To evaluate the PI-GRUs and the PIT we used the dataset described in chapter 3 to generate acoustic scenes with up to 3 simultaneously active sources. During the 20 s of every scene, sources could appear and disappear, so the models do not only need to estimate the DOA of the sources but also their onsets and offsets.

We used the icoCNN described in chapter 4.2 as the basis of our tracking models. Since the output of this model was a 3D vector pointing towards the DOA of the source and whose norm was proportional to the confidence of the model in the estimation, we can already interpret it as an ACCDOA. We used the model for maps with resolution $r = 3$ and modified it by increasing the number of kernels of the hidden convolutional

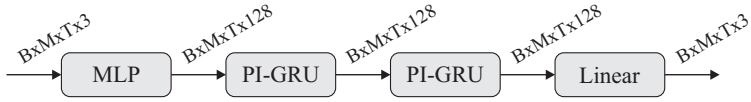


Figure 5.6: Architecture of the permutation invariant RNN used to evaluate the tracking capabilities of the PI-GRUs.

layers from 32 to 128 and the number of kernels of the last convolutional layer (and therefore the number of output ACCDOAs) from 1 to M_{det}.

After the convolutional part, and once generated M_{det} permutation invariant ACCDOAs, we used a small MLP to project every 3D ACCDOA into a space with d_x dimensions that we used to feed 2 PI-GRUs and then applied a final linear projection to convert every element of the output set into a 3D ACCDOA again as shown in Fig. 5.6. We used $M = M_{det} = 10$ as the number of ACCDOA outputs since we observed that it was beneficial to use a number higher than the maximum possible number of active sources in the dataset (i.e., 3) and $d_x = d_k = N_{heads} d_c = 128$ and $N_{heads} = 4$. With these hyperparameters, the permutation invariant RNN had 236 035 trainable parameters.

As baseline models, we studied the performance of the same icoCNN model without any RNN after it (so the tracking was done using only the temporal convolutions included in the icoCNN) and we also trained a model replacing the PI-GRUs of our permutation invariant RNN (Fig. 5.6) by conventional GRUs (Fig. 5.7). In order to feed the conventional GRUs with the set generated by the MLP, we needed to concatenate the M elements of the set into only one dimension and, to keep the number of trainable parameters in a reasonable range, we needed to reduce the size of the output of the GRUs. We used $d_h = 10 \cdot M = 100$ as output size and, even if it meant having a state vector smaller than the one used to represent just a single trajectory with the PI-GRUs, it resulted in an RNN with 509 153 trainable parameters, more than the double than with the PI-GRUs. Here, we can clearly see the compromise explained in section 5.3 between the model complexity and the amount of information that it can keep about every source and it would become even more extreme for higher values of M.

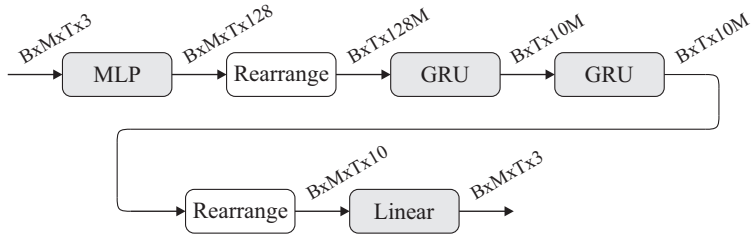


Figure 5.7: Architecture of the RNN used after the icoCNN as baseline in the evaluation.

We trained the three models using both conventional fPIT and the proposed sPIT, we did not include uPIT in the evaluation since it did not converge to any practical solution. We found that the model including the PI-GRUs was especially hard to train and in many cases it did not converge. Following some usual practices when training recurrent models, we used the AdamW algorithm [185] with gradient clipping [186] and reinitialized the state vectors to a learnable initial state every 25 time frames during the first 50 epochs and then every 50 frames during the following 100 epochs. For a fair comparison, we applied the same strategy when training the model with the conventional GRUs and, to facilitate the training of the convolutional part of the models, we also included a fPIT loss over the ACCDOAs generated by the icoCNN. All the results shown in the following section were obtained using $T_{avg} = 10$ frames (i.e., 2 s) for the sPIT, but no large changes were observed with longer or shorter windows.

5.5.2 Results

Fig. 5.8 shows the mean angular error (MAE), the identity switch (IDS) rate, and the receiver operating characteristic (ROC) and detection error tradeoff (DET) curves resulting from training each one of the three evaluated models. The MAEs and the IDS, true positive, and false positive rates were calculated following a procedure similar to the one defined by the CLEAR MOT metrics [187] typically used for multi-object tracking in computer vision. In every time frame, each estimate was associated with its closer ground-truth DOA and any estimate that could not be associated with a ground-truth DOA closer to a threshold (which we set at 30° as done in the evaluation of the results of the LOCATA challenge [188]) was considered a false positive. After

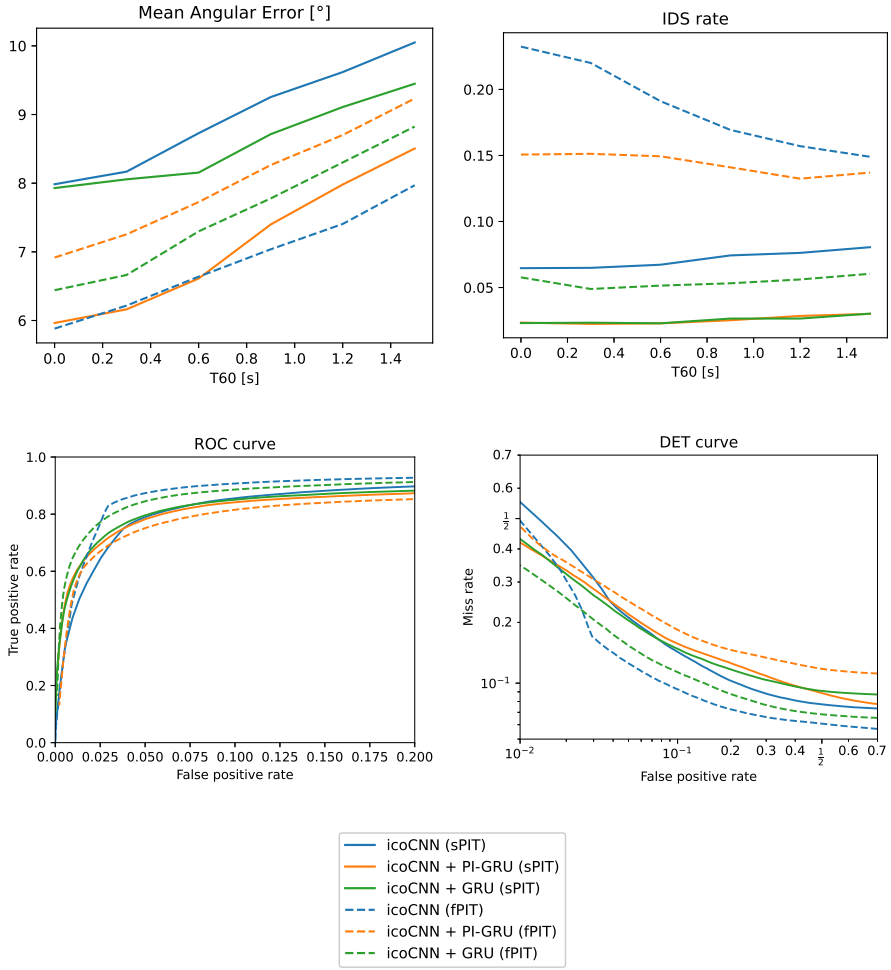


Figure 5.8: Evaluation metrics obtained using sPIT and fPIT to train the three analyzed models.

that, in every ground-truth trajectory that had already been assigned to an estimated trajectory in the previous time frame, an IDS was considered to have occurred if now it had been assigned to a different estimated trajectory. Finally, the MAE was defined as the sum of the angular distances of the true positives during the whole dataset divided by the total number of true positives and the IDS, true positive, and false positive rates, as the number of IDSs, true positives, and false positives in the whole dataset divided by the total number of ground-truth DOAs. The MAE and the IDS rate were computed considering a detection threshold of 0.5 over the norm of the estimated ACCDOA vectors and for computing the ROC curve we analyzed the true positive and false positive rates for a range of detection thresholds from 0 to 1.

As could be expected, the metric that is most affected by the PIT strategy is the IDS rate. We can see how using the proposed sPIT we can train purely convolutional models to obtain rates under 0.1 and how it reduces the number of IDSs of the model using conventional GRUs by more than a factor 2. The model using the proposed PI-GRUs had a quite high number of IDSs when trained with the fPIT, but when trained with the proposed sPIT the number was reduced to the same level as the model using conventional GRUs. Since the fPIT does not penalize the IDSs, the IDS rate of the models trained with it depends only on their ability to generate abrupt changes in their outputs because the switches are penalized only if they occur slowly and generate a high localization error till they are completed.

In terms of localization error, all the evaluated models have similar performance, with a difference of only 2° between the best and the worst MAE. For the model doing the tracking using only temporal convolutions and for the one using conventional GRUs the use of sPIT slightly degrades the localization accuracy but, in the case of the model using PI-GRUs, the sPIT improves its localization performance.

Finally, we can see in the ROC and DET curves how the compromise between the true and false positive rates for the models trained with sPIT are quite similar, especially in the models using recurrent layers for tracking. Again, we can see how sPIT slightly degrades the performance of the model using only convolutions and the model using GRUs but improves the results of the model using PI-GRUs.

As examples, figures 5.9, 5.10, and 5.11 show trajectories estimated by the evaluated models in acoustic scenes with up to 3 concurrent sources. We can see how the models trained with sPIT do not have the high number of IDSs that the models trained with fPIT have, though the model without recurrent layers still has some switches even when trained with sPIT. We can also see how the models using recurrent layers for tracking clearly outperform the model using only temporal convolutions but there are no clear differences between the model using conventional GRUs and the model using the proposed PI-GRUs. Finally, it is worth mentioning that any of the models seem to really be able to track three concurrent sources.

5.6 Conclusions

In this chapter, we have studied the permutation invariance of the multi-source tracking systems and proposed a new permutation invariant training (PIT) strategy to train tracking models and a new recurrent layer that takes an unordered set as input and that is invariant to the permutation of its elements.

We have proven how the proposed sPIT dramatically reduces the number of identity switches (IDSs) compared with the state-of-the-art fPIT, avoiding the need for using additional stages at the output of the tracking models to keep the identity of every tracked trajectory stable. The loss function of sPIT and the gradients that it generates are easy to interpret and it has a similar computational complexity to fPIT. When used over ACCDOAs, sPIT also optimizes the detection accuracy of the model, improving the ratio between the true and false positive rates and therefore optimizing all the features that we would expect from a sound source tracking (SST) system.

About the proposed permutation invariant gated recurrent unit (PI-GRU), even if we can argue that they outperformed the conventional GRUs since the models that used them obtained the same IDS rate and ROC curve as the model with the conventional GRUs and had a lower mean angular error (MAE), this difference probably does not worth the increase in the computational complexity. However, we think that the results obtained are promising since they represent the first attempt at developing

a recurrent layer for unordered sets and there is still room for improving and optimizing them. In the evaluated models, only geometrical information was provided to the recurrent layers to perform the tracking, but using models that include spectral information of every source would better exploit their ability to increase the size of the vectors representing every detection and tracked trajectory without needing to increase the number of trainable parameters in a quadratic way as happens with the conventional GRUs. Some ideas about how to improve both the PI-GRUs and the whole model are discussed in chapter 6.

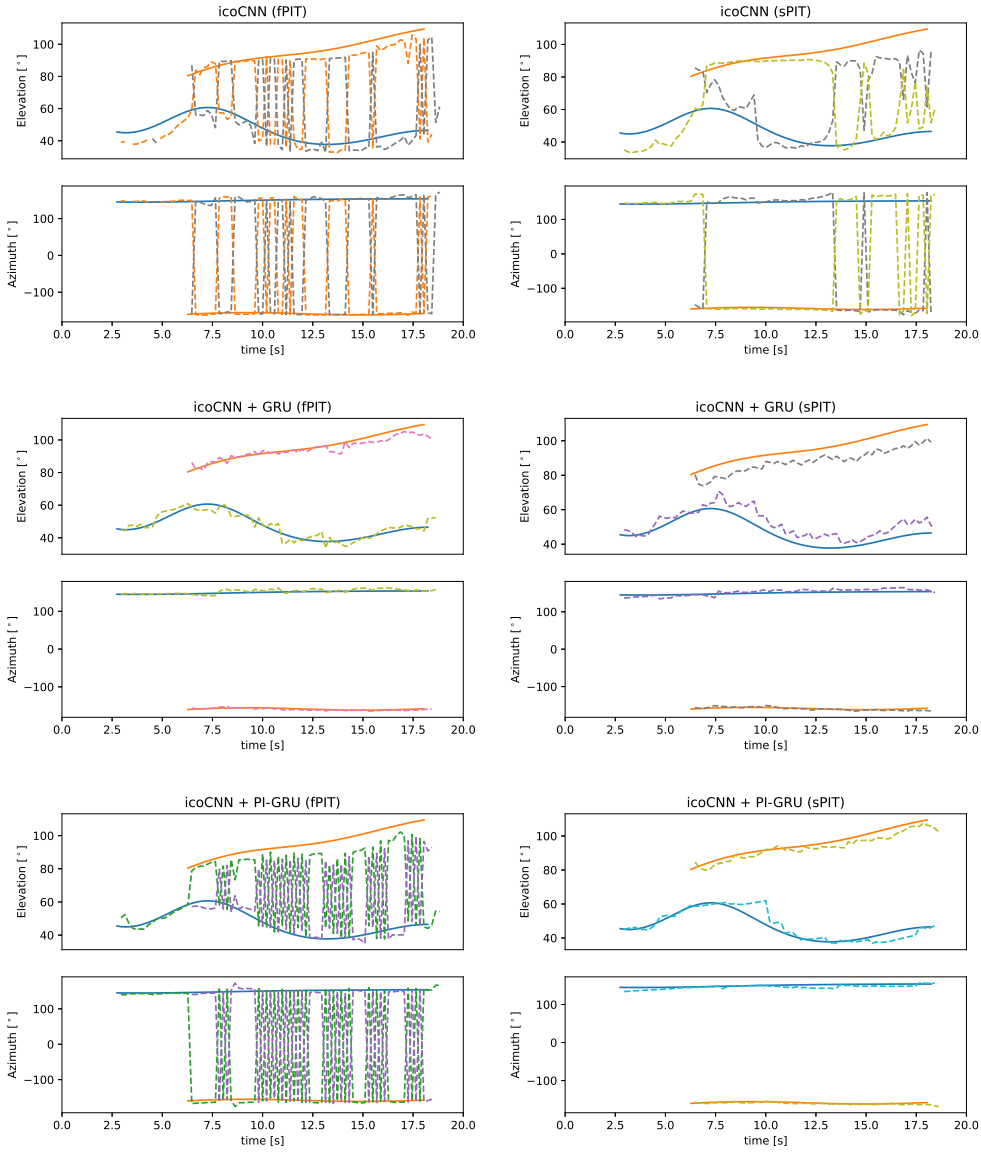


Figure 5.9: Example of the trajectories estimated by the evaluated models in an evaluation acoustic with up to 2 concurrent sound sources. The solid lines represent the ground-truth trajectories and the dashed lines the estimates.

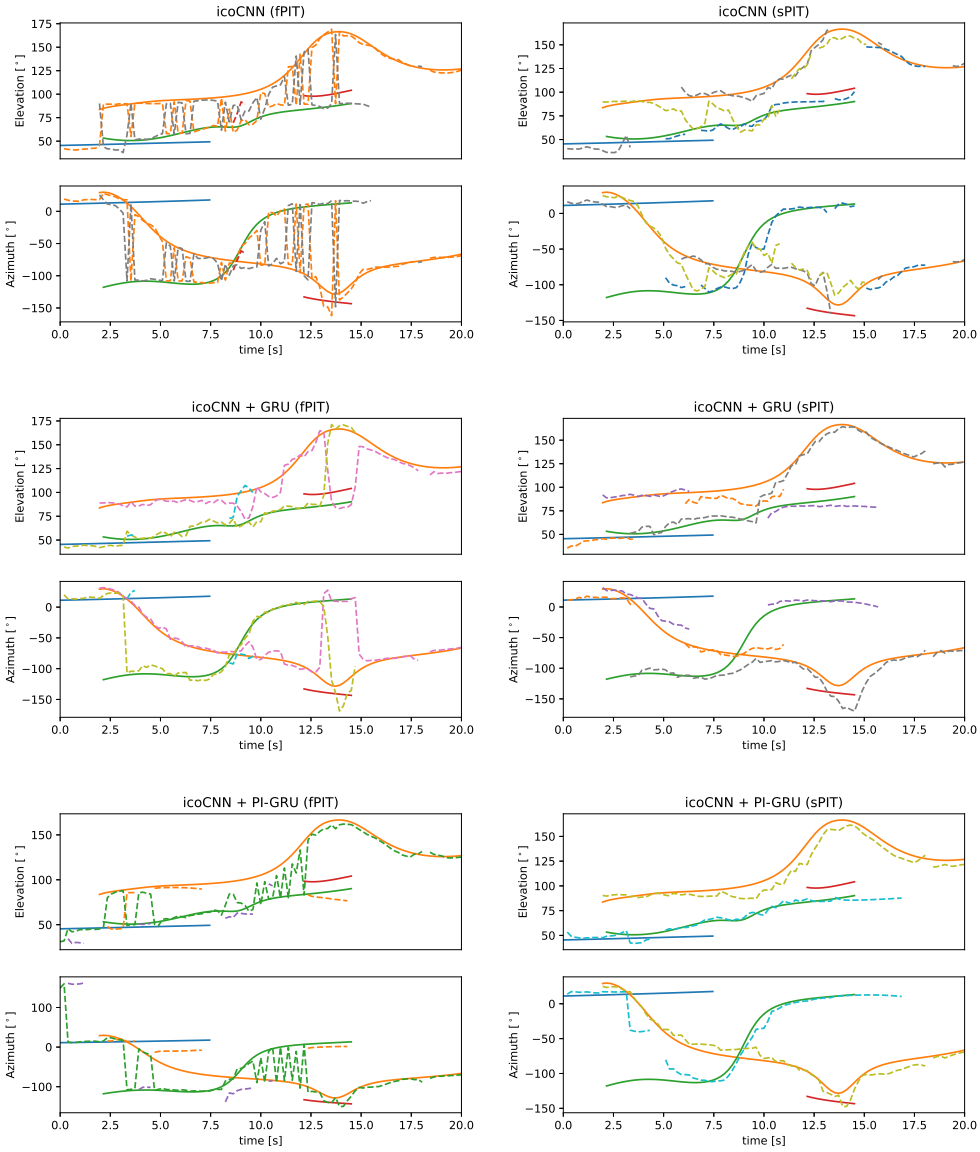


Figure 5.10: Example of the trajectories estimated by the evaluated models in an evaluation acoustic with up to 3 concurrent sound sources. The solid lines represent the ground-truth trajectories and the dashed lines the estimates.

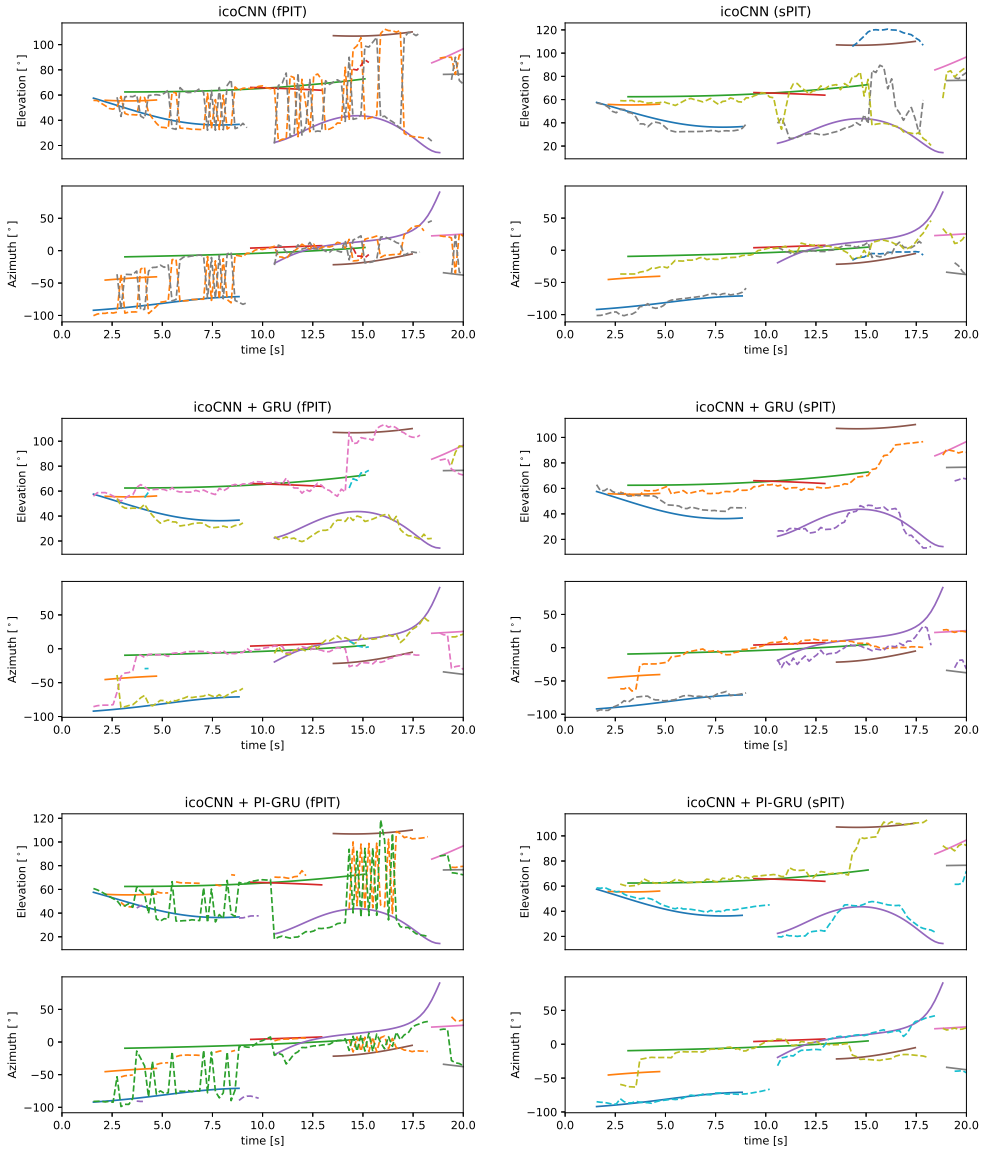


Figure 5.11: Another example of the trajectories estimated by the evaluated models in an evaluation acoustic with up to 3 concurrent sound sources. The solid lines represent the ground-truth trajectories and the dashed lines the estimates.

6

Conclusions and future work

In this chapter, we first summarize the conclusions of the thesis (section 6.1) and then we outline some of the research directions that we think have a greater potential to improve the performance the sound source localization (SSL) and sound source tracking (SST) systems based on deep learning in the near future (section 6.2).

6.1 Conclusions

- The signal processing community has studied the sound source localization (SSL) problem for decades and the classical techniques that were proposed before the emergence of the deep-learning solutions should not be neglected since they can provide acoustic representations with a higher correlation with the direction of arrival (DOA) of the sources and a better analytical understanding of the problem that should guide the design of the new deep-learning solutions.

- We can eliminate the effects that a source in a given position generates in the inter-microphone generalized cross-correlations (GCCs) by just doing a linear combination of time-shifted versions of the original GCC.

- When using synthetic datasets to train SSL models, we can generate unlimited different random trajectories and therefore we do not need to limit our datasets to a finite number of trajectories. Although this should be used with caution,

with the models employed in this work we have not observed signs of over-fitting to the trajectory generation procedure even if it was analytically quite simple.

- Many of the stages of the image source method (ISM) for room acoustic simulation can be parallelized and efficiently implemented in graphics processing units (GPUs). This is essential if we want to train our models with infinite-size datasets simulated on the fly, since the ISM is computationally expensive and its sequential implementation would slow down the trainings too much. The proposed GPU implementation reduces the simulation time by two orders of magnitude compared with other state-of-the-art implementations.

- The acoustic power maps generated with the classic steered response power with phase transform (SRP-PHAT) algorithm are a robust input representation for neural networks. Though their main source of information for one-source localization is the position of their maximum, neural networks are able to extract additional information from them.

- For one-source tracking, it is possible to obtain competitive results using only causal temporal convolutions avoiding the use of recurrent layers. Bi-directional recurrent layers should be avoided if we want our models to be useful for real-time applications.

- Using models that exploit the rotational invariance of the DOA estimation problem allows us to reduce the size of our models. Several architectures are invariant to the continuous space of spherical rotations, but the icosahedral convolutional neural networks (CNNs) provide a good approximation with their 60 rotational symmetries while having an efficient implementation based on conventional 2D convolutional layers.

- The proposed soft-argmax layer can transform a classification output into a regression one by interpreting the classification output as the probability distribution of the DOA and computing its expected value. It strongly reduces the number of trainable parameters of the model and avoids breaking its rotational equivariance.

- In permutation invariant training (PIT) strategies for multi-source tracking, the temporal context that we consider when choosing the best permutation of the estimated sources is crucial to reduce the number of identity switches (IDSs) in the tracked trajectories. Using a sliding window as we have proposed with the sliding permutation invariant training (sPIT) is a good trade-off between the high number of IDSs obtained with the frame-level permutation invariant training (fPIT) and the convergence issues presented by the utterance-level permutation invariant training (uPIT) for long acoustic scenes.

- We can design permutation invariant recurrent layers for deep learning models that operate over unordered sets instead of over ordered vectors by using a multi-head attention module to match the elements of the input set with the elements of the state set and then using element-wise gates to update the information of every state element.

- The use of permutation invariant recurrent layers, such as the proposed permutation invariant gated recurrent unit (PI-GRU), for SST of multiple sources is a promising research direction, but further work is needed to make them competitive with the traditional gated recurrent units (GRUs).

6.2 Future work

6.2.1 Training data

In section 4.1.4, we saw how all the evaluated models had a worse performance when evaluated in the LOCATA dataset than when evaluated with the synthetic dataset used to train them and the performance gap became ever wider in the case of the model proposed in 4.2 using icosahedral CNNs. The optimal solution would be training the models using actual recordings, but there are no multichannel-audio datasets with ground-truth position labels large enough to do this. Contrary to image datasets, position labels in audio datasets cannot be obtained after the signals are recorded and special hardware is needed to accurately label the source position of moving sources [138]. Therefore, it does not seem to be reasonable to expect the appearance of larger

position-labeled datasets in the near future and techniques to improve the performance of models trained with synthetic signals should be found.

There exist some datasets with recorded multichannel room impulse responses (RIRs) and position labels (e.g., [189, 190, 191, 192]) that could be used to replace the simulated RIRs and get more realistic synthetic training datasets, but they are usually not large enough, especially in terms of acoustic conditions. Another approach to improve the quality of the synthetic datasets would be improving RIRs simulated with the ISM by including more acoustic effects [193] such as source directivity [194] or acoustic diffraction on the walls or the array [195].

Finally, an approach that is in growing popularity is reducing the need for position labels by using semi-supervised or weakly-supervised training strategies. For example, some domain adaptation techniques have been proposed where the models are trained with synthetic signals following a supervised approach but then actual recordings without position labels are used to reduce the performance gap between the synthetic and the recorded signals [46, 64, 65, 196]. A different approach is followed in [49, 50], where a large dataset of unlabeled multichannel recordings is used to train an autoencoder to reconstruct the phase of the relative transfer functions between microphones from two latent variables while a smaller dataset with labeled recordings is used to train one of those variables to be the DOA of the acoustic source.

6.2.2 Rotation-equivariant models for SSL

In this thesis, we have proposed an SSL model based on applying an icosahedral CNN over SRP-PHAT power maps. However, other approaches could also be followed to design rotation-equivariant models. There are several signal processing techniques, such as the multiple signal classification (MUSIC) algorithm, that can be used to extract positional information of multichannel signals that can be computed on arbitrary grids and that, therefore, could be used to replace the SRP-PHAT power maps as model inputs or to complement them by stacking them as additional input channels. There are also other network architectures designed to be equivariant to the spherical rotations, such as those based on the spherical-harmonic domain [77, 168, 169, 170] or

on graphs [171, 172, 173], so a study comparing the performance of different input representations and network architectures would be interesting.

A more radically different approach would be finding other kinds of acoustic representations where it is possible to exploit the rotational equivariance. For example, a rotation-equivariant network is proposed in [197] that works over the ambisonic representation of the audio signals.

6.2.3 Permutation invariant recurrent networks

In section 5.3, we introduced the PI-GRU, which is, to the best of our knowledge, the first recurrent neural layer whose input and recurrent state are unordered sets and which is invariant to the permutations of the elements of the input set and equivariant to the permutations of the state set. We designed the architecture of the PI-GRU based on a simplified version of the minimal GRU [177] but other architectures, such as the ones shown in Fig. 6.1, could also be interesting. A further evaluation of permutation invariant recurrent architectures would be interesting in order to analyze the advantages and disadvantages of every one of them.

Actually, even with the proposed PI-GRU architecture, we did not conduct any systematic analysis of how its hyperparameters (such as the number of heads or the embedding sizes) affected the tracking performance of the model, so this would also need further study in the future.

We also found that the PI-GRUs were considerably harder to train than the conventional GRUs, exhibiting a slower convergence and even not converging to any solution during some trainings. Therefore, it would be also interesting to study which optimization algorithms, regularization techniques, and training strategies could facilitate the training of these models.

Finally, it is worth mentioning that, during the experiments conducted in section 5.5, the recurrent state of the PI-GRUs (and also of the baseline GRUs) was initialized to a learnable set of embedding (or to a learnable vector) at the beginning of each training or evaluation acoustic scene and we let it to evolve during the scene independently of whether new sources were found or not. A different approach that would be worth studying would be, as done with the TrackFormer [178] for visual

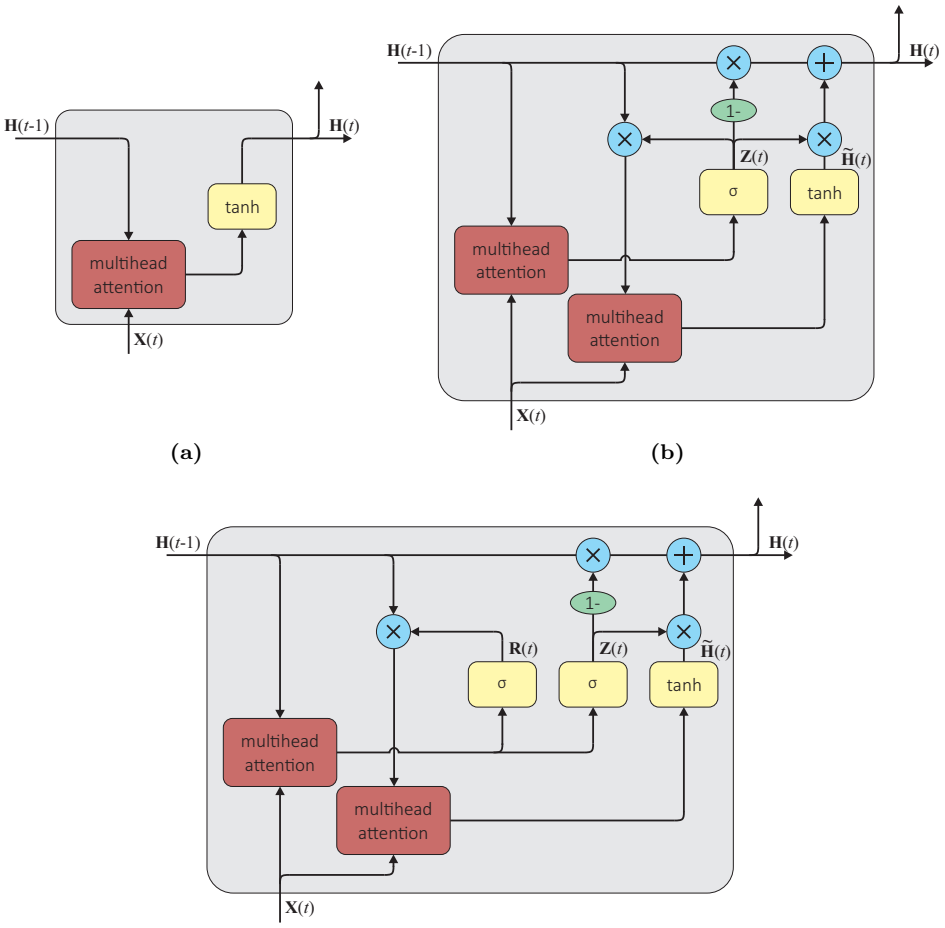


Figure 6.1: Different permutation invariant recurrent architectures. (a) is just a minimal recurrent unit, (b) is based on the minimal GRU and (c) is based on a complete GRU.

multi-object tracking (MOT) on videos, keeping always the learned embeddings in the recurrent state set and, after every time frame, adding to it the elements of the output that led to active sources in the previous frame. If the new elements are added to the state set without removing the elements that had generated them as done in [178], this would mean that the size of the state set varies over time, but this is compatible with the PI-GRU architecture presented in section 5.3 and also with the alternative architectures shown in Fig. 6.1.

6.2.4 Eliminating the information bottlenecks of the proposed model

In section 4.1, we argued that combining localization and tracking into a single stage by using 3D convolutions that extended their receptive field over the temporal dimension was preferable to first doing the localization using the acoustic information of just one time frame and then filtering this result with a tracking algorithm considering the localization results of the previous frames. However, our models also present some information bottlenecks where we could also be losing potentially useful information.

The first information bottleneck of our model happens at its very beginning: at the input feature selection. By using SRP-PHAT maps as the only input of our model, we are losing all the spectral information of the signals. This could be reasonable for single-source localization since we wanted to locate the only directional source present in the signals regardless of its spectral content, but for multi-source tracking, the spectral information provides crucial information to, for example, avoid IDSs.

One of the easiest ways to integrate spectral information into the model proposed in chapter 5 would be using classic beamformers to obtain the acoustic spectra at every DOA estimated by the icosahedral CNN and then concatenate it to the corresponding ACCDOA so every element in the input set of the PI-GRU contains both spatial and spectral information. This way, the correlation matrix computed in the multi-head attention module would match the elements of the input and state sets not only based on geometrical information but also based on the similarities of their spectrum.

A different approach would be replacing the input power maps with another icosahedral signal that, at every point of the grid, included spectral information obtained

using a beamformer steered at the corresponding direction. Considering every frequency bin as a convolution channel, the same convolutional architecture could be easily used, though we would probably need to increase the number of channels in the hidden layers to fully exploit the spectral information. However, feeding the PI-GRU with the spectral information processed by the icosahedral CNN would be more difficult, since every element of the input set should contain information from only one of the detected sources. An option could be increasing the number of channels of the last convolutional layer from M to $M + S$, so the first M channels could still be processed with the soft-argmax layer to obtain the ACCDOAs of the detected sources and filter each one of the remaining S channels with every one of the first M channels to finally integrate the $M \times S$ resulting maps over their spatial dimensions so we would obtain M vectors of length S and concatenate every one of them to the corresponding ACCDOA.

References

[1] Richard W. Carlisle and Arnold Schwartz. **Evaluation of a Stereophonic Loudspeaker by Multiple Microphone Arrays**. *The Journal of the Acoustical Society of America*, **31**(10):1348–1351, October 1959. 1

[2] Michael E. Austin and Richard B. Gomez. **Azimuth and Elevation Errors Inherent in Sound-Ranging Calculations**. *The Journal of the Acoustical Society of America*, **44**(1):25–27, July 1968. 1

[3] Michael Brandstein and Darren Ward. *Microphone Arrays: Signal Processing Techniques and Applications*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. 1

[4] Pierre-Amaury Grumiaux, Srđan Kitić, Laurent Girin, and Alexandre Guérin. **A Survey of Sound Source Localization with Deep Learning Methods**. *The Journal of the Acoustical Society of America*, **152**(1):107–151, July 2022. 3

[5] Dmitry Suvorov, Ge Dong, and Roman Zhukov. **Deep Residual Network for Sound Source Localization in the Time Domain**, August 2018. 3

[6] Juan Manuel Vera-Diaz, Daniel Pizarro, and Javier Macias-Guarasa. **Towards End-to-End Acoustic Localization Using Deep Learning: From Audio Signals to Source Position Coordinates**. *Sensors*, **18**(10):3418, October 2018. 3

[7] Hadrien Pujol, Éric Bavu, and Alexandre Garcia. **BeamLearning: An End-to-End Deep Learning Approach for the Angular Localization of Sound Sources Using Raw Multichannel Acoustic Pressure Data**. *The Journal of the Acoustical Society of America*, **149**(6):4248–4263, June 2021. 3

[8] Pasi Pertilä and Emre Cakir. **Robust Direction Estimation with Convolutional Neural Networks Based Steered Response Power**. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6125–6129, March 2017. 4, 6

[9] Nelson Yalta, Kazuhiro Nakadai, and Tetsuya Ogata. **Sound Source Localization Using Deep Learning Models**. *Journal of Robotics and Mechatronics*, **29**(1):37–48, 2017. 4

[10] Zhong-Qiu Wang, Xueliang Zhang, and DeLiang Wang. **Robust Speaker Localization Guided by Deep Learning-Based Time-Frequency Masking**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **27**(1):178–188, January 2019. 4

[11] Wangyou Zhang, Ying Zhou, and Yanmin Qian. **Robust DOA Estimation Based on Convolutional Neural Network and Time-Frequency Masking**. In *Interspeech 2019*, pages 2703–2707. ISCA, September 2019. 4

[12] S. Chakrabarty and E. A. P. Habets. **Multi-Speaker DOA Estimation Using Deep Convolutional Networks Trained With Noise Signals**. *IEEE Journal of Selected Topics in Signal Processing*, **13**(1):8–21, March 2019. 4, 9

[13] Aswin Shanmugam Subramanian, Chao Weng, Shinji Watanabe, Meng Yu, and Dong Yu. **Deep Learning Based Multi-Source Localization with Source Splitting and Its Effectiveness in Multi-Talker Speech Recognition**. *Computer Speech & Language*, **75**:101360, September 2022. 4, 6, 10, 121

[14] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen. **Sound Event Localization and Detection of Overlapping Sources Using Convolutional Recurrent Neural Networks**. *IEEE Journal of Selected Topics in Signal Processing*, **13**(1):34–48, March 2019. 4, 6, 84, 86, 117

[15] Karim Guirguis, Christoph Schorn, Andre Guntoro, Sherif Abdulatif, and Bin Yang. **SELD-TCN: Sound Event Localization & Detection via Temporal Convolutional Networks**. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 16–20, Amsterdam, Netherlands, January 2021. IEEE. 4

[16] Weipeng He, Petr Motlicek, and Jean-Marc Odobez. **Neural Network Adaptation and Data Augmentation for Multi-Speaker Direction-of-Arrival Estimation**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **29**:1303–1317, 2021. 4, 6

[17] Christopher Schymura, Benedikt Bönninghoff, Tsubasa Ochiai, Marc Delcroix, Keisuke Kinoshita, Tomohiro Nakatani, Shoko Araki, and Dorothea Kolossa. **PILOT: Introducing Transformers for Probabilistic Sound Event Localization**. In *Interspeech 2021*, pages 2117–2121. ISCA, August 2021. 4

[18] Weipeng He, Petr Motlicek, and Jean-Marc Odobez. **Joint Localization and Classification of Multiple Sound Sources Using a Multi-task Neural Network**. In *Interspeech 2018*, pages 312–316. ISCA, September 2018. 4

[19] Guillaume Le Moing, Phongtharin Vinayavekhin, Tadanobu Inoue, Jayakorn Vongkulbhisal, Asim Munawar, Ryuki Tachibana, and Don Joven Agravante. **Learning Multiple Sound Source 2D Localization**. In *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6, September 2019. 4

[20] Yiya Hao, Abdullah Küçük, Anshuman Ganguly, and Issa M. S. Panahi. **Spectral Flux-Based Convolutional Neural Network Architecture for Speech Source Localization and Its Real-Time Implementation**. *IEEE Access*, **8**:197047–197058, 2020. 4

[21] Daniel Krause, Archontis Politis, and Konrad Kowalczyk. **Comparison of Convolution Types in CNN-based Feature Extraction for Sound Source Localization**. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 820–824, Amsterdam, Netherlands, January 2021. IEEE. 4

[22] X. Xiao, S. Zhao, X. Zhong, D. L. Jones, E. S. Chng, and H. Li. **A Learning-Based Approach to Direction of Arrival Estimation in Noisy and Reverberant Environments**. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2814–2818, April 2015. 4, 5

[23] F. Vesperini, P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza. **A Neural Network Based Algorithm for Speaker Localization in a Multi-Room Environment**. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, September 2016. 4

[24] Luca Comanducci, Federico Borra, Paolo Bestagini, Fabio Antonacci, Stefano Tubaro, and Augusto Sarti. **Source Localization Using Distributed Microphones in Reverberant Environments Based on Deep Learning and Ray Space Transform**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **28**:2238–2251, 2020. 4

[25] Qinglong Li, Xueliang Zhang, and Hao Li. **Online Direction of Arrival Estimation Based on Deep Learning**. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2616–2620, April 2018. 4

[26] Lauréline Perotin, Romain Serizel, Emmanuel Vincent, and Alexandre Guérin. **CRNN-Based Multiple DoA Estimation Using Acoustic Intensity Features for Ambisonics Recordings**. *IEEE Journal of Selected Topics in Signal Processing*, **13**(1):22–33, March 2019. 4, 117

[27] Pierre-Amaury Grumiaux, Srdan Kitić, Prerak Srivastava, Laurent Girin, and Alexandre Guérin. **Saladnet: Self-Attentive Multisource Localization in the Ambisonics Domain**. In *2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 336–340, October 2021. 4

[28] Thi Ngoc Tho Nguyen, Ngoc Khanh Nguyen, Huy Phan, Lam Pham, Kenneth Ooi, Douglas L. Jones, and Woon-Seng Gan. **A General Network Architecture for Sound Event Localization and Detection Using Transfer Learning and Recurrent Neural Network**. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 935–939, June 2021. 4

[29] R. Schmidt. **Multiple Emitter Location and Signal Parameter Estimation**. *IEEE Transactions on Antennas and Propagation*, **34**(3):276–280, March 1986. 4, 16

[30] R. Roy and T. Kailath. **ESPRIT-estimation of Signal Parameters via Rotational Invariance Techniques**. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **37**(7):984–995, July 1989. 4, 16

[31] Joseph Hector DiBiase. *A High-Accuracy, Low-Latency Technique for Talker Localization in Reverberant Environments Using Microphone Arrays*. PhD thesis, Brown University, 2000. 4, 16, 19

[32] Joseph Hector DiBiase, Harvey F. Silverman, and Michael Brandstein. **Robust Localization in Reverberant Rooms**. In *Microphone Arrays: Signal Processing Techniques and Applications*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. 4, 15, 16, 19

[33] R. Takeda and K. Komatani. **Discriminative Multiple Sound Source Localization Based on Deep Neural Networks Using Independent Location Model**. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 603–609, December 2016. 4

[34] Thi Ngoc Tho Nguyen, Woon-Seng Gan, Rishabh Ranjan, and Douglas L. Jones. **Robust Source Counting and DOA Estimation Using Spatial Pseudo-Spectrum and Convolutional Neural Network**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **28**:2626–2637, 2020. 4

[35] Daniele Salvati, Carlo Drioli, and Gian Luca Foresti. **Exploiting CNNs for Improving Acoustic Source Localization in Noisy and Reverberant Conditions**. *IEEE Transactions on Emerging Topics in Computational Intelligence*, **2**(2):103–116, April 2018. 4

[36] E. L. Ferguson, S. B. Williams, and C. T. Jin. **Sound Source Localization in a Multipath Environment Using Convolutional Neural Networks**. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2386–2390, April 2018. 4

[37] Paolo Vecchiotti, Emanuele Principi, Stefano Squartini, and Francesco Piazza. **Deep Neural Networks for Joint Voice Activity Detection and Speaker Localization**. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 1567–1571, September 2018. 4

[38] Sharath Adavanne, Archontis Politis, and Tuomas Virtanen. **Differentiable Tracking-Based Training of Deep Learning Sound Source Localizers**. In *2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 211–215, New Paltz, NY, USA, October 2021. IEEE. 4, 6, 117, 121

[39] Daniel Krause, Archontis Politis, and Konrad Kowalczyk. **Feature Overview for Joint Modeling of Sound Event Detection and Localization Using a Microphone Array**. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 31–35, Amsterdam, Netherlands, January 2021. IEEE. 4, 9

[40] Luca Comanducci, Maximo Cobos, Fabio Antonacci, and Augusto Sarti. **Time Difference of Arrival Estimation from Frequency-Sliding Generalized Cross-Correlations Using Convolutional Neural Networks**. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4945–4949, May 2020. 4

[41] Yin Cao, Qiuqiang Kong, Turab Iqbal, Fengyan An, Wenwu Wang, and Mark Plumbley. **Polyphonic Sound Event Detection and Localization Using a Two-Stage Strategy**. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 30–34, New York University, NY, USA, October 2019. 4, 117

[42] Pranay Pratik, Wen Jie Jee, Srikanth Nagisetty, Rohith Mars, and Chongsoon Lim. **Sound Event Localization and Detection Using CRNN Architecture with Mixup for Model Generalization**. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 199–203. New York University, 2019. 4, 117

[43] Huy Phan, L. Pham, P. Koch, Ngoc Q. K. Duong, I. Mcloughlin, and A. Mertins. **On Multitask Loss Function for Audio Event Detection and Localization**. *undefined*, 2020. 5

[44] Wolfgang Mack, Ullas Bharadwaj, Soumitro Chakrabarty, and Emanuël A. P. Habets. **Signal-Aware Broadband DOA Estimation Using Attention Mechanisms**. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4930–4934, May 2020. 5

[45] Christopher Schymura, Tsubasa Ochiai, Marc Delcroix, Keisuke Kinoshita, Tomohiro Nakatani, Shoko Araki, and Dorothea Kolossa. **Exploiting Attention-based Sequence-to-Sequence Architectures for Sound Event Localization**. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 231–235, January 2021. 5

[46] Guillaume Le Moing, Phongtharin Vinayavekhin, Don Joven Agravante, Tadanobu Inoue, Jayakorn Vongkulbhisal, Asim Munawar, and Ryuki Tachibana. **Data-Efficient Framework for Real-World Multiple Sound Source 2d Localization**. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3425–3429, June 2021. 5, 6, 108, 138

[47] Yifan Wu, Roshan Ayyalasomayajula, Michael J. Bianco, Dinesh Bharadia, and Peter Gerstoft. **SSLIDE: Sound Source Localization for Indoors Based on Deep Learning**. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4680–4684, June 2021. 5

[48] Juan Manuel Vera-Diaz, Daniel Pizarro, and Javier Macias-Guarasa. **Acoustic Source Localization with Deep Generalized Cross Correlations**. *Signal Processing*, **187**:108169, October 2021. 5, 6

[49] Michael J. Bianco, Sharon Gannot, and Peter Gerstoft. **Semi-Supervised Source Localization with Deep Generative Modeling**. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, September 2020. 5, 138

[50] Michael J. Bianco, Sharon Gannot, Efren Fernandez-Grande, and Peter Gerstoft. **Semi-Supervised Source Localization in Reverberant Environments With Deep Generative Modeling**. *IEEE Access*, **9**:84956–84970, 2021. 5, 6, 138

[51] Sharath Adavanne, Archontis Politis, and Tuomas Virtanen. **Direction of Arrival Estimation for Multiple Sound Sources Using Convolutional Recurrent Neural Network**. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 1462–1466, September 2018. 5

[52] Yingxiang Sun, Jiajia Chen, Chau Yuen, and Susanto Rahardja. **Indoor Sound Source Localization With Probabilistic Neural Network**. *IEEE Transactions on Industrial Electronics*, **65**(8):6403–6413, August 2018. 5

[53] Lauréline Perotin, Alexandre Défossez, Emmanuel Vincent, Romain Serizel, and Alexandre Guérin. **Regression Versus Classification for Neural Network Based Audio Source Localization**. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 343–347, October 2019. 6, 84, 98

[54] Zhenyu Tang, John D. Kanu, Kevin Hogan, and Dinesh Manocha. **Regression and Classification for Direction-of-Arrival Estimation with Convolutional Recurrent Neural Networks**. In *Interspeech 2019*, pages 654–658. ISCA, September 2019. 6, 84, 98

[55] Sławomir Kapka and Mateusz Lewandowski. **Sound Source Detection, Localization and Classification Using Consecutive Ensemble of CRNN Models**. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 119–123. New York University, 2019. 6

[56] Sotirios Panagiotis Chytas and Gerasimos Potamianos. **Hierarchical Detection of Sound Events and Their Localization Using Convolutional Neural Networks with Adaptive Thresholds**. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 50–54, New York University, NY, USA, October 2019. 6

[57] Sooyoung Park. **TrellisNet-Based Architecture for Sound Event Localization and Detection with Reassembly Learning**. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 179–183, New York University, NY, USA, October 2019. 6

[58] Kazuki Shimada, Yuichiro Koyama, Naoya Takahashi, Shusuke Takahashi, and Yuki Mitsufuji. **Accdoa: Activity-Coupled Cartesian Direction of Arrival Representation for Sound Event Localization And Detection**. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 915–919, June 2021. 6, 98, 117

[59] Junhyeong Pak and Jong Won Shin. **Sound Localization Based on Phase Difference Enhancement Using Deep Neural Networks**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **27**(8):1335–1345, August 2019. 6

[60] Yankun Huang, Xihong Wu, and Tianshu Qu. **A Time-domain Unsupervised Learning Based Sound Source Localization Method**. In *2020 IEEE 3rd International Conference on Information Communication and Signal Processing (ICICSP)*, pages 26–32, September 2020. 6

[61] Jont B. Allen and David A. Berkley. **Image Method for Efficiently Simulating Small-room Acoustics**. *The Journal of the Acoustical Society of America*, **65**(4):943–950, April 1979. 7, 29, 56, 59, 60, 101

[62] Daniel Krause, Archontis Politis, and Konrad Kowalczyk. **Data Diversity for Improving DNN-based Localization of Concurrent Sound Events**. In *2021 29th European Signal Processing Conference (EUSIPCO)*, pages 236–240, August 2021. 7, 117, 121

[63] Yi Luo and Jianwei Yu. **FRA-RIR: Fast Random Approximation of the Image-source Method**, August 2022. 7, 49

[64] Ryu Takeda, Yoshiki Kudo, Kazuki Takashima, Yoshifumi Kitamura, and Kazunori Komatani. **Unsupervised Adaptation of Neural Networks for Discriminative Sound Source Localization with Eliminative Constraint**. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3514–3518, April 2018. 7, 108, 138

[65] Pasi Pertilä, Mikko Parviainen, Ville Myllylä, Anu Huttunen, and Petri Jarske. **Time Difference of Arrival Estimation with Deep Learning – From Acoustic Simulations to Recorded Data**. In *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6, September 2020. 7, 108, 138

[66] Weipeng He, Petr Motlicek, and Jean-Marc Odobez. **Neural Network Adaptation and Data Augmentation for Multi-Speaker Direction-of-Arrival Estimation**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **29**:1303–1317, 2021. 7, 108

[67] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. **Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges**, May 2021. 7

[68] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. **Backpropagation Applied to Handwritten Zip Code Recognition**. *Neural Computation*, **1**(4):541–551, December 1989. 8

[69] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. **ImageNet Classification with Deep Convolutional Neural Networks**. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, **25**. Curran Associates, Inc., 2012. 8

[70] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. **Deep Residual Learning for Image Recognition**. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, USA, June 2016. IEEE. 8

[71] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. **U-Net: Convolutional Networks for Biomedical Image Segmentation**. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Lecture Notes in Computer Science, pages 234–241, Cham, 2015. Springer International Publishing. 8, 115

[72] Jonathan Long, Evan Shelhamer, and Trevor Darrell. **Fully Convolutional Networks for Semantic Segmentation**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 8

[73] Taco Cohen and Max Welling. **Group Equivariant Convolutional Networks**. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 2990–2999. PMLR, June 2016. 8

[74] Emiel Hoogeboom, Jorn WT Peters, Taco S. Cohen, and Max Welling. **Hexaconv**. *arXiv preprint arXiv:1803.02108*, 2018. 8

[75] Daniel Worrall and Gabriel Brostow. **Cubenet: Equivariance to 3d Rotation and Translation**. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 567–584, 2018. 8

[76] Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. **Gauge Equivariant Convolutional Networks and the Icosahedral CNN**. In *Proceedings of the 36th International Conference on Machine Learning*, pages 1321–1330. PMLR, May 2019. 8, 12, 93, 94, 95, 104

[77] Taco S. Cohen, Mario Geiger, Jonas Köhler, and Max Welling. **Spherical CNNs**. In *International Conference on Learning Representations*, February 2018. 8, 93, 138

[78] Thomas N. Kipf and Max Welling. **Semi-Supervised Classification with Graph Convolutional Networks**. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 8

[79] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. **Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering**. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, **29**. Curran Associates, Inc., 2016. 8

[80] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M. Bronstein. **Geometric Deep Learning on Graphs and Manifolds Using Mixture Model Cnns**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5115–5124, 2017. 8

[81] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. **Graph Attention Networks**. In *International Conference on Learning Representations*, 2018. 8

[82] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. **Neural Message Passing for Quantum Chemistry**. In *International Conference on Machine Learning*, pages 1263–1272. PMLR, 2017. 8

[83] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. **Deep Sets**. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, **30**. Curran Associates, Inc., 2017. 8

[84] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. **Attention Is All You Need**. In *Advances in Neural Information Processing Systems*, **30**. Curran Associates, Inc., 2017. 8, 118, 119

[85] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. **E (n) Equivariant Graph Neural Networks**. In *International Conference on Machine Learning*, pages 9323–9332. PMLR, 2021. 8

[86] Danilo Comminiello, Marco Lella, Simone Scardapane, and Aurelio Uncini. **Quaternion Convolutional Neural Networks for Detection and Localization of 3D Sound Events**. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8533–8537, May 2019. 10

[87] Michela Ricciardi Celsi, Simone Scardapane, and Danilo Comminiello. **Quaternion Neural Networks for 3D Sound Source Localization in Reverberant Environments**. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, September 2020. 10

[88] Boaz Rafaely. *Fundamentals of Spherical Array Processing*, **8** of *Springer Topics in Signal Processing*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015. 16

[89] Daniel P. Jarrett, Emanuël A.P. Habets, and Patrick A. Naylor. *Theory and Applications of Spherical Microphone Array Processing*, **9** of *Springer Topics in Signal Processing*. Springer International Publishing, Cham, 2017. 16

[90] C. Knapp and G. Carter. **The Generalized Correlation Method for Estimation of Time Delay**. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **24**(4):320–327, August 1976. 16

[91] H. Wang and P. Chu. **Voice Source Localization for Automatic Camera Pointing System in Videoconferencing**. In *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, **1**, pages 187–190 vol.1, April 1997. 16

[92] M. Omologo and P. Svaizer. **Use of the Crosspower-Spectrum Phase in Acoustic Event Location**. *IEEE Transactions on Speech and Audio Processing*, **5**(3):288–292, May 1997. 16

[93] C. M. Zannini, A. Cirillo, R. Parisi, and A. Uncini. **Improved TDOA Disambiguation Techniques for Sound Source Localization in Reverberant Environments**. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 2666–2669, May 2010. 16

[94] Kevin D. Donohue, Jens Hannemann, and Henry G. Dietz. **Performance of Phase Transform for Detecting Sound Sources with Microphone Arrays in Reverberant and Noisy Environments**. *Signal Processing*, **87**(7):1677–1691, 2007. 16

[95] B. Champagne, S. Bedard, and A. Stephenne. **Performance of Time-Delay Estimation in the Presence of Room Reverberation**. *IEEE Transactions on Speech and Audio Processing*, **4**(2):148–152, March 1996. 16

[96] H. Do and H. F. Silverman. **A Fast Microphone Array SRP-PHAT Source Location Implementation Using Coarse-To-Fine Region Contraction(CFRC)**. In *2007 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 295–298, October 2007. 16

[97] H. Do and H. F. Silverman. **Stochastic Particle Filtering: A Fast SRP-PHAT Single Source Localization Algorithm**. In *2009 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 213–216, October 2009. 16, 19

[98] L. O. Nunes, W. A. Martins, M. V. S. Lima, L. W. P. Biscainho, M. V. M. Costa, F. M. Gonçalves, A. Said, and B. Lee. **A Steered-Response Power Algorithm Employing Hierarchical Search for Acoustic Source Localization Using Microphone Arrays**. *IEEE Transactions on Signal Processing*, **62**(19):5171–5183, October 2014. 16, 19, 20

[99] Amparo Marti, Maximo Cobos, Jose J. Lopez, and Jose Escolano. **A Steered Response Power Iterative Method for High-Accuracy Acoustic Source Localization**. *The Journal of the Acoustical Society of America*, **134**(4):2627–2630, October 2013. 16

[100] M. V. S. Lima, W. A. Martins, L. O. Nunes, L. W. P. Biscainho, T. N. Ferreira, M. V. M. Costa, and B. Lee. **A Volumetric SRP with Refinement Step for Sound Source Localization**. *IEEE Signal Processing Letters*, **22**(8):1098–1102, August 2015. 16

[101] S. Zhao, S. Ahmed, Y. Liang, K. Rupnow, D. Chen, and D. L. Jones. **A Real-Time 3D Sound Localization System with Miniature Microphone Array for Virtual Reality**. In *2012 7th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 1853–1857, July 2012. 16

[102] Taewoo Lee, Sukmoon Chang, and Dongsuk Yook. **Parallel SRP-PHAT for GPUs**. *Computer Speech & Language*, **35**(Supplement C):1–13, January 2016. 16

[103] H. R. Zarghi, M. Sharifkhani, and I. Gholampour. **Implementation of a Cost Efficient SSL Based on an Angular Beamformer SRP-PHAT**. In *2011 18th IEEE International Conference on Electronics, Circuits, and Systems*, pages 49–52, December 2011. 16

[104] R. Roy, A. Paulraj, and T. Kailath. **Estimation of Signal Parameters via Rotational Invariance Techniques - ESPRIT**. In *IEEE Military Communications Conference - Communications-Computers: Teamed for the 90's, 1986. MILCOM 1986*, **3**, pages 41.6.1–41.6.5, October 1986. 16

[105] M. A. Doran, E. Doron, and A. J. Weiss. **Coherent Wide-Band Processing for Arbitrary Array Geometry**. *IEEE Transactions on Signal Processing*, **41**(1):414–, January 1993. 17

[106] A. Johansson, G. Cook, and S. Nordholm. **Acoustic Direction of Arrival Estimation, a Comparison between Root-Music and SRP-PHAT**. In *TENCON 2004. 2004 IEEE Region 10 Conference*, **B**, pages 629–632, November 2004. 17

[107] J. P. Dmochowski, J. Benesty, and S. Affes. **Broadband Music: Opportunities and Challenges for Multiple Source Localization**. In *2007 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 18–21, October 2007. 17

[108] J. Krolik and D. Swingler. **Multiple Broad-Band Source Location Using Steered Covariance Matrices**. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **37**(10):1481–1494, October 1989. 17

[109] M. Souden, J. Benesty, and S. Affes. **Broadband Source Localization From an Eigenanalysis Perspective**. *IEEE Transactions on Audio, Speech, and Language Processing*, **18**(6):1575–1587, August 2010. 17

[110] Satish Mohan, Michael E. Lockwood, Michael L. Kramer, and Douglas L. Jones. **Localization of Multiple Acoustic Sources with Small Arrays Using a Coherence Test**. *The Journal of the Acoustical Society of America*, **123**(4):2136–2147, April 2008. 17

[111] M. Ren and Y. X. Zou. **A Novel Multiple Sparse Source Localization Using Triangular Pyramid Microphone Array**. *IEEE Signal Processing Letters*, **19**(2):83–86, February 2012. 17

[112] D. Pavlidi, A. Griffin, M. Puigt, and A. Mouchtaris. **Real-Time Multiple Sound Source Localization and Counting Using a Circular Microphone Array**. *IEEE Transactions on Audio, Speech, and Language Processing*, **21**(10):2193–2206, October 2013. 17

[113] O. Nadiri and B. Rafaely. **Localization of Multiple Speakers under High Reverberation Using a Spherical Microphone Array and the Direct-Path Dominance Test**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **22**(10):1494–1505, October 2014. 17

[114] H. Pessentheiner, M. Hagmüller, and G. Kubin. **Localization and Characterization of Multiple Harmonic Sources**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **24**(8):1348–1363, August 2016. 17

[115] Maoshen Jia, Jundai Sun, Changchun Bao, and Christian Ritz. **Multiple-to-Single Sound Source Localization by Applying Single-Source Bins Detection**. *Applied Acoustics*, **138**:28–38, September 2018. 17

[116] S. Rickard and O. Yilmaz. **On the Approximate W-disjoint Orthogonality of Speech**. In *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, **1**, pages I–529–I–532, May 2002. 17

[117] J. Traa and P. Smaragdis. **A Wrapped Kalman Filter for Azimuthal Speaker Tracking**. *IEEE Signal Processing Letters*, **20**(12):1257–1260, December 2013. 17

[118] Ye Tian, Zhe Chen, and Fuliang Yin. **Distributed Kalman Filter-Based Speaker Tracking in Microphone Array Networks**. *Applied Acoustics*, **89**:71–77, March 2015. 17

[119] D. B. Ward, E. A. Lehmann, and R. C. Williamson. **Particle Filtering Algorithms for Tracking an Acoustic Source in a Reverberant Environment**. *IEEE Transactions on Speech and Audio Processing*, **11**(6):826–836, November 2003. 17

[120] Wing-Kin Ma, Ba-Ngu Vo, S. S. Singh, and A. Baddeley. **Tracking an Unknown Time-Varying Number of Speakers Using TDOA Measurements: A Random Finite Set Approach**. *IEEE Transactions on Signal Processing*, **54**(9):3291–3304, September 2006. 17

[121] C. Evers, Y. Dorfan, S. Gannot, and P. A. Naylor. **Source Tracking Using Moving Microphone Arrays for Robot Audition**. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6145–6149, March 2017. 18

[122] O. Schwartz, Y. Dorfan, E. A. P. Habets, and S. Gannot. **Multi-Speaker DOA Estimation in Reverberation Conditions Using Expectation-Maximization**. In *2016 IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*, pages 1–5, September 2016. 18

[123] H. Do, H. F. Silverman, and Y. Yu. **A Real-Time SRP-PHAT Source Location Implementation Using Stochastic Region Contraction(SRC) on a Large-Aperture Microphone Array**. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, **1**, pages I–121–I–124, April 2007. 19

[124] Maximo Cobos, Amparo Marti, and Jose J. Lopez. **A Modified SRP-PHAT Functional for Robust Real-Time Sound Source Localization With Scalable Spatial Sampling**. *IEEE Signal Processing Letters*, **18**(1):71–74, January 2011. 20

[125] I. J. Clarke. **Efficient Maximum Likelihood Using Higher Rank Spectral Estimation**. In *Workshop on Higher-Order Spectral Analysis, 1989*, pages 229–234, June 1989. 22, 23

[126] M. A. Zatman, H. J. Strangeways, and E. M. Warrington. **Resolution of Multimoded HF Transmissions Using the DOSE Superresolution Direction Finding Algorithm**. In *, Eighth International Conference on Antennas and Propagation, 1993*, pages 415–417 vol.1, 1993. 22, 23

[127] A. Morrison, B. S. Sharif, S. Sali, and O. R. Hinton. **An Iterative DOA Algorithm for a Space-Time DS-CDMA RAKE Receiver**. In *3G Mobile Communication Technologies, 2000. First International Conference on (Conf. Publ. No. 471)*, pages 208–212, 2000. 22, 23

[128] M. A. Guérard and D. Grenier. **Direction of Arrival Estimation of Acoustic Echoes Using Source Elimination Method**. *Canadian Journal of Electrical and Computer Engineering*, **40**(3):246–252, 2017. 22

[129] Alessio Brutti, Maurizio Omologo, and Piergiorgio Svaizer. **Multiple Source Localization Based on Acoustic Map De-Emphasis**. *EURASIP Journal on Audio, Speech, and Music Processing*, **2010**(1):147495, December 2010. 22, 34, 35, 36, 40, 45, 46

[130] Angelo Farina and Lamberto Tronchin. **3D Sound Characterisation in Theatres Employing Microphone Arrays**. *Acta Acustica united with Acustica*, **99**(1):118–125, January 2013. 25

[131] A. O'Donovan, R. Duraiswami, and D. Zotkin. **Imaging Concert Hall Acoustics Using Visual and Audio Cameras**. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5284–5287, March 2008. 25

[132] Cha Zhang, D. Florencio, and Zhengyou Zhang. **Why Does PHAT Work Well in Lownoise, Reverberative Environments?** In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2565–2568, March 2008. 26, 37

[133] E A Lehmann and A M Johansson. **Diffuse Reverberation Model for Efficient Image-Source Simulation of Room Impulse Responses**. *IEEE Transactions on Audio, Speech, and Language Processing*, **18**(6):1429–1439, August 2010. 29, 56, 60, 69

[134] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. **Curriculum Learning**. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48, Montreal, Quebec, Canada, June 2009. Association for Computing Machinery. 49, 84, 101

[135] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. **Librispeech: An ASR Corpus Based on Public Domain Audio Books**. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, April 2015. 51, 101

[136] John Wiseman. **Wiseman/Py-Webrtcvad**. https://github.com/wiseman/py-webrtcvad, November 2019. 51, 101

[137] H. W. Löllmann, C. Evers, A. Schmidt, H. Mellmann, H. Barfuss, P. A. Naylor, and W. Kellermann. **The LOCATA Challenge Data Corpus for Acoustic Source Localization and Tracking**. In *2018 IEEE 10th Sensor Array and Multichannel Signal Processing Workshop (SAM)*, pages 410–414, July 2018. 53, 81, 90, 91, 101, 106

[138] Heinrich. W. Löllmann, Christine Evers, Alexander Schmidt, Heinrich Mellmann, Hendrik Barfuss, Patrick A. Naylor, and Walter Kellermann. **IEEE-AASP Challenge on Acoustic Source Localization and Tracking: Documentation of Final Release**. https://locata.lms.tf.fau.de/datasets/, January 2020. 53, 101, 137

[139] Jason L. Williams. **Marginal Multi-Bernoulli Filters: RFS Derivation of MHT, JIPDA, and Association-Based Member**. *IEEE Transactions on Aerospace and Electronic Systems*, **51**(3):1664–1687, July 2015. 53

[140] Karl Granström, Lennart Svensson, Yuxuan Xia, Jason Williams, and Ángel F. García-Femández. **Poisson Multi-Bernoulli Mixture Trackers: Continuity Through Random Finite Sets of Trajectories**. In *2018 21st International Conference on Information Fusion (FUSION)*, pages 1–5, July 2018. 53

[141] Chao Weng, Dong Yu, Shinji Watanabe, and Biing-Hwang Fred Juang. **Recurrent Deep Neural Networks for Robust Speech Recognition**. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5532–5536, May 2014. 55

[142] Anthony Griffin, Anastasios Alexandridis, Despoina Pavlidi, Yiannis Mastorakis, and Athanasios Mouchtaris. **Localizing Multiple Audio Sources in a Wireless Acoustic Sensor Network**. *Signal Processing*, **107**:54–67, February 2015. 55, 69

[143] D. S. Williamson and D. Wang. **Time-Frequency Masking in the Complex Domain for Speech Dereverberation and Denoising**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **25**(7):1492–1501, July 2017. 55, 69

[144] Manfred R. Schroeder. **Natural Sounding Artificial Reverberation**. *Journal of the Audio Engineering Society*, **10**(3):219–223, July 1962. 55

[145] Zhong-hua Fu and Jian-wei Li. **GPU-based Image Method for Room Impulse Response Calculation**. *Multimedia Tools and Applications*, **75**(9):5205–5221, May 2016. 56

[146] Emanuël A.P. Habets. **Room Impulse Response Generator**. Technical report, International Audio Laboratories Erlangen, September 2010. 56, 69

[147] Robin Scheibler, Eric Bezzam, and Ivan Dokmanić. **Pyroomacoustics: A Python Package for Audio Room Simulation and Array Processing Algorithms**. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 351–355, April 2018. 56, 69

[148] Patrick M. Peterson. **Simulating the Response of Multiple Microphones to a Single Acoustic Source in a Reverberant Room**. *The Journal of the Acoustical Society of America*, **80**(5):1527–1529, November 1986. 59

[149] B. D. Radlovic, R. C. Williamson, and R. A. Kennedy. **Equalization in an Acoustic Reverberant Environment: Robustness Results**. *IEEE Transactions on Speech and Audio Processing*, **8**(3):311–319, May 2000. 59

[150] J Antonio, L Godinho, and A Tadeu. **Reverberation Times Obtained Using a Numerical Model Versus Those Given by Simplified Formulas and Measurements**. *ACTA ACUSTICA UNITED WITH ACUSTICA*, **88**:10, 2002. 59

[151] Eric A. Lehmann and Anders M. Johansson. **Prediction of Energy Decay in Room Impulse Responses Simulated with an Image-Source Model**. *The Journal of the Acoustical Society of America*, **124**(1):269–277, July 2008. 60, 69

[152] Wallace Clement Sabine. *Collected Papers on Acoustics*. Cambridge : Harvard University Press, 1922. 60

[153] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. **Scalable Parallel Programming with CUDA**. In *ACM SIGGRAPH 2008 Classes*, SIGGRAPH '08, pages 16:1–16:14, New York, NY, USA, 2008. ACM. 61

[154] Tor G. J. Myklebust. **Computing Accurate Horner Form Approximations to Special Functions in Finite Precision Arithmetic**. *arXiv:1508.03211 [cs, math]*, August 2015. 67

[155] Marvin182. **Room Impulse Response Generator**. https://github.com/Marvin182/rir-generator, August 2018. 69

[156] Amin Hassani, Jorge Plata-Chaves, Mohamad Hasan Bahari, Marc Moonen, and Alexander Bertrand. **Multi-Task Wireless Sensor Network for Joint Distributed Node-Specific Signal Enhancement, LCMV Beamforming and DOA Estimation**. *IEEE Journal of Selected Topics in Signal Processing*, **11**(3):518–533, April 2017. 69

[157] Shmulik Markovich, Sharon Gannot, and Israel Cohen. **Multichannel Eigenspace Beamforming in a Reverberant Noisy Environment With Multiple Interfering Speech Signals**. *IEEE Transactions on Audio, Speech, and Language Processing*, **17**(6):1071–1086, August 2009. 69

[158] Xiaoyi Qin, Danwei Cai, and Ming Li. **Far-Field End-to-End Text-Dependent Speaker Verification Based on Mixed Training Data with Transfer Learning and Enrollment Data Augmentation**. In *Interspeech 2019*, pages 4045–4049. ISCA, September 2019. 69

[159] Ladislav Mosner, Minhua Wu, Anirudh Raju, Sree Hari Krishnan Parthasarathi, Kenichi Kumatani, Shiva Sundaram, Roland Maas, and Bjorn Hoffmeister. **Improving Noise Robustness of Automatic Speech Recognition via Parallel Data and Teacher-student Learning**. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6475–6479, Brighton, United Kingdom, May 2019. IEEE. 69

[160] Marco Severini, Daniele Ferretti, Emanuele Principi, and Stefano Squartini. **Automatic Detection of Cry Sounds in Neonatal Intensive Care Units by Using Deep Learning and Acoustic Scene Simulation**. *IEEE Access*, **7**:51982–51993, 2019. 69

[161] Eric A. Lehmann. **Fast Simulation of Acoustic Room Impulse Responses (Image-Source Method)**. https://www.mathworks.com/matlabcentral/fileexchange/25965-fast-simulation-of-acoustic-room-impulse-responses-image-source-method, March 2012. 69

[162] Despoina Pavlidi, Matthieu Puigt, Anthony Griffin, and Athanasios Mouchtaris. **Real-Time Multiple Sound Source Localization Using a Circular Microphone Array Based on Single-Source Confidence Measures**. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2625–2628, March 2012. 69

[163] Anastasios Alexandridis, Anthony Griffin, and Athanasios Mouchtaris. **Capturing and Reproducing Spatial Audio Based on a Circular Microphone Array**. *Journal of Electrical and Computer Engineering*, **2013**:e718574, March 2013. 69

[164] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. **Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification**. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, December 2015. 82

[165] Diederik P. Kingma and Jimmy Ba. **Adam: A Method for Stochastic Optimization**. In *3rd International Conference for Learning Representations (ICLR)*, San Diego, 2015. 84, 101

[166] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. **PyTorch: An Imperative Style, High-Performance Deep Learning Library**. In *Advances in Neural Information Processing Systems 32*, pages 8026–8037, 2019. 84, 101

[167] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. **Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation**. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. 86

[168] Carlos Esteves, Christine Allen-Blanchette, Ameesh Makadia, and Kostas Daniilidis. **Learning SO(3) Equivariant Representations with Spherical CNNs**. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–68, 2018. 93, 138

[169] Oliver Cobb, Christopher G. R. Wallis, Augustine N. Mavor-Parker, Augustin Marignier, Matthew A. Price, Mayeul d'Avezac, and Jason McEwen. **Efficient Generalized Spherical CNNs**. In *International Conference on Learning Representations*, September 2020. 93, 138

[170] Jason McEwen, Christopher Wallis, and Augustine N. Mavor-Parker. **Scattering Networks on the Sphere for Scalable and Rotationally Equivariant Spherical CNNs**. In *International Conference on Learning Representations*, September 2021. 93, 138

[171] Renata Khasanova and Pascal Frossard. **Graph-Based Classification of Omnidirectional Images**. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 869–878, 2017. 93, 139

[172] Nathanaël Perraudin, Michaël Defferrard, Tomasz Kacprzak, and Raphael Sgier. **DeepSphere: Efficient Spherical Convolutional Neural Network with HEALPix Sampling for Cosmological Applications**. *Astronomy and Computing*, **27**:130–146, April 2019. 93, 139

[173] Michaël Defferrard, Martino Milani, Frédérick Gusset, and Nathanaël Perraudin. **DeepSphere: A Graph-Based Spherical CNN**. In *International Conference on Learning Representations*, September 2019. 93, 139

[174] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. **Layer Normalization**. In *NIPS 2016 Deep Learning Symposium*, Barcelona, July 2016. 99

[175] Sepp Hochreiter and Jürgen Schmidhuber. **Long Short-Term Memory**. *Neural Computation*, **9**(8):1735–1780, November 1997. 117

[176] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. **On the Properties of Neural Machine Translation: Encoder–Decoder Approaches**. In *SSST@EMNLP*, 2014. 117

[177] Guo-Bing Zhou, Jianxin Wu, Chen-Lin Zhang, and Zhi-Hua Zhou. **Minimal Gated Unit for Recurrent Neural Networks**. *International Journal of Automation and Computing*, **13**(3):226–234, June 2016. 119, 139

[178] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. **TrackFormer: Multi-Object Tracking with Transformers**. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8834–8844, New Orleans, LA, USA, June 2022. IEEE. 120, 139, 141

[179] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. **End-to-End Object Detection with Transformers**. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 213–229, Cham, 2020. Springer International Publishing. 120

[180] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. **Deformable DETR: Deformable Transformers for End-to-End Object Detection**. In *International Conference on Learning Representations*, February 2022. 120

[181] Yihong Xu, Aljosa Osep, Yutong Ban, Radu Horaud, Laura Leal-Taixé, and Xavier Alameda-Pineda. **How to Train Your Deep Multi-Object Tracker**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6787–6796, 2020. 121

[182] Dong Yu, Morten Kolbæk, Zheng-Hua Tan, and Jesper Jensen. **Permutation Invariant Training of Deep Models for Speaker-Independent Multi-Talker Speech Separation**. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 241–245, March 2017. 121

[183] Harold W. Kuhn. **The Hungarian Method for the Assignment Problem**. *Naval research logistics quarterly*, **2**(1-2):83–97, 1955. 122

[184] Morten Kolbæk, Dong Yu, Zheng-Hua Tan, and Jesper Jensen. **Multitalker Speech Separation With Utterance-Level Permutation Invariant Training of Deep Recurrent Neural Networks**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **25**(10):1901–1913, October 2017. 123

[185] Ilya Loshchilov and Frank Hutter. **Decoupled Weight Decay Regularization**. In *International Conference on Learning Representations*, 2018. 126

[186] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. **On the Difficulty of Training Recurrent Neural Networks**. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1310–1318. PMLR, May 2013. 126

[187] Keni Bernardin and Rainer Stiefelhagen. **Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics**. *EURASIP Journal on Image and Video Processing*, **2008**(1):1–10, December 2008. 126

[188] Christine Evers, Heinrich W. Löllmann, Heinrich Mellmann, Alexander Schmidt, Hendrik Barfuss, Patrick A. Naylor, and Walter Kellermann. **The LOCATA Challenge: Acoustic Source Localization and Tracking**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **28**:1620–1643, 2020. 126

[189] Luca Cristoforetti, Mirco Ravanelli, Maurizio Omologo, Alessandro Sosi, Alberto Abad, Martin Hagmueller, and Petros Maragos. **The DIRHA Simulated Corpus**. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2629–2634, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA). 138

[190] Elior Hadad, Florian Heese, Peter Vary, and Sharon Gannot. **Multichannel Audio Database in Various Acoustic Environments**. In *2014 14th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pages 313–317, September 2014. 138

[191] Igor Szöke, Miroslav Skácel, Ladislav Mošner, Jakub Paliesek, and Jan Černocký. **Building and Evaluation of a Real Room Impulse Response Dataset**. *IEEE Journal of Selected Topics in Signal Processing*, **13**(4):863–876, August 2019. 138

[192] Efren Fernandez-Grande. **DTU Three-Channel Room Impulse Response Dataset for Direction of Arrival Estimation 2020**, June 2021. 138

[193] Prerak Srivastava, Antoine Deleforge, Archontis Politis, and Emmanuel Vincent. **How to (Virtually) Train Your Sound Source Localizer**. *HAL Archives Ouvertes*, November 2022. 138

[194] Femke B. Gelderblom, Yi Liu, Johannes Kvam, and Tor Andre Myrvoll. **Synthetic Data For Dnn-Based Doa Estimation of Indoor Speech**. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4390–4394, June 2021. 138

[195] Kadi Bouatouch, Olivier Deille, Julien Maillard, Jacques Martin, and Nicolas Noé. **Real Time Acoustic Rendering of Complex Environments Including Diffraction and Curved Surfaces**. In *Audio Engineering Society Convention 120*. Audio Engineering Society, May 2006. 138

[196] Ryu Takeda and Kazunori Komatani. **Unsupervised Adaptation of Deep Neural Networks for Sound Source Localization Using Entropy Minimization**. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2217–2221, March 2017. 138

[197] Ryotaro Sato, Kenta Niwa, and Kazunori Kobayashi. **Ambisonic Signal Processing DNNs Guaranteeing Rotation, Scale and Time Translation Equivariance**. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **29**:1449–1462, 2021. 139

Appendix A

Network architecture of the proposed and baseline models

The following tables, some of them extracted from the supplementary material of [V], detail the network architecture of the proposed models and baseline models analyzed along the thesis:

A.1 Cross3D models

Table A.1: Architecture of Cross3D for 4x8 power maps. The output sizes correspond to a sequence of 103 maps, i.e. an input size of 3x103x4x8.

	Layer	Number of kernels	Kernel size	Dilation	Number of parameters	Output size
Input layer	3D causal conv	32	5x5x5	1x1x1	12 032	32x103x4x8
	PReLU				32	32x103x4x8
θ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x4x8
	Max pooling		1x1x2			32x103x4x4
	PReLU				32	32x103x4x4
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x4x4
	Max pooling		1x1x2			32x103x4x2
	PReLU				32	32x103x4x2
	Transpose					103x32x4x2
	Reshape					103x256
φ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x4x8
	Max pooling		1x2x1			32x103x2x8
	PReLU				32	32x103x2x8
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x2x8
	Max pooling		1x2x1			32x103x1x8
	PReLU				32	32x103x1x8
	Transpose					103x32x1x8
	Reshape					103x256
Output layers	Concatenate					103x512
	Transpose					512x103
	1D causal conv	128	5	2	327 808	128x103
	PReLU				128	128x103
	1D causal conv	3	5	2	1923	3x103
	tanh					3x103

Table A.2: Architecture of Cross3D for 8x16 power maps. The output sizes correspond to a sequence of 103 maps, i.e. an input size of 3x103x8x16.

	Layer	Number of kernels	Kernel size	Dilation	Number of parameters	Output size
Input layer	3D causal conv	32	5x5x5	1x1x1	12 032	32x103x8x16
	PReLU				32	32x103x8x16
θ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x8x16
	Max pooling		1x1x2			32x103x8x8
	PReLU				32	32x103x8x8
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x8x8
	Max pooling		1x1x2			32x103x8x4
	PReLU				32	32x103x8x4
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x8x4
	Max pooling		1x1x2			32x103x8x2
	PReLU				32	32x103x8x2
	Transpose					103x32x8x2
	Reshape					103x512
φ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x8x16
	Max pooling		1x2x1			32x103x4x16
	PReLU				32	32x103x4x16
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x4x16
	Max pooling		1x2x1			32x103x2x16
	PReLU				32	32x103x2x16
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x2x16
	Max pooling		1x2x1			32x103x1x16
	PReLU				32	32x103x1x16
	Transpose					103x32x1x16
	Reshape					103x512
Output layers	Concatenate					103x1024
	Transpose					1024x103
	1D causal conv	128	5	2	655 488	128x103
	PReLU				128	128x103
	1D causal conv	3	5	2	1923	3x103
	tanh					3x103

A. NETWORK ARCHITECTURE OF THE PROPOSED AND BASELINE MODELS

Table A.3: Architecture of Cross3D for 16x32 power maps. The output sizes correspond to a sequence of 103 maps, i.e. an input size of 3x103x16x32.

	Layer	Number of kernels	Kernel size	Dilation	Number of parameters	Output size
Input layer	3D causal conv	32	5x5x5	1x1x1	12 032	32x103x16x32
	PReLU				32	32x103x16x32
θ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x16x32
	Max pooling		1x1x2			32x103x16x16
	PReLU				32	32x103x16x16
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x16x16
	Max pooling		1x1x2			32x103x16x8
	PReLU				32	32x103x16x8
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x16x8
	Max pooling		1x1x2			32x103x16x4
	PReLU				32	32x103x16x4
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x16x4
	Max pooling		1x1x2			32x103x16x2
	PReLU				32	32x103x16x2
	Transpose					103x32x16x2
	Reshape					103x1024
φ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x16x32
	Max pooling		1x2x1			32x103x8x32
	PReLU				32	32x103x8x32
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x8x32
	Max pooling		1x2x1			32x103x4x32
	PReLU				32	32x103x4x32
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x4x32
	Max pooling		1x2x1			32x103x2x32
	PReLU				32	32x103x2x32
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x2x32
	Max pooling		1x2x1			32x103x1x32
	PReLU				32	32x103x1x32
	Transpose					103x32x1x32
	Reshape					103x1024
Output layers	Concatenate					103x2048
	Transpose					2048x103
	1D causal conv	128	5	2	1 310 848	128x103
	PReLU				128	128x103
	1D causal conv	3	5	2	1923	3x103
	tanh					3x103

Table A.4: Architecture of Cross3D for 32x64 power maps. The output sizes correspond to a sequence of 103 maps, i.e. an input size of 3x103x32x64.

	Layer	Number of kernels	Kernel size	Dilation	Number of parameters	Output size
Input layer	3D causal conv	32	5x5x5	1x1x1	12 032	32x103x32x64
	PReLU				32	32x103x32x64
θ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x32x64
	Max pooling		1x1x2			32x103x32x32
	PReLU				32	32x103x32x32
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x32x32
	Max pooling		1x1x2			32x103x32x16
	PReLU				32	32x103x32x16
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x32x16
	Max pooling		1x1x2			32x103x32x8
	PReLU				32	32x103x32x8
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x32x8
	Max pooling		1x1x2			32x103x32x4
	PReLU				32	32x103x32x4
	Transpose					103x32x32x4
	Reshape					103x4096
φ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x32x64
	Max pooling		1x2x1			32x103x16x64
	PReLU				32	32x103x16x64
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x16x64
	Max pooling		1x2x1			32x103x8x64
	PReLU				32	32x103x8x64
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x8x64
	Max pooling		1x2x1			32x103x4x64
	PReLU				32	32x103x4x64
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x4x64
	Max pooling		1x2x1			32x103x2x64
	PReLU				32	32x103x2x64
	Transpose					103x32x2x64
	Reshape					103x4096
Output layers	Concatenate					103x8192
	Transpose					8192x103
	1D causal conv	128	5	2	5 243 008	128x103
	PReLU				128	128x103
	1D causal conv	3	5	2	1923	3x103
	tanh					3x103

A. NETWORK ARCHITECTURE OF THE PROPOSED AND BASELINE MODELS

Table A.5: Architecture of Cross3D for 64x128 power maps. The output sizes correspond to a sequence of 103 maps, i.e. an input size of 3x103x64x128.

	Layer	Number of kernels	Kernel size	Dilation	Number of parameters	Output size
Input layer	3D causal conv	32	5x5x5	1x1x1	12 032	32x103x64x128
	PReLU				32	32x103x64x128
θ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x64x128
	Max pooling		1x1x2			32x103x64x64
	PReLU				32	32x103x64x64
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x64x64
	Max pooling		1x1x2			32x103x64x32
	PReLU				32	32x103x64x32
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x64x32
	Max pooling		1x1x2			32x103x64x16
	PReLU				32	32x103x64x16
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x64x16
	Max pooling		1x1x2			32x103x64x8
	PReLU				32	32x103x64x8
	Transpose					103x32x64x8
	Reshape					103x16384
φ branch	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x64x128
	Max pooling		1x2x1			32x103x32x128
	PReLU				32	32x103x32x128
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x32x128
	Max pooling		1x2x1			32x103x16x128
	PReLU				32	32x103x16x128
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x16x128
	Max pooling		1x2x1			32x103x8x128
	PReLU				32	32x103x8x128
	3D causal conv	32	5x3x3	1x1x1	46 112	32x103x8x128
	Max pooling		1x2x1			32x103x4x128
	PReLU				32	32x103x4x128
	Transpose					103x32x4x128
	Reshape					103x16384
Output layers	Concatenate					103x32768
	Transpose					32768x103
	1D causal conv	128	5	2	20 971 648	128x103
	PReLU				128	128x103
	1D causal conv	3	5	2	1923	3x103
	tanh					3x103

A.2 icoCNN models

Table A.6: Architecture of the icoCNN model for r=1 power maps. The output sizes correspond to a sequence of 103 maps, i.e. an input size of 1x103x1x5x2x4.

Layer	Number of kernels	Kernel size	Number of parameters	Output size
icoConv	32	Hexagonal	256	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	1	5	161	1x103x6x5x2x4
R-pooling				1x103x1x5x2x4
soft-max				103x3

Table A.7: Architecture of the icoCNN model for r=2 power maps. The output sizes correspond to a sequence of 103 maps, i.e. an input size of 1x103x1x5x4x8.

Layer	Number of kernels	Kernel size	Number of parameters	Output size
icoConv	32	Hexagonal	256	32x103x6x5x4x8
1D causal conv	32	5	5152	32x103x6x5x4x8
Layer norm			64	32x103x6x5x4x8
ReLU				32x103x6x5x4x8
icoConv	32	Hexagonal	43 040	32x103x6x5x4x8
1D causal conv	32	5	5152	32x103x6x5x4x8
Layer norm			64	32x103x6x5x4x8
ReLU				32x103x6x5x4x8
icoPooling		Hexagonal		32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	1	5	161	1x103x6x5x2x4
R-pooling				1x103x1x5x2x4
soft-max				103x3

Table A.8: Architecture of the icoCNN model for r=3 power maps. The output sizes correspond to a sequence of 103 maps, i.e. an input size of 1x103x1x5x8x16.

Layer	Number of kernels	Kernel size	Number of parameters	Output size
icoConv	32	Hexagonal	256	32x103x6x5x8x16
1D causal conv	32	5	5152	32x103x6x5x8x16
Layer norm			64	32x103x6x5x8x16
ReLU				32x103x6x5x8x16
icoConv	32	Hexagonal	43 040	32x103x6x5x8x16
1D causal conv	32	5	5152	32x103x6x5x8x16
Layer norm			64	32x103x6x5x8x16
ReLU				32x103x6x5x8x16
icoPooling		Hexagonal		32x103x6x5x4x8
icoConv	32	Hexagonal	43 040	32x103x6x5x4x8
1D causal conv	32	5	5152	32x103x6x5x4x8
Layer norm			64	32x103x6x5x4x8
ReLU				32x103x6x5x4x8
icoConv	32	Hexagonal	43 040	32x103x6x5x4x8
1D causal conv	32	5	5152	32x103x6x5x4x8
Layer norm			64	32x103x6x5x4x8
ReLU				32x103x6x5x4x8
icoPooling		Hexagonal		32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	32	5	5152	32x103x6x5x2x4
Layer norm			64	32x103x6x5x2x4
ReLU				32x103x6x5x2x4
icoConv	32	Hexagonal	43 040	32x103x6x5x2x4
1D causal conv	1	5	161	1x103x6x5x2x4
R-pooling				1x103x1x5x2x4
soft-max				103x3

A.3 Baseline models

Table A.9: Architecture of the 1D CNN for DOA estimation from 858 sequences of GCCs. The output sizes corresponds to sequence of length 103, i.e. an input size of 858x103.

Layer	Number of kernels	Kernel size	Dilation	Number of parameters	Output size
1D causal conv	1024	5	1	4 393 984	1024x103
PReLU				1024	1024x103
1D causal conv	512	5	1	2 621 952	512x103
PReLU				512	512x103
1D causal conv	512	5	1	1 311 232	512x103
PReLU				512	512x103
1D causal conv	512	5	1	1 311 232	512x103
PReLU				512	512x103
1D causal conv	512	5	1	1 311 232	512x103
PReLU				512	512x103
1D causal conv	128	5	2	327 808	128x103
PReLU				512	128x103
1D causal conv	3	5	2	1923	3x103
tanh					3x103

Table A.10: Architecture of the 1D CNN for DOA estimation from the coordinates of the maximums of the SRP-PHAT maps. The output sizes correspond to sequences of length 103, i.e. an input size of 2x103.

Layer	Number of kernels	Kernel size	Dilation	Number of parameters	Output size
1D causal conv	1024	5	1	11 264	1024x103
PReLU				1024	1024x103
1D causal conv	512	5	1	2 621 952	512x103
PReLU				512	512x103
1D causal conv	512	5	1	1 311 232	512x103
PReLU				512	512x103
1D causal conv	512	5	1	1 311 232	512x103
PReLU				512	512x103
1D causal conv	512	5	1	1 311 232	512x103
PReLU				512	512x103
1D causal conv	128	5	2	327 808	128x103
PReLU				512	128x103
1D causal conv	3	5	2	1923	3x103
tanh					3x103

A. NETWORK ARCHITECTURE OF THE PROPOSED AND BASELINE MODELS

Table A.11: Architecture of the 2D CNN for DOA estimation from the spectrograms of the 12 microphone signals (window size $K = 2048$). The output sizes correspond to spectrograms with 103 time frames, i.e. an input size of 24x103x2048.

Layer	Number of kernels	Kernel size	Dilation	Number of parameters	Output size
2D causal conv	128	5x5	1	76 928	1024x103
PReLU				128	128x103x2048
Max pooling		1x4			128x103x512
2D causal conv	128	5x5	1	409 728	128x103x512
PReLU				128	128x103x512
Max pooling		1x4			128x103x128
2D causal conv	128	5x5	1	409 728	128x103x128
PReLU				128	128x103x128
Max pooling		1x4			128x103x32
2D causal conv	128	5x5	1	409 728	128x103x32
PReLU				128	128x103x32
Max pooling		1x4			128x103x8
2D causal conv	128	5x5	1	409 728	128x103x8
PReLU				128	128x103x8
1D causal conv	128	5	2	163 968	128x103
PReLU				128	128x103
1D causal conv	3	5	2	1923	3x103
tanh					3x103

Table A.12: Architecture of SELDnet for DOA estimation from spectrograms of the 12 microphone signals (window size $K = 512$). The output sizes correspond to spectrograms with 833 time frames, i.e. an input size of 24x833x256.

Layer	Number of kernels	Kernel size	Dilation	Number of parameters	Output size
2D conv	64	3x3	1	13 888	64x833x256
ReLU					64x833x256
Max pooling		1x8			64x833x32
2D conv	64	3x3	1	36 928	64x833x32
ReLU					64x833x32
Max pooling		1x8			64x833x4
2D conv	64	3x3	1	36 928	64x833x4
PReLU					64x833x4
Max pooling		1x2			64x833x2
GRU bi-directional	128			74 496	833x128
GRU bi-directional	128			74 496	833x128
Fully Connected	128			16 512	833x128
Fully Connected	3			387	833x3
tanh				128	833x3

Appendix B

Results in every recording of the LOCATA dataset

The following tables, extracted from the supplementary materials of [VI], show the root mean squared angular error (RMSAE) obtained with the proposed models in every recording of the single source tasks of the development and evaluation partitions of the LOCATA dataset:

Table B.1: RMSAE [°] of the DOA estimated for the development partition of the LOCATA dataset with the proposed models using several map resolutions. The second (gray) numbers indicate the RMSAE without taking into account the frames when the sound source was silent.

	Model:	IcoCNN				Cross3D					1D CNN
	Input:	Icosahedral SRP-PHAT maps				Equiangular SRP-PHAT maps					GCCs
		r=1	r=2	r=3	r=4	4x8	8x16	16x32	32x64	64x128	
Task 1	Recording 1	9.34 8.85	3.63 3.44	6.52 5.74	6.71 5.65	17.93 18.67	11.92 14.18	8.30 8.56	4.62 4.58	5.16 4.98	16.18 16.93
	Recording 2	8.07 8.36	6.70 6.77	6.53 6.39	6.15 5.99	18.90 20.22	7.68 8.86	6.68 7.11	4.90 4.92	3.91 3.96	12.60 12.73
	Recording 3	2.59 2.59	3.24 3.15	5.18 5.52	5.19 5.26	10.35 11.39	6.34 9.30	2.98 3.36	3.25 3.20	2.24 2.29	11.57 12.09
	Average	6.67 6.60	4.52 4.45	5.74 5.88	6.02 5.63	15.72 16.79	8.65 10.78	5.99 6.34	4.26 4.23	3.77 3.74	13.45 13.92
Task 3	Recording 1	8.19 7.45	8.08 5.79	7.70 5.42	7.52 5.93	23.06 23.34	18.11 20.89	13.79 14.94	12.43 12.98	9.92 10.06	13.59 13.62
	Recording 2	8.15 6.86	8.30 7.51	8.80 8.04	9.61 9.10	20.97 17.72	13.71 12.11	10.01 9.29	8.36 8.31	9.22 9.06	14.17 13.34
	Recording 3	10.25 7.57	7.82 5.33	7.18 4.69	7.50 5.60	21.05 24.92	12.74 12.62	9.83 8.71	7.69 6.56	6.60 5.02	15.21 13.09
	Average	8.86 7.29	8.07 6.21	7.89 6.05	8.21 6.88	21.69 21.99	14.85 15.21	11.21 10.98	9.49 9.28	8.58 8.05	14.32 13.35
Task 5	Recording 1	13.63 13.31	9.74 9.86	7.43 7.50	8.42 8.26	11.93 11.10	10.83 9.40	7.25 7.41	5.74 5.70	5.49 5.49	10.93 10.96
	Recording 2	14.04 12.07	10.64 9.94	9.72 8.49	9.39 8.15	20.92 20.67	16.16 15.56	16.08 15.84	12.18 11.57	13.59 13.67	17.33 17.38
	Recording 3	21.56 11.87	20.10 8.26	18.37 8.50	19.47 8.62	23.57 18.42	18.25 14.72	13.58 10.88	15.64 12.86	15.49 12.11	20.14 17.33
	Average	16.41 12.42	13.49 9.35	11.84 8.16	12.43 8.34	18.81 16.73	15.08 13.23	12.31 11.38	11.19 10.04	11.52 10.42	16.13 15.22
Average		10.65 8.77	8.69 6.67	8.61 6.70	8.88 6.95	18.74 18.50	12.86 13.07	9.83 9.57	8.31 7.85	7.96 7.41	14.64 14.16
Median		9.34 8.36	8.08 6.77	7.43 6.39	7.52 5.99	20.92 18.67	12.74 12.62	9.83 8.71	7.69 6.56	6.60 5.49	14.17 13.34
Standard deviation		5.00 3.10	4.66 2.35	3.67 1.37	3.98 1.45	4.40 4.45	3.98 3.62	3.87 3.67	3.99 3.53	3.20 3.72	2.77 2.28

Table B.2: RMSAE [°] of the DOA estimated for the evaluation partition of the LOCATA dataset with the proposed models using several map resolutions. The second (gray) numbers indicate the RMSAE without taking into account the frames when the sound source was silent.

	Model:	IcoCNN				Cross3D					1D CNN
	Input:	Icosahedral SRP-PHAT maps				Equiangular SRP-PHAT maps					GCCs
		r=1	r=2	r=3	r=4	4x8	8x16	16x32	32x64	64x128	
Task 1	Recording 1	12.24 12.05	7.17 7.64	5.87 5.76	5.67 4.63	23.78 22.31	19.62 18.75	5.46 5.17	4.45 4.42	3.37 3.33	6.43 6.78
	Recording 2	2.33 2.26	3.74 3.75	5.43 5.63	5.30 5.32	8.55 8.69	6.64 6.56	2.70 2.76	2.14 2.18	1.75 1.70	6.25 6.29
	Recording 3	10.70 11.75	3.78 4.04	5.11 5.52	4.41 4.42	14.65 13.99	10.78 10.64	7.32 7.60	8.45 8.52	5.17 5.17	27.04 27.44
	Recording 4	4.73 4.92	3.85 3.78	4.74 4.70	3.68 3.63	15.53 15.52	9.83 9.91	6.25 6.21	4.45 4.45	3.23 3.25	4.42 4.50
	Recording 5	8.09 8.17	4.92 4.89	4.99 5.13	4.80 4.73	77.58 62.77	53.83 43.74	22.18 24.82	6.55 7.27	4.84 5.35	9.34 9.27
	Recording 6	3.25 3.27	3.33 3.34	3.67 3.71	3.58 3.61	17.19 17.48	8.52 8.61	5.47 5.50	3.82 3.87	2.92 2.95	14.11 14.30
	Recording 7	11.38 11.59	12.00 11.80	8.29 8.03	7.01 7.04	18.39 18.57	15.61 15.75	7.59 7.58	3.80 3.77	4.14 4.14	19.30 19.43
	Recording 8	11.40 11.40	6.36 6.79	6.16 6.04	5.35 5.37	48.34 50.26	44.39 46.88	9.97 10.45	6.42 6.72	4.55 4.74	7.19 7.49
	Recording 9	6.36 6.58	5.37 5.58	5.08 5.26	5.96 5.94	12.15 11.77	7.44 7.21	2.50 2.40	3.91 3.85	1.85 1.92	8.60 8.38
	Recording 10	4.61 4.51	4.21 4.56	6.66 7.48	7.79 7.41	41.80 44.21	38.27 40.17	42.57 44.44	25.79 27.74	16.32 17.52	19.87 21.11
	Recording 11	4.43 5.00	5.47 6.13	4.49 4.83	3.70 4.17	52.15 52.11	49.90 51.77	58.48 58.92	31.99 33.24	14.92 15.44	20.95 21.25
	Recording 12	8.35 10.21	7.31 7.27	4.38 5.32	4.36 5.04	23.10 20.82	11.80 12.36	5.04 5.35	2.98 3.08	3.07 3.20	11.91 12.34
	Recording 13	16.72 16.66	8.95 9.52	7.65 7.79	6.60 6.71	31.34 32.23	15.53 15.72	4.98 4.94	4.74 4.93	2.50 2.52	7.58 7.86
	Average	8.04 8.34	5.88 6.08	5.57 5.78	5.25 5.23	28.62 27.09	22.47 22.16	13.89 14.32	9.84 8.51	5.28 5.48	12.54 12.80
Task 3	Recording 1	9.91 7.35	7.68 6.95	7.92 6.53	10.28 7.82	19.66 19.59	15.32 14.91	12.92 11.97	9.04 8.16	7.40 6.91	12.18 11.62
	Recording 2	16.10 10.32	15.62 9.44	14.57 7.82	15.58 9.58	17.43 13.11	14.49 11.70	11.99 7.53	12.41 8.14	14.19 8.94	13.47 11.46
	Recording 3	8.92 9.27	8.29 7.50	14.39 7.52	7.71 8.13	23.34 24.39	26.86 28.12	17.65 18.50	16.77 17.57	11.88 12.34	13.71 14.23
	Recording 4	8.28 7.99	6.78 6.24	6.99 6.64	7.84 7.48	14.63 14.59	14.45 14.43	9.90 9.83	8.21 8.10	7.95 7.87	12.10 12.00
	Recording 5	9.45 9.94	6.31 6.34	6.47 6.14	6.43 6.33	17.49 18.22	16.68 18.36	11.36 11.99	9.49 9.70	8.16 8.26	8.98 8.78
	Average	10.53 8.97	8.94 7.29	10.07 6.93	9.57 7.87	18.51 17.98	17.56 17.50	12.76 12.16	11.18 10.33	9.92 8.86	12.09 11.62
Task 5	Recording 1	25.15 21.33	15.72 12.80	19.45 18.26	17.29 14.31	25.87 22.16	28.02 24.93	24.08 21.08	24.94 24.58	25.31 23.87	22.27 16.83
	Recording 2	12.47 11.47	10.02 8.62	8.97 7.53	9.08 7.59	12.86 12.78	9.74 9.54	8.01 7.65	7.58 7.09	7.77 7.02	14.40 14.70
	Recording 3	14.30 9.42	13.20 8.52	12.11 7.79	11.90 7.50	17.07 12.30	14.80 10.07	14.51 8.79	11.73 6.22	12.45 6.67	15.41 10.65
	Recording 4	12.55 10.65	9.53 7.65	9.47 7.30	9.31 6.45	18.51 18.25	11.66 11.48	9.91 9.60	9.42 8.92	10.96 10.34	12.72 12.06
	Recording 5	12.91 12.30	7.19 6.77	7.71 7.43	7.17 6.62	21.32 13.60	23.21 10.88	8.62 6.63	7.33 6.33	5.97 5.74	17.57 12.29
	Average	15.48 13.03	11.13 8.87	11.54 9.66	10.95 8.49	19.13 15.82	17.49 13.38	13.03 10.75	12.20 10.63	12.49 10.73	16.47 13.31
Average		10.20 9.50	7.69 6.97	7.85 6.87	7.43 6.51	24.90 23.47	20.32 19.24	13.45 13.03	9.84 9.52	7.86 7.36	13.30 12.65
Median		9.91 9.94	7.17 6.79	6.66 6.53	6.60 6.45	18.51 18.25	15.32 14.43	9.90 7.65	7.58 7.09	5.97 5.74	12.72 12.00
Standard deviation		5.00 4.17	3.53 2.40	3.82 2.69	3.51 2.26	15.65 14.39	13.43 13.24	12.87 13.26	7.71 8.07	5.71 5.37	5.67 5.44

Appendix C

Conclusiones

Following the regulations of the University of Zaragoza, this appendix includes a translated version of the conclusions of the thesis presented in section 6.1:

- La comunidad del procesado de señal ha estudiado el problema de la localización de fuentes sonoras durante décadas y las técnicas que fueron propuestas antes del afloramiento de soluciones basadas en *deep learning* no deberían ser ignoradas, ya que aportan representaciones acústicas con una gran correlación con la dirección de llegada de las fuentes sonoras y una mejor comprensión del problema que debería guiar el diseño de las nuevas soluciones basadas en *deep learning*.

- Podemos eliminar el efecto que una fuente en una posición dada genera en las funciones de correlación cruzada de los micrófonos de una agrupación mediante una combinación lineal de las funciones originales con retardos temporales.

- Cuando usamos *datasets* sintéticos para entrenar modelos de localización de fuentes sonoras, podemos generar trayectorias aleatorias ilimitadamente y por tanto no necesitamos limitar nuestros *datasets* a un número finito de trayectorias. Pese a que esto debería usarse con precaución, en los modelos entrenados con esta técnica no se han observado indicios de *over-fitting* al procedimiento de generación de las trayectorias pese a que este era bastante simple.

- Muchas partes del método de las imágenes para la simulación de acústica de salas pueden ser paralelizadas e implementadas eficientemente en GPUs. Esto

es esencial si queremos entrenar nuestros modelos con un *dataset* infinito simulando las trayectorias conforme son necesarias, dado que el método de las imágenes es computacionalmente costoso y su implementación secuencial ralentizaría demasiado los entrenamientos. La implementación propuesta reduce el tiempo de simulación en más de dos órdenes de magnitud comparada con otras implementaciones del estado del arte.

- Los mapas acústicos generados con el algoritmo SRP-PHAT son una representación robusta para ser usados como entrada de redes neuronales. Pese a que la principal fuente de información para la localización de una fuente sonora está en la posición de sus máximos, las redes neuronales son capaces de extraer información adicional de ellos.

- Es posible obtener resultados competitivos en la localización de una única fuente sonora usando únicamente convoluciones causales y evitando las redes recurrentes. Las redes recurrentes bidireccionales deberían ser evitadas si se quiere que los modelos sean útiles en aplicaciones en tiempo real.

- El uso de modelos que explotan la invariancia rotacional del problema de la estimación de la dirección de llegada nos permite reducir el tamaño de nuestros modelos. Múltiples arquitecturas son invariantes al grupo continuo de rotaciones esféricas, pero las convoluciones icosaédricas suponen una buena aproximación con sus 60 simetrías y tienen una implementación eficiente basada en redes convolucionales 2D convencionales.

- La capa soft-argmax propuesta permite transformar una salida de clasificación en una salida de regresión interpretándola como la distribución de probabilidad de la dirección de llegada y calculando su valor esperado. Esto reduce considerablemente el número de parámetros entrenables de los modelos y evita romper su equivariancia a las rotaciones.

- En las estrategias de entrenamiento invariantes a las permutaciones para *tracking* de múltiples fuentes, el contexto temporal que consideramos para elegir la permutación optima de las fuentes estimadas es crucial para reducir el número

de cambios de identidad en las trayectorias estimadas. El uso de una ventana deslizante presenta un buen compromiso entre el alto número de cambios de identidad obtenido cuando se elige una permutación diferente para cada instante temporal y los problemas de convergencia que se dan cuando se intenta entrenar modelos usando una única permutación para cada escena acústica.

- Podemos diseñar capas recurrentes invariantes a las permutaciones que operan sobre conjuntos en lugar de vectores mediante el uso de un módulo *multi-head attention* para asignar cada elemento del set de entrada a los elementos del set de estado y después actualizando cada elemento del conjunto de estado de forma independiente.

- El uso de capas recurrentes invariantes a las permutaciones, como la PI-GRU propuesta, para el tracking de múltiples fuentes es un campo de investigación prometedor, pero aún necesita un mayor estudio para que sea competitivo con las capas GRU convencionales.