Universidad de Alcalá

PhD. Program in Electronics: Advanced Electronic Systems. Intelligent Systems

Acoustic localization of people in reverberant environments using deep learning techniques

PhD. Thesis Presented by

Juan Manuel Vera Díaz

2023

Universidad de Alcalá

PhD. Program in Electronics: Advanced Electronic Systems. Intelligent Systems

Acoustic localization of people in reverberant environments using deep learning techniques

PhD. Thesis Presented by

Juan Manuel Vera Díaz

Advisors

Dr. Javier Macías Guarasa

Dr. Daniel Pizarro Pérez

Alcalá de Henares, June 27th, 2023

A mi familia y a Leti

"Deux ex machina"

Acknowledgements

Aerodynamically, a bumble bee's body is not made to fly;
the good thing is that the bee doesn't know it

Anonymous

Creo que únicamente la gente que ha pasado por el proceso de realizar una tesis doctoral con sus experimentos, sus articulos, su escritura del libro, sus agobios y sus alegrias, puede comprender al 100% lo que significa estar escribiendo unos agradecimientos que cierran una etapa académica completa, y la sensación de satisfacción que ello conlleva.

En primer lugar me gustaría agradecer a mis tutores Daniel Pizarro y Javier Macías el apoyo ofrecido durante todo el proceso de la tesis, ya no tanto en lo académico, si no también en lo personal.

A Dani le agradezco todo el conocimiento que me ha enseñado sobre optimización y redes neuronales durante estos años, así como lo pendiente que ha estado, tanto de Leti como de mi, en todos las aventuras en las que nos hemos metido y por alegrarse de cada paso que hemos ido dando. Sinceramente creo que echaré de menos esas reuniones donde me explotaba la cabeza al escucharte. Espero que nos sigamos viendo aunque sea para tomar una cerveza y celebrar.

A Javi le agradezco también todo el conocimiento aportado sobre localización acústica, latex y bash entre otras cosas. Tengo que reconocerte que muchas de las manías que te he criticado a lo largo de estos años las he acabado viendo útilies y las he añadido a mi cosecha de manías propias. Estoy esperando a ver cuando echamos otro partidillo de pádel para ver cuanto has mejorado.

También me gustaría agradecer del grupo a Marta Marrón por haber sido la primera en darme una oportunidad en algo relacionado con la investigación a la hora de ofrecerme el TFG hace ya unos añitos... y abrirme las puertas del ISPACE. Quiero mencionar también a Frank Sanabría y las horas de "por el poder de la localización acústica!". "Chamaco" sabes que te seguimos debiendo una visita a Tenerife. Así como a Marcos Baptista, gracias por decir siempre las cosas como son y sin anestesia.

A mis padres y mi hermano agradecerles el aguante de soportar a alguien malhumorado mucho tiempo, diciendo que el tema de la tesis es un tema tabú y aún así, tener la

paciencia de seguir aconsejándome y apoyándome. Os pido perdón por ello, lo habéis hecho lo mejor que se podía hacer. Creo que parte de los resultados, tanto de la tesis como de los siguientes pasos dados, son en parte vuestros. También agradecerle a mis suegros que siempre hayan estado dispuestos a aconsejarme y apoyarme.

Y por último Leti, ¿qué nos queda ya por hacer juntos?. A ti agradecerte infinito el animarme cuando tú has necesitado los mismos ánimos que yo. Creo que habría dejado la tesis varias veces si no hubiese sido por tu apoyo. Sólo nosotros dos sabemos lo difícil que ha sido realizar las dos tesis en paralelo y aún así (aunque suene pretencioso y a tí no te guste...) ¡vaya dos pedazo de tesis que nos hemos marcado!. Espero que sigamos avanzando y consiguiendo más metas juntos. Al menos ya estamos empezando a recojer los frutos de esta apuesta que ha sido el doctorado.

Gracias a todos.

Resumen

La localización de las personas a partir de información acústica es cada vez más importante en aplicaciones del mundo real como la seguridad, la vigilancia y la interacción entre personas y robots. En muchos casos, es necesario localizar con precisión personas u objetos en función del sonido que generan, especialmente en entornos ruidosos y reverberantes en los que los métodos de localización tradicionales pueden fallar, o en escenarios en los que los métodos basados en análisis de vídeo no son factibles por no disponer de ese tipo de sensores o por la existencia de oclusiones relevantes.

Por ejemplo, en seguridad y vigilancia, la capacidad de localizar con precisión una fuente de sonido puede ayudar a identificar posibles amenazas o intrusos. En entornos sanitarios, la localización acústica puede utilizarse para controlar los movimientos y actividades de los pacientes, especialmente los que tienen problemas de movilidad. En la interacción entre personas y robots, los robots equipados con capacidades de localización acústica pueden percibir y responder mejor a su entorno, lo que permite interacciones más naturales e intuitivas con los humanos. Por lo tanto, el desarrollo de sistemas de localización acústica precisos y robustos utilizando técnicas avanzadas como el aprendizaje profundo es de gran importancia práctica.

Es por esto que en esta tesis doctoral se aborda dicho problema en tres líneas de investigación fundamentales: *(i)* El diseño de un sistema extremo a extremo (*end-to-end*) basado en redes neuronales capaz de mejorar las tasas de localización de sistemas ya existentes en el estado del arte. *(ii)* El diseño de un sistema capaz de localizar a uno o varios hablantes simultáneos en entornos con características y con geometrías de arrays de sensores diferentes sin necesidad de re-entrenar. *(iii)* El diseño de sistemas capaces de refinar los mapas de potencia acústica necesarios para localizar a las fuentes acústicas para conseguir una mejor localización posterior.

A la hora de evaluar la consecución de dichos objetivos se han utilizado diversas bases de datos realistas con características diferentes, donde las personas involucradas en las escenas pueden actuar sin ningún tipo de restricción. Todos los sistemas propuestos han dis evaluados bajo las mismas condiciones consiguiendo superar en términos de error de localización a los sistemas actuales del estado del arte.

Palabras clave: Localización de Fuentes Acústicas, Steered Response Power, Diferencia de Tiempos de LLegada, Redes Neuronales, Aprendizaje Profundo.

Abstract

Locating people from acoustic information is becoming increasingly important in real-world applications such as security, surveillance, and human-robot interaction. In many cases, there is a need to accurately locate people or objects based on the sound they produce, especially in noisy and reverberant environments where traditional localization methods may fail, or in scenarios where video analytics-based methods are not feasible due to lack of such sensors or relevant occlusions.

For instance, in security and surveillance, the ability to accurately locate a sound source can help identify potential threats or intruders. In healthcare, acoustic localization can be used to monitor the movements and activities of patients, especially those with mobility issues. In human-robot interaction, robots equipped with acoustic localization capabilities can better sense and respond to their environment, enabling more natural and intuitive interactions with humans. Hence, the development of accurate and robust acoustic localization systems using advanced techniques such as deep learning is of great practical importance.

Therefore, this thesis addresses this problem in three fundamental research lines: *(i)* The design of an end-to-end system based on neural networks capable of improving the localization accuracy rates of existing state-of-the-art systems. *(ii)* The design of a system capable of simultaneously localizing one or more speakers in environments with different characteristics and sensor array geometries without the need for retraining. *(iii)* The design of systems capable of refining the acoustic power maps required for acoustic source localization in order to achieve better localization rates later.

In order to evaluate the achievement of these objectives, several realistic databases with different characteristics have been used, where the people involved in the scenes can act without any constraints. All the proposed systems have been evaluated under the same conditions and have outperformed the current state-of-the-art systems in terms of localization error.

Keywords: Acoustic Source Localization, Steered Response Power, Time Diference of Arrival, Neural Networks, Deep Learning.

Contents

List of Figures

List of Tables

List of Acronyms

ADDA	Adversarial Discriminate Domain Adaptation.
ASL	Acoustic Source Localization.
CNN	Convolutional Neural Network.
CRNN	Convolutional Recurrent Neural Network.
DAS	Distributed Acoustic Sensing.
DCASE	Detection and Classification of Acoustic Scenes and Events.
DeepGCC	Deep Generalized Cross-Correlation.
DNN	Deep Neural Network.
DoA	Direction of Arrival.
ESPRIT	Estimation of Signal Parameters via Rotational Invariance Technique.
FFT	Fast Fourier Transform.
GCC	Generalized Cross-Correlation.
GCC-PHAT	Generalized Cross-Correlation with Phase Transform.
GMM	Gaussian Mixture Model.
GMR	Gaussian Mixture Regression.
GRU	Gated Recurrent Unit.
ICA	Independent Component Analysis.
IMU	Inertial Measurement Unit.
LASSO	Least Absolute Shrinkage Selection Operator.
LSTM	Long Sort-Term Memory.

MLP Multi-Layer Perceptron.
MOTP Multiple Object Tracking Precision.
MSE Mean Squared Error.
MUSIC Multiple Signal Clasification.
MVDR Minimum Variance Distortionless Response.

PSD Power Spectral Density.

RIR Room Impulse Response.
RNN Recurrent Neural Network.
RoI Region of Interest.

SRP Steered Response Power.
SVM Support Vector Machine.

TDoA Time Difference of Arrival.

Chapter 1

Introduction

Good ideas are always crazy until they're not.

Elon Musk

These days, data is essential to people's day-to-day lives. The insights that can be extracted from such data are crucial for countless applications and can be worth millions. However, this data must be able to be obtained and processed. When thinking in applications related to the presence of users in given environments, or those that have to do with the interaction between users and their surroundings, specific sensor schemes are required. A so-called *smart room* or *intelligent space* is an environment capable of collecting information through various types of sensors and analyzing it to understand what is happening within it and even to interact with the people inside.

These complex systems comprise various modules that perform tasks such as event detection, people or objects localization, activity detection, people re-identification, speech recognition, and so on, which brings various challenges across different disciplines. As previously said, the environmental information necessary for the tasks described above is extracted by means of sensing, which can be of two kinds: *(i)* invasive sensors, such as Inertial Measurement Unit (IMU) devices, which must be worn by the person or object inside the intelligent space or *(ii)* non-invasive sensors, such as RGB cameras or microphone arrays, so that the information is extracted without the need for the individual or object to carry any device. We are not discussing here issues related to users' privacy considerations, which should be fully addressed in real world deployments.

This thesis focuses on the Acoustic Source Localization (ASL) task, which addresses the estimation of the position of one or several acoustic sources, by exploiting the multichannel signals recorded with a microphone array within a closed environment, such as an office or a home room. In most practical cases, the ASL task can be simplified to estimate the Direction of Arrival (DoA) of the acoustic sources rather than their three-dimensional position. ASL has many different practical applications including source separation [1], speech recognition [2], speech enhancement [3] or human-robot interaction [4].

Although ASL is a long-standing and widely researched topic [5–7], it is still considered an open and challenging problem these days. Traditional ASL methods rely on acoustic propagation models and signal processing techniques to find the sources' position. Even though they have shown remarkable advances in this field over the years, they perform poorly in realistic scenarios where noise, reverberation, and signal multi-path effects exist. Recently, deep learning approaches have appeared to solve the ASL posed as a regression problem, showing advantages over the traditional methods in complex environments [8].

Both, traditional and Deep Neural Network (DNN)-based methods roughly follow a simple workflow shown in Figure 1.1. A multi-channel input signal is recorded with one or several microphone arrays and is then processed by a feature extraction module, which provides the input features. Those input features are passed to an ASL model to estimate the acoustic source position.

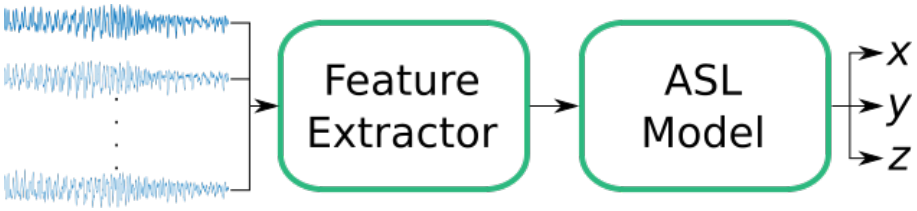


Figure 1.1: General workflow of an ASL system.

Under free-field conditions, the multi-channel signals only differ up to a time delay and amplitude variation, representing enough information to estimate the source position. However, in realistic environments, signals are contaminated with noise and multi-path distortion, and thus the problem becomes considerably more complex. Learning-based techniques usually outperform traditional methods in these complex scenarios since they can tell the valuable information that correlates with the source position from the noise and distortion. In contrast, conventional methods often suffer from over-simplistic assumptions and mathematical models in order to be tractable. However, the lack of generality is the major drawback of the DNN-based approaches. A learned model designed for and trained in a specific environment configuration will not provide accurate localization results if the set-up changes. Effective domain adaptation methods are still an open problem in deep learning in general.

1.1 Motivation and Objectives

In this thesis we develop ASL systems capable of improving the most consolidated systems in the state-of-the-art. We focus on reverberant environments where speakers can move and speak without any kind of restriction. Our datasets contain recordings of meetings or lectures, or interactions of users freely moving in the monitored environments while speaking. Furthermore, we explore methods capable to cope with several simultaneously active speakers.

Accordingly, we set three major objectives for this thesis:

1. **Improving the acoustic source localisation accuracy of state-of-the-art systems**. One of the main challenges of the ASL task is to obtain a position estimate in a reverberant environment. This is a significant challenge as these environments suffer from the effects of noise and multipath, which contaminate the signals used and make it difficult to obtain a correct estimate. Several methods address this problem using different techniques, but this thesis aims to develop a system based on deep learning that is able to achieve better results than other state-of-the-art methods.

2. **Addressing the problem of domain independence in ASL systems based on deep learning techniques**. As in many other areas of machine learning research, the accuracy of the systems developed is highly dependent on the data they are trained on. A variation in the distribution of the inference data can cause a system with good accuracy rates in the training or validation phase to start working incorrectly. Furthermore, in the specific case of ASL tasks, a change in the geometry of the environment, as well as that of the array of sensors that collect the information, can also affect the performance of the system, since the propagation of the signals changes and so does the way in which they reach the microphones. Part of the thesis therefore aims to develop a system that is independent of the data it has been trained on and the geometry of the room in which it operates.

3. **Improving source localisation through acoustic power map refinement techniques**. One of the main problems when estimating the position of speakers within an environment, given an acoustic power map, is that this map is very dense and it is difficult to determine which local maximum corresponds to the acoustic source. As a final objective of this thesis, it is proposed to implement machine-learning based techniques that are able to represent these maps in a sparse way, facilitating the estimation of the acoustic sources within them.

1.2 Summary of Contributions

Briefly stated these are the main contributions of the PhD. Thesis, related to the objetives described above:

1. We propose a new learning-based method that is able to obtain better localization scores than the existing state-of-the-art, comprising both classical and learning-based methods. For this purpose, we use a Convolutional Neural Network (CNN) model that takes as input the raw signal collected by the microphones and estimates the (x, y, z) position of a single acoustic source, which is not frequent in the literature as most studies focus on Direction of Arrival (DoA) estimation. This model is trained

with synthetic data and is then fine tuned with real data using different domain adaptation strategies.

2. We propose a system capable of addressing the problem of domain independence. For this purpose, we propose a CNN encoder-decoder network that takes as input the Generalized Cross-Correlations (GCCs) of the microphone signals and returns a domain-independent version of them. We then use a classical approach to construct acoustic power maps describing the acoustic activity of the environment and where the acoustic sources are easily obtained as local minima. With this approach we obtain better results in locating acoustic sources in multi-domain environments.

3. We investigate two acoustic map refinement techniques. One of them is based on optimization techniques and the other one is based on learning techniques. These methods allow us to obtain a cleaner version of the acoustic power maps previously calculated with our methods so that we can obtain more accurate acoustic source locations.

1.3 Publications

Every result obtained from the objectives set for this thesis, described before, has been published in the following research articles:

- **Towards End-to-End Acoustic Localization using Deep Learning: from Audio Signal to Source Position Coordinates**
  Sensors
  October 2018
  cites: 101
  DOI: 10.3390/s18103418
  JCR: *Q2*

- **Towards Domain Independence in CNN-based Acoustic Localization using Deep Cross Correlations**
  IEEE EUSIPCO
  January 2021
  cites: 4
  DOI: 10.23919/Eusipco47968.2020.9287466
  JCR: —

- **Acoustic Source Localization with Deep Generalized Cross Correlations**
  Elsevier Signal Processing
  May 2021
  cites: 9

DOI: 10.1016/j.sigpro.2021.108169
JCR: *Q1*

1.4 Book Organization

This thesis is organized in seven chapters:

1. **Introduction**: (Current chapter) it describes the definition of the ASL task and the context, motivation and objectives of the thesis. Additionally, it summarizes the Thesis contributions to the state-of-the-art and define the mathematical notation to be used for the rest of chapters.

2. **State-of-the-art**: it provides a thorough review of the most relevant works in ASL that we divided into two major groups, referred to as conventional methods and learning-based methods.

3. **Datasets**: it describes four different datasets used to train and evaluate the ASL methods. In this chapter we provide all the relevant details of these databases, including characteristics such as the dimensions of the room where the events take place, the configuration of the microphone arrays, the description of the available sequences or the total number of hours or minutes of data available.

4. **Acoustic Source Localization from audio signal waveforms**: it addresses the first research line proposed in this thesis, where we design an end-to-end system capable of giving estimates of the (x, y, z) position of an acoustic source from the raw audio signals as the result of the first defined contribution. This system is compared both with conventional and learning-based methods.

5. **Acoustic Source Localization from deep cross correlations**: it describes the second research line concerning the model domain independence in ASL, where we define the concept of Deep Generalized Cross-Correlation (DeepGCC) signals and apply them to build acoustic power maps. This is framed in the second defined contribution. Here we show that our method can be used in various environments with different characteristics without retraining the neural network and while maintaining performance.

6. **Acoustic map refinement techniques**: it addresses the third research line that arises from building acoustic power maps from correlation signals, which results are the last of the contributions of this thesis. Usually these maps are contaminated by multipath and reverberation effects. In this chapter we propose two different approaches to refine these map to improve the localization of the acoustic sources.

7. **Conclusions and future lines**: it provides a summary of all the work achieved during the thesis period with some global conclusions, as well as some possible research lines on which to continue working in the future.

Chapter 2

State of the art

Insanity: doing the same thing over and over again and expecting different results.

Albert Einstein

2.1 Introduction

Despite a large number of papers published in recent years, Acoustic Source Localization (ASL) is considered an unresolved problem with new publications every year. In this section, we review the state-of-the-art in ASL to motivate the proposals of this thesis. Existing ASL methods can be divided into two major groups: *(i)* conventional methods and *(ii)* learning-based methods.

2.2 Conventional Methods

Conventional methods exploit the acoustic properties of the recorded signals and use mathematical models and optimization techniques to estimate the positions of one or several acoustic sources. Four different groups can be distinguished in this category: *(i)* Time Difference of Arrival (TDoA) methods *(ii)* Beamforming methods *(iii)* Subspace methods and *(iv)* Mixtures based methods

2.2.1 TDoA methods

When the geometry of the sensor array is known, and for the particular case of $3D$ space, source position estimation can be performed by computing the TDoAs of the sources between at least three different pairs of microphones [9,10] followed by hyperbolic trilateration [11–13]. This process involves solving a system of quadratic equations with iterative optimization methods.

One of the most popular techniques to estimate the TDoA is the Generalized Cross-Correlation with Phase Transform (GCC-PHAT) [14], which is computed as the inverse Fourier transform of a weighted cross-spectrum between the signals of two microphones. The TDoA is usually estimated as the position of the maximum value of the GCC-PHAT between the signals.

TDoA estimations degrade with noise and signal multipath effects, which translates to poor localization accuracy. Recent proposals improve localization accuracy by exploiting redundancy in the estimated TDoA set [15] and the geometrical limitations imposed by the sensor array [16]. Nevertheless, in practical scenarios TDoA methods show a lower localization accuracy than other state-of-the-art solutions, being extremely inefficient when facing multi-source scenarios.

2.2.2 Beamforming Methods

Building an acoustic power map or an acoustic energy map [17] is another common strategy to obtain the position of one or multiple acoustic sources. These approaches are based on estimating the acoustic power at a discrete set of potential source positions by combining the signals received at the microphones.

The most common approach is the Steered Response Power (SRP) algorithm [18–23] due to its simplicity and high performance in noisy scenarios, usually combined with the Phase Transform (PHAT). The acoustic energy is estimated from the GCC-PHAT between signals received at pairs of sensors that are time-shifted to steer the beamformer at each potential source position. The accumulated energy at each source position from all microphone pairs generates the SRP map. Those points where the SRP map becomes locally maximal are the estimations of the acoustic source positions.

Minimum Variance Distortionless Response (MVDR) [24] is another popular beamforming method. It assumes that every captured signal is the addition of an interest signal, such as human speech, and an interference signal that includes reverberations, noise, or interference from other simultaneous acoustic sources. In this case, the beamformer is steered by adjusting a weight factor that is obtained through an analysis of the interference signal Power Spectral Density (PSD). Proper estimation of the interference PSD is crucial to perform accurate localization with this method, which is also a challenging task.

Methods based on beamforming are more robust than those relying on TDoA estimation. However, their localization accuracy degrades due to noise, signal multipath and those properties inherited from the GCC-PHAT signals, such as the mismatch between the TDoA and the location of the maximum in the correlation signals. In order to improve the source localization accuracy, some methods [25, 26] impose sparse constraints and use generative models to improve the SRP map, assuming that only a reduced number of simultaneous active sources are expected in a scene.

2.2.3 Subspace Methods

Subspace methods assume that the signal recorded by the microphones can be decomposed as a weighted sum of basis vectors. These basis vectors are obtained through the eigenvalue decomposition of a covariance matrix obtained from the microphone signals. One of the most popular subspace methods is Multiple Signal Clasification (MUSIC) [27], where the basis decomposition is used to steer a beamformer at different positions to estimate the presence of a source [28].

The Estimation of Signal Parameters via Rotational Invariance Technique (ESPRIT) algorithm [29] avoids the beamformer steering process by exploiting the structure of the source subspace to infer the source Direction of Arrival (DoA) end-to-end. However, it often produces more inaccurate estimations than MUSIC [30].

Both algorithms assume environments where narrowband signals are emitted, although there are some proposals that incorporate wideband signals [31,32]. In any case, subspace methods are robust to noise and are able to produce reliable predictions, but they are highly sensitive to reverberation effects.

2.2.4 Mixtures Based Methods

The methods based on mixtures use probabilistic generative models [33–38] to describe the signals. Typically, those models are extensions or variants of Gaussian Mixture Models (GMMs), using a single Gaussian component per acoustic source. The parameters of these models are estimated by the expectation-maximization (EM) algorithms and exploit the sparsity of sound sources in the time-frequency domain [39].

Another mixture method is the Gaussian Mixture Regression (GMR), which can be used for both single and multiple source scenarios [40, 41]. GMR methods are locally linear but globally non-linear and the parameters are fit on training data, in a similar manner to learning-based ASL methods. These methods generalize well to speech signals which are sparser than noise in the time-frequency domain.

Finally, the Independent Component Analysis (ICA) algorithm is originally designed to separate different signal sources from a mixture signal, also known as blind source separation, by exploiting their mutual statistical independence. This method has been applied for multisource ASL tasks [42] by analyzing the signals from different sources independently and using a parallel single source localization method [43, 44].

Mixture methods are used in ASL systems to deal with multiple source environments, and are usually combined with TDoA and beamforming methods to solve the ASL task. They thus inherit the limitations and benefits of these previous algorithms.

2.3 Learning-based Methods

In the last decade, several works have been proposed that learn to solve the ASL problem from data. We can distinguish between shallow learning methods, such as Support Vector Machines (SVMs) and Deep Neural Networks (DNNs) methods, following their rise in almost every scientific and technological application [45, 46].

In the shallow learning category we highlight [47], where they use SVMs to learn signal features that improve beamforming algorithms, and [48], where they use SVM vectors as subspaces to build acoustic space mapping systems. The scope of these early learning-based methods is limited and they show a strong dependence on the data used for training.

The deep learning category is by far the most populated, and can be organized as a function of the neural architecture used to address the ASL task: *(i)* Multi-Layer Perceptron (MLP) networks, *(ii)* Convolutional Neural Networks (CNNs), *(iii)* Residual Convolutional Neural Networks (CNNs), *(iv)* Convolutional Recurrent Neural Networks (CRNNs), *(v)* encoder-decoder networks and *(vi)* Self-attention networks. We describe them next in detail.

2.3.1 Multi-layer Perceptron Networks

MLP networks are composed of several fully-connected neuron layers with non-linear activation functions. *Kim et al.* [49] proposed in 2011 one of the first MLP architectures for ASL. They used the mono-pulse beamforming signals received in a four-element sensor array to estimate the Direction of Arrivals (DoAs) of up to 5 different static acoustic sources. The model was trained under synthetic anechoic conditions (assuming direct-path sound propagation only) and tested in realistic scenarios.

Tsuzuki et al. [50] proposed in 2013 a MLP-based method to estimate the azimuth of a single static source from the time delay of the received signals and assuming anechoic conditions. In 2015, *Ma et al.* [51] also proposed a system that retrieved the azimuth of a maximum of three static sources from the signals' cross-correlations. The source space of solutions is discrete, which allowed them to use a classifier network. This method was tested in realistic environments. A similar method based on cross-correlations was proposed in 2017 by *Yiwere et al.* [52] to obtain the azimuth and distance coordinates of a single static source in a realistic environment. The approach also uses a classifier for a discretized space of source positions.

Takeda et al. [53–56] published a series of works between 2016 and 2018, describing one of the most consolidated MLP based methods. They use the MLP to infer the eigenvectors of the correlation matrix obtained from the microphone signals. They used these estimations with the MUSIC algorithm to achieve a better azimuth localization of a static source. In [54] they extended the work to perform a simultaneous two-source localization, whereas in [55] they retrieved the source's elevation and the azimuth. The

proposed system was developed for a real environment where the space was discretized to enable a classification approach.

Some other methods rely on GCC-PHAT signals as the neural inputs. In 2015, *Xiao et al.* [57] followed a classification approach to estimate the azimuth of a single static source using these signals captured inside reverberant rooms. *Vesperini et al.* [58] proposed in 2016 a regression method that retrieved the $2D$ Cartesian coordinates (x, y) of a static acoustic source using as input the GCC-PHAT signals also within reverberant scenarios. Finally, *Gelderblom et al.* [59] presented in 2021 a regression MLP model capable of estimating the azimuth of up to two active static sources in reverberant environments.

2.3.2 Convolutional Neural Networks

CNNs, which are based on various designs of convolutional layers, have been successfully applied to several tasks, such as image classification, natural language processing, or automatic speech recognition. ASL systems based on these architectures are commonly used nowadays.

Hirvonen et al. [60] proposed the first CNN-based ASL system in 2015. They used the magnitude component of the signals' spectrum to estimate the DoA of a static acoustic source through a classification approach. This system works in reverberant environments and has been trained with synthetic signals.

In 2017, *Chakrabarty et al.* proposed a classification CNN that used the spectrum's phase component of the captured signals to retrieve the azimuth of one or several static sources. In [61], they used this approach to single source localization, whereas in [62], they extended the system to localize two sources. In 2019, they improved the multi-source ASL system to achieve a better localization accuracy in [63, 64].

He et al. [65] proposed in 2018 a classification CNN that used the GCC-PHAT signals to estimate the azimuth of a maximum of two sources that are static inside a reverberant room. In the same year, *Vecchiotti et al.* [66] estimated the $2D$ Cartesian coordinates of a single static source using as inputs the GCC-PHAT signals and the Mel spectrogram captured in a reverberant environment. This system accuracy was further improved [67] in 2019. Finally, in 2018, *Vera-Diaz et al.* [68] introduced an ASL system that retrieved end-to-end the $3D$ Cartesian coordinates (x, y, z) of a moving acoustic source from the raw audio waveform. This system was tested in realistic scenarios and is explained in chapter 4.

In 2019, *Chytas et al.* [69] developed a CNN-based system that estimated the DoA of a static source from the signal waveform captured within reverberant environments. Also, *Vecchiotti et al.* [70] proposed a classification CNN that used the waveform signals to estimate the azimuth of a single static source inside a realistic scenario. Similarly, *Zhang et al.* [71] used the phase component of the signals' spectrum to achieve the same

goal. They used clear speech recordings convolved with Room Impulse Responses (RIRs) in order to generate those synthetic reverberant signals later used as inputs.

In 2020, *Hübner et al.* [72] used the phase component of the frequency spectrum to estimate the azimuth of a single static source. In 2021, *Bologni et al.* [73] proposed a classification architecture to retrieve the azimuth and the distance of a static source from the microphone array by using as input the signals' waveform. *Vargas et al.* [74] used the phase component of the signal spectrum as neural inputs and estimated the azimuth of the static source. Finally, *Diaz-Guerra et al.* [75] used the SRP acoustic map to estimate the $3D$ Cartesian coordinates of a single moving source, whereas *He et al.* [76] used the signals waveform. Finally, in 2022 *SongGong et al.* [77] proposed a CNN-based method to better estimate DoA of a single speaker by using new circular harmonic features that are frequency invariant as the input to the model. This approach is trained with synthetically generated data and further tested with real recordings.

2.3.3 Residual Convolutional Neural Networks

Residual CNNs are composed of convolutional layers with residual connections. These connections are designed to enable a feature to bypass a layer block. This allows the gradients to flow through the network, leading to a better convergence and enabling deeper networks.

To our knowledge, the first use of a neural network with residual connections for ASL was proposed by *Yalta et al.* [78] in 2017. They use a classification residual CNN that takes the magnitude of the signal frequency spectrum to estimate the azimuth of a single static source inside a reverberant environment.

In 2018, *Suvorov et al.* [79] developed a system that returns the azimuth of a static acoustic source from the raw signals recorded in a reverberant room. They approached this task as a classification problem. Also, *He et al.* [80] proposed a similar system where they supplied the recorded signals spectrum to estimate the position of up to three static sources. Further, they extended this work in 2021 to detect a maximum of four simultaneous static sources [81].

A residual CNN that retrieved the $2D$ position of a static acoustic source from the waveform of the recorded signals was proposed by *Pujol et al.* [82] in 2019. Those signals were synthetically generated and virtually propagated along a reverberant room. *Ranjan et al.* [83] presented a model which used the log-Mel spectrum to classify the possible DoA of a static source inside reverberant scenarios.

Finally, in the year 2020, *Shimada et al.* [84] proposed a similar approach but using the spectrogram's magnitude component of the captured signals for regressing the moving single-source DoA instead. In the same year, *Sundar et al.* [85] estimated the azimuth and the distance of a maximum of three moving sources just from the recorded raw signals.

This work was able to train the model under synthetic conditions and then the trained models were used to properly carry out locatization inside realistic scenarios.

2.3.4 Convolutional Recurrent Neural Networks

Recurrent Neural Networks (RNNs) model data sequences over time. Particular types of RNNs include Long Sort-Term Memory (LSTM) cells [86] and Gated Recurrent Units (GRUs) [87]. They have become prevalent in the deep learning literature due to their ability to avoid the training issues that regular RNN have, such as the vanishing gradient problem.

Convolutional Recurrent Neural Networks (CRNNs) are specific neural networks that contain one or more convolutional layers and one or more recurrent layers. CRNNs have been widely used for ASL tasks since 2018 because of the characteristics of these layers: *(i)* convolutional layers are suitable to extract relevant features and *(ii)* recurrent layers exploit those features over time.

In 2018 and 2019, *Adavanne et al.* [88–90] developed GRU-based recurrent network for a sound event localization and multi-task detection named as `SELDNet`. They used the magnitude and phase spectrogram of synthetically generated signals to estimate the DoA of a maximum of three simultaneous moving sources and categorized them into different acoustic events. This system was the baseline for task 3 of the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge in 2019 [91], 2020 [92] and 2021 [93]. Therefore, it has been used as a baseline system in many other works, with various modifications, extensions, and improvements. Also, in 2018 *Li et al.* [94] proposed a LSTM-based CRNN that approached an azimuth estimation of a single static source as a classification problem in noisy and reverberant environments, using as input the GCC-PHAT signals over the time.

In 2019, *Kapka et al.* [95] presented a modification of *Adavanne et al.* [90], where two static sources DoAs could be estimated inside reverberant scenarios. A similar approach was followed by *Zhang et al.* [96] where they used the Mel-spectrogram to retrieve the DoA of a single static acoustic source. *Cao et al.* [97] also proposed an approach based on `SELDNet` in 2020. They retrieved the DoA of two moving sources along with their class from the raw waveform.

Finally, in 2021, several CRNN-based systems were proposed. Among them, *Bohlender et al.* [98], and *Subramanian et al.* [99] followed a similar strategy to estimate the azimuth of up to three or two static sources respectively, approaching it as a classification problem from the phase spectrogram of the recorded signals. *Guirguis et al.* [100] developed a network that used as inputs the spectrogram of the captured signals to estimate the DoA of a single moving source.

2.3.5 Encoder-Decoder Networks

Encoder-decoder neural networks are networks which follow a specific topology. They are made of two blocks: *(i)* an encoder, which is fed by input features, and outputs a representation of the input data, known as the latent space, and *(ii)* a decoder, which transforms the latent space into the desired output data. These types of models can be made of different kinds of layers, such as convolutional layers or dense fully-connected layers.

The `U-Net` architecture is a particular fully-convolutional neural network originally proposed in [101] for biomedical image segmentation. This network has been proven to be also suitable for ASL estimations. The `V-Net` model [102] is also another fully-convolutional encoder-decoder network for biological image segmentation. These topologies allow information to directly propagate from the encoder to the decoder via skip connections, thus improving estimations.

In 2019, *Chazan et al.* [1] proposed an `U-Net` based network that estimates the azimuth, using a classification approach, of a maximum of two static sources. The network is trained using the spectrogram of synthetically generated signals. Further, they used the estimated azimuths to perform a signal source separation task.

Several works were proposed in 2020 that include encoder-decoder architectures. *Comanducci et al.* [103] modified an `U-Net` model to retrieve the DoA of a single static source through the GCC-PHAT of the captured signals. *Patel et al.* [104] approached a $3D$ Cartesian coordinates localization of a single static source by using as input the GCC-PHAT signals over time and the log-Mel spectrogram. They included GRUs in the `U-Net` scheme to allow temporal information to be exploited. *Jenrungrot et al.* [105] used an `U-Net` topology to estimate the azimuths of up to eight moving sources using the raw waveforms of the captured signals. *Huang et al.* [106] proposed a method that used the synthetically generated raw signals to estimate the azimuth of a single static source. *Le Moing et al.* [107] retrieved the $2D$ coordinates of a maximum of three static sources from the signals spectrogram with a similar model. Finally, in 2021 *Vera-Diaz et al.* [108, 109] proposed a encoder-decoder based system that returned the $3D$ position x, y, z of a single moving source from the GCC-PHAT of the recorded signals, achieving high level of domain invariance in terms of environment and sensor array geometry. This work is presented in chapter 5.

2.3.6 Self-Attention Networks

An attention mechanism is a modern technique which allows neural networks to put emphasis on relevant vectors of a temporal sequence for a given task. *Bahdanau et al.* [110] originally proposed an attention-based method to improve sequence-to-sequence models for machine translation. The general principle is to allocate a different weight to the

vectors of the input sequence when using a combination of those vectors for estimating a vector of the output sequence. The model is trained to compute the optimal weights that reflect both the link between vectors of the input sequence (self-attention) and the relevance of the input vectors to explain each output vector (attention at the decoder). The popular `Transformer` architecture [111] is inspired by this technique. Nowadays, attention models are popular in recent deep learning proposals, including those applied in ASL tasks.

In 2021, *Cao et al.* [112] presented a modification for the CRNN-based model `SELDNet` where they included a self-attention module at the end of the network to improve the estimations of two moving DoA estimations by using as inputs the log-Mel spectrogram. *Wang et al.* [113] followed a similar goal. They used a recent method for speech recognition named as `Conformer` [114] and a residual CNN to determine the DoA of a single moving source through the Mel-spectrogram along with the GCC-PHAT signals over time.

2.3.7 Deep-Learning Based Methods Comparison

Table 2.1 compares all the methods mentioned above, where we summarize the most relevant deep learning-based systems that address the ASL task along the last ten years. We show the year of each proposal, the type of network used (Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), Residual Convolutional Neural Network (Res. CNN), Convolutional Recurrent Neural Network (CRNN), Encoder decoder networks (Enc-Dec) or self attention networks (Self Att.)), the kind of approach the authors followed (Regression (R) or Classification (C)), the input fed to the network, the output given, the maximum number of sources the system can simultaneously locate, whether those sources can move or not (✔ or ✘), if the system works within real or synthetic environments (✔ or ✘) and if the system can operate in reverberant or anechoic scenarios (✔ or ✘).

As presented in Table 2.1, in the last ten years, systems have evolved from simpler MLP models to more complex self-attention schemes. The most popular way to retrieve the localization of sources is through a classification approach. Nevertheless, regression approaches are recently becoming more and more popular. The azimuths of the sources are the location coordinates retrieved by most authors, followed by the DoA. Only a few methods give the sources' $2D$ position (x, y), while only our proposals can estimate the $3D$ position (x, y, z) of the sources.

Either single-speaker or multi-speaker approaches have been widely studied during the past ten years. However, only since 2018 moving speakers have been taken into account. One of our proposals was one of the first to address this task, along with very few other works that year.

Finally, systems are designed to perform within simulated scenarios as well as real-world environments. In the former, clear speech recordings are propagated along the

room according to propagation models. These models tend to include reverberation and noise effects in order generate more realistic signals.

In both of them, recordings could have different constraints as a maximum number of simultaneous speakers, whether the speakers have no movement restrictions or they have to consider some limitations as if they should always face the sensors or not, or if they should be standing still or not.

2.4 Notation

We include here the general mathematical notation used along the document.

Real scalar values are represented by lowercase letters (e.g. α, c). Vectors are by default arranged column-wise and are represented by lowercase bold letters (e.g. $\mathbf{x}$). Matrices are represented by uppercase bold letters (e.g. $\mathbf{M}$). Upper-case letters are reserved to define vector and set sizes (e.g. vector $\mathbf{x} = (x_1, \ldots, x_N)^\top$ is of size N), and $\mathbf{x}^\top$ denotes transpose of vector $\mathbf{x}$. Calligraphic fonts are reserved to represent sets (e.g., $\mathbb{R}$ for the set of real numbers or generic sets such as $\mathcal{G}$.) The l_p norm ($p > 0$) of a vector is depicted as $\|.\|_p$, e.g., $\|\mathbf{x}\|_p = (|x_1|^p + \cdots + |x_N|^p)^{\frac{1}{p}}$, where $|.|$ is reserved to represent absolute values of scalars, or the module operation for complex values. The l_2 norm $\|.\|_2$ (euclidean distance) will be written by default as $\|.\|$ for simplicity. The Discrete Fourier transform of a discrete signal $x[n]$ is represented with complex function $X[\omega]$, with $X^*[\omega]$ being the complex-conjugate of $X[\omega]$. We refer to $\lfloor . \rfloor$, $\lceil . \rceil$ and $\lceil . \rfloor$ as the floor, ceiling and nearest rounding operators respectively.

Proposal	Year	Network	Type	Input	Output	# S	Movement	Real	Reverb
[49]	2011	MLP	R	Beamforming signals	DoA	5	✘	✔	✔
[50]	2013	MLP	R	Signals time dalay	θ	1	✘	✔	✘
[60]	2015	CNN	C	Magnitude spectro	DoA	1	✘	✘	✔
[51]	2015	MLP	C	Cross-correlations	θ	3	✘	✔	✔
[57]	2015	MLP	C	GCC-PHAT	θ	1	✘	✔	✔
[53]	2016	MLP	C	R-eigenvectors	θ	1	✘	✔	✔
[54]	2016	MLP	C	R-eigenvectors	θ	2	✘	✔	✔
[58]	2016	MLP	R	GCC-PHAT	x, y	1	✘	✔	✔
[61]	2017	CNN	C	Phase spectro	θ	1	✘	✔	✔
[62]	2017	CNN	C	Phase spectro	θ	2	✘	✘	✔
[55]	2017	MLP	C	R-eigenvectors	DoA	1	✘	✔	✔
[78]	2017	Res. CNN	C	Magnitude spectro	θ	1	✘	✔	✔
[52]	2017	MLP	C	Cross-correlations	θ, r	1	✘	✔	✔
[88]	2018	CRNN	C	Spectro	DoA	3	✘	✘	✔
[65]	2018	CNN	C	GCC-PHAT	θ	2	✘	✔	✔
[80]	2018	Res. CNN	C	Spectro	θ	3	✘	✔	✔
[94]	2018	CRNN	C	GCC-PHAT	θ	1	✘	✘	✔
[79]	2018	Res. CNN	C	Waveform	θ	1	✘	✔	✔
[56]	2018	MLP	C	R-egeinvectors	θ	1	✘	✔	✔
[66]	2018	CNN	R	GCC-PHAT+Mel spectro	x, y	1	✘	✔	✔
[68]*	2018	CNN	R	Waveform	x, y, z	1	✔	✔	✔
[89]	2019	CRNN	R	Spectrogram	DoA	3	✘	✔	✔
[90]	2019	CRNN	R	Spectrogram	DoA	3	✔	✔	✔
[63]	2019	CNN	C	Phase spectro	θ	2	✘	✘	✔
[64]	2019	CNN	C	Phase spectro	θ	2	✘	✘	✔
[1]	2019	Enc-Dec	C	Spectrogram	θ	2	✘	✘	✔
[69]	2019	CNN	R	Waveform	DoA	1	✘	✔	✔
[95]	2019	CRNN	R	Spectro	DoA	2	✘	✔	✔
[82]	2019	Res. CNN	R	Waveform	x, y	1	✘	✘	✔
[83]	2019	Res. CNN	C	Log-mel spectro	DoA	1	✘	✔	✔
[67]	2019	CNN	R	GCC-PHAT+Mel spectro	x, y	1	✘	✔	✔
[70]	2019	CNN	C	Waveform	θ	1	✘	✔	✔
[71]	2019	CNN	C	Phase spectro	θ	1	✘	✘	✔
[96]	2019	CRNN	R	Spectrogram	DoA	1	✘	✔	✔
[97]	2020	CRNN	R	Waveform	DoA	2	✔	✔	✔
[103]	2020	Enc-Dec	C	GCC-PHAT	DoA	1	✘	✔	✔
[106]	2020	Enc-Dec	R	Waveform	θ	1	✘	✘	✘
[72]	2020	CNN	C	Phase spectro	θ	1	✘	✔	✔
[105]	2020	Enc-Dec	R	Waveform	θ	8	✔	✔	✔
[107]	2020	Enc-Dec	C	Spectro	x, y	3	✘	✔	✔
[104]	2020	Enc-Dec	R	GCC-PHAT+Log-mel spectrogram	DoA	1	✔	✔	✔
[84]	2020	Res. CNN	R	Magnitude spectro	DoA	1	✔	✔	✔
[85]	2020	Res. CNN	R	Waveform	θ, r	3	✔	✔	✔
[98]	2021	CRNN	C	Phase spectro	θ	3	✘	✔	✔
[73]	2021	CNN	C	Waveform	θ, r	1	✘	✘	✔
[112]	2021	Self Att.	R	Log-mel Spectro	DoA	2	✔	✔	✔
[75]	2021	CNN	R	SRP map	DoA	1	✔	✔	✔
[59]	2021	MLP	R	GCC-PHAT	θ	2	✘	✔	✔
[59]	2021	CRNN	R	Spectro	DoA	1	✔	✔	✔
[81]	2021	Res. CNN	C	Spectro	θ	4	✘	✔	✔
[76]	2021	CNN	R	Waveform	DoA	1	✔	✔	✔
[99]	2021	CRNN	C	Phase spectro	θ	2	✘	✘	✔
[74]	2021	CNN	C	Phase spectro	θ	1	✘	✔	✔
[108]*	2021	Enc-Dec	R	GCC-PHAT	x, y, z	1	✔	✔	✔
[109]*	2021	Enc-Dec	R	GCC-PHAT	x, y, z	1	✔	✔	✔
[113]	2021	Self Att.	R	GCC-PHAT+Mel spectro	DoA	1	✔	✔	✔
[77]	2022	CNN	C	Circular Hatmonic Features	DoA	1	✘	✔	✔

Table 2.1: State-of-the-art deep learning based methods comparison (systems proposed in this thesis dissertation are marked with $[\cdot]^*$).

Chapter 3

Datasets

Data! Data Data! I can't make bricks without clay!.

Arthur Conan Doyle

3.1 Introduction

Adequate data availability is essential in order to develop robust learning-based systems. These data is used to train and evaluate the models in specific tasks. Data is usually stored as datasets where at least two elements are required: (i) input data and (ii) target data. The former is the data which learning-based models have to learn from. Typically, input data stored in datasets is not used itself but a preprocessed version of it. The latter is the output value that the models are meant to estimate and return. These outputs can be defined as categorical outputs (e.g. models have to predict a class or some classes over a set of them), or non-categorical outputs (e.g. models have to predict an exact position of a speaker).

One of the key challenges in acoustic localization is dealing with the wide variety of acoustical and geometrical conditions that can be encountered in real-world environments. To address this challenge, we used datasets that were recorded under a range of different conditions, including various microphone array geometries, reverberation times, and room geometries. We also included datasets with varying numbers of sound sources and microphone arrays, so that they were representative of real-world scenarios, comprising a comprehensive set of acoustical and geometrical conditions,

In this section we describe the used datasets that were made available for the scientific community. In further sections we will also describe some the dataset we synthetically generate fro some specific proposals.

3.2 Albayzin Phonetic Corpus

The *Albayzin Phonetic Corpus* [115] consists of three sub-corpora of audio files captured at $16kHz$ with 16 bits resolution, and recorded by 304 Castilian Spanish speakers in a professional recording studio using high quality, close-talk microphones.

Each of the sub-corpora available in the database has the following features:

- ***Phonetic corpus***: This corpus is composed of utterances of phonetically balanced sentences. It is divided in a train and a test set.
  - *Training set*: Composed of 200 phonetically balanced sentences, where 4 speakers have been recorded saying the full set of sentences and 160 speakers have been recorded uttering a subset of 25 phonetically balanced sentences. Thus, 24 utterances of each sentence are obtained, which means 4800 different recordings.
  - *Testing set*: Composed of 500 phonetically balanced sentences, where 40 speakers have been recorded saying a group of 50 sentences. This results in 4 utterances of each sentence, or 2000 recordings.
- ***Application corpus***: Composed by 3900 sentences, the corpus is divided in 2700 sentences for training and 1200 sentenes for testing. This corpus is divided in 78 subsets of 50 sentences each and is uttered by 136 speakers.
- ***Lombard corpus***: This database is a subset of the application corpus which is utterd by 40 speakers. During the recordings, a noise source is applied to produce the Lombard effect.

We used this dataset the in proposal described in Chapter 4 to train the end-to-end network in order to directly estimate speaker positions from acoustic waveforms. From the three sub-corpora, we only used the so-called *phonetic corpus* due to the fact that it uses the same number of male and female speakers, as well as for being the one that contains the most different recordings. We have not proceeded to merge the three corpora because of the different nature of the recordings provided.

The *phonetic corpus* signals are used to simulate recordings with the same microphone array geometries that those used in the real data. The simulations included noise addition and anechoic propagation for random source positions.

3.3 AV16.3 dataset

The AV16.3 dataset [116] is an audio-visual dataset recorded in the *Smart Meeting Room* of the *IDIAP* research institute. As it can be shown in Figure 3.1, the room consists of a $3.6m \times 8.2m \times 2.4m$ rectangular space containing a centrally located $4.8m \times 1.2m$

rectangular table. It features two circular microphone arrays of radius $10cm$, each of them composed of 8 microphones, placed on top of the table. The two arrays centers are separated by $80cm$, and the room's coordinate origin is located in the middle point between the two arrays.

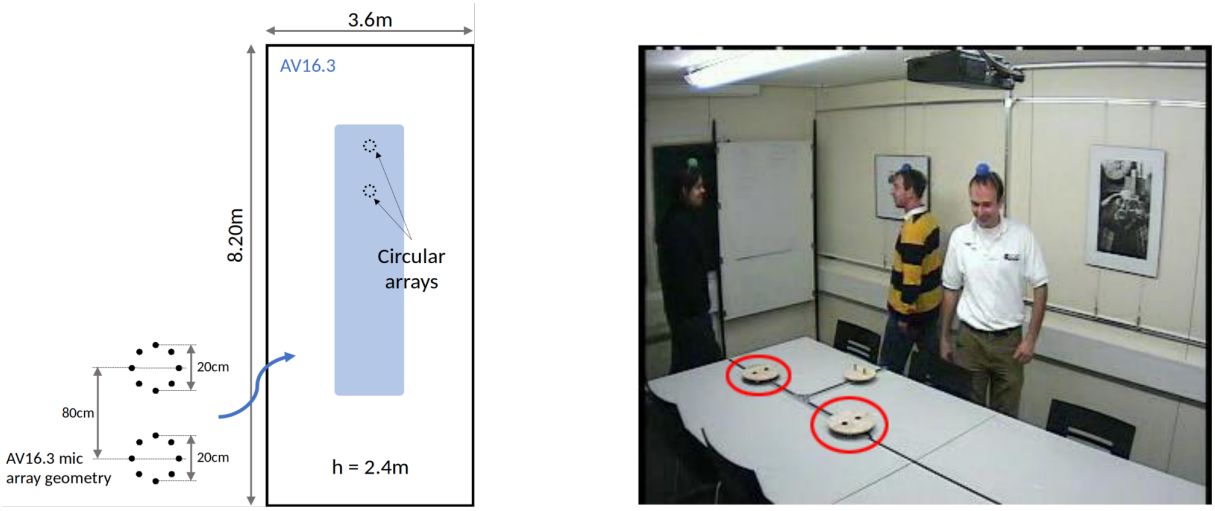


Figure 3.1: AV16.3 dataset environment. a) room layout. b) environment picture.

Sequences	Features	Time length
Aseq01	Male speaking at 16 different static locations, always facing the microphone array.	$3m$ $44s$
Aseq02	Female speaking at 16 different static locations, always facing the microphone array.	$3m$ $09s$
Aseq03	Male speaking at 16 different static locations, always facing the microphone array.	$4m$ $02s$
Aseq11	Male moving while speaking always facing the microphone array	$32s$
Aseq15	Male moving while speaking unconstrained alternating long periods of silence.	$36s$
Total		$12m$ $03s$

Table 3.1: Description of the AV16.3 dataset sequences.

The room is also equipped with 3 video cameras providing different angle views. The audio signals are recorded at a sampling frequency of 16 kHz, and this dataset also provides the speakers mouth ground-truth location. Table 3.1 shows the sequences that are later used in Chapters 4, 5 and 6.

3.4 CAV3D Dataset

The *CAV3D* (Co-located Audio-Visual streams with $3D$ tracks) dataset [117] is a people localization multi-modal database released in 2019 by the Center for Information and Communication Technology *ICT* in the Foundazione Bruno Kessler. *CAV3D* dataset was collected using a sensing platform consisting of a monocular color camera co-located with

an 8-element circular microphone array (uniformly distributed). The sensing platform was placed on a 0.72 cm high table in a room with dimensions $4.77m \times 5.95m \times 4.5m$ as it is shown in Figure 3.2.

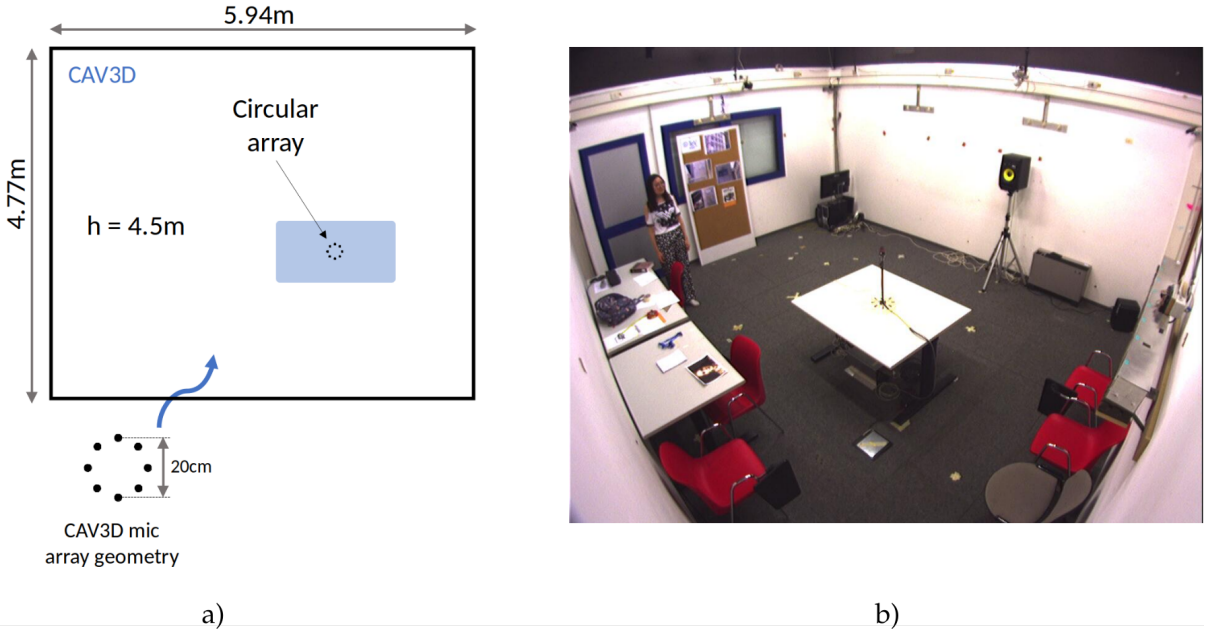


Figure 3.2: *CAV3D* dataset environment. (a) room layout. (b) environment picture.

The audio sampling rate is $96kHz$, and video was recorded at 15 frames per second. Sequences showed in Table 3.2 are used to evaluate our proposals described in Chapters 5 and 6.

3.5 CHIL-CLEAR Dataset

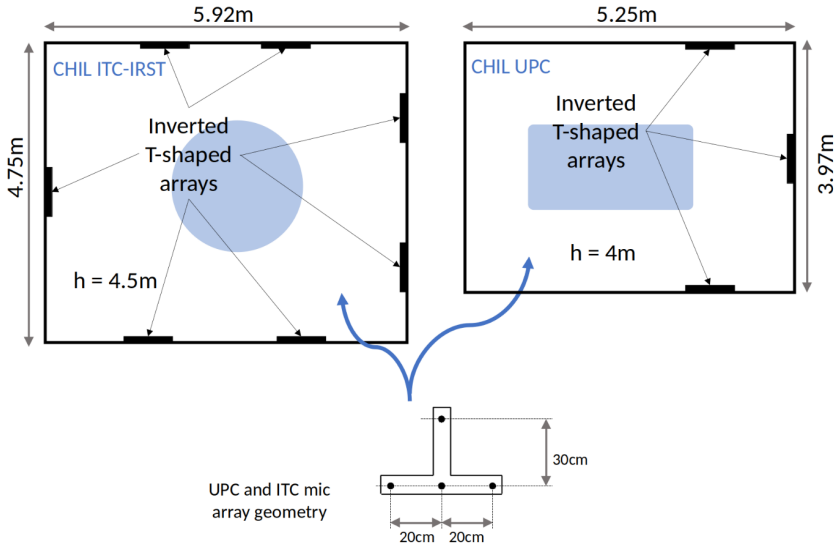
The *CHIL* (Computers in the Human Interaction Loop) project [118] generated a rich set of datasets, which were later used in several competitions under the *CLEAR* name. The latest one, organized in 2007 [119], assembled several tracks. In this Thesis we have used the "*lecture sessions*" subset, where a person is speaking in front of a sometimes numerous audience. Specifically, there are recordings from 5 different rooms (*RESIT-AIT*, *IBM*, *ITC-IRST*, *UKA-ISL*, and UPC) where 10 minutes of recording have been collected for 4 different sessions per room.

The *CLEAR* evaluations explicitly defined the datasets partitions into development and testing subsets. In our work we focus on the UPC and *ITC-IRST* datasets, as they provide two extreme scenarios, shown in Figure 3.3, with different number of microphone arrays, 3 and 7, respectively. The UPC recording room ($3.97m \times 5.25m \times 4.00m$ in size) was equipped with 3 inverted T shaped 4-microphone arrays. The *ITC-IRST* recording room ($4.75m \times 5.92m \times 4.50m$) was equipped with 7 inverted T shaped 4-microphone arrays. In both cases the microphone arrays were located on the walls at a height of $2.38m$.

Sequences	Features	Time length
Cseq06	Female moving along a reduced Region of Interest (RoI) inside the room.	52s
Cseq07	Male moving along a reduced RoI inside the room.	59s
Cseq08	Male moving along a reduced RoI inside the room.	1m 10s
Cseq09	Male moving along a reduced RoI inside the room with clapping/noising situations	50s
Cseq10	Male moving along a reduced RoI inside the room with clapping/noising situations.	50s
Cseq11	Male moving along a reduced RoI inside the room with clapping/noising and bending/sitting situations.	1m 10s
Cseq12	Male moving along a reduced RoI inside the room with clapping/noising and bending/sitting situations.	1m 29s
Cseq13	Male moving along the whole room.	1m 25s
Cseq20	Male moving along a reduced RoI inside the room.	46s
Cseq21	Female moving along a reduced RoI inside the room with bending/sitting situations.	46s
Cseq22	A female and a male moving along the whole room simultaneously speaking.	39s
Cseq23	A female and a male moving along the whole room simultaneously speaking.	1m 04s
Cseq24	A female and a male moving along the whole room simultaneously speaking.	1m 09s
Cseq25	Two females and a male moving along the whole room simultaneously speaking.	1m 02s
Cseq26	Two females and a male moving along the whole room simultaneously speaking.	36s
Total		15m 51s

Table 3.2: CAV3D dataset sequences descriptions.

Table 3.3 summarizes the sequences we used for the evaluation tasks described in Chapter 5. The whole dataset is recorded at 44.1kHz where all the speakers in the sequences moved without restrictions and the listeners were supposed not to speak at anytime.



a)

b)

Figure 3.3: *CHIL-CLEAR* dataset environment. a) room layout (left UPC, right ITC). b) environment picture (left UPC, right ITC).

***UPC* Sequences**	**Time length**	***ITC* Sequences**	**Time length**
UPCdev	$22m\ 56s$	ITCdev	$30m\ 36s$
UPC01A	$5m\ 00s$	ITC01A	$5m\ 07s$
UPC01B	$5m\ 21s$	ITC01B	$5m\ 14s$
UPC02A	$5m\ 04s$	ITC02A	$5m\ 11s$
UPC02B	$5m\ 05s$	ITC02B	$5m\ 03s$
UPC03A	$5m\ 09s$	ITC03A	$5m\ 05s$
UPC03B	$5m\ 14s$	ITC03B	$5m\ 05s$
UPC04A	$5m\ 02s$	ITC04A	$5m\ 27s$
UPC04B	$5m\ 27s$	ITC04B	$5m\ 06s$
Total	$1h\ 4m\ 18s$	**Total**	$1h\ 11m\ 54s$

Table 3.3: CHIL dataset sequences descriptions.

Chapter 4

Acoustic Source Localization from Audio Signal Waveforms

Have the courage to follow your heart and intuition...

Steve Jobs

4.1 Summary

This chapter presents the work published in the research paper entitled "*Towards End-to-End Acoustic Localization Using Deep Learning: From Audio Signals to Source Position Coordinates*" [68]. It presents a novel indoor acoustic source localization approach using microphone arrays based on a Convolutional Neural Network (CNN).

In the proposed solution, the CNN is designed to directly estimate the $3D$ position of a single acoustic source using the raw audio signals as the input information and without using hand-crafted audio features.

Given the limited amount of available localization data, we proposed a training strategy based on two steps:

1. We first train our network using semi-synthetic data from close talk speech recordings. For the synthetic audio generation, we simulate the time delays, and the distortion suffered in the signal that propagates from the source to the array of microphones.

2. We then fine-tune this network using a small amount of real data.

Our experimental results, evaluated on a publicly available dataset recorded in a real room (AV16.3 dataset), show that our approach significantly improved existing localization methods based on Steered Response Power (SRP) strategies and also coexisting proposals

based on Convolutional Recurrent Neural Networks (CRNNs). In addition, our experiments show that the performance of our CNN-based method does not show a relevant dependency on the speaker's gender or the signal window size.

4.2 Introduction

The development of advanced perceptual systems has notably grown during the last decades and has experienced a tremendous rise in recent years due to the availability of increasingly sophisticated sensors, the use of computing nodes with increasing computational power, and the development of powerful algorithmic strategies based on deep learning (all of them entering the mass consumer market). Perceptual systems aim to automatically analyze complex and rich information from sensors to obtain refined information, such as human activities, on the sensed environment. Scientific works in these environments cover from sensor technologies to signal processing and pattern recognition. Also, they open the pathway to systems being able to analyze human activities, providing us with advanced interaction capabilities and services. In this scope, the localization of humans (being the most interesting element for perceptual systems) is a fundamental task so that the systems can provide higher-level information on human activities. Without precise localization, further advanced interactions between humans and their physical environment cannot be fulfilled successfully. The scientific community has devoted much effort to building robust and reliable indoor localization systems relying on different sensors [120–122]. Non-invasive technologies are preferred in this context, so no electronic or passive devices need to be carried by humans for localization. The two non-invasive technologies that have been mainly used in indoor localization are those based on video systems and acoustic sensors.

This chapter focuses on audio-based localization from unknown wide-band audio sources (e.g., the human voice) captured by a set of microphone arrays placed in known positions. The objective of this work is to directly use the raw signals captured by the microphone arrays to automatically obtain the position of the acoustic source detected in the environment. Even though there have been several proposals in this area, Acoustic Source Localization (ASL) is still a hot research topic. Thus, we propose a CNN architecture that is trained end-to-end to solve the acoustic localization problem. Our CNN takes the raw signals captured by the microphones as input and yields the $3D$ position of the acoustic source as its output. The idea of using neural networks for sound processing is not new and has recently gained popularity (especially for speech recognition [123]). In the context of ASL, deep learning methods have been recently developed [55,60,62,65,78,88,124–127]. Most of these works focus on obtaining the Direction of Arrival (DoA) of the acoustic source. They also feed the network with feature vectors extracted from the audio signals.

This proposal was the first in the literature that directly used the speech signal as input and aims to estimate the source position coordinates in the room in $3D$ space. Avoiding hand-crafted features has increased the accuracy of classification and regression methods based on CNNs in other fields, such as computer vision. We evaluate our method using both semi-synthetic and real data. It outperforms traditional solutions based on SRP, and also shows better results than a recent proposal based on a CRNN [88].

The contributions to this research line are as follows:

- We have designed a method capable of determining the position of an acoustic source within a reverberant environment from the acoustic signal obtained by the sensors.
- We can pre-train the network with synthetically generated data in such a way that the model learns general aspects of the task to be solved. Subsequently, we can fine tune the network so that it adapts to a specific scenario.
- Our proposal is robust against gender variations of speakers, as well as against variations in the window size used to analyze the data.

4.3 Problem Statement

Our system obtains the position of a single acoustic source from the audio signals captured by an array of M microphones located at $3D$ coordinates $\mathbf{m} = (m_{i,x}, m_{i,y}, m_{i,z})^\top$, with $i = 0, \ldots, M-1$, with respect to a reference coordinate origin. In the same coordinate system, the source $3D$ position is defined as $\mathbf{r} = (r_x, r_y, r_z)^\top$. The signal emitted by this source is received by the M microphones. The signal captured by the i^{th} microphone is modeled as follows:

$$x_i(t) = \alpha_i(t)\, s(t - \tau_i) + \eta_i(t) \tag{4.1}$$

where $s(t)$ is the signal emitted at the source position $\mathbf{r}$, $\eta_i(t)$ is the additive noise effects between the position $\mathbf{r}$ and the microphone $\mathbf{m}_i$, $\alpha_i(t)$ is the reverberation due to multipath effects added to the source signal and τ_i is the time distance between the acoustic source $\mathbf{r}$ and the i^{th} microphone, which is defined as:

$$\tau_i = \frac{\|\mathbf{m}_i - \mathbf{r}\|}{c} \tag{4.2}$$

with c as the sound air propagation velocity with a nominal value of $343 m/s$. Typically, $x_i(t)$ is discretized with a sampling frequency f_s and is defined as $x_i[n]$. For simplicity, we assume that $x_i[n]$ is of finite length with N samples, which corresponds to a small audio window with a duration of $w_s = N/f_s$. The window length w_s is a design parameter in our system.

The objective in this work is to find a regression function that obtains the speaker's position given the signals recorded from the microphones as follows:

$$\hat{\mathbf{r}} = f\left(x_0\left[n\right], \ldots, x_{M-1}\left[n\right], \mathbf{m}_1, \ldots, \mathbf{m}_{M-1}\right) \tag{4.3}$$

In classical approaches f is found by assuming that signals received from different microphones mostly differ by a delay that directly depends on the relative position of the source with respect to the microphones. However, this assumption does not hold in environments where the signal is severely affected by the effects of reverberation due to multi-path propagation and the presence of diffuse and ambient noise. Given the aforementioned effects and the random nature of the audio signal, the regression function of Equation (4.3) is unknown. We present an approach to describe f as a CNN whose parameters are trained end-to-end from the microphone signals. In our system, we assume that the microphone positions are fixed during training. We drop them from the arguments of Equation (4.3) and they are implicitly learned by our network. We thus find the following ASL regression function:

$$\hat{\mathbf{r}} = f\left(x_0\left[n\right], \ldots, x_{M-1}\left[n\right]\right) \tag{4.4}$$

4.4 Model Topology

The topology of our neural network is shown in Figure 4.1. It is based on two phases: filter-and-sum enhancement by means of $1D$ convolutional *FIR* filters, followed by a standard fully connected network. We believe that this architecture is well suited for audio analysis, especially when the window size is fixed, as it is in our case.

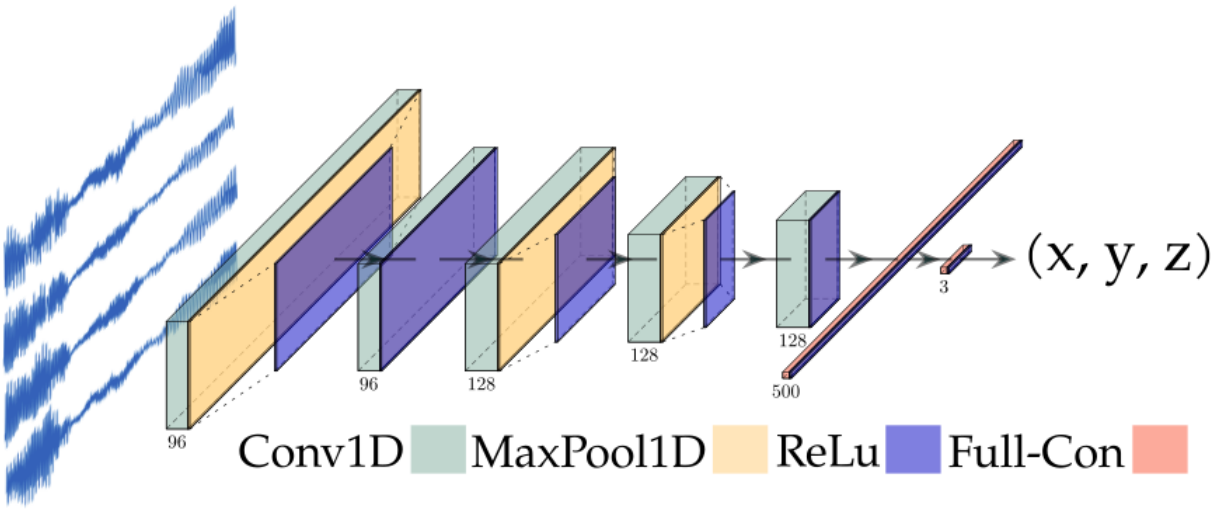


Figure 4.1: `ASLNet` network topology.

The network is composed of five one-dimensional, convolutional blocks and two fully connected blocks. In accordance with Equation (4.4), the network inputs are the set of windowed signals from the microphones, and the network output is the estimated position of the acoustic source. Table 4.1 shows the details of each layer in the proposed network topology.

ID	Layer	Kernel size	Input size	Output size
Layer 1	*Conv1D*	7	$L \times C$	$L \times 96$
Layer 2	*MaxPooling1D*	—	$L \times 96$	${}^{L}/_{7} \times 96$
Layer 3	*ReLu*	—	${}^{L}/_{7} \times 96$	${}^{L}/_{7} \times 96$
Layer 4	*Conv1D*	7	${}^{L}/_{7} \times 96$	${}^{L}/_{7} \times 96$
Layer 5	*ReLu*	—	${}^{L}/_{7} \times 96$	${}^{L}/_{7} \times 96$
Layer 6	*Conv1D*	5	${}^{L}/_{7} \times 96$	${}^{L}/_{7} \times 128$
Layer 7	*MaxPooling1D*	—	${}^{L}/_{7} \times 128$	${}^{L}/_{35} \times 128$
Layer 8	*ReLu*	—	${}^{L}/_{35} \times 128$	${}^{L}/_{35} \times 128$
Layer 9	*Conv1D*	5	${}^{L}/_{35} \times 128$	${}^{L}/_{35} \times 128$
Layer 10	*MaxPooling1D*	—	${}^{L}/_{35} \times 128$	${}^{L}/_{175} \times 128$
Layer 11	*ReLu*	—	${}^{L}/_{175} \times 128$	${}^{L}/_{175} \times 128$
Layer 12	*Conv1D*	3	${}^{L}/_{175} \times 128$	${}^{L}/_{175} \times 128$
Layer 13	*ReLu*	—	${}^{L}/_{175} \times 128$	${}^{L}/_{175} \times 128$
Layer 14	*FC*	—	${}^{128L}/_{175}$	500
Layer 15	*ReLu*	—	500	500
Layer 16	*FC*	—	500	3
Layer 17	*ReLu*	—	3	3

Table 4.1: `ASLNet` layers summary.

Filters of size 7 (layers 1 and 4), size 5 (layers 6 and 9) and size 3 (layer 12) are used. The number of filters is 96 in the first two convolutional layers and 128 in the rest. As seen in Figure 4.1, some of the layers are equipped with *MaxPooling* filters with the same pool size as their corresponding convolutional filters. The last two layers are fully-connected layers, one hidden with 500 nodes and the output layer. The activation functions of all layers are *ReLUs*. During training, we included a dropout with a probability of 0.5 in the fully-connected layers to prevent over-fitting.

4.5 Experimental Work

In this section, we describe the general conditions of the experimental setup to evaluate the proposed method, the training strategy and the error metrics used for comparing our proposal with other state-of-the-art algorithms, and our experimental results.

4.5.1 Experimental Setup

We used a simple microphone array configuration to evaluate our proposal in a resource-restricted environment, with the same configuration than that used in [25]. In order to

do so, we used four microphones (referenced as 1, 5, 11, and 15, out of the 16 available in the AV16.3 dataset), grouped in two microphone pairs. This configuration of four microphones is the same as that selected in [25] to provide two orthogonal microphone pairs. The selected microphone pair configurations are shown in Figure 4.2, in which microphones of the same microphone pair share the same color. We evaluated results for acoustic frame lengths of $80ms$, $160ms$, and $320ms$ to accurately assess the extent to which improvements were consistent with different acoustic temporal resolutions.

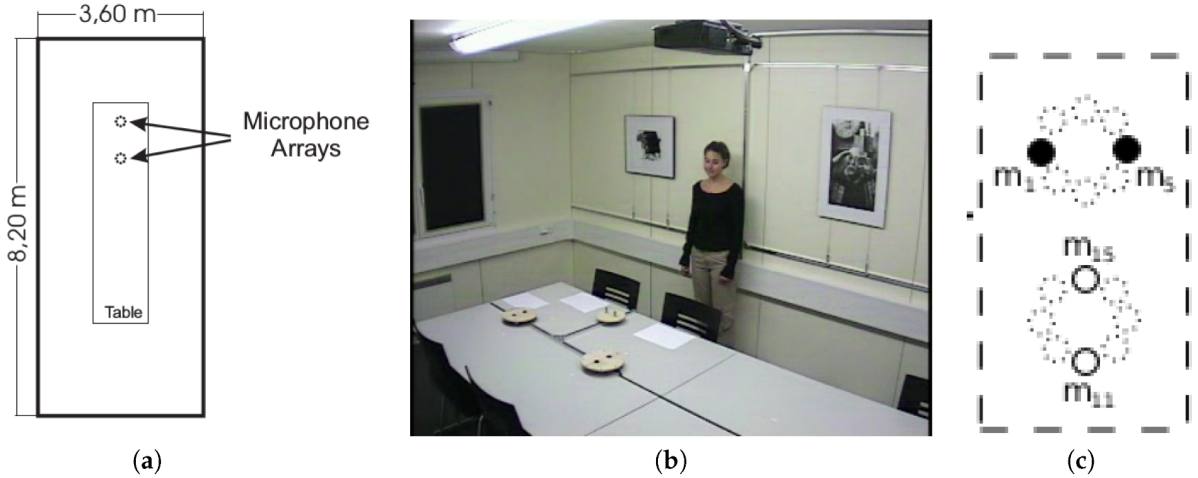


Figure 4.2: (a) Simplified top view of the *IDIAP Smart Meeting Room*; (b) a real picture of the room extracted from a video frame; (c) microphone setup used in this proposal.

The main interest of our experimental work was to assess whether the end-to-end CNN based strategy (that we will refer to as `ASLNet`) could be competitive with other traditional localization methods. We compared the `ASLNet` approach with the standard SRP method and the strategy proposed in [25] that we refer to as `GMBF`. `GMBF` is based on fitting a generative model to the GCC-PHAT signals using sparse constraints, and it was associated with significant improvements over SRP in the *IDIAP* dataset. The `GMBF` fitting procedure does not require training, as opposed to the CNN approach. We also compare our method with another ASL strategy based on a CRNN [88], with a similar scope. We define four different experiments, that we briefly summarize here:

- **Experiment 1**: Baseline results. We provide the results of the comparison between `SRP`, `GMBF`, and our proposal `ASLNet` without applying the fine tuning procedure.
- **Experiment 2**: We evaluate the performance improvements when using a single moving speaker sequence for the fine tuning procedure.
- **Experiment 3**: We evaluate the impact of adding an additional moving speaker sequence for the fine tuning procedure.
- **Experiment 4**: We evaluate the final performance improvements when also adding static sequences to the refinement process.

Table 4.2 shows the train and test sequence partitions for each experiment. Note that Exp 4.1, Exp 4.2 and Exp 4.3 are relative to experiment 4 where all the sequences, moving and static, are used for fine tuning the model except the one used for testing.

Experiment	Train seqs	Fine-tune seqs	Test seqs
Exp 1	*Albayzin Corpus*	—	Aseq01, Aseq02, Aseq03
Exp 2	*Albayzin Corpus*	Aseq15	Aseq01, Aseq02, Aseq03
Exp 3	*Albayzin Corpus*	Aseq11, Aseq15	Aseq01, Aseq02, Aseq03
Exp 4.1	*Albayzin Corpus*	Aseq11, Aseq15, Aseq02, Aseq03	Aseq01
Exp 4.2	*Albayzin Corpus*	Aseq11, Aseq15, Aseq01, Aseq03	Aseq02
Exp 4.3	*Albayzin Corpus*	Aseq11, Aseq15, Aseq01, Aseq02	Aseq03

Table 4.2: Training and testing sequences used in the evaluation of `ASLNet`.

After these experiments, we evaluate the differences between the semi-synthetic training in addition of the fine tuning approach versus just training the network from scratch, to assess the actual contribution of the fine tuning strategy. Finally, we provide a comparison between our proposal and that described in [88].

4.5.2 Training Strategy

We use the *Albayzin Phonetic Corpus* and the AV16.3 dataset to train the proposed model, described in Sections 3.2 and 3.3, respectively. Due to the reduced amount of labeled single speaker real data included in the AV16.3 dataset, we propose a training strategy comprising two steps:

1. Semi-Synthetic Dataset Generation: The network is trained with semi-synthetic data. We use close-talk speech recordings to generate simulated versions of the signals captured by a set of microphones from a set of randomly generated source positions. The microphones have the same geometry as in the real data recordings. We also take into account additional considerations on the acoustic behavior of the target environment (specific noise types, noise levels, etc.) to generate the data. A dataset of this type can virtually be made as big as required to train a network.

2. Fine Tuning Procedure: The network is fine-tuned with real data. The network is trained on a reduced subset of the database captured in the target physical environment using the weights obtained in Step 1 for initialization.

4.5.2.1 Semi-Synthetic Dataset Generation

In this step, audio signals are extracted from any available close-talk (anechoic) corpus and used to generate semi-synthetic data. There are many available datasets that are suitable for this task (freely or commercially distributed). Our semi-synthetic dataset can thus be made as big as required to train the CNN.

For this task, $N_{\mathcal{Q}}$ position vectors are randomly generated as $\mathbf{q}_i = (q_{i,x}, q_{i,y}, q_{i,z})^\top \in \mathcal{Q}$ with $i = 1, \ldots, N_{\mathcal{Q}}$, using a uniform distribution that covers the whole physical space (room) used. In order to realistically simulate the signals received in the microphones from a given source position, we have to consider two main issues:

- **The acoustic noise conditions of the room and the recording process conditions**: These can result from additional equipment (computers, fans, air conditioning systems, etc.) present in the room, and from problems in the signal acquisition setup. They can be addressed by assuming additive noise conditions and selecting the noise type and acoustic effects that should be preferably estimated in the target room.
- **Signal propagation considerations**: This is affected by the impulse response of the target room. Different alternatives can be used to simulate this effect, such as convolving the anechoic signals with real room impulse responses [54], which can be difficult to acquire for general positions in big environments, or by using room response simulation methods, such as the image method [128, 129].

In our case, and regarding the first issue, we simulated noise and disturbances in the signals arriving to the microphones so that the signal-to-noise ratio and the spectral content of the signals were as similar as possible to those found in real data.

Electrical noise usually appears within realistic environments recordings. These effects are due to the equipment used in the recording process beside the fans, computers and other devices that are located inside the room while recording. Typically frequencies for these tones actually varied in a range between 20 Hz and 30 Hz. So, in the synthetic data generation process, we contaminated the signals from the phonetic corpus with an additive tone of a random frequency in this established range, and we also added white Gaussian noise in accordance with the expression:

$$x_{ct_{\text{new}}}[n] = x_{ct}[n] + k_e sin(2\pi f_0 n/f_s + \phi_0) + k_\eta \eta[n] \tag{4.5}$$

where $x_{ct}[n]$ is a windowed signal from the close-talk dataset, k_e is a scaling factor for the electrical noise, set up to 0.1 in our case, f_s is the sampling frequency used, $f_0 \in [20, 30]$ Hz, $\phi_0 \in [0, \pi]$ rad, $\eta[n]$ is a white Gaussian noise signal and k_η is a noise scaling factor.

Regarding the second issue, we use the simplest approach as our initial alternative, just taking into account the propagation delay from the source position to each of the microphones which depends on their relative positions and the speed of sound in the room. Our simulation model does not consider other effects, such as reverberation of the signals in the room or other environmental noise conditions. We thus do not require more specific knowledge about the room, such as the positions and materials of the walls and furniture.

The delay from an acoustic source to the i^{th} microphone in samples is calculated as $N_{s_i} = c^{-1} f_s d_i$ where d_i is the Euclidean distance between them and c is the speed of sound

in air ($c = 343m/s$ in a room at 20^oC). In general, N_{s_i} is not an integer number. Thus, a method to simulate sub-sample shifts in the signal is required. In order to implement the delay from N_{s_i} on $x_{ct}[n]$ to obtain $x_i[n]$, the following transformation is used:

$$\begin{cases} \mathbf{X}_{ct_{\text{new}}}[k] = \mathcal{F}\{x_{ct_{\text{new}}}[n]\} \\ \mathbf{D}_{s_i}[k] = e^{-jk\frac{2\pi N_{s_i}}{N_k}} \\ x_i[n] = A_i\left(\mathcal{F}^{-1}\{\mathbf{X}_{ct_{\text{new}}}[k] \odot \mathbf{D}_{s_i}[k]\}\right) \end{cases} \quad \text{with } k = 0, \ldots, N_k - 1 \tag{4.6}$$

where $\mathbf{X}_{ct_{\text{new}}}[k]$ is the k^{th} element of the Fast Fourier Transform (FFT) of N_k frequency bins of the $x_{ct_{\text{new}}}(t)$ time signal by using the operator $\mathcal{F}\{\cdot\}$, $\mathbf{D}_{s_i}[k]$ is the frequency phase shifting vector according to the N_{s_i} samples delay, $\odot$ is the element-wise product and A_i is an amplitude factor that is applied to the signal that follows an uniform random distribution, and it is different for each microphone ($A_i \in [0.01, 0.03]$ in our case). We used random amplitudes because we explicitly wanted the network to focus on phase or time-delay differences between the microphones. It was intended that these random amplitudes would take away the effects of the directionality of the microphones, and this is so because we assumed that they had omnidirectional responses

4.5.2.2 Training and Fine Tuning Details

Our proposal is fitted with the geometrical configuration of the *IDIAP Smart Meeting Room* where AV16.3 was recorded (See section 3.3). For this work we use the whole room space but just a reduced number of available microphones as shown in figure 4.2, with two orthogonal microphone pairs.

Accordingly, in order to generate the semi-synthetic dataset, the $N_{\mathcal{Q}}$ random positions $\mathbf{q}_i$ were uniformly distributed in the following intervals: $q_{i,x} \in [0, 3.6]\,m$, $q_{i,y} \in [0, 8.2]\,m$ and $q_{i,z} \in [0.92, 1.53]\,m$, which correspond to the possible distributions of the speaker's mouth positions in the *IDIAP* room.

Regarding the optimization loss function, we used the mean squared error between the estimated position estimated by the model ($\hat{\mathbf{r}}_i$) and the target position ($\mathbf{q}_i$) in order to train the proposed network. This function is denoted as:

$$\mathcal{L}(\Theta) = \frac{1}{N_{\mathcal{Q}}} \sum_{i=1}^{N_{\mathcal{Q}}} \|\mathbf{q}_i - \hat{\mathbf{r}}_i\|^2 \tag{4.7}$$

In order to minimize Equation (4.7), we employ the *ADAM* optimizer [130] (variant of *SGD* optimizer with a variable learning rate) along 200 epochs with a batch size of 100 samples. The learning rate of the *ADAM* optimizer is fixed at $\alpha = 10^{-3}$, and the other parameters are set with the recommended values ($\beta_1 = 0.9$, $\beta_2 = 0.99$ and $\epsilon = 10^{-8}$). A total of 7200 different frames of input data per epoch are randomly generated during the

training phase, and another 800 are generated for validation, as it is described above in section 4.5.2.1.

We study the impact of different window lengths in the ASLNet input. We train three different models, once per window length (80ms, 160ms and 320ms). In each training session, 200 audio close-talk recordings from *Albayzin Phonetic Corpus* are randomly chosen and 40 different windows are randomly extracted from each. In the same way, 200 acoustic source position vectors ($\mathbf{q}_i$) are randomly generated, so each position generates 40 windows.

For the fine tuning procedure we use AV16.3 sequences never used in testing phase, mainly focusing on those in which speakers are moving. We leave 20% of acoustic positions for fine tuning the model as validation data. The *ADAM* optimizer was also used for fine tuning. In this case, we fixed the learning rate at $\alpha = 10^{-4}$, while the rest of the parameters were set to the recommended values.

4.5.3 Evaluation Metrics

ASLNet yield a set of spatial coordinates $\hat{\mathbf{r}}_k = (\hat{r}_{k,x}, \hat{r}_{k,y}, \hat{r}_{k,z})^\top$ that are estimations of the current speaker's position at the k^{th} acoustic frame. These position estimates are compared, by means of the Euclidean distance, to the ones labeled in the ground truth file containing the real positions, $\mathbf{r}_k$ (ground truth), of the speaker. We evaluated performance by adopting the same metric used in [25] and developed under the *CHIL* project [119], which is known as Multiple Object Tracking Precision (MOTP) and is defined as:

$$\text{MOTP} = \frac{\sum_{k=1}^{N_\mathcal{P}} \|\mathbf{r}_k - \hat{\mathbf{r}}_k\|}{N_\mathcal{P}} \quad [m] \tag{4.8}$$

where $N_\mathcal{P}$ denotes the total number of position estimations along time. We also compared the results of ASLNet, GMBF and SRP by measuring the relative improvement in MOTP as compared to SRP, which is defined as follows:

$$\Delta_r^{\text{MOTP}} = 100\,\frac{\text{MOTP}_{\text{SRP}} - \text{MOTP}_{\text{proposal}}}{\text{MOTP}_{\text{SRP}}} \quad [\%] \tag{4.9}$$

4.5.4 Baseline Results

The baseline results for sequences Aseq01, Aseq02 and Aseq03 are shown in Table 4.3 as well as the evaluated time window sizes (in all the tables showing results in this section, **bold font** highlights the best ones for a given data sequence and window length). The table shows the results achieved by the SRP standard algorithm strategy (column SRP), the alternative described in [25] (column GMBF), and the ASLNet proposal without applying

the fine tuning procedure (column ASLNet). We also show the relative improvements of GMBF and ASLNet as compared with SRP.

	80ms			160ms			320ms		
	SRP	GMBF	ASLNet	SRP	GMBF	ASLNet	SRP	GMBF	ASLNet
Aseq01 MOTP	1.020	**0.795**	1.615	0.910	**0.686**	1.526	0.830	**0.588**	1.464
Aseq01 Δ_r^{MOTP}		**22.1**%	-58.3%		**24.6**%	-67.7%		**29.1**%	-76.4%
Aseq02 MOTP	0.960	**0.864**	2.124	0.840	**0.759**	1.508	0.770	**0.694**	1.318
Aseq02 Δ_r^{MOTP}		**10.0**%	-121.3%		**9.6**%	-79.5%		**9.9**%	-71.2%
Aseq03 MOTP	0.900	**0.686**	1.559	0.770	**0.563**	1.419	0.690	**0.484**	1.379
Aseq03 Δ_r^{MOTP}		**23.8**%	-73.2%		**26.9**%	-84.3%		**29.9**%	-99.9%
Average MOTP	0.957	**0.778**	1.763	0.836	**0.666**	1.481	0.760	**0.585**	1.385
Average Δ_r^{MOTP}		**18.7**%	-84.3%		**20.4**%	-77.1%		**22.9**%	-82.3%

Table 4.3: Baseline results for the SRP strategy, the GMBF method and the CNN trained with synthetic data without applying the fine tuning procedure (column ASLNet) for sequences Aseq01, Aseq02 and Aseq03 for different window sizes. Relative improvements compared to SRP are shown below the MOTP values.

From the baseline results the main conclusions are as follows:

- The MOTP values improved as the window length increased, as expected, given that better correlation values will be estimated for longer window signal lengths. The best MOTP values for the standard SRP algorithm were around 69cm, and for the GMBF, around 48cm.
- The average MOTP value for the standard SRP algorithm was between 76cm and 96cm, and for the GMBF, it was between 59cm and 78cm.
- The GMBF strategy, as described in [25], achieves very relevant improvements compared with SRP, with average relative improvements of around 20% and peak improvement values of almost 30%.
- The ASLNet strategy, which, at this point, is only trained with semi-synthetic data, was shown to be very far from reaching SRP or GMBF in terms of performance. This result leads us to think that there are other effects that are only present in real data such as reverberation that are affecting the network, as they have not been properly modeled in the training data. This could be addressed by introducing simulation algorithms that can model room propagation effects (such as the image source method [128, 131]) to generate more realistic semi-synthetic data. This will be evaluated in future work.

4.5.5 Fine Tuning Results

The second experiment in which we first apply the fine tuning procedure uses the Aseq15 sequence as the fine tuning subset. Table 4.4 shows the results obtained by GMBF and ASLNet with this fine tuning strategy. The results in the table shows that fine tuned

ASLNet is, most of the time, better than the SRP baseline (except in two cases for Aseq03 in which there was a slight degradation). The average performance shows a consistent improvement in ASLNet compared with SRP, between 1.8% and 11.3%. However, ASLNet is still behind GMBF in all cases but one (for Aseq03 and $80ms$).

	$80ms$		$160ms$		$320ms$	
	GMBF	ASLNet	GMBF	ASLNet	GMBF	ASLNet
Aseq01 MOTP	**0.795**	0.875	**0.686**	0.833	**0.588**	0.777
Aseq01 Δ_r^{MOTP}	**22.1**%	14.2%	**24.6**%	8.5%	**29.1**%	6.4%
Aseq02 MOTP	0.864	**0.839**	**0.759**	0.801	**0.694**	0.731
Aseq02 Δ_r^{MOTP}	10.0%	**12.6**%	**9.6**%	4.6%	**9.9**%	5.1%
Aseq03 MOTP	**0.686**	0.835	**0.563**	0.806	**0.484**	0.734
Aseq03 Δ_r^{MOTP}	**23.8**%	7.2%	**26.9**%	-4.7%	**29.9**%	-6.4%
Average MOTP	**0.778**	0.849	**0.666**	0.813	**0.585**	0.746
Average Δ_r^{MOTP}	**18.7**%	11.3%	**20.4**%	2.8%	**22.9**%	1.8%

Table 4.4: Experiment 2 results for the stratgy GMBF and ASLNet that was fine-tuned with sequence Aseq15.

Our conclusion is that the fine tuning procedure is able to effectively complement the trained models from synthetic data, leading to results that outperform SRP. This is specially relevant due to the following points.

- The amount of fine tuning data is limited (only 36 seconds, corresponding to 436 frames), thus opening the path to further improvements with a limited data recording effort.

- The speaker used for fine tuning is mostly moving while speaking, while in the testing sequences, the speakers are static while speaking. This means that the fine tuning material includes far more active positions than the testing sequences, and the network is able to extract the relevant information for the tested positions.

- The improvements obtained by ASLNet decrease for longer signal window sizes, suggesting the speaker's speed (and thus, the displacement of the speaker across the signal window) might be having an impact on the results. We have evaluated the average speed of the speakers for the moving speaker sequences, $0.72m/s$ for Aseq11, and $0.48m/s$ for Aseq15. They do not seem to have a significant relevant impact on position estimation. We have also evaluated the source displacement distribution within individual signal frames across the different sequences. The average displacement distances are $4-6cm$ for the $80ms$ window, $7-11cm$ for the $160ms$ window and $15-20cm$ for the $320ms$ window. When we have considered the maximum displacement distances, these values have turned out to be $7-27cm$ for the $80ms$ window, $14-34cm$ for the $160ms$ window, and $28-46cm$ for the $320ms$ case. These

displacements could have a visible impact on the results, and they might be the reason for the lower improvements achieved by our method for longer window sizes.

- The speaker used for fine tuning is male, and the obtained results for male speakers (sequences Aseq01 and Aseq03) and the female one (sequence Aseq02) do not seem to show any gender-dependent bias, which means that the gender issue does not seem to play a role in the adequate adaptation of the network models.

In spite of the relevant improvements with the fine tuning approach, they are still far from making this method suitable for further competitive exploitation in the ASL scenario (provided we have the GMBF alternative), so we next aim to increase the amount of fine tuning material.

In our third experiment, we apply the fine tuning procedure using an additional moving speaker sequence, that is, by including Aseq15 and Aseq11 in the fine tuning subset. Table 4.5 shows the results obtained by GMBF and ASLNet after fine tuning with Aseq15 and Aseq11 sequences. In this case, additional improvements over using only Aseq15 for fine tuning occurred, and there is only one case in which ASLNet does not outperform SRP (with a marginal degradation of −0.3%).

	80ms		160ms		320ms	
	GMBF	ASLNet	GMBF	ASLNet	GMBF	ASLNet
Aseq01 MOTP	**0.795**	0.805	**0.686**	0.750	**0.588**	0.706
Aseq01 Δ_r^{MOTP}	**22.1**%	21.1%	**24.6**%	17.6%	**29.1**%	14.9%
Aseq02 MOTP	0.864	**0.809**	0.759	**0.716**	**0.694**	0.712
Aseq02 Δ_r^{MOTP}	10.0%	**15.7**%	9.6%	**14.8**%	**9.9**%	7.5%
Aseq03 MOTP	**0.686**	0.792	**0.563**	0.732	**0.484**	0.692
Aseq03 Δ_r^{MOTP}	**23.8**%	12.0%	**26.9**%	4.9%	**29.9**%	-0.3%
Average MOTP	**0.778**	0.802	**0.666**	0.732	**0.585**	0.703
Average Δ_r^{MOTP}	**18.7**%	16.2%	**20.4**%	12.4%	**22.9**%	7.5%

Table 4.5: Experiment 3 results for the strategy GMBF and ASLNet that was fine-tuned with sequences Aseq15 and Aseq11.

The ASLNet approach again shows an average and consistent improvement compared with SRP of between 7.5% and 16.2%. In this case, the newly added sequence (Aseq11, with a duration of only 33s) for fine tuning corresponds to a randomly moving male speaker, and the results show that its addition contributes to further improvements in ASLNet proposal, but it is still behind the GMBF method in all cases but two, but with results getting closer. This suggests that a further increment in the fine tuning material should be considered.

Our last experiment consists of fine tuning the network, including additional static speaker sequences. To assure that the training (including fine tuning) and testing material are fully independent, we fine-tune with Aseq15, Aseq11 and with the static sequences that

are not tested in each experiment run, as it is shown in Table 4.2 experiments 4.1, 4.2 and 4.3.

	$80ms$		$160ms$		$320ms$	
	GMBF	ASLNet	GMBF	ASLNet	GMBF	ASLNet
Aseq01 MOTP	0.795	**0.607**	0.686	**0.540**	0.588	**0.485**
Aseq01 Δ_r^{MOTP}	22.1%	**40.5%**	24.6%	**40.7%**	29.1%	**41.6%**
Aseq02 MOTP	0.864	**0.669**	0.759	**0.579**	0.694	**0.545**
Aseq02 Δ_r^{MOTP}	10.0%	**30.3%**	9.6%	**31.1%**	9.9%	**29.2%**
Aseq03 MOTP	**0.686**	0.707	**0.563**	0.617	**0.484**	0.501
Aseq03 Δ_r^{MOTP}	**23.8%**	21.4%	**26.9%**	19.9%	**29.9%**	27.4%
Average MOTP	0.778	**0.664**	0.666	**0.581**	0.585	**0.511**
Average Δ_r^{MOTP}	18.7%	**30.6%**	20.4%	**30.6%**	22.9%	**32.8%**

Table 4.6: Experiment 4 results for the strategy GMBF and ASLNet that was fine-tuned with sequences Aseq15 and Aseq11 + static sequences.

Table 4.6 shows the results obtained for this fine tuning scenario, where the main conclusions are as follows:

- The ASLNet proposal exhibits much better average behavior than GMBF for all window sizes. The average absolute improvement against SRP for the CNN is more than 10 points higher than for GMBF, reaching 32.8% in the ASLNet case and 22.9% in GMBF.
- Considering the individual sequences, ASLNet is shown to be significantly better than GMBF for sequences Aseq01 and Aseq02, and slightly worse for Aseq03.
- Considering the best individual result, the maximum improvement for the ASLNet is 41.6% (Aseq01, $320ms$), while the top result for GMBF is 29.9% (Aseq03, $320ms$).
- The effect of adding static sequences is shown to be beneficial, as expected, provided that the acoustic tuning examples are generated from similar, but not identical, positions, as the speakers have varying heights and their positions in the room are not strictly equal from sequence to sequence.
- The improvements obtained are significant and come at the cost of additional fine tuning sequences. However, this extra cost is still reasonable, as the extra fine tuning material is of limited duration, around $400s$ on average ($6.65min$).

Summarizing, Figure 4.3 shows the average MOTP relative improvements over SRP obtained by ASLNet using different fine tuning subsets and its comparison with the GMBF results for all of the signal window sizes.

From the results obtained by our proposal, it is clear that the highest contribution to the improvements from the bare ASLNet training was the fine tuning procedure with

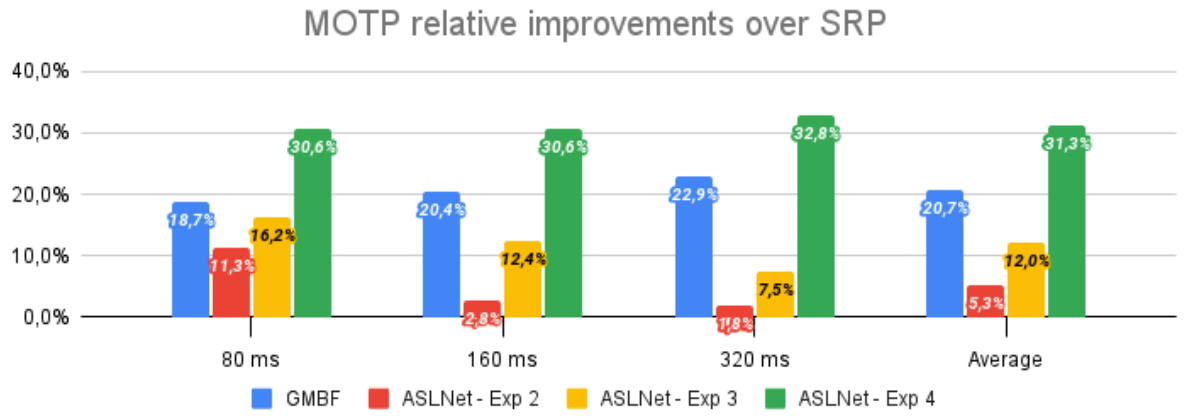


Figure 4.3: MOTP relative improvements over SRP for GMBF and ASLNet using different fine tuning subsets (for all window sizes).

limited data (Exp 2, compare Table 4.3 and Table 4.4), while the use of additional fine tuning material consistently improved the results (Table 4.5 and Table 4.6). It is again worth noticing that these improvements are consistently independent of the height and gender of the considered speaker and whether there is a match or not between the static or dynamic activity of the speakers being used in the fine tuning subsets. This suggests that the network actually learns the acoustic cues that are related to the localization problem. Thus, we conclude that our proposal is a suitable and promising strategy for solving the ASL task.

4.5.6 Fine Tuning Strategy Validation

When comparing the results of Table 4.3 and Table 4.4, and given the large improvement when applying the fine tuning strategy, it could be assumed that the initial training with semi-synthetic data is limited. Based on this argument, we run additional training experiments in which we just train the ASLNet network from scratch with the same sequences used in the experiments shown in Table 4.4, Table 4.5 and Table 4.6, with the objective of assessing the actual effect of combining semi-synthetic training and fine tuning versus just training with real room data. The training strategy and parameters are the same as those used when training the network from semi-synthetic data, described in Section 4.5.2.2.

Table 4.7 shows a comparison between these two options using different sequences. The metrics shown are the average values across all testing sequences for each case. The results for the training from scratch approach are included in column "ASLNet (fs)", and those for our proposed combined semi-synthetic training and fine tuning strategy are included in column "ASLNet (ft)".

When using Aseq15 in the training and fine tuning procedures (first row of Table 4.7), the average improvement of the ASLNet (ft) approach varies between 1.8% and 11.3%

	80ms		160ms		320ms	
	ASLNet (fs)	ASLNet (ft)	ASLNet (fs)	ASLNet (ft)	ASLNet (fs)	ASLNet (ft)
Exp 2 MOTP	0.915	**0.849**	0.875	**0.813**	0.916	**0.746**
Exp 2 Δ_r^{MOTP}	4.3%	**11.3**%	-4.6%	**2.8**%	-20.6%	**1.8**%
Exp 3 MOTP	0.951	**0.802**	0.900	**0.732**	0.983	**0.703**
Exp 3 Δ_r^{MOTP}	0.6%	**16.2**%	-7.6%	**12.4**%	-29.4%	**7.5**%
Exp 4 MOTP	0.791	**0.664**	0.724	**0.581**	0.742	**0.511**
Exp 4 Δ_r^{MOTP}	17.3%	**30.6**%	13.4%	**30.6**%	2.3%	**32.8**%

Table 4.7: Results for ASLNet, either trained from scratch (column "ASLNet (fs)") or using semi-synthetic training + fine tuning (column "ASLNet (ft)"), for different training/fine tuning sequences.

with an average improvement over all window lengths of 5.3%, while the ASLNet (fs) average improvement varies between -20.6% and 4.3% with an average value of -7.0%.

When using Aseq15 and Aseq11 in the training and fine tuning procedures (second row of Table 4.7), the average improvement of the ASLNet (ft) approach varies between 7.5% and 16.3% with an average improvement over all window lengths of 12.0%, while the ASLNet (fs) average improvement varies between -29.4% and 0.6% with an average value of -12.1%.

Finally, when using sequences from experiment 4 (third row of Table 4.7), the average improvement of the ASLNet (ft) approach varies between 30.6% and 32.8% with an average improvement over all window lengths of 31.3%, while the ASLNet (fs) average improvement varies between 2.3% and 17.3% with an average value of 11.0%.

So, in all of the evaluated cases, the combined semi-synthetic training and fine tuning approach clearly outperforms the training from scratch strategy, thus validating our methodology.

4.5.7 Deep learning Methods Comparison

We also provide a comparison between our proposal and a widely used deep learning ASL method known as SELDnet [88]. SELDnet is a CRNN architecture that uses the signal spectrum of the audio signals as inputs (phase and magnitude components of the spectrogram calculated on each audio channel) and is able to deal with multiple overlapping sound events. SELDnet generates two different outputs:

1. **Classification output**: The first output of the SELDnet is able to classify the sound events among a list of classes for each consecutive frame in the input audio signals.

2. **Regression output**: The second output estimates the Direction of Arrival (DoA) vector detected on each of the consecutive frames in the audio input. This vector is parameterized as the x, y, z axis coordinates of the DoA on a unit sphere around the microphone, which is claimed to lead to a network that learns better than one that uses a parametrization based on angles.

As suggested by the authors, we use the default values of the SELDnet design parameters regarding the feature extraction, network model, and training process, and in order to carry out the comparison with our method, the following issues are taken into account:

- SELDnet uses an audio window of size w_s for each microphone and extracts consecutive overlapped frames to compute the spectral components that are used as inputs. To compare this with our network, we perform experiments with different values of w_s: $80ms$, $160ms$, and $320ms$.

- Due to the fact that we use sequences of audio where only a single speaker appears simultaneously, we assign the same label ("*speech*") to all the audio windows used for training.

- We need SELDnet to infer the x, y, z coordinates of the target source, instead of the DoA vector. This only require us to change the target output during training, as the network model does not change it at all. Our spatial coordinates are also normalized in the interval $[-1, 1]$ which is compatible with the regression output of the SELDnet. The final output coordinates are again denormalized back to metric coordinates to proceed with the MOTP calculations.

- We follow the same experimental procedure as in our proposal (initial semi-synthetic training followed by fine tuning) in a resource-restricted scenario using only two microphone pairs. The experimental conditions are those for which we got the best performance (included in Table 4.6), relative to the experiment 4.

Table 4.8 shows the relative improvements of the proposal in [88] (column SELDnet) and our CNN approach (column ASLNet) over SRP.

	$80ms$		$160ms$		$320ms$	
	SELDnet	ASLNet	SELDnet	ASLNet	SELDnet	ASLNet
Aseq01 MOTP	1.037	**0.607**	1.039	**0.540**	1.076	**0.485**
Aseq01 Δ_r^{MOTP}	-1.7%	**40.5%**	-14.2%	**40.7%**	-29.6%	**41.6%**
Aseq02 MOTP	1.035	**0.669**	1.003	**0.579**	0.981	**0.545**
Aseq02 Δ_r^{MOTP}	-7.8%	**30.3%**	-19.4%	**31.1%**	-27.4%	**29.2%**
Aseq03 MOTP	1.017	**0.707**	0.991	**0.617**	1.020	**0.501**
Aseq03 Δ_r^{MOTP}	-13.0%	**21.4%**	-28.7%	**19.9%**	-47.8%	**27.4%**
Average MOTP	1.029	**0.664**	1.010	**0.581**	0.585	**0.511**
Average Δ_r^{MOTP}	-7.6%	**30.6%**	-20.7%	**30.6%**	-34.9%	**32.8%**

Table 4.8: Relative improvements over SRP for SELDnet and fine-tuned ASLNet.

It can be clearly seen that the SELDnet produces worse results than ASLNet approach in terms of localization accuracy and it actually performs worse than the standard SRP algorithm.

4.6 Conclusions

In this chapter, we present an audio localization CNN, `ASLNet` that is trained end-to-end from the raw audio signals to the source position. We show that this method is very promising, as it outperforms the state-of-the-art methods [25] and those using SRP when sufficient fine tuning data is available. It also performed better than a widely used proposal based on CRNNs. In addition, our experiments show that `ASLNet` exhibits good resistance against varying gender of the speaker and different window sizes compared with the baseline methods.

Given that the amount of data recordings for audio localization is limited, we propose in the chapter to first train the network using semi-synthetic data followed by fine tuning, using a small amount of real data. This has been a common strategy in other fields to prevent over-fitting, and we show that it significantly improves the system performance as compared with training the network from scratch using real data.

Chapter 5

Acoustic Source Localization from Deep Cross Correlations

. . . They somehow already know what you truly want to become. . .

Steve Jobs

5.1 Summary

This chapter includes the work published in the papers "*Towards Domain Independence in CNN-based Acoustic Localization using Deep Cross Correlations*" [108] and "*Acoustic source localization with deep generalized cross correlations*" [109].

Acoustic source localization techniques have advanced over the years, with the combination of Generalized Cross-Correlation (GCC) and Steered Response Power (SRP) being the most popular method. However, classical methods are now being surpassed by deep learning strategies. The downside is that these deep learning strategies rely on specific room geometry and sensor setups during training, making them less adaptable to new environments. This poses a practical problem as retraining involves labeling new data and running complex training algorithms. In this chapter, we introduce a simpler approach called Deep Generalized Generalized Cross Correlation (`DeepGCC`) that uses a Convolutional Neural Network (CNN) to transform the Generalized Cross-Correlation (GCC) into a Gaussian-shaped signal. By combining `DeepGCC` estimates, we can create a $3D$ acoustic map similar to SRP methods. An advantage of our method is that we can adapt the acoustic map to different microphone geometries without retraining the `DeepGCC` network. In real and simulated scenarios with varying training and test conditions, our method outperforms classical approaches and recent deep learning strategies without the need for retraining with different sensor configurations or room environments.

5.2 Introduction

Estimating the three-dimensional position of an acoustic source from a microphone set is a crucial task within many applications, such as human-machine interaction, smart rooms, or surveillance systems. This is referred to as Acoustic Source Localization (ASL), and it has been a long-standing research problem. A critical task in ASL systems is to estimate the delay between signals received at microphones that are placed at different spatial locations [9, 11]. Existing works have considered various scenarios.

In Time Difference of Arrival (TDoA) methods, the source signal is unknown (e.g., human speech). Thus, they measure the delay between the received signals at each microphone pair. The source position can be estimated with at least three TDoA measurements by using hyperbolic trilateration methods. Signals captured in everyday scenarios are contaminated with noise and multipath effects. Directly measuring TDoA in those cases is a challenging task that produces inaccurate localization results.

Other ASL techniques, such as the well-known SRP [5, 12, 18–20] or Minimum Variance Distortionless Response (MVDR) [24, 47], compute an acoustic power map, where the maximum of which reveals the position of the acoustic source. These methods use the GCC function [14] to assess the acoustic power from received signals at microphone pairs. This strategy is more robust to noise and multipath effects than TDoA methods. However, its results remain inaccurate in general scenarios with a reduced number of microphones.

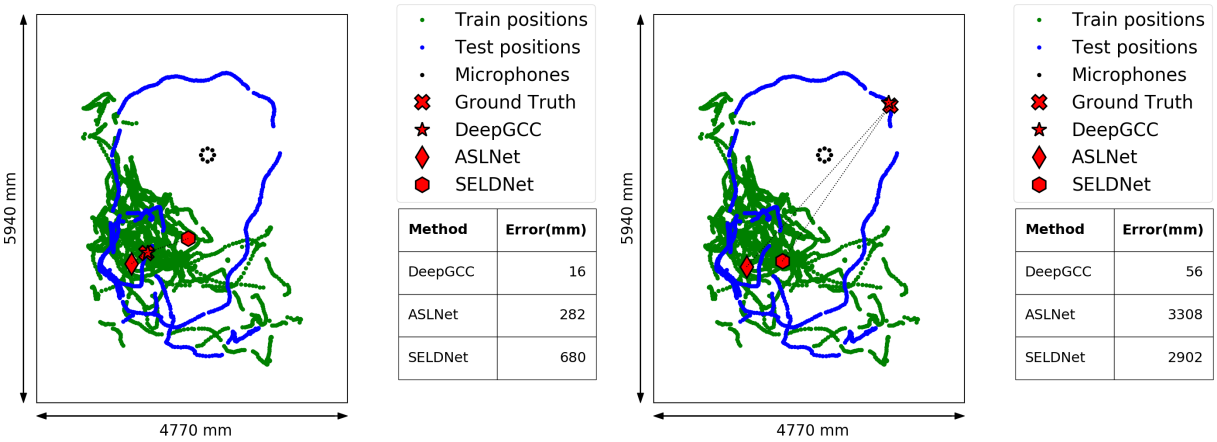
In the last years, ASL approaches based on Deep Learning techniques have appeared in the literature, following their previous success in several other related problems, such as in speech recognition and signal classification tasks. One of the commonly cited techniques from ASL literature is the one described in Chapter 4 of this book, where we use the raw acoustic signals to directly estimate the three-dimensional position of an acoustic source. In [88] *Adavanne et al.* they use the signal spectra to estimate the Direction of Arrival (DoA) of the acoustic source. The results of both approaches are promising, as they report a higher estimation accuracy than classical ASL methods. However, they have substantial limitations:

1. They involve a large amount of labeled training data, while the availability of large and public datasets for the ASL task is limited.
2. Such techniques are highly dependent on the room and sensor geometry used for training.

Consequently, they require structural changes and retraining when the room or microphone arrays vary, drastically degrading their accuracy when tested in environments that differ from the training conditions.

To illustrate this issue, Figure 5.1 compares the accuracy of our proposal (referred to as `DeepGCC` in the figure) with the state-of-the-art methods `ASLNet` (described in

Chapter 4) and SELDNet (proposed in [88])xs. This figure shows a $2D$ representation of an environment for one of our ASL datasets, where the microphone positions are shown as black dots, the green dots describe the training positions and the blue dots are the testing positions.



(a) (b)

Figure 5.1: Example results comparison of our proposal (DeepGCC) and the DNN-based state-of-the-art methods ASLNet and SELDNet. (a) shows the accuracy of all methods in testing positions included in the training material, while (b) shows the performance when the testing position is far from the distribution of the training positions.

In the left plot (Figure 5.1a), the methods are evaluated at a position that is within the area included in the training subset, where it can be seen that all approaches produce sub-meter accuracy, especially DeepGCC with an error of less than $2cm$. In the right plot right (Figure 5.1b), they are evaluated at a position that is outside the area included in the training subset. In this case, both ASLNet and SELDNet produce larger errors (above $3m$), while DeepGCC still achieves a good accuracy (below $6cm$), therefore showing a higher robustness to cope with geometrical constraints that are not found in the training material.

In this chapter, we propose a method based on a CNN, that takes as input the GCC of the signals received by a pair of microphones and estimates a Gaussian-shaped function, where its maximum appears at the time-delay between the two signals. We call this likelihood function the Deep Generalized Cross-Correlation (DeepGCC) of the original GCC signal.

Our proposal consists on a two-stage method to solve the ASL problem relying on DeepGCC. We first compute the DeepGCC signals from multiple microphone pairs using our trained neural network. We show that the DeepGCC model achieves a high degree of robustness, and is accurate with signals captured with different types of microphones and in different physical environments. Second, we combine the correlation signals in a $3D$ spatial grid, similarly to classical SRP methods, but replacing the GCC with the DeepGCC in the beamformer function. The crucial advantage of this design strategy is

that the microphone geometry is used as input in this step, so that our approach does not require re-training for different microphone geometries.

The contribution of this work are as follows:

- The DeepGCC method is largely independent of the environment and sensor geometry.
- We can use small sized datasets to train the CNN model.
- Our proposal is consistently more accurate than both classical and deep learning methods, specially under conditions where the testing room and/or the sensor array are physically different, or the source position significantly differs from those available in the training data (e.g. Figure 5.1).

Therefore, our proposal is more accurate and applicable than the current state-of-the-art and represents a step forward in this area.

5.3 Problem Statement

Let us consider a closed environment where a set of $N_{\mathcal{M}}$ microphones is placed at positions $\mathcal{M} = \{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_{N_{\mathcal{M}}}\}$, where $\mathbf{m}_k$ is a known $3D$ vector $\mathbf{m}_k = (m_{k,x}, m_{k,y}, m_{k,z})^\top$, denoting the position of the microphones from a reference coordinate origin. We group all these microphones in pairs, forming the set $\mathcal{P} = \left\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{N_{\mathcal{P}}}\right\}$, where $\mathbf{p}_j = (\mathbf{m}_{j,1}, \mathbf{m}_{j,2})$ contains two $3D$ vectors $\mathbf{m}_{j,1}$ and $\mathbf{m}_{j,2}$, with $\mathbf{m}_{j,1}, \mathbf{m}_{j,2} \in \mathcal{M}$ with $\mathbf{m}_{j,1} \neq \mathbf{m}_{j,2}$. If all microphone pair combinations are allowed then $N_{\mathcal{P}} = N_{\mathcal{M}}\left(N_{\mathcal{M}} - 1\right)/2$.

According to this setup, we can assume that several acoustic sources are located at unknown positions $\mathbf{r}_i = (r_{i,x}, r_{i,y}, r_{i,z})^\top$, with $i = 1, \ldots, N_{\mathcal{R}}$ being $N_{\mathcal{R}}$ the total number of acoustic sources in the scene. If each source is emitting an acoustic signal $s_i(t)$, a weighted and delayed addition of all of them is received by each microphone obtaining a time signal $x_k(t)$ following the propagation model:

$$x_k(t) = \sum_{i=1}^{N_{\mathcal{R}}} h_{i,k}(t) * s_i(t) + \eta_{i,k}(t) \tag{5.1}$$

with $h_{i,k}(t)$ being the Room Impulse Response (RIR) between the acoustic source position $\mathbf{r}_i$ and the k^{th} microphone, $*$ the time convolution operator, and $\eta_{i,k}(t)$ a signal that models all the adverse effects not included in $h_{i,k}(t)$ (noise, interference, etc.).

Under free-field conditions, the signals received by each microphone can be considered as the addition of the delayed and attenuated version of each acoustic source signals as:

$$x_k(t) = \sum_{i=1}^{N_{\mathcal{R}}} \alpha_{i,k} s_i\left(t - \tau_k(\mathbf{r}_i)\right) \tag{5.2}$$

where $\tau_k(\mathbf{r}_i)$ is the propagation delay between $\mathbf{m}_k$ and $\mathbf{r}_i$, calculated as $\tau_k(\mathbf{r}_i) = c^{-1}\|\mathbf{r}_i - \mathbf{m}_k\|$, and $\alpha_{i,k} = \frac{1}{4\pi c \tau_k(\mathbf{r}_i)}$, which is a distance-related attenuation assuming spherical propagation, with c as the sound propagation velocity in air.

Considering the received signals from two of the microphones of a microphone pair $\mathbf{p}_j$, and assuming both signals to have only phase component (not attending to their amplitude), they will differ in a TDoA $\Delta\tau\left(\mathbf{r}_i, \mathbf{p}_j\right)$, defined as:

$$\Delta\tau\left(\mathbf{r}_i, \mathbf{p}_j\right) = \tau_{j,1}(\mathbf{r}_i) - \tau_{j,2}(\mathbf{r}_i) = c^{-1}\left(\|\mathbf{r}_i - \mathbf{m}_{j,1}\| - \|\mathbf{r}_i - \mathbf{m}_{j,2}\|\right) \tag{5.3}$$

By means of the GCC function of the signals received by microphone pair $\mathbf{p}_j$, it is known that those TDoAs can be estimated. In order not to attend to the signals amplitude, its Generalized Cross-Correlation with Phase Transform (GCC-PHAT) filtered version of this function can be computed in practice as follows:

$$\mathbf{gcc}\left(\mathbf{p}_j\right) = \mathbf{gcc}_j = \mathcal{FFT}^{-1}\left\{\frac{X_{j,1}[\omega]\, X_{j,2}^*[\omega]}{|\, X_{j,1}[\omega]\,||\, X_{j,2}^*[\omega]\,|}\right\} \tag{5.4}$$

where we define $\mathbf{gcc}_j$ to simplify the notation, and $\mathcal{FFT}^{-1}$ is the *Inverse Fast Fourier Transform*.

Note that the $\mathbf{gcc}_j$ function is a time discrete signal, with time index n, that depends on f_s. For simplicity, we assume that the length of $\mathbf{gcc}_j$ with $j = 1, \ldots, N_{\mathcal{P}}$ is equal to $N+1$ (considering N even) and thus, $n_j \in [-N/2, \ldots, N/2]$. Accordingly, N should be chosen so that it covers the maximum TDoA (in samples) from any possible source position to any of the available microphones as follows:

$$N \geq \max\left(n_1, \ldots, n_{N_{\mathcal{P}}}\right), \quad \text{with} \quad n_j = \left\lceil \frac{f_s\|\mathbf{m}_{j,1} - \mathbf{m}_{j,2}\|}{c} \right\rceil \;\; \forall\, j \in [1, N_{\mathcal{P}}] \tag{5.5}$$

Once the maximum length for all $\mathbf{gcc}_j$ signals is defined, estimates $\widehat{\Delta\tau_i}\left(\mathbf{p}_j\right)$ of the actual $\Delta\tau\left(\mathbf{r}_i, \mathbf{p}_j\right)$ can be obtained from $\mathbf{gcc}_j$ according to an iterative process, where we assume that the number of active sources within the scene is known, as follows:

1. The most powerful source TDoA is estimated as:

$$\widehat{\Delta\tau_i}\left(\mathbf{p}_j\right) = \underset{n}{\operatorname{argmax}}\left(\mathbf{gcc}_j[n]\right) \tag{5.6}$$

2. This estimated TDoA contribution is canceled out from the $\mathbf{gcc}_j$ signal:

$$\mathbf{gcc}_j[n] = \mathbf{gcc}_j[n] - e^{-\frac{\left(n - \widehat{\Delta\tau_i}\left(\mathbf{p}_j\right)\right)^2}{2\sigma^2}} \tag{5.7}$$

3. This process is repeated as many times as sources are inside the environment.

Considering ideal conditions and discarding discretization effects, $\widehat{\Delta\tau}_i\left(\mathbf{p}_j\right) = \Delta\tau\left(\mathbf{r}_i, \mathbf{p}_j\right)$. Nevertheless, multipath, reverberation and noise effects make the anechoic assumption from Equation (5.2) no longer valid, and thus, the estimates $\widehat{\Delta\tau}_i\left(\mathbf{p}_j\right)$ differ from the ideal delay $\Delta\tau\left(\mathbf{r}_i, \mathbf{p}_j\right)$. Despite this mismatch, the $\mathbf{gcc}_j$ functions are usually used as the basis of the SRP beamformer. It aims for building an environment acoustic power map where the acoustic sources are to be located. It proceeds by defining a search space of $N_{\mathcal{Q}}$ locations, with $\mathcal{Q} = \left[\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_{N_{\mathcal{Q}}}\right]$, where $\mathbf{q}_k = (q_{kx}, q_{ky}, q_{kz})^\top$. Then, the SRP algorithm yields a value for each of the $\mathbf{q}_k$ positions as:

$$\text{srp}\left(\mathbf{q}_k\right) = \sum_{\forall \mathbf{p}_j \in \mathcal{P}} \mathbf{gcc}_j\left[\Delta\tau\left(\mathbf{q}_k, \mathbf{p}_j\right)\right] \;\forall\, \mathbf{q}_k \in \mathcal{Q} \tag{5.8}$$

where $srp\left(\mathbf{q}_k\right)$ represents an acoustic power map whose local maximums are related to the presence of active acoustic sources, thus allowing its localization as:

$$\hat{\mathbf{r}}_i = \underset{\mathbf{q}_k}{\text{argmax}}\left(\text{srp}\left(\mathbf{q}_k\right)\right) \tag{5.9}$$

Each acoustic source $\mathbf{r}_i$ is estimated by canceling the presence of the previous one from the $\mathbf{gcc}_j$ signals as detailed in Equation (5.7), where $\widehat{\Delta\tau}_i\left(\mathbf{p}_j\right) = \Delta\tau\left(\hat{\mathbf{r}}_{i-1}, \mathbf{p}_j\right)$ for each iteration. However, noise and reverberation effects have an impact on the accuracy of the $\mathbf{gcc}_j$ functions with respect to the free-field scenario, and therefore generate inaccuracies in the estimates derived from them, such as $\widehat{\Delta\tau}_i\left(\mathbf{p}_j\right)$ or $\text{srp}\left(\mathbf{q}_k\right)$.

The main goal of the method proposed in this chapter is to improve the SRP algorithm by training a CNN that transform each $\mathbf{gcc}_j$ into an ideal multiple-mean Gaussian-shaped signal (named as DeepGCC) where each peak is centered at the correct time delay $\Delta\tau_i\left(\mathbf{p}_j\right)$. This DeepGCC function will not only improve the further source localization through the acoustic power maps $\text{srp}\left(\mathbf{q}_k\right)$, but it also will allow us to refine those maps by applying diverse techniques and leading to an even better localization accuracy[1].

5.4 Model Topology

The main goal of the system described in this chapter is to develop an ASL system capable of outperforming the estimation of other state-of-the-art deep learning approaches in this task, specially when facing mismatched geometrical conditions or multi-speaker scenarios. This proposal relies on a two-stage method which uses the $\mathbf{gcc}_j$ functions as input and follows the architecture shown in Figure 5.2, with two main stages: (i) first computing the DeepGCC signals from every available microphone pair and (ii) combining them into a $3D$ acoustic power map. These stages are fully explained next.

[1] We will detail these processes in Chapter 6

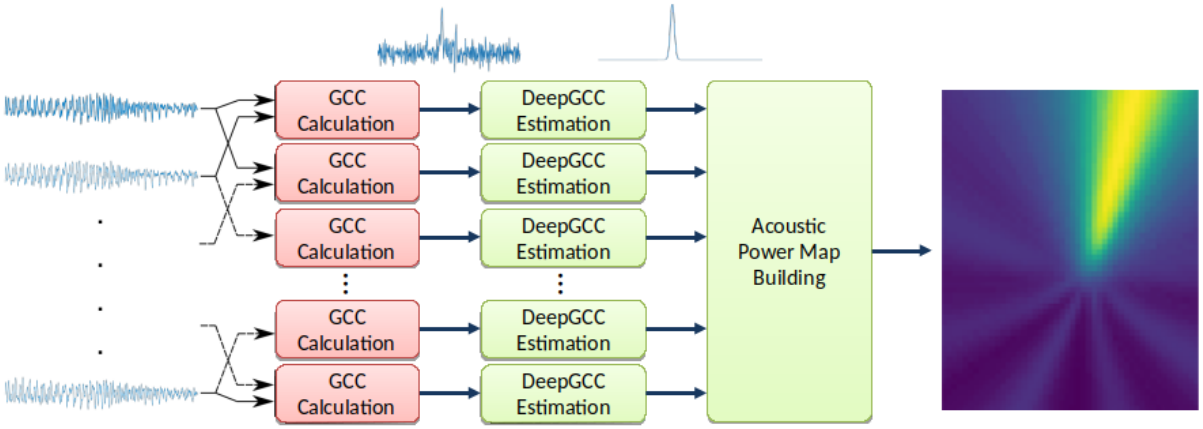


Figure 5.2: System Architecture with sample subsystem outputs.

5.4.1 DeepGCC Estimation

The first step of this proposal relies on an encoder-decoder CNN architecture referred to as DeepGCC. DeepGCC takes as input the $\mathbf{gcc}_j$ function for a given microphone pair $\mathbf{p}_j$ and yields a mixture of Gaussian-shaped signals with the same variance σ^2, and a mean equal to the corresponding source TDoA and microphone pair $\mathbf{p}_j$ $(\Delta\tau\left(\mathbf{r}_i, \mathbf{p}_j\right))$ as follows:

$$f_{\texttt{DeepGCC}}\left(\mathbf{gcc}_j\right) = f_{\texttt{DeepGCC}_j} = \frac{1}{N_{\mathcal{R}}} \sum_{\forall i \in N_{\mathcal{R}}} e^{-\frac{\left(n-\Delta\tau\left(\mathbf{r}_i,\mathbf{p}_j\right)\right)^2}{2\sigma^2}} \tag{5.10}$$

where $f_{\texttt{DeepGCC}_j}$ is defined to simplify the notation, being a function in the discrete time domain, with time index n in the range where N exits defined in Equation (5.5).

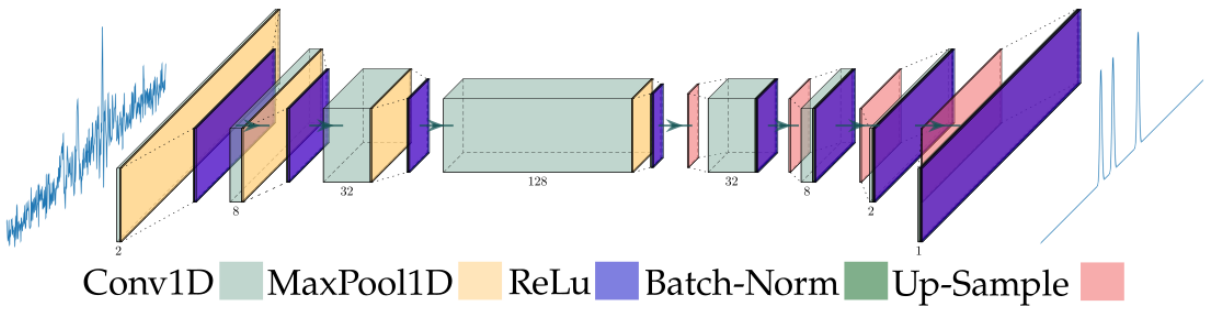


Figure 5.3: DeepGCC estimation architecture.

As shown in Figure 5.3, the DeepGCC model is divided in two parts, following an encoder-decoder architecture. The encoder side is composed by four blocks: Each block contains an $1D$ convolutional layer with filter size of 4 samples and stride of a single sample, max pooling layer, a batch normalization layer and a $ReLU$ activation layer. The depth of each block is 2, 8, 32 and 128 respectively. The decoder is also composed by four blocks, where each one is composed by an upsampling layer, a $1D$ convolutional layer with filter size of 4 samples and a stride of a single sample, a batch normalization layer and a

ReLU activation. The four blocks have a depth of 32, 8, 2 and 1 respectively. Table 5.1 summarizes the structure of the DeepGCC layers.

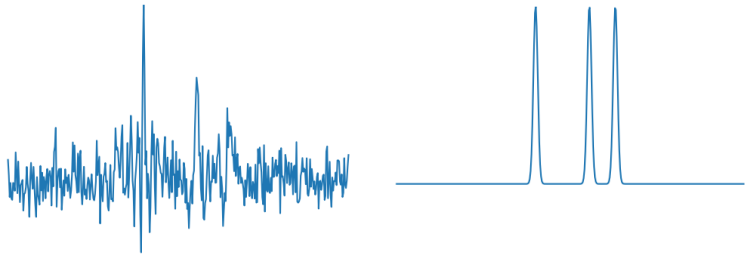
ID	Layer	Kernel size	Input size	Output size
Layer 1	*Conv1D*	4	$N \times 1$	$N \times 2$
Layer 2	*MaxPooling1D*	—	$N \times 2$	$N/2 \times 2$
Layer 3	*Batch Norm*	—	$N/2 \times 2$	$N/2 \times 2$
Layer 4	*ReLU*	—	$N/2 \times 2$	$N/2 \times 2$
Layer 5	*Conv1D*	4	$N/2 \times 2$	$N/2 \times 8$
Layer 6	*MaxPooling1D*	—	$N/2 \times 8$	$N/4 \times 8$
Layer 7	*Batch Norm*	—	$N/4 \times 8$	$N/4 \times 8$
Layer 8	*ReLu*	—	$N/4 \times 8$	$N/4 \times 8$
Layer 9	*Conv1D*	4	$N/4 \times 8$	$N/4 \times 32$
Layer 10	*MaxPooling1D*	—	$N/4 \times 32$	$N/8 \times 32$
Layer 11	*Batch Norm*	—	$N/8 \times 32$	$N/8 \times 32$
Layer 12	*ReLU*	—	$N/8 \times 32$	$N/8 \times 32$
Layer 13	*Conv1D*	4	$N/8 \times 32$	$N/8 \times 128$
Layer 14	*MaxPooling1D*	—	$N/8 \times 128$	$N/16 \times 128$
Layer 15	*Batch Norm*	—	$N/16 \times 128$	$N/16 \times 128$
Layer 16	*Relu*	—	$N/16 \times 128$	$N/16 \times 128$
Layer 17	*UpSampling1D*	—	$N/16 \times 128$	$N/8 \times 128$
Layer 18	*Conv1D*	—	$N/8 \times 128$	$N/8 \times 32$
Layer 19	*Batch Norm*	—	$N/8 \times 32$	$N/8 \times 32$
Layer 20	*Relu*	—	$N/8 \times 32$	$N/8 \times 32$
Layer 21	*UpSampling1D*	—	$N/8 \times 32$	$N/4 \times 32$
Layer 22	*Conv1D*	—	$N/4 \times 32$	$N/4 \times 8$
Layer 23	*Batch Norm*	—	$N/4 \times 8$	$N/4 \times 8$
Layer 24	*Relu*	—	$N/4 \times 8$	$N/4 \times 8$
Layer 25	*UpSampling1D*	—	$N/4 \times 8$	$N/2 \times 8$
Layer 26	*Conv1D*	—	$N/2 \times 8$	$N/2 \times 2$
Layer 27	*Batch Norm*	—	$N/2 \times 2$	$N/2 \times 2$
Layer 28	*Relu*	—	$N/2 \times 2$	$N/2 \times 2$
Layer 29	*UpSampling1D*	—	$N/2 \times 2$	$N \times 2$
Layer 30	*Conv1D*	—	$N \times 2$	$N \times 1$
Layer 31	*Batch Norm*	—	$N \times 1$	$N \times 1$
Layer 32	*Relu*	—	$N \times 1$	$N \times 1$

Table 5.1: DeepGCC layers summary.

Figure 5.4 shows a comparison between an example of a $\mathbf{gcc}_j$ function and its $f_{\texttt{DeepGCC}_j}$ estimate. The mixture Gaussian-shaped $f_{\texttt{DeepGCC}_j}$ is meant to be smoother than $\mathbf{gcc}_j$. As we will show in the experimental results section, the local maximums of $f_{\texttt{DeepGCC}_j}$ are better located around the source TDoAs than those of $\mathbf{gcc}_j$.

5.4.2 Acoustic Power Map Building

The second stage of the proposed system is in charge of building an acoustic power map (apm) by following the same approach used in classical SRP methods, but using as the beamformer basis the DeepGCC signals ($f_{\texttt{DeepGCC}_j}$) instead of the GCCs ($\mathbf{gcc}_j$). Therefore,



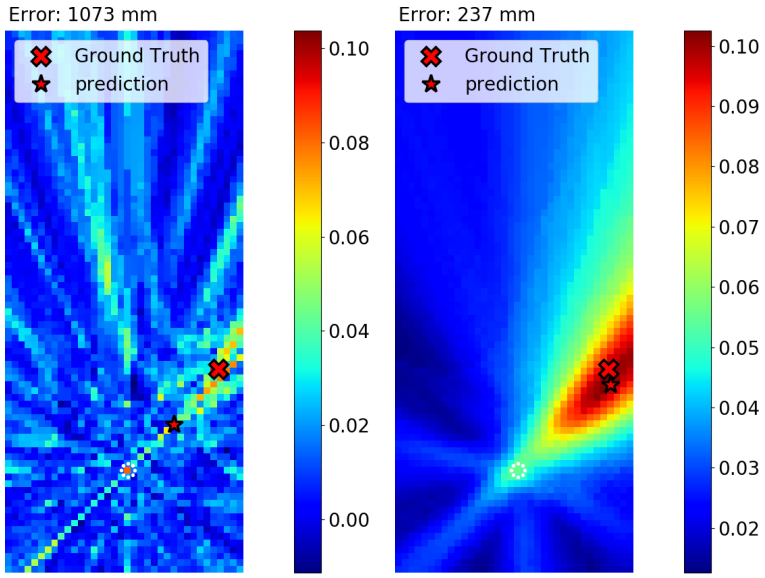
(a) (b)

Figure 5.4: Example comparison between (a) the $\mathbf{gcc}_j$ function, and (b) its corresponding $f_{\texttt{DeepGCC}_j}$ estimation.

following the structure of Equation (5.8), we can build an acoustic power map as:

$$\text{apm}\left(\mathbf{q}_k\right) = \sum_{\forall \mathbf{p}_j \in \mathcal{P}} f_{\texttt{DeepGCC}_j}\left[\Delta\tau\left(\mathbf{q}_k, \mathbf{p}_j\right)\right] \;\forall\, \mathbf{q}_k \in \mathcal{Q} \tag{5.11}$$

Figure 5.5 shows a real example of an acoustic power map extracted from the AV16.3 dataset. $\text{srp}\left(\mathbf{q}_k\right)$ is obtaine by applying Equation (5.8) is shown in Figure 5.5a. Also, $\text{apm}\left(\mathbf{q}_k\right)$ it shown in Figure 5.5b, which has been computed according to Equation (5.11). Again, the smoothing effect earlier discussed when comparing $\mathbf{gcc}_j$ and $f_{\texttt{DeepGCC}_j}$ is extended also to the acoustic power maps built from them.



(a) (b)

Figure 5.5: Example comparison between (a) an acoustic power map with basis functions $\mathbf{gcc}_j$ and (b) its corresponding apm based on $f_{\texttt{DeepGCC}_j}$ estimations.

5.5 Experimental Work

In this section, we describe the general conditions of the experimental setup to test the proposed system, the training strategy and the error metrics used for comparing our

proposal with other state-of-the-art algorithms. Finally, we present our experimental results.

5.5.1 Experimental Setup

The DeepGCC method based system is compared against ASLNet [68] and SELDNet [88], two state-of-the-art learning-based methods addressing the ASL problem. ASLNet, described in Chapter 4, is a neural network that directly estimates the $3D$ coordinates of a single acoustic source. The SELDNet approach estimates the source position of multiple sources in azimuth and elevation format, after being previously categorized into different acoustic classes. For our experimental work, we have modified the SELDNet output layer to yield $3D$ coordinates, and also removing the classification process, since the human speech can only be categorized as a single class. This method is referred as $\texttt{SELDNet}_{\texttt{XYZ}}$ in our experiments.

Our principal aim is to test the performance of the DeepGCC method under two different scenarios: (i) scenarios with mismatched conditions between training and testing conditions, such as the room geometry or the geometrical setup of the microphone arrays, and (ii) scenarios with multiple simultaneous speakers.

Table 5.2 shows the training, evaluation and testing sequences of the databases used to evaluate the two scenarios: CAV3D, AV16.3, UPC and ITC.

Data Partition	Training	Validation	Testing	Time per set
$\text{CAV3D}_{\text{DP1}}$	$\text{C}_{\text{seq[06-09, 11, 20, 21]}}$	C_{seq13}	$\text{C}_{\text{seq[10, 12]}}$	$6m\ 33s/1m\ 25s/2m\ 19s$
$\text{CAV3D}_{\text{DP2}}$	$\text{C}_{\text{seq[06-08, 10, 12, 20, 21]}}$	C_{seq13}	$\text{C}_{\text{seq[09, 11]}}$	$6m\ 51/1m\ 25s/2m$
$\text{CAV3D}_{\text{DP3}}$	$\text{C}_{\text{seq[06-10, 12, 20, 21]}}$	C_{seq11}	C_{seq13}	$7m\ 42s/1m\ 10s/1m\ 25s$
$\text{CAV3D}_{\text{DP4}}$	$\text{C}_{\text{seq[06-10, 12]}}$	$\text{C}_{\text{seq[11, 20, 21]}}$	$\text{C}_{\text{seq[13, 22-26]}}$	$6m\ 7s/2m\ 42s/5m\ 55s$
$\text{AV16.3}_{\text{DP1}}$	$\text{A}_{\text{seq[02, 03, 11]}}$	A_{seq15}	A_{seq1}	$8m\ 18s/36s/3m\ 44s$
$\text{AV16.3}_{\text{DP2}}$	$\text{A}_{\text{seq[01, 03, 11]}}$	A_{seq15}	A_{seq2}	$8m\ 16s/36s/3m\ 9s$
$\text{AV16.3}_{\text{DP3}}$	$\text{A}_{\text{seq[01, 02, 11]}}$	A_{seq15}	A_{seq3}	$7m\ 25s/36s/4m\ 2s$
UPC_{DP}	UPC_{dev}	$\text{UPC}_{\text{[01A, 01B]}}$	$\text{UPC}_{\text{[02A, 02B, 03A, 03B, 04A, 04B]}}$	$22m\ 56s/10m\ 21s/36m\ 22s$
ITC_{DP}	ITC_{dev}	$\text{ITC}_{\text{[01A, 01B]}}$	$\text{ITC}_{\text{[02A, 02B, 03A, 03B, 04A, 04B]}}$	$30m\ 36s/10m\ 21s/30m\ 56s$

Table 5.2: Data split used in the experiments, showing sequences.

According to the description provided in Table 3.2, the test sequences of the first two CAV3D partitions are relatively generic and assess overall aspects of the ASL task. As for the third partition, the sequence 13 contains positions never seen in the training set. Finally, the $\text{CAV3D}_{\text{DP4}}$ data partition is used for the multi-speaker scenario leaving the others for the mismatched conditions scenarios.

The rationale followed for the AV16.3 database split was the same as explained in Section 4.5.1. It is always tested with one of the static sequences, leaving the other two for training. Moreover, sequence 11, which is a dynamic sequence covering a large spatial region and with slow movements, is left for training, while sequence 15, since it has a higher movement speed, is left for validation.

Finally, for the UPC and ITC databases, the split recommended by the database developers has been followed. Note that the amount of available training data is relatively limited for the CAV3D and AV16.3 datasets (on average, less than 7.5 minutes of labeled speech per training partition), whereas the UPC and ITC datasets are much larger (more than 20 minutes each). In this case, we propose five different experiments to assess the performance of our proposal under the two scenarios described above:

- **Experiment 1. Simulated data:** This experiment evaluates the proposed methods with simulated data, generated with a realistic room simulator [132] that takes into account room reverberation properties to simulate the signal received by the microphone arrays. This experiment consists on three steps:

  1. Design a dataset of positions, evenly generated within the CAV3D and AV16.3 room environments, and with an amount of data similar in size to the real datasets, allowing fair comparisons. This dataset allow us to analyze the properties of learning-based localization methods without restrictions of having uneven localization data distributions, which were present in our real datasets.
  2. Verify the accuracy of these learning-based methods when tested in the same environment used for training, and also in mismatched room geometrical conditions.
  3. Analyze the synthetically trained learning-based methods accuracy on real data, which could be used to assess the feasibility of adapting those methods to a new room without the requirement of a data annotation effort.

- **Experiment 2. Matched room and microphone array geometries:** This experiment is focused on evaluating the maximum achievable performance for each of the evaluated methods, so that the room and microphone geometry are the same in the training and testing subsets, thus using data from the same dataset. In these experiments, the whole set of microphones is used in the localization task, except for the AV16.3 dataset where one of the two arrays is used.

- **Experiment 3. Mismatched room geometry:** The main goal of this experiment is to evaluate the algorithm robustness in a new environment where the room geometry changes, but the microphone array geometry is the same as the one used in the training phase.

- **Experiment 4. Mismatched room and microphone array geometries:** This experiment focuses on evaluating the performance of the algorithms in a fully new environment where both the room and the microphone array geometries vary.

- **Experiment 5. Multi-speaker scenarios:** We assess the capability of `DeepGCC` method when facing multiple speakers situations.

It is important to note that the SELDNet$_{XYZ}$ and ASLNet networks are trained for a fixed amount of microphones (which affects their input size), and for a particular array geometry. Changing these conditions requires retraining for these methods. Therefore, some of the tests in Experiment 3, and all of them in Experiment 4, will only be carried out for the SRP and DeepGCC algorithms, since they are largely independent on the geometrical conditions. In Experiment 5 also SRP and DeepGCC can be only evaluated, since the ASLNet and SELDNet$_{XYZ}$ methods are not suitable for multi-source human speech acoustic environments.

5.5.2 Training Strategy

To evaluate our proposal, we use the CAV3D, AV16.3, UPC and ITC datasets, described in Chapter 3.

5.5.2.1 Dataset Generation

The generation of the training, validation, and testing data is identical in all cases, and consists in computing the GCC-PHATs for all the possible signals pairs. We extract each acoustic frame (signal window) from the whole sequence using a $133ms$ window length with 50% overlapping and Blackman windowing. Assuming a $96kHz$ sampling rate, the window size is $12,800$ samples long.

For the GCCs computation, the FFT length is equal to the window size. For the correlation calculation, and with the same $96kHz$ sampling rate, we use windows of 400 samples, which implies that the network can model time-delays up to approximately $\pm 2ms$. This delay is more than enough for our requirements, given the maximum separation between microphones in all datasets ($40cm$ for the inverted T-shaped microphone arrays). The SRP map is built using an homogeneously distributed grid evaluated every $10cm$ in the three dimensions.

For each GCC-PHAT signal, we generate the supervised network output according to Equation (5.10) (page 49), where each ground truth $\Delta\tau\left(\mathbf{r}_i, \mathbf{p}_j\right)$ is computed in terms of the microphones positions and the labeled acoustic source location, as it is shown in Equation (5.3) (page 47). This signal is also generated to be 400 samples in length and has a standard deviation of $\sigma = 5$ samples that we empirically selected in preliminary experiments.

5.5.2.2 Training Details

With respect to the training strategy, we again follow the same procedure to train all the methods. The batch size (N_b) is equal to 100 samples and we use validation data in order to stop the training if the validation loss does not improve along 50 epochs. The

loss function used to train DeepGCC consists in the Mean Squared Error (MSE) between the network estimation $\hat{f}_{\texttt{DeepGCC}}$, and the actual Ground Truth $f_{\texttt{DeepGCC}}$, described in Equation (5.10), as:

$$\text{MSE}\left(f_{\texttt{DeepGCC}}, \hat{f}_{\texttt{DeepGCC}}\right) = \sum_{\forall b \in N_b} \|f_{\texttt{DeepGCC}_b} - \hat{f}_{\texttt{DeepGCC}_b}\| \tag{5.12}$$

Also, the MSE function is used to train the ASLNet and $\texttt{SELDNet}_{\texttt{XYZ}}$ methods, where the actual x, y, z position ground truth ($\mathbf{r}$) is compared with the estimation given by the model ($\hat{\mathbf{r}}$) according to:

$$\text{MSE}\left(\mathbf{r}, \hat{\mathbf{r}}\right) = \sum_{\forall b \in N_b} \|\mathbf{r}_b, \hat{\mathbf{r}}_b\| \tag{5.13}$$

To minimize Equation (5.12) and Equation (5.13), we use the *ADAM* optimizer [130] with a learning rate of 10^{-4} and a decay of 10^{-8}, setting default values for the rest of parameters. Table 5.3 summarizes the architectures of the evaluated deep learning-based methods, their size (number of parameters), and the general learning parameters used.

Method	Architecture	Parameters	Learning Parameters
DeepGCC	CNN Encoder-Decoder	36025	Batch size = 100 Window length = 133ms
ASLNet [68]	CNN encoder + Fully Connected	4680979	decay = 10^{-8} lr = 10^{-4}
$\texttt{SELDNet}_{\texttt{XYZ}}$ [88]	CNN Feature extraction + GRU layer	562884	Optimizer = Adam Early stopping = 50

Table 5.3: Details on the architectural complexity for each of the evaluated methods, and the learning parameters used.

5.5.3 Evaluation Metrics

We evaluate performance adopting the same metric developed under the *CHIL* project [119]. It is known as Multiple Object Tracking Precision (MOTP) and defined as:

$$\text{MOTP} = \frac{\sum_{k=1}^{N_\mathcal{K}} \|\mathbf{r}_k - \hat{\mathbf{r}}_k\|}{N_\mathcal{K}} \quad [mm] \tag{5.14}$$

where $N_\mathcal{K}$ denotes the total number of position estimations along time. Note that the lower the MOTP, the better. We also measure the relative improvement of a given *Method* in MOTP with respect to a different *BaseMethod* as:

$$\Delta^{\text{MOTP}}_{r_{BaseMethod}} = 100\,\frac{\text{MOTP}_{BaseMethod} - \text{MOTP}_{Method}}{\text{MOTP}_{BaseMethod}} \quad [\%] \tag{5.15}$$

Positive values for $\Delta^{\text{MOTP}}_{r_{BaseMethod}}$ mean that *Method* performs better than *BaseMethod*.

5.5.4 DeepGCC Model Justification

In this section we describe some experiments that provide an idea on the influence of the DeepGCC architecture complexity and the sampling frequency in the localization accuracy. We also evaluate how the DeepGCC strategy would compare with a simple low-pass filtering approach, to demonstrate that DeepGCC does not simply smooth the acoustic maps.

The experiments are carried out using the $\mathsf{CAV3D}_{\mathsf{DP3}}$ partition, comparing the localization accuracy with the proposed DeepGCC in the same data-set. The partition is selected as it contains the sequence $\mathsf{C}_{\mathsf{seq13}}$, which is the most complex sequence within the single speaker available data collection.

5.5.4.1 DeepGCC Model Complexity Influence

We evaluate a simpler version of DeepGCC, namely $\texttt{DeepGCC}_{\texttt{Simpler}}$, obtained by removing the last encoder layer and the first decoder layer.

The first and second columns of Table 5.4 show the localization accuracy for DeepGCC and $\texttt{DeepGCC}_{\texttt{Simpler}}$, respectively, showing a slight degradation in the later due to its reduced architectural complexity. This suggest that we could study a further increase in the complexity of the model to possibly improve our results; or a further reduction in its complexity to allow for lower computational demands with an admissible degradation in performance.

DeepGCC	$\texttt{DeepGCC}_{\texttt{Simpler}}$	$\texttt{DeepGCC}_{@16kHz}$
798 mm	811 mm	1288 mm

Table 5.4: MOTP localization errors for DeepGCC, $\texttt{DeepGCC}_{\texttt{Simpler}}$, and that downsampling the signals to $16kHz$.

5.5.4.2 DeepGCC Model Sampling Frequency Influence

We also evaluated the localization accuracy of DeepGCC with acoustic signals downsampled from $96kHz$ to $16kHz$, to assess the relevance of this parameter.

Column 3 of Table 5.4 shows that the DeepGCC localization error significantly increases by a 60% with the downsampled signals, thus suggesting that high sampling rates are critical in our experimental approach.

5.5.4.3 DeepGCC Model vs Low-Pass Filtering Approach

As described in Section 5.4.1, our DeepGCC model has a smoothing effect over the GCC signal. Obviously, the highly non-linear nature of the deep CNN is responsible for such filtering/smoothing process in DeepGCC, but we could think of less complex alternatives

if the advantage of the proposal were just based on this smoothing process. In this experiment we want to analyze if such level of complexity (that of a CNN) is actually needed. Our approach is comparing DeepGCC with a simpler method based on low-pass filtering the GCC signals. Hence, we apply 3 different low-pass filters to the GCC signal, with cut-off frequencies at 2, 4, and 8 kHz respectively.

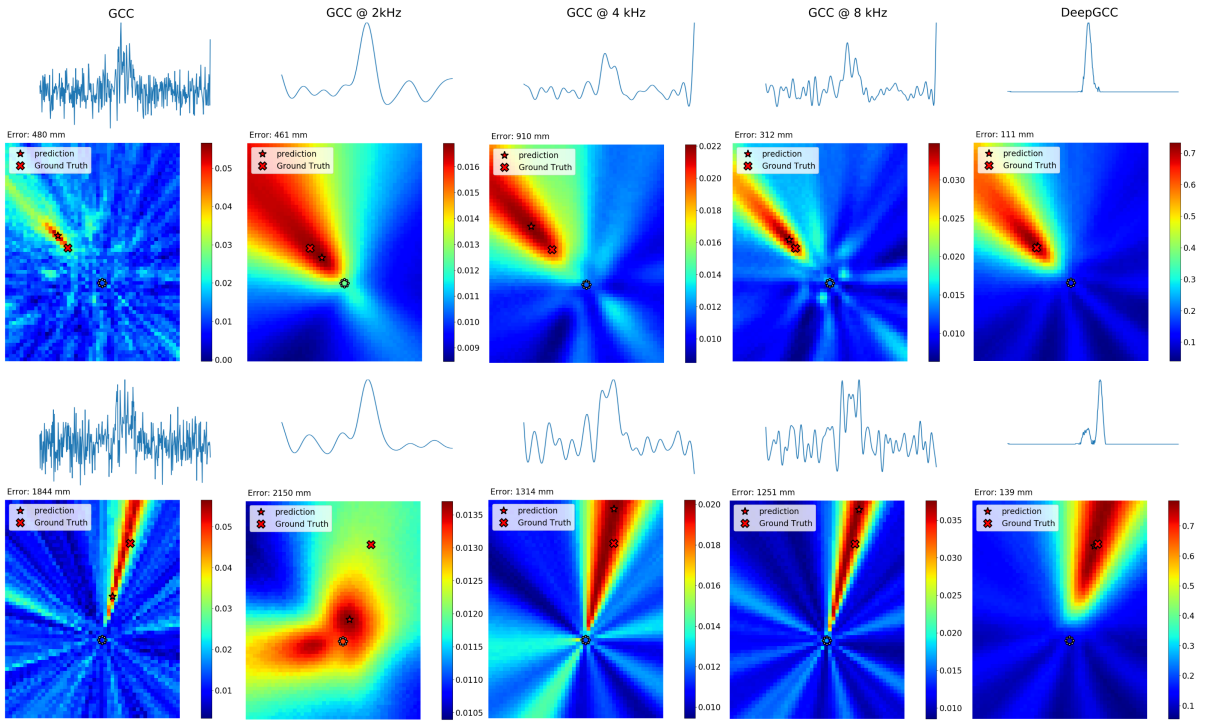


Figure 5.6: Qualitative comparison examples of GCC, GCC with low-pass filtering at $2kHz$, $4kHz$, $8kHz$ and DeepGCC signals with their corresponding acoustic map building.

Figure 5.6 shows two examples of acoustic power maps obtained from the low-pass filtered GCC signals, compared to the maps obtained by the standard GCC function and those obtained by our DeepGCC model. In the graphics we include the ground-truth source positions, the source position estimations, the localization error, and an example of one particular GCC function and its filtered versions. In the two examples, DeepGCC obtains more accurate source localization than the filtered GCC signals, with the localization error increasing with decreasing values of the filters cut-off frequency.

GCC	GCC$_{@2kHz}$	GCC$_{@4kHz}$	GCC$_{@8kHz}$	DeepGCC
867 mm	1141 mm	1013 mm	1000 mm	**798 mm**

Table 5.5: MOTP localization errors when using the standard GCC function, its low-pass filtered versions (at different cutoff frequencies) and the DeepGCC estimation. Evaluation is done on of the test sequence of the CAV3D$_{DP3}$ partition.

Table 5.5 shows the resulting average localization error of the alternatives based on GCC, its filtered versions, and DeepGCC, evaluated in the CAV3D$_{DP3}$ data partition. In all configurations, DeepGCC obtains significantly better results than those achieved by

the filtered GCC signals. This shows that DeepGCC, in addition to its smoothing effect, is thus effectively removing information from the GCC signals to produce more accurate source localization results than when using simple linear filters.

5.5.5 Experiment 1: Simulated Data

In this experiment we randomly generate 5000 acoustic source positions uniformly distributed in each of the AV16.3 and CAV3D environments (see Sections 3.3 and 3.4 respectively). We use the "pyroomacoustics" simulation tool [132] to generate the signals captured by the microphones, using the real reverberation time for the corresponding rooms. To generate the source acoustic signals, we use the *Albayzin Phonetic Corpus* [115] (Section 3.2), which consists of high quality close talk speaking recordings. We generated 4000 frames for training, 500 frames for validation and 500 for testing. We refer to the synthetic data-sets generated for this experiment as $\text{CAV3D}_{\text{Sim}}$ and $\text{AV16.3}_{\text{Sim}}$.

Our first test is aimed at comparing all the evaluated methods in matched conditions, meaning we use the same environment for training and testing. Table 5.6 shows the MOTP localization error of all compared methods in the simulated data-sets.

$\text{CAV3D}_{\text{Sim}}$	SRP	DeepGCC	ASLNet	$\text{SELDNet}_{\text{XYZ}}$
MOTP	868	777	**756**	1065
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	10.5%	**12.9**%	−22.7%
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	−11.7%	—	**2.7**%	−37.1%
$\Delta^{\text{MOTP}}_{r_{ASLNet}}$	**14.8**%	−2.8%	—	−40.9%
$\Delta^{\text{MOTP}}_{r_{SELDNet_{XYZ}}}$	18.5%	27.0%	**29.0**%	—

$\text{AV16.3}_{\text{Sim}}$	SRP	DeepGCC	ASLNet	$\text{SELDNet}_{\text{XYZ}}$
MOTP	891	**880**	974	1442
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	**1.2**%	−9.3%	−61.8%
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	**-1.3**%	—	−10.7%	−63.9%
$\Delta^{\text{MOTP}}_{r_{ASLNet}}$	8.5%	**9.7**%	—	−48.0%
$\Delta^{\text{MOTP}}_{r_{SELDNet_{XYZ}}}$	38.2%	**39.0**%	32.5%	—

Table 5.6: Experiment 1 results with matched conditions on $\text{CAV3D}_{\text{Sim}}$ (left) and $\text{AV16.3}_{\text{Sim}}$ (right): MOTP localization error in *mm*.

Results show that DeepGCC consistently performs better than SRP algorithm on simulated data. DeepGCC also obtains better results than the rest of learning-based methods in $\text{AV16.3}_{\text{Sim}}$, but ASLNet is the best method on $\text{CAV3D}_{\text{Sim}}$, closely followed by DeepGCC. Finally, $\text{SELDNet}_{\text{XYZ}}$ is the worst performing method in this experiment. This might be caused by the CNN-based feature extractor, particularly designed to work in realistic environments.

We also evaluate the performance of all the methods on simulated data when tested in mismatched room conditions. Table 5.5.7 shows the localization error MOTP of all compared methods. Note that this assessment of room geometry mismatches between the CAV3D and AV16.3 rooms exploits the fact that there are arrays with the same microphone geometry in both rooms, plus the fact that only the top array is used in AV16.3.

trn: AV16.3$_{Sim}$ tst: CAV3D$_{Sim}$	SRP	DeepGCC	ASLNet	SELDNet$_{XYZ}$
MOTP	868	**803**	1203	2173
$\Delta^{MOTP}_{r_{SRP}}$	—	**7.5**%	−38.6%	−150.3%
$\Delta^{MOTP}_{r_{DeepGCC}}$	**-8.1**%	—	−49.8%	−170.3%
$\Delta^{MOTP}_{r_{ASLNet}}$	27.8%	**33.3**%	—	−80.6%
$\Delta^{MOTP}_{r_{SELDNet_{XYZ}}}$	60.1%	**63.0**%	44.6%	—

trn: CAV3D$_{Sim}$ tst: AV16.3$_{Sim}$	SRP	DeepGCC	ASLNet	SELDNet$_{XYZ}$
MOTP	891	**785**	3328	3040
$\Delta^{MOTP}_{r_{SRP}}$	—	**11.9**%	−273.5%	−242.2%
$\Delta^{MOTP}_{r_{DeepGCC}}$	**-13.5**%	—	−323.9%	−287.3%
$\Delta^{MOTP}_{r_{ASLNet}}$	73.2%	**76.4**%	—	8.7%
$\Delta^{MOTP}_{r_{SELDNet_{XYZ}}}$	70.7%	**74.2**%	−9.5%	—

Table 5.7: Experiment 1 results with mismatched room conditions: Trained on AV16.3$_{Sim}$ and tested on CAV3D$_{Sim}$ (left), and trained on CAV3D$_{Sim}$ and tested on AV16.3$_{Sim}$ (right): MOTP localization error in *mm*.

DeepGCC obtains better results than the SRP algorithm, even when trained in a different room, while the rest of deep learning based methods significantly degrade their performance. It is important to note that ASLNet and SELDNet$_{XYZ}$ show a significant degradation in localization error when tested in environments different from those used during training. This experiment confirms that our method seems to be more robust to mismatched room conditions than the other state of the art deep learning based proposals.

Finally, we investigate the effect of training a model with simulated data and then testing it in real data sequences. In particular, we use the CAV3D C$_{seq13}$ sequence (most complex sequence from the CAV3D static sequence set) and the AV16.3 A$_{seq2}$ sequence. Table 5.8 shows the MOTP results of all models, including combinations where the simulated data used for training corresponds to a different environment from the one used during testing.

	SRP	DeepGCC	ASLNet	SELDNet$_{XYZ}$
Train: CAV3D$_{Sim}$ Test: C$_{seq13}$	**867**	2224	1698	2115
Train: AV16.3$_{Sim}$ Test: A$_{seq2}$	**693**	1072	1620	1588
Train: AV16.3$_{Sim}$ Test: C$_{seq13}$	**867**	1917	1736	2165
Train: CAV3D$_{Sim}$ Test: A$_{seq2}$	**693**	1011	4118	4710

Table 5.8: Experiment 1 on real data results with models trained with simulated data: MOTP localization error in *mm*. Training and testing sequences are indicated in the Table.

Results in Table 5.8 shows that all learning-based methods which are trained on simulated data, including DeepGCC, do not perform well when tested on real data. This large gap between simulated data and real data is also well known in other fields such as computer vision. Using only simulated data to train our localization systems thus requires further study which is out of the scope of this work.

5.5.6 Experiment 2: Matched Room and Microphone Array Geometries

This experiment evaluates the methods using the same environment where they were trained. The results for each CAV3D sequence and its average results and its relative improvements are shown in Table 5.9.

CAV3D	SRP	DeepGCC	ASLNet	SELDNet$_{XYZ}$
C$_{seq09}$	692	728	814	**620**
C$_{seq10}$	815	806	925	**552**
C$_{seq11}$	1092	1069	1333	**946**
C$_{seq12}$	1080	1025	1035	**734**
C$_{seq13}$	867	**798**	1367	905
CAV3D$_{DP1}$	908	881	1108	**757**
CAV3D$_{DP2}$	882	857	1195	**834**
CAV3D$_{DP3}$	867	**798**	1367	905

CAV3D	SRP	DeepGCC	ASLNet	SELDNet$_{XYZ}$
MOTP	935	903	1130	**776**
$\Delta^{MOTP}_{r_{SRP}}$	—	3.4%	-20.8%	**17.0%**
$\Delta^{MOTP}_{r_{DeepGCC}}$	-3.5%	—	-25.1%	**14.0%**
$\Delta^{MOTP}_{r_{ASLNet}}$	17.2%	20.1%	—	**31.3%**
$\Delta^{MOTP}_{r_{SELDNet_{XYZ}}}$	-20.4%	**-16.3%**	-45.5%	—

Table 5.9: Experiment 2 CAV3D detailed results: MOTP localization error in *mm* for each testing sequence (left, with last three rows showing average results for the three defined partitions), and average MOTP localization error and relative improvements (right)

Taking into account the CAV3D data partitions defined in Table 5.2 (page 52) and shown in Figure 5.7, SELDNet$_{XYZ}$ is the best proposal, in average MOTP performance, with top values in sequences C$_{seq09}$ and C$_{seq10}$ (simplest test sequences in CAV3D dataset) where the speaker movements are limited, but with worse results in the other sequences, where more complicated movement activity is carried out. The DeepGCC model is the second best method, outperforming the SRP algorithm in every sequence, with the exception of C$_{seq09}$.

Table 5.10 shows the AV16.3 precision results, with all methods outperforming the SRP algorithm, and with DeepGCC also outperforming the other Deep Learning approaches (except for the A$_{seq01}$ sequence where SELDNet$_{XYZ}$ performs better). Figure 5.8 shows the AV16.3 data partitions defined in Table 5.2.

AV16.3	SRP	DeepGCC	ASLNet	SELDNet$_{XYZ}$
A$_{seq01}$ AV16.3$_{DP1}$	1036	**680**	771	1000
A$_{seq02}$ AV16.3$_{DP2}$	693	623	677	**573**
A$_{seq03}$ AV16.3$_{DP3}$	739	**577**	730	634

AV16.3	SRP	DeepGCC	ASLNet	SELDNet$_{XYZ}$
MOTP	815	**679**	725	726
$\Delta^{MOTP}_{r_{SRP}}$	—	**16.7%**	11.1%	10.9%
$\Delta^{MOTP}_{r_{DeepGCC}}$	-20.0%	—	**-6.7%**	-6.9%
$\Delta^{MOTP}_{r_{ASLNet}}$	-12.4%	**6.3%**	—	-0.2%
$\Delta^{MOTP}_{r_{SELDNet_{XYZ}}}$	-12.2%	**6.5%**	0.2%	—

Table 5.10: Experiment 2 AV16.3 detailed results: MOTP localization error in *mm* for each testing sequence (left, in this case, the partition test set coincides with each sequence) and average MOTP localization error and relative improvements (right)

For the UPC environment, Table 5.11 clearly shows that, again, DeepGCC outperforms ASLNet and SELDNet$_{XYZ}$. In this case, the data distribution described in Table 5.2 is shown in Figure 5.9.

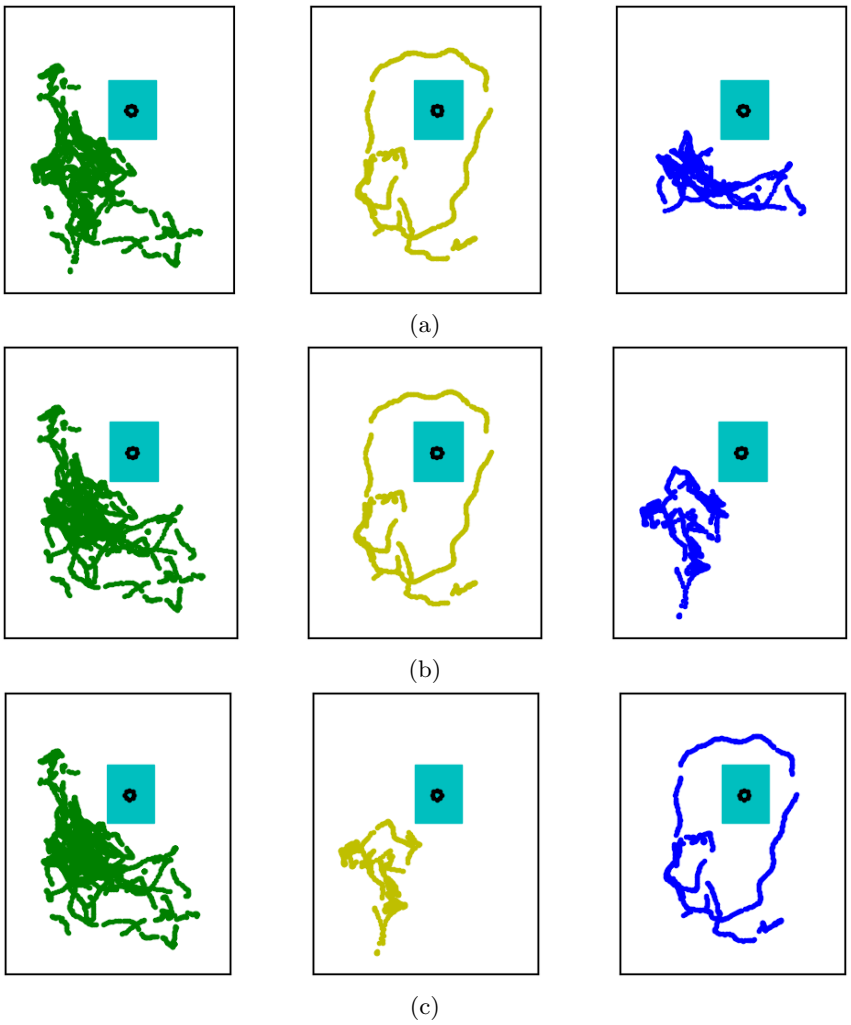


Figure 5.7: CAV3D positions used in each data split (green = Train, yellow = Validation, blue = Test): (a) data partition 1 ($\text{CAV3D}_{\text{DP1}}$), (b) data partition 2 ($\text{CAV3D}_{\text{DP2}}$), and (c) data partition 3 ($\text{CAV3D}_{\text{DP3}}$).

UPC	SRP	DeepGCC	ASLNet	$\text{SELDNet}_{\text{XYZ}}$
UPC_{02A}	1288	**622**	651	863
UPC_{02B}	1362	874	**840**	994
UPC_{03A}	1300	**1025**	1034	1129
UPC_{03B}	1302	910	**834**	933
UPC_{04A}	1032	**765**	994	1127
UPC_{04B}	1336	**1313**	1664	1731
UPC_{DP}	1278	**941**	1001	1127

UPC	SRP	DeepGCC	ASLNet	$\text{SELDNet}_{\text{XYZ}}$
MOTP	1278	**941**	1001	1127
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	**26.4**%	21.6%	11.8%
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	-35.8%	—	**-6.4**%	-19.8%
$\Delta^{\text{MOTP}}_{r_{ASLNet}}$	-27.6%	**6.0**%	—	-12.6%
$\Delta^{\text{MOTP}}_{r_{SELDNet_{XYZ}}}$	-13.4%	**16.5**%	11.2%	—

Table 5.11: Experiment 2 UPC detailed results: MOTP localization error in *mm* for each testing sequence (left, with partition average in the last row) and average MOTP localization error and relative improvements (right)

Finally, Table 5.12 shows the results for the ITC dataset. Once again, DeepGCC system is, on average, the most accurate method. It should be noted that, in this case, the ASLNet and $\text{SELDNet}_{\text{XYZ}}$ models have a bad performance due to the data partition characteristics, in which most of the positions in the testing subset did not appear in the training subset, as it can be clearly seen in Figure 5.10.

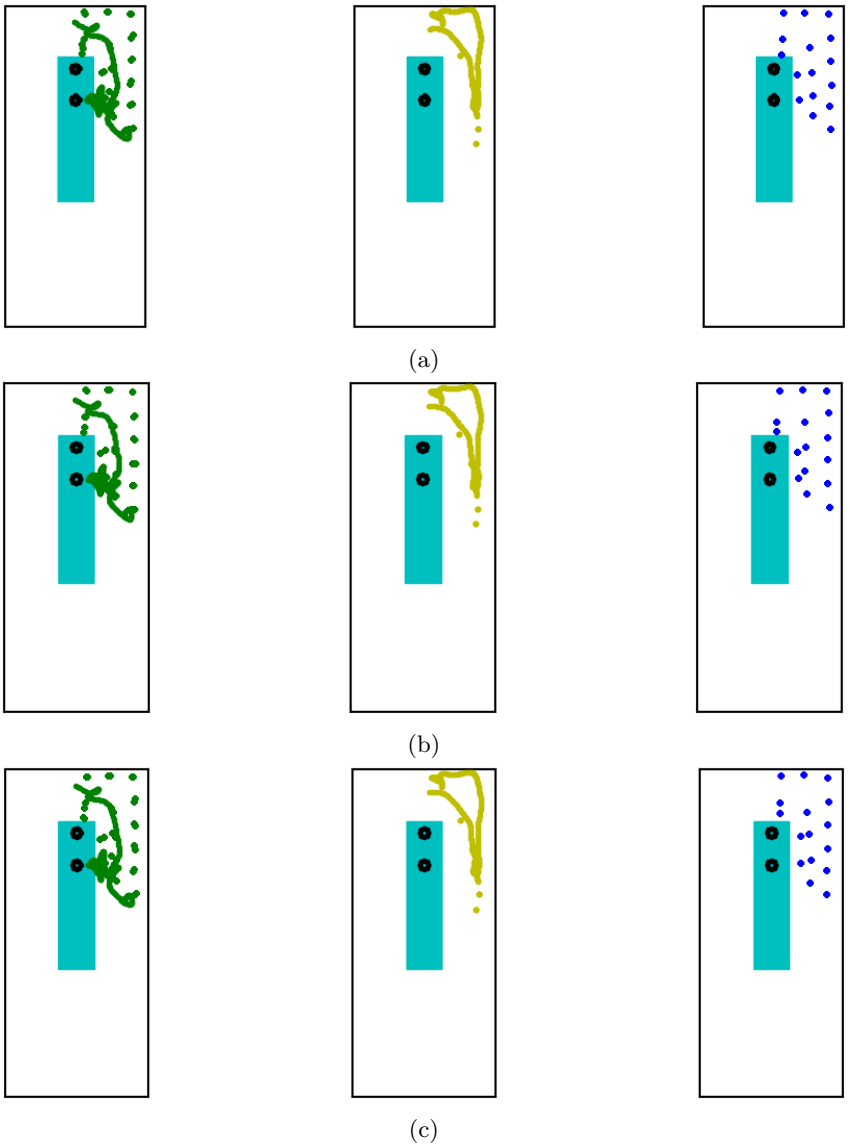


Figure 5.8: AV16.3 positions used in each data split (green = Train, yellow = Validation, blue = Test): (a) data partition 1 ($\text{AV16.3}_{\text{DP1}}$), (b) data partition 2 (($\text{AV16.3}_{\text{DP2}}$)), and (c) data partition 3 (($\text{AV16.3}_{\text{DP3}}$)).

ITC	SRP	DeepGCC	ASLNet	$\text{SELDNet}_{\text{XYZ}}$
ITC_{02A}	1081	**801**	1732	1877
ITC_{02B}	1390	**1111**	1550	1664
ITC_{03A}	1117	**741**	1548	1712
ITC_{03B}	1419	**1015**	1398	1467
ITC_{04A}	1183	**900**	1667	1774
ITC_{04B}	1206	**873**	1603	1750
ITC_{DP}	1229	**904**	1578	1711

ITC	SRP	DeepGCC	ASLNet	$\text{SELDNet}_{\text{XYZ}}$
MOTP	1229	**904**	1578	1711
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	**26.5**%	-29.1%	-39.2%
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	**-36.0**%	—	-75.6%	-89.3%
$\Delta^{\text{MOTP}}_{r_{ASLNet}}$	22.6%	**43.1**%	—	-7.8%
$\Delta^{\text{MOTP}}_{r_{SELDNet_{XYZ}}}$	28.2%	**47.2**%	7.3%	—

Table 5.12: Experiment 2 ITC detailed results: MOTP localization error in *mm* for each testing sequence (left, with partition average in the last row) and average MOTP localization error and relative improvements (right)

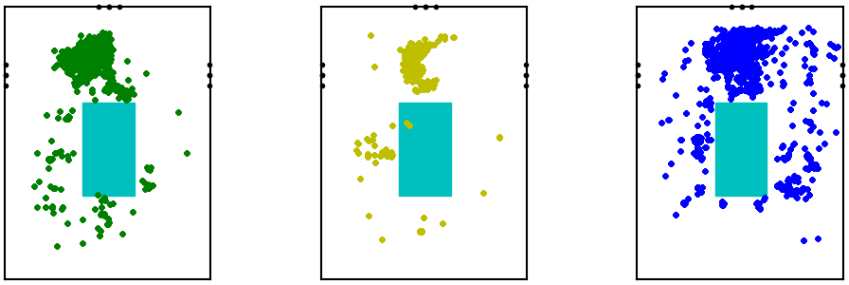


Figure 5.9: UPC positions used in the UPC_{DP} data partition (green = Train, yellow = Validation, blue = Test)

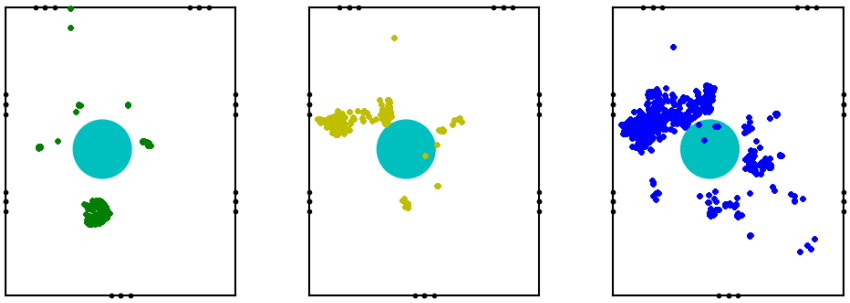


Figure 5.10: ITC positions used in the ITC_{DP} data partition (green = Train, yellow = Validation, blue = Test)

Figure 5.11 summarizes the performance results for experiment 2, where the DeepGCC gets the best performance in all datasets but CAV3D (where it comes second, with a slight degradation compared to SELDNet(1.7cm)). Although the other Deep Learning techniques achieve good accuracy results in areas where they have been trained in compared to DeepGCC, in cases like the ITC test sequences or the CAV3D C_{seq13} sequence, these methods perform poorly due to the training-testing conditions mismatch, even within the same data-set. In these scenarios, the DeepGCC system has a better behavior than SRP, getting an even more substantial improvement when the map refinement is applied, as will be discussed in Chapter 6.

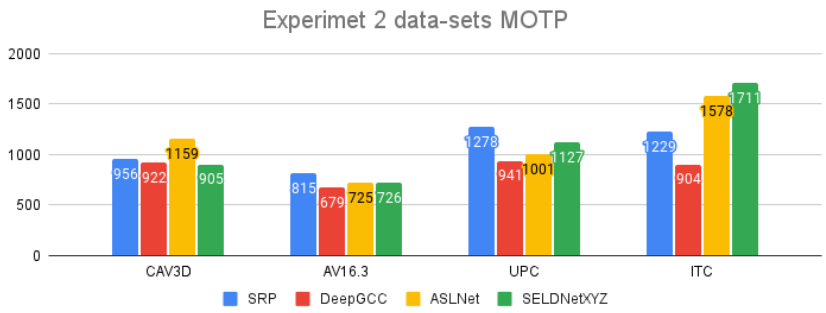


Figure 5.11: Experiment 2 summary results: MOTP localization error in mm for each dataset.

5.5.7 Experiment 3: Mismatched Room Geometry

In this experiment we evaluate the accuracy of all the methods in a new room environment that shares the same array geometry used for training. In order to achieve this goal, we first used the training subset of the best CAV3D partition to train the DeepGCC, ASLNet and $\texttt{SELDNet}_{XYZ}$ models, evaluating them on the AV16.3 testing sequences (which is similar in conditions to the results shown in Table (page 64), but using real data this Mon May 1 18:00:09 2023). The results are shown in Table 5.13. In all evaluated sequences, the best accuracy is obtained by the DeepGCC model.

trn: CAV3D tst: AV16.3	SRP	DeepGCC	ASLNet	$\texttt{SELDNet}_{XYZ}$
A_{seq01}	1036	**845**	3483	3478
A_{seq02}	693	**650**	3504	3632
A_{seq03}	739	**593**	3523	3604

trn: CAV3D tst: AV16.3	SRP	DeepGCC	ASLNet	$\texttt{SELDNet}_{XYZ}$
MOTP	815	**690**	3504	3575
$\Delta^{MOTP}_{r_{SRP}}$	—	**15.4**%	-329.9%	-338.5%
$\Delta^{MOTP}_{r_{DeepGCC}}$	**-18.2**%	—	-408.2%	-418.4%
$\Delta^{MOTP}_{r_{ASLNet}}$	76.7%	**80.3**%	—	-2.0%
$\Delta^{MOTP}_{r_{SELDNet_{XYZ}}}$	77.2%	**80.7**%	2.0%	—

Table 5.13: Experiment 3 results with mismatched room conditions, Models are trained with CAV3D and tested with the AV16.3 testing sequences: MOTP localization error in mm for each sequence (left) and average MOTP localization error and relative improvements (right)

It is important to note that the other Deep Learning methods (ASLNet and $\texttt{SELDNet}_{XYZ}$) severely fail where the environment geometry changes, leading to MOTP errors in the 4 meters range while our DeepGCC proposal keeps a performance which is very close to that obtained in the conditions of Experiment 2: Average MOTP for DeepGCC was 679mm in Experiment 2 (see Table 5.10 in page 60), and is 690mm here, which means an error increase of 11mm (1.62% relative) with mismatched room geometry and the same number of microphones (we only used one circular microphone array in AV16.3).

Table 5.14 shows that despite training the DeepGCC model in a room with a different geometry from that of AV16.3, the degradation of the accuracy of the localisation method in terms of MOTP is only reduced by 1.62% on average.

	$\texttt{DeepGCC}_{EXP2}$	$\texttt{DeepGCC}_{EXP3}$	$\Delta^{MOTP}_{r_{EXP2vsEXP3}}$
A_{seq01}	680	845	-24.26%
A_{seq02}	623	650	-4.65%
A_{seq03}	577	593	-2.77%
MOTP	679	690	-1.62%

Table 5.14: MOTP degradation effect when training and testing on AV16 room (EXP2) versus training on CAV3D room and testing on AV16 room (EXP3)

To evaluate Experiment 3 on the *CHIL-CLEAR* datasets, we used the UPC training sequences to train the DeepGCC method, and the evaluation was done on the ITC testing recordings. The average results obtained are shown in Table 5.15, in which the results for

the DeepGCC algorithm are again close to that obtained in experiment 2: Average MOTP for DeepGCC was 904mm in Experiment 2 (see Table 5.12 in page 62), and is 1019mm here, which means an error increase of 115mm (12.72% relative) with mismatched room geometry and a very different number of microphone arrays.

In this case, the ASLNet and $\texttt{SELDNet}_{\texttt{XYZ}}$ methods could not even be tested due to their architecture definition, which only allows the same amount of input microphone signals as the ones used in the training phase. Thus, the penalty for the other Deep Learning methods is even worse here.

trn: UPC tst: ITC	SRP	DeepGCC
ITC_{02A}	1081	**854**
ITC_{02B}	1390	**1153**
ITC_{03A}	1117	**947**
ITC_{03B}	1419	**1191**
ITC_{04A}	1183	**1032**
ITC_{04B}	1206	**962**

trn: UPC tst: ITC	SRP	DeepGCC
MOTP	1229	**1019**
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	**17.1**%
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	**-20.6**%	—

Table 5.15: Experiment 3 detailed results with *UPC* trained models tested over *ITC*: MOTP localization error in mm for each sequence (left) and average MOTP localization error and relative improvements (right)

Additionally, Table 5.16 shows the results regarding the degradation of localization accuracy in terms of MOTP when going from a model trained in matched conditions to a model trained in mismatched room conditions. Although the degradation in this experiment is greater than in the previous case (Table 5.14), the error committed can be assumed, since it translates into a loss of approximately 10cm in localization accuracy on average. It should be remembered that this degradation may be due to the fact that although only the geometry of the room is changed, maintaining the typology of the microphone arrays used, the number of arrays used increases, so that DeepGCC may face data that it has never seen during training. However, it can be concluded that despite this, DeepGCC is able to solve this issue with minimal degradation of accuracy.

	DeepGCC_{EXP2}	DeepGCC_{EXP3}	$\Delta^{\text{MOTP}}_{r_{EXP2vsEXP3}}$
ITC_{02A}	801	854	-6.62%
ITC_{02B}	1111	1153	-3.78%
ITC_{03A}	741	947	-27.80%
ITC_{03B}	1015	1191	-17.34%
ITC_{04A}	900	1032	-14.66%
ITC_{04B}	873	962	-10.19%
MOTP	904	1019	-12.72%

Table 5.16: MOTP degradation effect when training and testing on ITC room (EXP2) versus training on UPC room and testing on ITC room (EXP3)

In this third experiment, which results are summarized in Figure 5.12, we can clearly see the robustness of the DeepGCC system when facing a new room geometry condition, especially when compared with the other Deep Learning methods. ASLNet and $\texttt{SELDNet}_{\texttt{XYZ}}$ show a significant performance degradation in this experiment, and need a full retraining step if there is a change in the number of microphones or the array type.

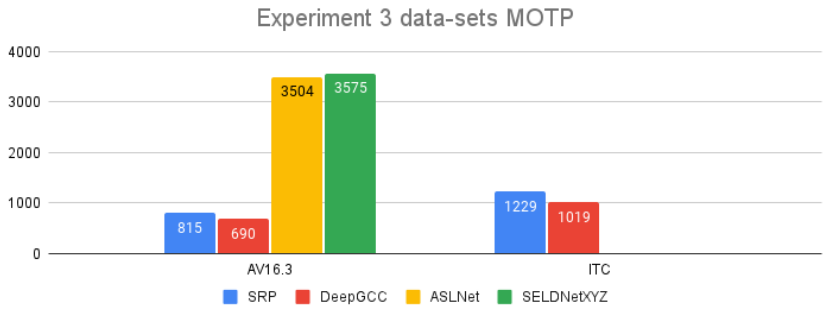


Figure 5.12: Experiment 3 summary results: MOTP localization error in mm for each dataset.

5.5.8 Experiment 4: Mismatched Room and Microphone Array Geometries

The fourth experiment carried out in this chapter aims at evaluating the robustness of the DeepGCC model when tested in an environment where both the room and the microphone array geometries have changed from the ones used during training. Standard methods ASLNet and $\texttt{SELDNet}_{\texttt{XYZ}}$ are not suitable under these constrains without a full re-training procedure, given their dependence on the microphone array configuration during the training phase, as already discussed before. This fact restricts again our comparison to the SRP vs. DeepGCC algorithms.

We first evaluate the models trained with the CAV3D training subset which have got better results on *Experiment* 2 on the UPC and ITC testing subsets. These results are shown in the upper and lower parts of Table 5.17, respectively. DeepGCC still get relevant improvements against the SRP algorithm. Recall that the amount of training material is relatively small here (less than 3.5 minutes of labeled speech for the CAV3D training partition, as discussed in Section 3.4).

trn: CAV3D$_{best}$ tst: UPC	SRP	DeepGCC
UPC$_{02A}$	1288	**1075**
UPC$_{02B}$	1362	**1153**
UPC$_{03A}$	1300	**1221**
UPC$_{03B}$	1302	**1136**
UPC$_{04A}$	1032	**838**
UPC$_{04B}$	1336	**1048**

trn: CAV3D$_{best}$ tst: UPC	SRP	DeepGCC
MOTP	1278	**1089**
$\Delta^{MOTP}_{r_{SRP}}$	—	**14.7%**
$\Delta^{MOTP}_{r_{DeepGCC}}$	**-17.3%**	—

trn: CAV3D$_{best}$ tst: ITC	SRP	DeepGCC
ITC$_{02A}$	1081	**909**
ITC$_{02B}$	1390	**1269**
ITC$_{03A}$	1117	**973**
ITC$_{03B}$	1419	**1305**
ITC$_{04A}$	1183	**1082**
ITC$_{04B}$	1206	**1097**

trn: CAV3D$_{best}$ tst: ITC	SRP	DeepGCC
MOTP	1229	**1102**
$\Delta^{MOTP}_{r_{SRP}}$	—	**10.3%**
$\Delta^{MOTP}_{r_{DeepGCC}}$	**-11.5%**	—

Table 5.17: Experiment 4 detailed results with models trained with the best *CAV3D* training partition, and evaluated in the *UPC* (upper) and *ITC* (lower) testing subsets: MOTP localization error in *mm* for each sequence (left) and average MOTP localization error and relative improvements (right)

Additionally, Table 5.18 shows a performance comparison of the DeepGCC model under *Experiments* 2, 3 and 4 on the ITC room. It should be noted that for these tests, our model has been trained with data extracted from the ITC, UPC and CAV3D room respectively. From mentioned results, it is observed that the model performance decreases as the mismatching conditions are increased.

Nevertheless, it should be noted that this performance degradation of the system proposed is about 20% for the hardest situation (Exp 4) with respect to the model trained in matched situations. This means that the difference between training in matched conditions (Exp 2) and full-mismatched conditions (Exp 4) is that for each centimeter estimated in the first case, in the second case an additional average error of 2 millimeters is committed.

	DeepGCC$_{EXP2}$	DeepGCC$_{EXP3}$	DeepGCC$_{EXP4}$	$\Delta^{MOTP}_{r_{EXP2vsEXP3}}$	$\Delta^{MOTP}_{r_{EXP2vsEXP4}}$
ITC$_{02A}$	801	854	909	-6.62%	-13.48%
ITC$_{02B}$	1111	1153	1269	-3.78%	-14.22%
ITC$_{03A}$	741	947	973	-27.80%	-31.31%
ITC$_{03B}$	1015	1191	1305	-17.34%	-28.57%
ITC$_{04A}$	900	1032	1082	-14.66%	-20.22%
ITC$_{04B}$	873	962	1097	-10.19%	-25.66%
MOTP	904	1019	1102	-12.72%	-21.90%

Table 5.18: MOTP degradation effect when training and testing on ITC room (EXP2: matched conditions) versus training on UPC room and testing on ITC room (EXP3: mismatched room geometry) versus training on CAV3D room and testing on ITC room (EXP4: mismatche room and sensor array geometry)

Next, we evaluated models trained with UPC training subset on CAV3D and AV16.3 test subsets, obtaining the results shown at the top and bottom of Table 5.19, respectively. Regarding the results of CAV3D when estimating with a model trained on UPC data, it is

observed that in this case it never improves to the results obtained by the SRP method. This may be due to the fact that the difference between the room geometries, as well as the topologies of the sensor arrays or the actors involved in the scenes are such that the DeepGCC network is not able to extract the insight needed to properly estimate.

	SRP	DeepGCC
C_{seq09}	**692**	801
C_{seq10}	**815**	972
C_{seq11}	**1092**	1435
C_{seq12}	**1080**	1215
C_{seq13}	**867**	1202

	SRP	DeepGCC
MOTP	**956**	1190
$\Delta^{MOTP}_{r_{SRP}}$	—	**-24.5%**
$\Delta^{MOTP}_{r_{DeepGCC}}$	**19.7%**	—

	SRP	DeepGCC
A_{seq01}	1036	**732**
A_{seq02}	693	**491**
A_{seq03}	739	**528**

	SRP	DeepGCC
MOTP	815	**578**
$\Delta^{MOTP}_{r_{SRP}}$	—	**29.03%**
$\Delta^{MOTP}_{r_{DeepGCC}}$	**-40.91%**	—

Table 5.19: Experiment 4 detailed results, training with UPC training sequences and evaluating with CAV3D (upper) and AV16.3 (lower) testing sequences: MOTP localization error in *mm* for each sequence (left) and average MOTP localization error and relative improvements (right)

Following the same reasoning, in the case of testing in the AV16.3 room, the effect is the opposite, so that in the UPC data there is some features that make the results better when estimating in the AV16.3 room than when DeepGCC is trained with data from the AV16.3 room itself as it can be seen in Table 5.20.

	DeepGCC_{EXP2}	DeepGCC_{EXP3}	DeepGCC_{EXP4}	$\Delta^{MOTP}_{r_{EXP2vsEXP3}}$	$\Delta^{MOTP}_{r_{EXP2vsEXP4}}$
A_{seq01}	680	845	732	-24.26%	-7.65%
A_{seq02}	623	650	491	-4.65%	21.19%
A_{seq03}	577	593	528	-2.77%	8.49%
MOTP	679	690	578	-1.62%	14.87%

Table 5.20: MOTP degradation effect when training and testing on AV16 room (EXP2: matched conditions) versus training on CAV3D room and testing on AV16 room (EXP3: mismatched room geometry) versus training on UPC room and testing on AV16 room (EXP4: mismatched room and sensor array geometry)

This experiment (which results are summarized in Figure 5.13) shows that DeepGCC method is able to reasonably cope with very different geometrical conditions, as compared to those used in the training stage. Our method still improves the results of the SRP algorithm (except in the mismatched CAV3D case), and, most importantly, avoiding the need of any type of re-training or fine tuning.

Although there is still work to be developed on this topic, the results obtained here are more than promising. The DeepGCC model has shown the ability to generalize the ASL problem in such a way that it becomes independent of the geometry of the room, as well as the topology of the sensor arrays used. This means that it is not necessary to

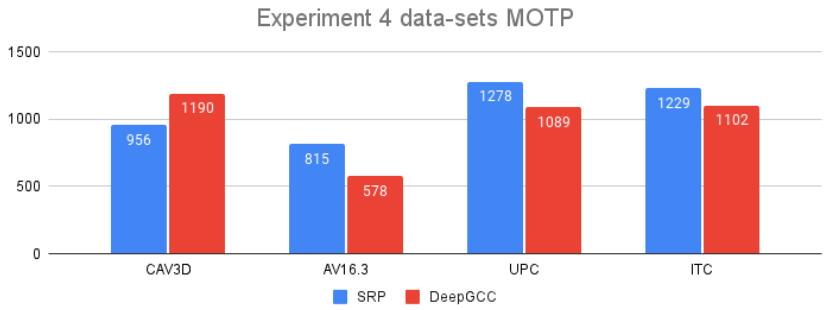


Figure 5.13: Experiment 4 summary results: MOTP localization error in mm for each dataset.

carry out data capture processes and their subsequent labeling when a new environment needs to be assessed, only assuming a minimum extra estimation error on the part of our model.

5.5.9 Experiment 5: Multi-Speaker Scenarios

In this last experiment we evaluate the performance of the DeepGCC method when facing scenarios with multiple and simultaneous speakers under the same constraints as Exp 2 (trainig data is extracted from the same room where testings are performed). As far as we know there are not any state-of-the art method yet capable of working under this conditions and yielding the x, y, z coordinates of each source. Therefore, we only compare the model proposed in this chapter against the SRP algorithm.

In order to train the model according to the above mentioned constraints, we use the $\mathsf{CAV3D}_{\mathsf{DP4}}$ partition, containing the multi-speaker sequences only in its testing split, as shown in Table 5.2 (page 52). This would mean that the DeepGCC model would be trained using only single speaker sequences, which would not allow for properly modeling the multi speaker sequences. To avoid this, we decided to generate artificial "multi-speaker" window signals, by mixing the windowed signals of randomly selected positions from the single-speaker training sequences. We uniformly select between 1, 2 or 3 of those sequences to generate a "virtual" multi-speaker acoustic frame. The process to generate the network inputs and outputs is the same we used in the previous experiments, as detailed in Section 5.5.2.1.

First, we evaluate the performance of this training procedure with the single speaker sequence $\mathsf{C}_{\mathsf{seq13}}$. Table 5.21 shows that multi-speaker DeepGCC model ($\texttt{DeepGCC}_{\texttt{MS}}$) outperforms the SRP algorithm and $\texttt{DeepGCC}_{\texttt{SS}}$ model trained for single speaker sequences (which is the same used for experiment 2, with the results shown in Table 5.9). The proposal is capable of better estimating the speaker position by including more complex sequences into the training stage.

	Speaker 1		
	SRP	DeepGCC$_{SS}$	DeepGCC$_{MS}$
C_{seq13}	867	798	**597**
$\Delta^{MOTP}_{r_{SRP}}$	—	7.96%	**31.14%**

Table 5.21: Experiment 5 results comparing the performance of DeepGCC$_{MS}$ model (DeepGCC trained for multi-speaker environments) against DeepGCC$_{SS}$ (DeepGCC trained for single-speaker environments): MOTP localization error in *mm*.

In a second step, we asses the accuracy of the DeepGCC model when facing scenarios where two simultaneous speech sources are active. Table 5.22 shows the precision error either for each speaker individually and in average. In all cases, DeepGCC gets better results than the SRP algorithm. Considering the localization results of the most powerful speaker (speaker 1), the average localization accuracy of the single speaker (Table 5.9 of Experiment 2) improves from 922mm to 827mm. This effect can also be seen when comparing the location accuracy of the second speaker against the single speaker scenario results of Experiment 2 in the same room, obtaining a result of 724mm. Focusing on the average localization accuracy of all speakers that appear within the scene (790mm), this is still better than in Experiment 2. This shows again that the introduction of more complex training sequences has a positive influence on the ASL task.

	Speaker 1		Speaker 2		Average	
	SRP	DeepGCC	SRP	DeepGCC	SRP	DeepGCC
C_{seq22}	939	**782**	1695	**803**	1317	**793**
C_{seq23}	1369	**1000**	986	**641**	1178	**821**
C_{seq24}	1025	**690**	1608	**830**	1317	**760**
Avg_{MOTP}	1136	**827**	1393	**724**	1265	**790**
$\Delta^{MOTP}_{r_{SRP}}$	—	27.16%	—	48.03%	—	37.51%

Table 5.22: Experiment 5 results for two simultaneous speakers sequences: MOTP localization error in *mm*.

Finally, in Table 5.23 we present the results when evaluating sequences with three simultaneous speakers in terms of precision accuracy. Comparing the results obtained with those from Experiment 2, we obtain the same conclusion as for the two-speaker case. The precision for the most powerful speaker increases from 922mm to 696mm. Moreover, when analyzing the performance of the DeepGCC model on the second most powerful speaker, the accuracy also improves to 595mm. On average, a better result is obtained than for the single speaker case with a accuracy of 875mm. Only when locating the third speaker a worse result is obtained. This is due to the fact that the acoustic power with which this speaker emits sound is much lower than the others, thus making it harder to be located.

	Speaker 1		Speaker 2		Speaker 3		Average	
	SRP	DeepGCC	SRP	DeepGCC	SRP	DeepGCC	SRP	DeepGCC
C_{seq25}	916	**633**	1023	**557**	2004	**1312**	1314	**834**
C_{seq26}	1357	**860**	1173	**692**	2692	**1395**	1741	**982**
Avg_{MOTP}	1039	**696**	1065	**595**	2196	**1335**	1433	**875**
$\Delta^{MOTP}_{r_{SRP}}$	—	32.98%	—	44.15%	—	39.20%	—	38.92%

Table 5.23: Experiment 5 results for three simultaneous speakers sequences: MOTP localization error in *mm*.

Overall, this final experiment, summarized in Figure 5.14, shows that the DeepGCC proposal is able to face multi-speaker realistic scenarios. Our method is capable of outperforming the SRP algorithm with up to three simultaneous speakers with relative improvements over SRP near 40%. It is important to notice that, beside to properly addressing the problem of localizing multiple speakers, it has been proven that the inclusion of more complex sequences during the training phase allows the proposed DeepGCC model to improve the localization of individual speakers.

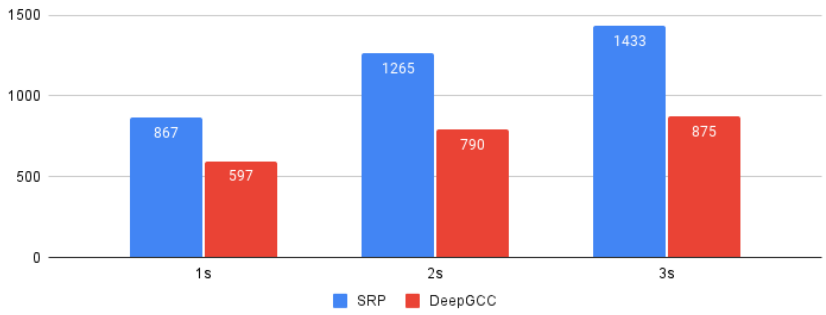


Figure 5.14: Experiment 5 summary results: MOTP localization error in mm for different number of simultaneous speakers.

5.6 Conclusions

In this chapter we have described a novel ASL method based on deep learning, conceived to deal with mismatched geometry conditions between the training and testing scenarios. Being capable of working outside training conditions is a necessity in real world applications, and it is not achieved in current state-of-the-art DNN methods. Our ASL system has proved its accuracy in diverse scenarios where other state-of-the-art methods fail. We also address multi-speaker environments.

Four main conclusions can be drawn from our experiments:

1. In scenarios with matched training-testing conditions, DeepGCC performs better than the other methods (classical or learning based), $\texttt{SELDNet}_{\texttt{XYZ}}$ gets a performance

similar to DeepGCC in all scenarios but in ITC room, where its localization acuracy becomes inconsistent.

2. Changing the room geometry significantly affects the performance of other DNN methods, while DeepGCC remains stable in terms of accuracy, outperforming the rest of DNN methods and the classical SRP method.

3. When both the room geometry and the microphone array configuration change, our system is still capable of outperforming the SRP algorithm (even in cases where the amount of training material is relatively small), while the other DNN methods cannot even be tested without a full retraining procedure.

4. Under multi-speaker conditions, DeepGCC method outperforms the classical SRP algorithm, where other DNN-based state-of-the-art model can not even be evaluated due to their model definition constraints. In addition. it has been proved that the inclusion of multispeaker sequences within training phases improves the single speaker localization accuracy.

From all of the above, we can finally conclude that our method is consistently more accurate than both the SRP classical method and other state of the art DNN strategies, specially in conditions where the testing room or microphone array configuration is physically different, or the source position significantly differs from those available in the training data. This is specially relevant, as no retraining is needed when there are such changes, thus opening the possibility of developing a multi-environment DNN system, which would only need to be trained once.

Chapter 6

Acoustic Map Refinement Techniques

. . . Everything else is secondary.

Steve Jobs

6.1 Summary

This chapter includes the description of part of the work published in the papers "Towards Domain Independence in CNN-based Acoustic Localization using Deep Cross Correlations" [108] and "Acoustic source localization with deep generalized cross correlations" [109].

Many existing methods for Acoustic Source Localization (ASL), as discussed in Chapter 5, focus on computing the acoustic power map from signals received by the microphone array. The acoustic power map provides information about the 3D position of the acoustic source by identifying its local maxima. However, in realistic scenarios with signal noise and reverberation effects, the acoustic power map becomes more challenging to analyze. It contains numerous local maxima that do not directly correspond to real acoustic sources. To address this issue, several acoustic power map refinement methods [25] have been proposed in the literature. The goal is to process the acoustic power map and obtain a refined map that is less affected by noise and reverberation, leading to improved localization estimation. However, these refinement methods involve complex iterative optimization techniques and subspace decomposition, resulting in high computational complexity and compromising real-time performance. In this chapter, we introduce two refinement methods for acoustic power map-based systems. The first method utilizes optimization techniques to leverage the characteristics of Deep Generalized Cross-Correlation (DeepGCC) signals described in Chapter 5. The second method involves a Convolutional Neural Network (CNN) model that refines the map end-to-end using training with synthetic data. As discussed later, the CNN-based refinement approach is comparable to the

optimization-based approach in terms of Acoustic Source Localization (ASL) accuracy but is computationally less demanding.

6.2 Introduction

An acoustic power map is created from the set of microphone signals to estimate the positions of active acoustic sources. Estimating source positions is straightforward under free-field conditions and single speaker sequences, requiring identification of the global maximum. However, in multi-speaker environments, iterative methods are necessary to search for N local maxima within the acoustic power map.

Finding maxima in an acoustic power map, whether global or local, becomes challenging when noise and reverberation effects are present. We aim to assess the feasibility of refining acoustic power maps by removing these adverse effects. We will evaluate two approaches: *(i)* optimization-based algorithms and *(ii)* learning-based methods. Optimization-based algorithms utilize subspace decomposition techniques with iterative optimization algorithms, such as the system proposed in [25]. Learning-based methods leverage the capability of Deep Neural Networks (DNNs) to analyze large amounts of data and distinguish relevant components from those influenced by noise and reverberation [133].

In this chapter, we introduce two approaches for refining acoustic power maps generated by the DeepGCC model, as described in Chapter 5. Firstly, we adapt the method proposed in [25] to incorporate the Gaussian basis component enforced in the DeepGCC signal. Secondly, we propose the use of a VNet-based model [102] for refining the acoustic maps, which achieves similary to the optimization-based method while requiring lower computational resources.

The contributions of this work are as follows:

- Adaptation of the method proposed in [25] to refine acoustic maps generated by the DeepGCC model, leveraging the Gaussian properties of the estimated signals.
- Development of a deep learning-based approach for refining acoustic power maps obtained from the DeepGCC model.
- Exploration of advanced machine learning techniques such as domain adaptation and training with synthetic data.

6.3 Problem Statement

Let us define the scalar function $f : \mathcal{X} \rightarrow \mathbb{R}$, with $\mathcal{X} = (\mathbf{r}, \mathbf{p}, \mathbf{q})$, that models the acoustic power generated from an acoustic source at position $\mathbf{r} = (r_x, r_y, r_z)^\top$, received in the pair

of microphones with coordinates $\mathbf{p} = (m_{1,x}, m_{1,y}, m_{1,z}, m_{2,x}, m_{2,y}, m_{2,z})^\top$ and when the pair is steered at position $\mathbf{q} = (q_x, q_y, q_z)^\top$. Discretizing $\mathcal{X}$ we define the set of function evaluations $\mathcal{F} = \left\{ f\left(\mathbf{r}_i, \mathbf{p}_j, \mathbf{q}_k\right) \;\; \mathbf{r}_i \in \mathcal{R}, \mathbf{p}_j \in \mathcal{P}, \mathbf{q}_k \in \mathcal{Q} \right\}$, where $\mathcal{R} = \{\mathbf{r}_1, \ldots, \mathbf{r}_{N_\mathcal{R}}\}$ is the set of potential source positions, $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_{N_\mathcal{P}}\}$ is the set of sensor pair positions and $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_{N_\mathcal{Q}}\}$ is the set of search space positions to explore. For simplicity, we assume that $\mathcal{R} = \mathcal{Q}$, meaning that the search space of possible source positions matches the steering search space to compute the acoustic power.

We approximate the acoustic power generated by a single acoustic source positioned at $\mathbf{r}_i$ at the steered position $\mathbf{q}_k$ as follows:

$$\widehat{\text{apm}}\left(\mathbf{q}_k, \mathbf{r}_i\right) = \frac{1}{N_\mathcal{P}} \sum_{\forall \mathbf{p}_j \in \mathcal{P}} f\left(\mathbf{r}_i, \mathbf{p}_j, \mathbf{q}_k\right), \tag{6.1}$$

We then define $\mathcal{R}_o = \{\mathbf{r}_{o,1}, \ldots, \mathbf{r}_{o,N_o}\}$ as the set of N_o positions actually occupied with an active accoustic source at any given time. We define the following binary occupancy function:

$$\omega(\mathbf{r}, \mathcal{R}_o) = \begin{cases} 1 & \mathbf{r} \in \mathcal{R}_o \\ 0 & \text{otherwise} \end{cases} \tag{6.2}$$

Using ω we approximate the acoustic power map generated by the set of active acoustic sources as follows:

$$\widehat{\text{apm}}\left(\mathbf{q}_k, \mathcal{R}_o\right) = \frac{1}{N_\mathcal{P}} \sum_{\forall \mathbf{r}_i \in \mathcal{R}} \omega\left(\mathbf{r}_i, \mathcal{R}_o\right) \sum_{\forall \mathbf{p}_j \in \mathcal{P}} f\left(\mathbf{r}_i, \mathbf{p}_j, \mathbf{q}_k\right), \tag{6.3}$$

Using the model proposed in Equation (6.1) over all the $N_\mathcal{Q}$ positions in $\mathcal{Q}$, the following vector $\hat{\mathbf{y}}$ of acoustic power map estimations is defined:

$$\hat{\mathbf{y}} = \left(\widehat{\text{apm}}\left(\mathbf{q}_1, \mathcal{R}_o\right) \cdots \widehat{\text{apm}}\left(\mathbf{q}_{N_\mathcal{Q}}, \mathcal{R}_o\right)\right) \tag{6.4}$$

By using Equation (6.1) and Equation (6.3), $\hat{\mathbf{y}}$ can be linearly decomposed as follows:

$$\hat{\mathbf{y}} = \mathbf{M}\boldsymbol{\omega} \quad \mathbf{M} \in \mathbb{R}^{N_\mathcal{Q} \times N_\mathcal{R}} \quad \mathbf{M}_{k,i} = \widehat{\text{apm}}\left(\mathbf{q}_k, \mathbf{r}_i\right) \tag{6.5}$$

and where

$$\boldsymbol{\omega} = \left(\omega(\mathbf{r}_1, \mathcal{R}_o), \ldots, \omega(\mathbf{r}_{N_\mathcal{R}}, \mathcal{R}_o)\right) \tag{6.6}$$

Note that $\boldsymbol{\omega}$ is ideally a sparse binary vector of size $N_\mathcal{R}$, where only those positions corresponding to the set $\mathcal{R}_o$ of active acoustic sources has a non-zero value. Our map refinement methods consists of using Equation (6.5) to approximate real acoustic measurements in terms of $\boldsymbol{\omega}$, which is later used as the refined map.

6.4 Model Proposals

The main goal of the proposals presented in this chapter is to develop a system capable of refining an acoustic power map in order to reduce the effect of noise and reverberation in the acoustic power maps and thus, improving the ASL accuracy of the original map. This process follows the scheme shown in Figure 6.1, where a dense acoustic power map is fed to the refining block. This allows us to obtain a sparse representation of this previous map. It can be observed that in dense representations is harder to find an accurate maximum value whereas in sparse representations this value is easier to locate.

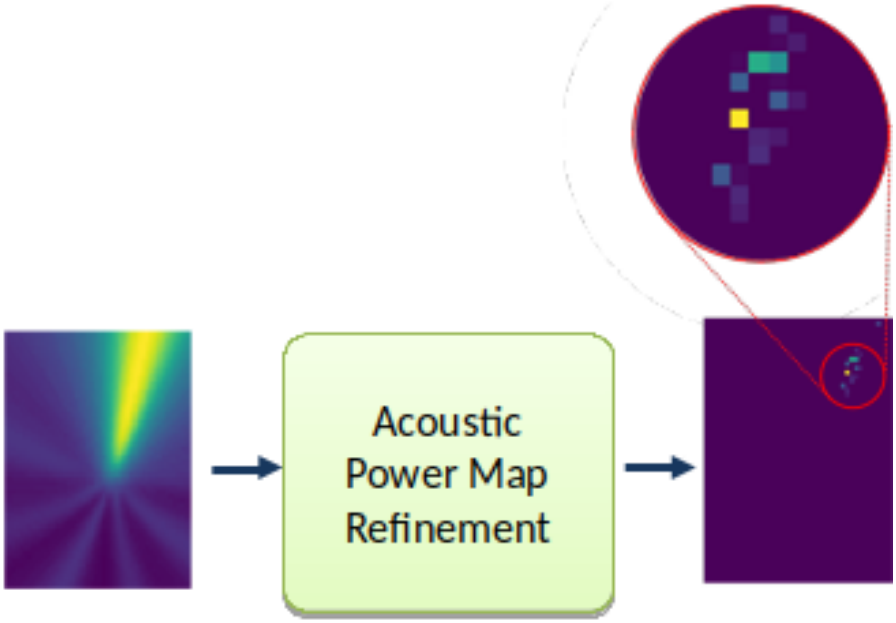


Figure 6.1: System Architecture with sample subsystem input and output.

Our proposal consists of a single module which refines the acoustic power maps built with DeepGCC signals. This will be addressed from two different approaches: *(i)* optimization approach and *(ii)* learning approach, which will be described next.

6.4.1 Optimization-Based proposal

We base this approach on the idea described in [25], where they present a sparse representation of the SRP acoustic map, that leads to better ASL accuracy. In this approach we define the f function in Equation (6.1) as:

$$f\left(\mathbf{r}_i, \mathbf{p}_j, \mathbf{q}_k\right) = e^{-\frac{\left(\Delta\tau\left(\mathbf{q}_k, \mathbf{p}_j\right) - \Delta\tau\left(\mathbf{s}_i, \mathbf{p}_j\right)\right)^2}{2\sigma^2}}, \tag{6.7}$$

where $\Delta\tau\left(\mathbf{q}_k, \mathbf{p}_j\right)$ is the Time Difference of Arrival (TDoA) between the steering point $\mathbf{q}_k$ and the microphone pair $\mathbf{p}_j$, and $\Delta\tau\left(\mathbf{s}_i, \mathbf{p}_j\right)$ is the TDoA between the acoustic source position $\mathbf{s}_i$ and the same microphone pair $\mathbf{p}_j$. This function correspond to the Gaussian-

like functions defined in Equation (5.10). This is a natural choice since we created the acoustic power map using the DeepGCC signals. In the case of Equation (6.5), $\boldsymbol{\omega}$ are the unknown parameters of the model, and vector $\hat{\mathbf{y}}$ can be seen as the acoustic power map data synthesized by the proposed model as a function of weight vector $\boldsymbol{\omega}$. Figure 6.2

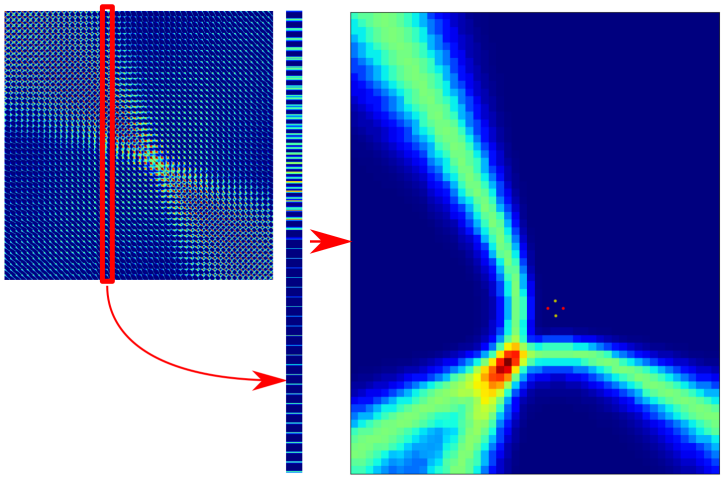


Figure 6.2: Geometrical interpretation of the contents of matrix $\mathbf{M}$, defined for an active acoustic source located at the position of the high activation area (red + black colors) and two microphone pairs. This shows an example where two orthogonal microphone pairs are shown as small red and yellow dots.

shows $\mathbf{M}$ as a heat map for a practical case. In this example $\mathcal{Q}$ is built as an uniform grid with positions corresponding to a plane parallel to the room floor at the speaker's mouth height and $N_{\mathcal{P}} = 2$. Note that the i^{th} column of $\mathbf{M}$ corresponds to the theoretical acoustic power map obtained from a virtually active source at position $\mathbf{q}_k \in \mathcal{Q}$.

We see that the acoustic power map, obtained from the selected column of the $\mathbf{M}$ basis, shows a geometrical pattern. In particular, this pattern corresponds to the intersection of two hyperbolas. The geometry of this pattern depends on the relative position between the source and the corresponding microphone pairs, depicted as yellow and red points in the right graphic of Figure 6.2.

In a real scenario, we would obtain $\hat{\mathbf{y}}$, containing the acoustic power maps measurements $apm\,(\mathbf{q}_k)$ by using the DeepGCC signals defined in Equation (5.11), as follows:

$$\hat{\mathbf{y}}_{DeepGCC} = \left(\text{apm}\,(\mathbf{q}_1) \cdots \text{apm}\left(\mathbf{q}_{N_{\mathcal{Q}}}\right)\right) \;\forall \mathbf{q}_i \in \mathcal{Q} \tag{6.8}$$

Our aim is to find a vector $\boldsymbol{\omega}$ that recovers $\hat{\mathbf{y}}_{DeepGCC}$ using model $\mathbf{M}$. Note that $\hat{\mathbf{y}}_{DeepGCC}$ includes the errors introduced by DeepGCC signals and thus $\hat{\mathbf{y}}_{DeepGCC} \approx \mathbf{M}\boldsymbol{\omega}$.

Forcing $\boldsymbol{\omega}$ to be only composed by the active positions where an acoustic source is present, is a property given by the construction of model $\mathbf{M}$. A straightforward and natural choice to find $\boldsymbol{\omega}$ is to solve the following linear least squares optimization problem:

$$\hat{\boldsymbol{\omega}} = \underset{\boldsymbol{\omega}}{\text{argmin}} \|\hat{\mathbf{y}}_{DeepGCC} - \mathbf{M}\boldsymbol{\omega}\|^2 \tag{6.9}$$

When $\hat{\mathbf{y}}_{DeepGCC}$ contains errors, $\mathbf{M}$ might not be an accurate model. Then, solving Equation (6.9) produces an estimated refined map $\hat{\boldsymbol{\omega}}$, where the source of the acoustic power is spread around a large amount of positions. We propose to exploit the fact that only a small number of simultaneous active sources are present. This condition implies forcing $\hat{\boldsymbol{\omega}}$ to have as many zeroes as possible, therefore being a sparse vector. This can be expressed as the following optimization problem:

$$\hat{\boldsymbol{\omega}} = \underset{\boldsymbol{\omega}}{\operatorname{argmin}} \|\hat{\mathbf{y}}_{DeepGCC} - \mathbf{M}\boldsymbol{\omega}\|^2 + \lambda\|\boldsymbol{\omega}\|_1 \tag{6.10}$$

where λ is the *Lagrange* multiplier that balances the relevance of the l_1 term in the optimization problem. Equation (6.10) is a convex optimization problem and can be efficiently solved with iterative algorithms independently presented and popularized under the names of Least Absolute Shrinkage Selection Operator (LASSO) [134] and Basis Pursuit Denoising [135].

We expect $\hat{\boldsymbol{\omega}}$ to be a *refined acoustic power map*, where the i^{th} position of $\hat{\boldsymbol{\omega}}$ represents the amount of acoustic power estimated at position $\mathbf{r}_i$. In figure 6.3, a visual refinement procedure example can be observed. A dense acoustic power map is vectorized in order to apply the LASSO algorithm on it, obtaining the sparse $\hat{\boldsymbol{\omega}}$ representation. This vector $\hat{\boldsymbol{\omega}}$ only assigns acoustic power where the source might be presented. Thus, we expect $\hat{\boldsymbol{\omega}}$ to be a vector with very few non-zero elements, corresponding to the source positions.

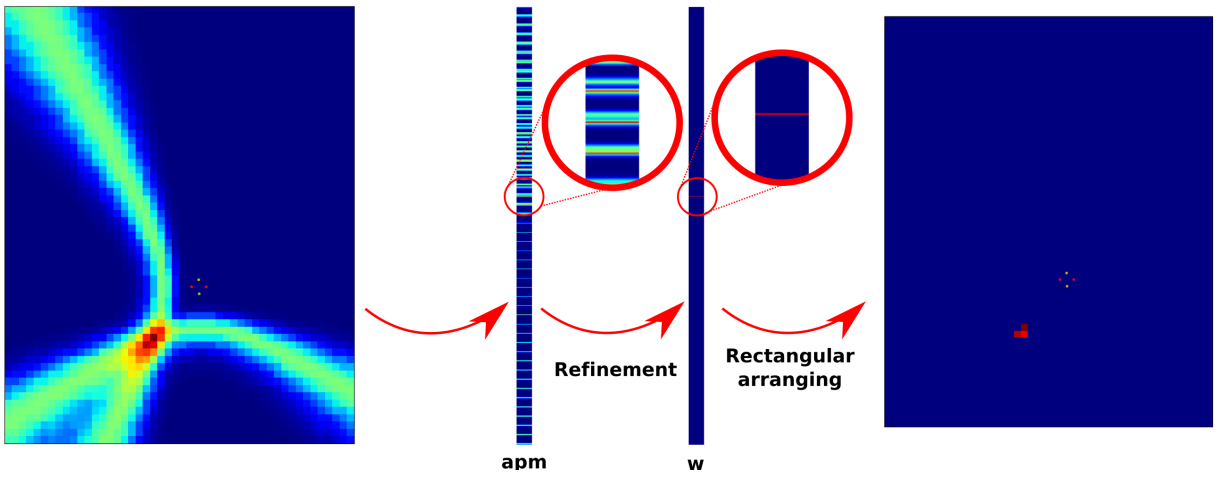
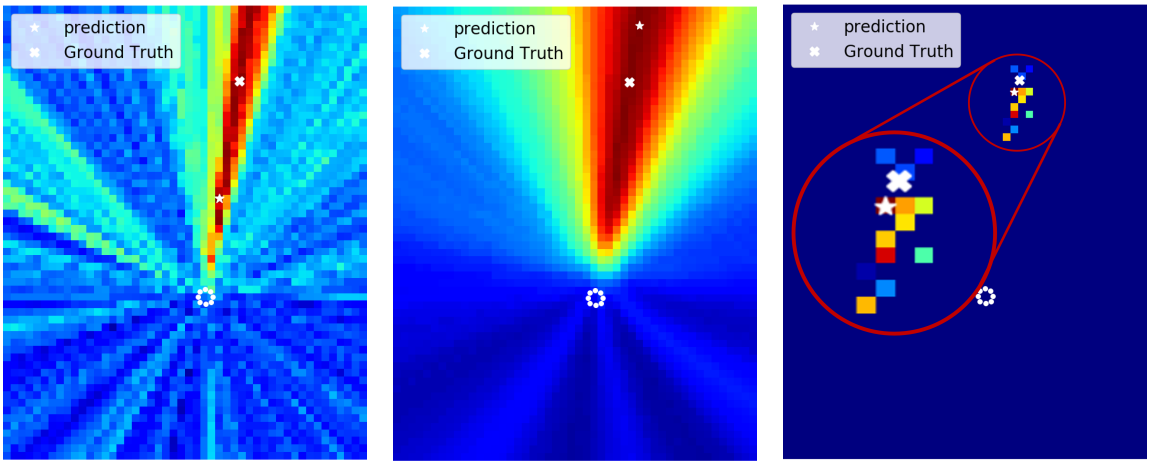


Figure 6.3: Refinement example.

Once $\hat{\boldsymbol{\omega}}$ is calculated by solving Equation (6.10), we obtain the source position estimate, in a single speaker scenario, as $\hat{\mathbf{r}} = \mathbf{q}_k$, where $k = \underset{k}{\operatorname{argmax}}\{\boldsymbol{\omega}\}$ with $\mathbf{q}_k \in \mathcal{Q}$. When locating multiple sources, this process is repeated as many times as active sources appear in the scene by searching for local maximums within the acoustic power map. In Section 6.5 we will refer to this refinement method as $\texttt{DeepGCC}_{\texttt{LASSO}}$.

Figure 6.4 shows an example of the LASSO-based refinement procedure. The acoustic power map generated with the GCC signals calculated directly from the acoustic signals



(a) GCC apm $(\mathbf{q}_k)$ (b) DeepGCC apm $(\mathbf{q}_k)$ (c) Refined apm $(\mathbf{q}_k)$: $\hat{\boldsymbol{\omega}}$

Figure 6.4: Map examples.

received by the sensors can be seen in Figure 6.4a. Acoustic maps generated with these signals have a noisy nature thus, finding an adequate estimation of the source location is challenging. Figure 6.4b, shows the equivalent acoustic map generated by using the DeepGCC signals calculated from GCC signals. Here it can be seen how the map becomes much cleaner and smoother than the previous one. However, there is still a large area where the estimated acoustic power is high, even though the estimation of the source position improves with respect to the GCC-based map. Finally, Figure 6.4c shows the refined version of the map obtained from DeepGCC signals by applying the LASSO optimization approach (and a zoomed version of the relevant area). It shows how the areas of high acoustic power are reduced to a few points in the searching space. Furthermore, the estimation of the acoustic source position improves in terms of location accuracy.

6.4.2 Learning-Based Proposal

Optimization-based refinement methods have two main drawbacks. First, they heavily rely on the ability of the model (i.e. $\mathbf{M}\boldsymbol{\omega}$) to accurately reproduce real acoustic power maps. Using an oversimplistic model that does not include real effects, such as distortion and signal multipath, can lead to unsuccesful map refinements. Second, solving the optimization problem of Equation (6.10) is computationally complex, compromising real-time performance.

In this section, a learning-based method is proposed to solve the limitations of optimization-based methods. It consists of a neural network regressor that estimates the refined acoustic map directly from the acoustic power map measurements. Our approach is based on the VNET neural model [102], widely used in 3D semantic segmentation and image classification. The VNET model is based on a encoder-decoder CNN and it is suitable to process 3D volumetric data for regression as is the case of acoustic map re-

finement. The VNET is also a relatively small model and can be run in real time during inference.

From Equation (6.8), $\hat{\mathbf{y}}_{DeepGCC}$ represents the input signal, arranged as a 3D matrix and assuming $\mathcal{Q}$ is sampled as a 3D regular grid. The output of the VNET is trained to output a refined power map $\mathbf{z}$ where *(i)* it is the same size as the input and *(ii)* each acoustic source inside the environment $\mathbf{r}_i \in \mathcal{R}_o$ with $i \in [1, N_o]$ is represented by a Gaussian function whose $3D$ mean $\boldsymbol{\mu}_i$ is centered at the actual source position ($\mathbf{r}_i = \boldsymbol{\mu}_i$), and with sigma σ:

$$\mathbf{z}\left(\mathbf{q}_k\right) = \sum_{\forall i \in N_{\mathcal{R}}} e^{-\frac{\|\mathbf{q}_k - \boldsymbol{\mu}_i\|^2}{2\sigma^2}} \quad \text{with } \mathbf{q}_k \in \mathcal{Q} \tag{6.11}$$

so that $\mathbf{z} = \{z\left(\mathbf{q}_1\right), \ldots, z\left(\mathbf{q}_{N_\mathcal{Q}}\right), \mathbf{q}_k \in \mathcal{Q}\}$. Therefore, we implement the VNET-based model (topology shown in Figure 6.5), in order to estimate $\mathbf{z}$ as $\hat{\mathbf{z}} = f_{\text{VNET}}\left(\mathbf{y}\right)$. In Section 6.5 we will refer to this refinement method as $\text{DeepGCC}_{\text{VNET}}$.

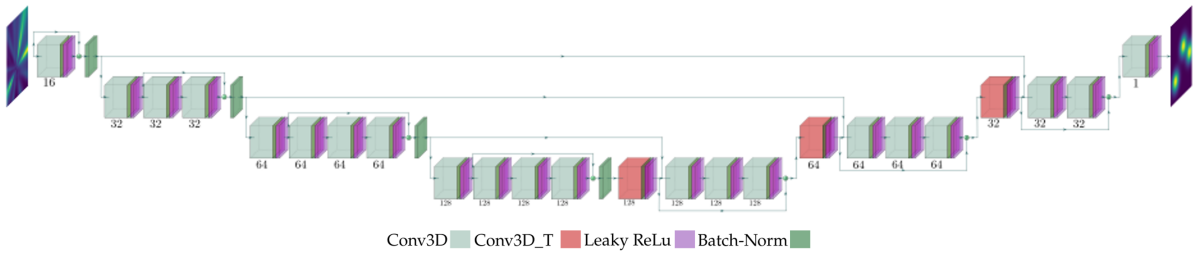
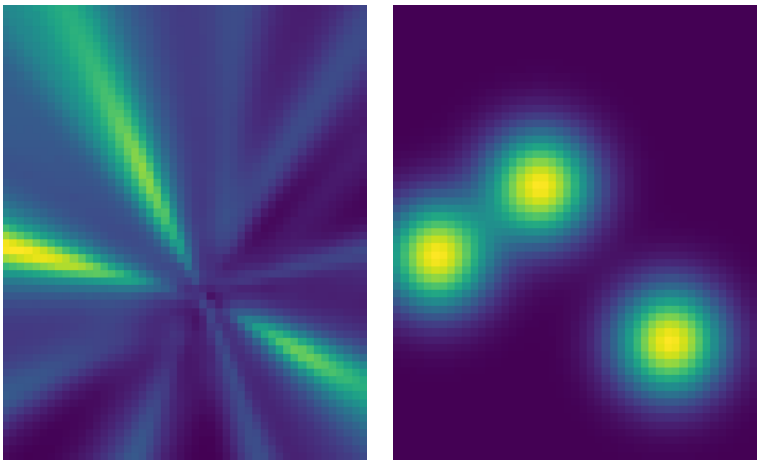


Figure 6.5: Model topology.

The model depicted in Figure 6.5 consists of four convolutional blocks and four deconvolutional blocks. Each convolutional block comprises a three-dimensional convolutional layer with a kernel size of 5 and stride of 1 step, followed by a batch normalization layer and a Leaky ReLU activation. The number of layers increases progressively in each block, starting with 1 set and progressing to 2, 3, and finally 3 sets. Additionally, the model gains depth in each block, with 16, 32, 64, and 128 layers respectively. On the other hand, the four deconvolutional blocks consist of a transposed three-dimensional convolutional layer, a batch normalization layer, and a Leaky ReLU activation, along with an equivalent number of sets of convolutional layers as found in the corresponding convolutional block.

Figure 6.6 shows an example of the refinement map retrieved by the proposed VNET-based model. Figure 6.16a map is generated by DeepGCC model and further used as an input of the VNET-based proposal. This approach transforms the speakers activation zones form beam-shaped areas into Gaussian-shaped areas as shown in Figure 6.16b. This Gaussian-shaped zones allow us to better locate sources inside the environment since this



(a) DeepGCC (b) $\text{DeepGCC}_{\text{VNET}}$

Figure 6.6: Map examples.

kind of acoustic power maps can be modeled by a mean vector and a variance vector according to Equation (6.11).

6.5 Experimental Work

In this section we discuss the experimental conditions we used for testing our proposals, the training strategy we followed and the relevant metrics to evaluate them. Finally, we present a summary of the results of the main experiments carried out.

Our aim is to assess the effect of the proposed refinement procedures in the ASL estimates for both single speaker and multi-speaker scenarios.

6.5.1 Experimental Setup

The refinement proposals based on LASSO [134] ($\text{DeepGCC}_{\text{LASSO}}$) and VNET [102] ($\text{DeepGCC}_{\text{VNET}}$, with additional variants) are compared to the SRP algorithm [18–23] and also the DeepGCC model estimations, where the DeepGCC model is the proposal detailed in Chapter 5.

The $\text{DeepGCC}_{\text{LASSO}}$ proposal needs no training apart from that required to generate the DeepGCC model, while the $\text{DeepGCC}_{\text{VNET}}$ system needs to train both the DeepGCC model and the refinement block (which is trained using a semi-synthetic strategy, as will be detailed in Section 6.5.2.1). Regarding the experimental data, we use the *CAV3D* sequences, splitted as shown in Table 6.1, where $\text{CAV3D}_{\text{DP4}}$ is a partition with multi-speaker sequences within it. "*Data Partitions*" and sequences used in further experiments are exactly the same as those used on Chapter 5 experiments. All of them are fully detailed in Table 3.2.

Data Partition	Training	Validation	Testing
$\text{CAV3D}_{\text{DP1}}$	$C_{\text{seq[06-09, 11, 20, 21]}}$	C_{seq13}	$C_{\text{seq[10, 12]}}$
$\text{CAV3D}_{\text{DP2}}$	$C_{\text{seq[06-08, 10, 12, 20, 21]}}$	C_{seq13}	$C_{\text{seq[09, 11]}}$
$\text{CAV3D}_{\text{DP3}}$	$C_{\text{seq[06-10, 12, 20, 21]}}$	C_{seq11}	C_{seq13}
$\text{CAV3D}_{\text{DP4}}$	$C_{\text{seq[06-10, 12]}}$	$C_{\text{seq[11, 20, 21]}}$	$C_{\text{seq[13, 22-26]}}$

Table 6.1: Data split used in the experiments, showing sequences.

Due to the fact that two acoustic map refinement strategies (optimization-based and learning-based) are proposed in this chapter, three experiments are carried out. The first two experiments focus on optimizing the model parameters in case of the proposed LASSO-based approach for the first experiment, while the second one explores different training approaches for the proposed VNET model. The last experiment evaluates and compares the performance of both methods under the same test set. The detailed description of these experiments is:

- **Experiment 1.** $\text{DeepGCC}_{\text{LASSO}}$ map refinement: This experiment evaluates the quality of the map refinement generated by the $\text{DeepGCC}_{\text{LASSO}}$ algorithm, determining the importance of the λ parameter as well as the global accuracy in terms of ASL error. We train the DeepGCC model as explained in Section 5.5.2 (page 54) with data partitions shown in Table 6.1. The LASSO method ($\text{DeepGCC}_{\text{LASSO}}$) is applied along the average acoustic map of the acoustic maps of the last half second.

- **Experiment 2.** $\text{DeepGCC}_{\text{VNET}}$ map refinement: The final goal of this experiment is also evaluating the global accuracy of the model ($\text{DeepGCC}_{\text{VNET}}$) in terms of ASL error. Additionally, we describe different strategies that have been applied in order to improve the refinement procedure, namely:

  1. We synthetically train the $\text{DeepGCC}_{\text{VNET}}$ model by generating virtual Gaussian-like maps with random positions within it.

  2. We train the model by means of a semi-synthetic strategy (using both synthetic and real data) in order to improve the previous fully synthetic approach.

  3. We test the influence of a different function loss when training the $\text{DeepGCC}_{\text{VNET}}$ model.

  4. We apply Adversarial Discriminate Domain Adaptation (ADDA) techniques to better refine the acoustic power maps.

- **Experiment 3.** $\text{DeepGCC}_{\text{LASSO}}$ vs. $\text{DeepGCC}_{\text{VNET}}$: In this experiment we compare the results of both proposed methods as well as their computational requirements.

6.5.2 DeepGCC$_{\text{VNET}}$ Training Strategy

This section details the process of generating the synthetic data used for training the DeepGCC$_{\text{VNET}}$ model as well as the details of both hyperparameters and loss functions used during the training.

6.5.2.1 Data-Sets Generation

We consider two types of data to train the models with: *i)* One composed by synthetically generated data. We use these data for training and validation data to train the DeepGCC$_{\text{VNET}}$ model, and *ii)* Another one generated according to the labeled data within *CAV3D* database. These data is used as the training and validation subsets for fine-tuning the previously synthetically trained model beside the testing sub-set, which is used either for testing the synthetically trained model or the fine-tuned one.

The generation of the synthetic datasets is the same for every data partition from Table 6.1 . It consists on choosing N random positions $\mathbf{r}_i = (r_{x,i}, r_{y,i}, r_{z,i})^\top$ with $i \in [1, N]$ and generating a volumetric map with a $3D$ Gaussian volume at each of the selected positions with $\sigma = 25cm$ (which has been empirically selected) as shown in Equation (6.11). In our experiments we used $N = 3$, since in used sequences the maximum number of simultaneous speakers is 3, and generated 3000 training maps that were uniformly distributed according to the number of simultaneous active sources (1000 for single speaker maps, 1000 for two speakers maps and 1000 for three speakers maps), and also 600 validation maps also uniformly distributed.

For the fine-tuning procedures in the DeepGCC$_{\text{VNET}}$ model, we also use the training, validation and testing sub-sets from the *CAV3D* data-set shown in Table 6.1. In every case we generate a $3D$ Gaussian volume at each of the ground-truth labeled positions, according to Equation (6.11), also with $\sigma = 25cm$.

6.5.2.2 Training details

In order to train the model, we follow the same procedure for both the synthetic training and the fine-tuning procedure (experiments 2.1 and 2.2). In every case we train the models with a data batch size (N_b) of 20 samples along 100 epochs, as long as the model improves within the last 10 epochs at least, otherwise the training will be ended.

We use two different loss functions to fit our proposed model. The first one is the well known Mean Squared Error (MSE) loss shown in Equation (6.12) where f_{VNET} is the labeled network input data and $\hat{f}_{\text{VNET}}$ is the network output prediction.

$$\text{MSE}\left(f_{\text{VNET}}, \hat{f}_{\text{VNET}}\right) = \sum_{\forall b \in N_b} \|f_{\text{VNET}_b} - \hat{f}_{\text{VNET}_b}\| \tag{6.12}$$

Also we train our model by using the Shrinkage focal loss proposed in [136] and shown in Equation (6.13). This loss function allows the network to pay more attention and focus on target values (active map areas with higher acoustic power levels in our specific case) when optimizing. We consider f_{VNET} as the labeled network input data and $\hat{f}_{\text{VNET}}$ as the network output prediction. We set the a and c parameters to 10 and 0.2 as the general case recommended by the authors.

$$\text{SHRINKAGE}\left(f_{\text{VNET}}, \hat{f}_{\text{VNET}}\right) = \sum_{\forall b \in N_b} \frac{\|f_{\text{VNET}_b} - \hat{f}_{\text{VNET}_b}\|^2}{1 + e^{a\left(c - \|f_{\text{VNET}_b} - \hat{f}_{\text{VNET}_b}\|\right)}} \tag{6.13}$$

To minimize Equation (6.12) and Equation (6.13) we use the *ADAM* optimizer [130] with a learning rate of 10^{-4} and a decay of 10^{-8}, setting the default values for the rest of the parameters.

6.5.3 Evaluation metrics

We evaluate the ASL performance adopting the Multiple Object Tracking Precision (MOTP) metric developed under the *CHIL* project [119], for which $N_{\mathcal{K}}$ positions are estimated as the $N_{\mathcal{K}}$ first local maxima of the map generated by the $\texttt{DeepGCC}_{\text{VNET}}$ network ($\hat{\mathbf{r}}_k$). Multiple Object Tracking Precision (MOTP) is defined as:

$$\text{MOTP} = \frac{\sum_{k=1}^{N_{\mathcal{K}}} \|\mathbf{r}_k - \hat{\mathbf{r}}_k\|}{N_{\mathcal{K}}} \quad [mm] \tag{6.14}$$

where $N_{\mathcal{K}}$ denotes the total number of position estimations along time. Note that the lower the MOTP, the better. We also measure the relative improvement of a given *Method* in MOTP with respect to a different *BaseMethod* as:

$$\Delta^{\text{MOTP}}_{r_{BaseMethod}} = 100\ \frac{\text{MOTP}_{BaseMethod} - \text{MOTP}_{Method}}{\text{MOTP}_{BaseMethod}} \quad [\%] \tag{6.15}$$

Positive values for $\Delta^{\text{MOTP}}_{r_{BaseMethod}}$ mean that *Method* performs better than *BaseMethod*.

6.5.4 Experiment 1: $\texttt{DeepGCC}_{\text{LASSO}}$ Map Refinement

In this section we evaluate the performance of the proposed map refinement method based on the LASSO optimization technique described in Section 6.4.1. We assess the influence of the λ parameter in this optimization method and finally the results in terms of MOTP error are shown.

In all experiments carried out with the $\texttt{DeepGCC}_{\text{LASSO}}$ algorithm defined in Equation (6.10), the basis have been generated by synthetically steering a DeepGCC-based

beamformer at every possible position within the environment as described in Section 6.5.2.1. Figure 6.7 represents this basis matrix $\mathbf{M}$ where each column corresponds to the acoustic power map generated by an acoustic source located at a given position and each row is the vectorized representation of the acoustic power map itself.

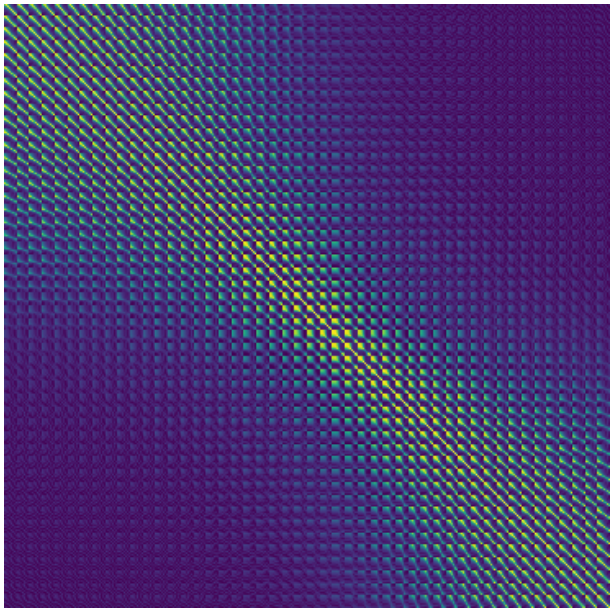


Figure 6.7: *CAV3D* environment basis used for all LASSO-based refinement experiments.

6.5.4.1 Lambda Estimation

According to Equation (6.10), the λ parameter sets how sparse the maps should be. The higher the λ, the more sparse the map will be. This means that larger values of this parameter are translated into a better source localization in terms of MOTP error. However, if λ is set to a too large value, the localization process could have inaccuracies, as the optimization process will only search to minimize the number of active acoustic sources.

In order to avoid this effect, we need to select a proper λ value for which the MOTP error is minimized. Therefore, we run a hyperparameter optimal search procedure, by obtaining the average MOTP error for all the sequences in the corresponding validation sets used in this chapter.

In Figure 6.8 we see a comparison between the average MOTP error obtained with the SRP and DeepGCC models and the average MOTP error obtained by the DeepGCC maps refined with the $\text{DeepGCC}_{\text{LASSO}}$ technique. We can see that for λ values larger than $1e^{-3}$ the LASSO optimization strategy is able to outperform both the SRP and DeepGCC methods. In our experiments we finally selected $\lambda = 1e^{-2}$.

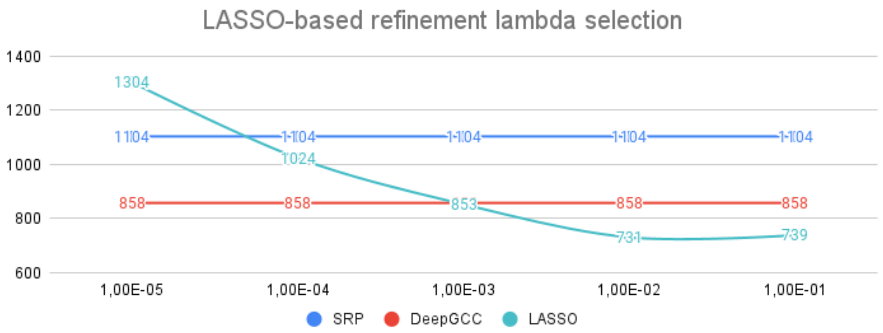


Figure 6.8: MOTP errors variations based on the selected lambda for the LASSO algorithm.

6.5.4.2 Acoustic Source Localization Performance

Once the λ parameter is experimentally set to an specific value ($\lambda = 1e^{-2}$ in our case), we then assess the performance of the proposed $\texttt{DeepGCC}_{\texttt{LASSO}}$ method, by refining the maps generated by the DeepGCC model, and compare it with the SRP algorithm and the DeepGCC method. We evaluated the performance of the $\texttt{DeepGCC}_{\texttt{LASSO}}$ method in scenarios with one, two and three simultaneous speakers inside the environment.

The results for the single speaker scenario are shown in Table 6.2 where we can see that the $\texttt{DeepGCC}_{\texttt{LASSO}}$ refinement outperforms all the other methods in every sequence but one. Regarding the first sequence, where the SRP method obtains the best results, this is a low noise sequence, where the speaker moves slow and always facing sensors. These conditions are suitable for the performance of this classical algorithm. Nevertheless, in other more complex sequences $\texttt{DeepGCC}_{\texttt{LASSO}}$ obtains better results, achieving a very relevant average improvement, with around 30% relative performance improvements when comparing with both methods.

	Speaker 1		
	SRP	DeepGCC	$\texttt{DeepGCC}_{\texttt{LASSO}}$
C_{seq09}	**692**	728	814
C_{seq10}	815	806	**775**
C_{seq11}	1092	1069	**597**
C_{seq12}	1080	1025	**668**
C_{seq13}	867	798	**565**
Avg_{MOTP}	955	921	**653**
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	3.56%	**31.62%**
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	-3.69%	—	**29.10%**

Table 6.2: Experiment 1 results comparing the performance of $\texttt{DeepGCC}_{\texttt{LASSO}}$ model (DeepGCC maps refined with the LASSO algorithm) against the DeepGCC model and SRP algorithm when facing single speaker sequences: MOTP localization error in *mm*.

Table 6.3 shows the results obtained by each method when facing the two speakers sequences. In every scenario the refinement of the first speaker position is better than the

second one. This happens due to the fact that first detected speaker is always the most acoustically energetic one, thus the second speaker will have a worse localization. However, the performance of DeepGCC and $\text{DeepGCC}_{\text{LASSO}}$ methods are very close in terms of MOTP error. The average relative improvement of $\text{DeepGCC}_{\text{LASSO}}$ as compared with the DeepGCC is around 37%, while there is a non-significant average relative degradation of −0.13% as compared with DeepGCC

	Speaker 1			Speaker 2			Average		
	SRP	DeepGCC	$\text{DeepGCC}_{\text{LASSO}}$	SRP	DeepGCC	$\text{DeepGCC}_{\text{LASSO}}$	SRP	DeepGCC	$\text{DeepGCC}_{\text{LASSO}}$
C_{seq22}	939	782	**780**	1695	**803**	852	1317	**793**	816
C_{seq23}	1369	1000	**983**	986	641	**640**	1178	821	**812**
C_{seq24}	1025	690	**684**	1608	**830**	832	1317	760	**758**
Avg_{MOTP}	1136	827	**812**	1393	**724**	764	1265	**790**	791
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	27.16%	**28.52%**	—	**48.03%**	45.15%	—	**37.51%**	37.47%
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	-37.36%	—	**1.81%**	-92.40%	—	**-5.52%**	-60.13%	—	**-0.13%**

Table 6.3: Experiment 1 results comparing the performance of $\text{DeepGCC}_{\text{LASSO}}$ model (DeepGCC maps refined with the LASSO algorithm) against DeepGCC model and SRP algorithm when facing two speakers sequences: MOTP localization error in *mm*.

Finally, Table 6.4 shows the results for scenarios with three simultaneous active speakers. We see very similar behavior for the DeepGCC and $\text{DeepGCC}_{\text{LASSO}}$ models. This is caused by the fact that the acoustic maps estimated by the DeepGCC method are accurate and clean enough to perform a proper localization. Applying the $\text{DeepGCC}_{\text{LASSO}}$ method only leads to an average improvement of 3 mm for the case of sequence 25, while for sequence 26 it even deteriorates by 4 mm. Regarding speaker 3 of both sequences, the LASSO variant of the DeepGCC approach fails to improve on DeepGCC itself. This is because this speaker has so little acoustic power that it is difficult to enhance its localization by this technique. Nevertheless,we can state that DeepGCC and $\text{DeepGCC}_{\text{LASSO}}$ have a similar performance, being $\text{DeepGCC}_{\text{LASSO}}$ slightly better, and outperforming SRP algorithm.

	Speaker 1			Speaker 2			Speaker 3			Average		
	SRP	DeepGCC	$\text{DeepGCC}_{\text{LASSO}}$	SRP	DeepGCC	$\text{DeepGCC}_{\text{LASSO}}$	SRP	DeepGCC	$\text{DeepGCC}_{\text{LASSO}}$	SRP	DeepGCC	$\text{DeepGCC}_{\text{LASSO}}$
C_{seq25}	916	633	**630**	1023	557	**549**	2004	**1312**	1313	1314	834	**831**
C_{seq26}	1357	**860**	864	1173	**692**	**692**	2692	**1395**	1401	1741	**982**	986
Avg_{MOTP}	1039	696	**695**	1065	595	**589**	2196	**1335**	1338	1433	875	**874**
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	32.98%	**33.11%**	—	44.15%	**44.69%**	—	**39.20%**	39.07%	—	38.92%	**39.01%**
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	-49.28%	—	**0.14%**	-78.99%	—	**1.01%**	-64.49%	—	**-0.22%**	-63.77%	—	**0.11%**

Table 6.4: Experiment 1 results comparing the performance of $\text{DeepGCC}_{\text{LASSO}}$ model (DeepGCC maps refined with the LASSO algorithm) against DeepGCC model and SRP algorithm when facing three speakers sequences: MOTP localization error in *mm*.

The LASSO-based map refinement strategy outperforms the localization accuracy in terms of MOTP errors in most cases, especially in single speakers scenarios. However, it has been proven that the lower the acoustic power of the speakers, the worse the localization of them. Moreover, if the maps provided by the DeepGCC method are sufficiently

accurate, this LASSO based approach will improve the localization of the acoustic sources in a minor percentage, and may even worsen it.

Finally, Figure 6.9 shows a graphical comparison of the average MOTP error for sequences with one, two and three simultaneous active speakers.

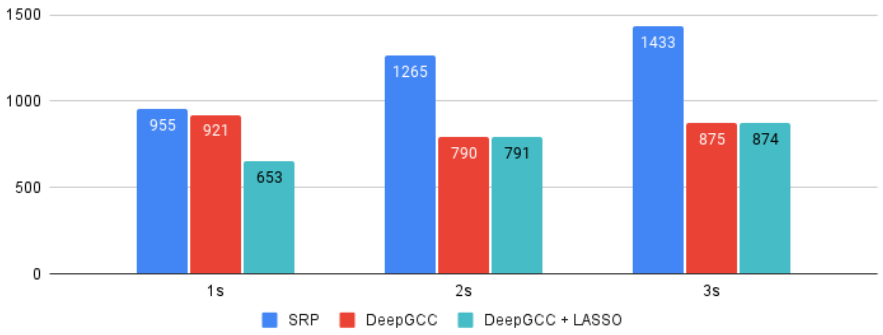


Figure 6.9: MOTP errors (mm) for the LASSO map refinement algorithm according to the maximum number of simultaneous active speakers.

6.5.5 Experiment 2: $\texttt{DeepGCC}_{\texttt{VNET}}$ Map Refinement

This section shows the results achieved with the map refinement method based on the VNET model. This method takes the maps estimated with the DeepGCC network outputs, and it refines them aiming to achieve a better localization accuracy in terms of MOTP errors.

6.5.5.1 Experiment 2.1: Synthetic Training

In this experiment we train and validate the $\texttt{DeepGCC}_{\texttt{VNET}}$ model with the synthetic datasets that were generated as explained in Section 6.5.2.1 and using the standard MSE loss for this purpose. After that, we evaluate its ASL accuracy in terms of the MOTP performance metric with the testing sequences included in Table 6.1.

The localization performance of the synthetically trained model (referred to as $\texttt{DeepGCC}_{\texttt{VNET}_S}$) is compared against the SRP algorithm and the DeepGCC network under the same conditions as the previous experiments with the $\texttt{DeepGCC}_{\texttt{LASSO}}$ approach (described in Section 6.5.4), showing results for one, two and three simultaneous active speakers scenarios.

In the case of single-speaker environments, Table 6.5 shows the obtained results. Again, the map refinement strategy shows a very important relative performance improvement of around 42% when compared with the SRP and DeepGCC alternatives. Also, in this case, the first sequence analyzed has a better result for the SRP algorithm. This is because this particular sequence has more favorable properties for this method, as described above.

	Speaker 1		
	SRP	DeepGCC	$\text{DeepGCC}_{\text{VNet}_S}$
C_{seq09}	**692**	728	708
C_{seq10}	815	806	**669**
C_{seq11}	1092	1069	**486**
C_{seq12}	1080	1025	**551**
C_{seq13}	867	798	**450**
Avg_{MOTP}	955	921	**540**
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	3.56%	**43.46%**
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	-3.69%	—	**41.37%**

Table 6.5: Experiment 2.1 results comparing the performance of the $\text{DeepGCC}_{\text{VNET}_S}$ model (DeepGCC maps refined with the synthetically trained VNET model) against the DeepGCC model and the SRP algorithm when facing single speaker sequences: MOTP localization error in *mm*.

Table 6.6 shows the results when facing two speakers scenarios. It is observed how speaker 1 (the most energetic one) is properly located, while for speaker 2 the localization error severely increases up to 40*cm* as compared with that obtained with the DeepGCC model. This is due to the synthetic data generation process, where no realistic acoustic power level difference between speakers has been simulated when building the training maps. Hence, this inaccuracy leads to a worse localization performance for the second speaker.

	Speaker 1			Speaker 2			Average		
	SRP	DeepGCC	$\text{DeepGCC}_{\text{VNet}_S}$	SRP	DeepGCC	$\text{DeepGCC}_{\text{VNet}_S}$	SRP	DeepGCC	$\text{DeepGCC}_{\text{VNet}_S}$
C_{seq22}	939	782	**761**	1695	**803**	956	1317	**793**	859
C_{seq23}	1369	**1000**	1083	986	**641**	1172	1178	**821**	1128
C_{seq24}	1025	690	**614**	1608	**830**	1231	1317	**760**	923
Avg_{MOTP}	1136	827	**823**	1393	**724**	1148	1265	**790**	986
$\Delta^{\text{MOTP}}_{r_{SRP}}$	—	27.16%	**27.55%**	—	**48.03%**	17.59%	—	**37.51%**	22.05%
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	-37.36%	—	**0.48%**	-92.40%	—	**-58.56%**	-60.13%	—	**-24.81%**

Table 6.6: Experiment 2.1 results comparing the performance of the $\text{DeepGCC}_{\text{VNET}_S}$ model (DeepGCC maps refined with the synthetically trained VNET model) against DeepGCC model and SRP algorithm when facing two speakers sequences: MOTP localization error in *mm*.

The results for the experiment facing a three speakers environment are presented in Table 6.7. Here we observe how the performance of speakers 1 and 2 is very similar (approximately 600 cm of localization error) achieving a relative improvement over DeepGCC of 5% on average. However, it is also seen that the results for speaker 3 are far worse, being consistently better the DeepGCC method. The reason for this is the same as previously discussed, where due to the inaccuracies concerning the acoustic powers of each speaker introduced by the training map synthetic generator, those acoustic sources with less acoustic intensity are poorly localized.

	Speaker 1			Speaker 2			Speaker 3			Average		
	SRP	DeepGCC	DeepGCC$_{VNet_S}$	SRP	DeepGCC	DeepGCC$_{VNet_S}$	SRP	DeepGCC	DeepGCC$_{VNet_S}$	SRP	DeepGCC	DeepGCC$_{VNet_S}$
C_{seq25}	916	633	**692**	1023	557	**623**	2004	**1312**	1971	1314	**834**	1095
C_{seq26}	1357	860	**600**	1173	692	**394**	2692	**1395**	1454	1741	982	**816**
Avg_{MOTP}	1039	696	**666**	1065	595	**559**	2196	**1335**	1827	1433	**875**	1017
$\Delta^{MOTP}_{r_{SRP}}$	—	32.98%	**35.90%**	—	44.15%	**47.51%**	—	**39.20%**	16.80%	—	**38.92%**	29.03%
$\Delta^{MOTP}_{r_{DeepGCC}}$	-49.28%	—	**4.31%**	-78.99%	—	**6.05%**	-64.49%	—	**-36.85%**	-63.77%	—	**-16.22%**

Table 6.7: Experiment 2.1 results comparing the performance of DeepGCC$_{VNET_S}$ model (DeepGCC maps refined with the VNET model) against the DeepGCC model and the SRP algorithm when facing three speakers sequences: MOTP localization error in *mm*.

As a general conclusion, the DeepGCC$_{VNET_S}$ model clearly outperforms the localization performance of the SRP method in any situation, while with when comparing it with respect to the DeepGCC model, it tends to improve the localization of the more energetic speakers. Bearing in mind that average MOTP errors are increased by the inaccuracies of the least energetic speakers due to the generation of the training maps, Figure 6.10 shows a graphical comparison of these errors.

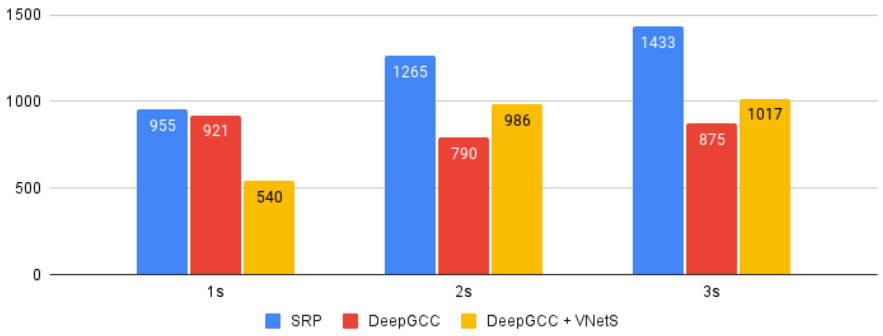


Figure 6.10: MOTP errors for the synthetically trained VNET map refinement method according to the maximum number of simultaneous active speakers.

6.5.5.2 Experiment 2.2: Semi-synthetic training

The semi-synthetic training strategy consists of keeping the previously synthetically trained DeepGCC$_{VNET_S}$ model, and fine-tuning it with real sequences, using again the standard MSE loss function. The sequences used for the fine tuning procedure are the training partitions shown in Table 6.1. The process for generating the DeepGCC signals for building the acoustic power maps in multi-speaker sequences is identical to that explained in Section 5.5.2.1.

This method, referred to as DeepGCC$_{VNET+FT}$, is compared against the DeepGCC and DeepGCC$_{VNET_S}$ models in order to asses the improvement of the fine-tuning process in terms of MOTP errors. We again evaluate the performance of the neural network in scenarios with one, two or three simultaneous active speakers.

When facing single-speaker environments we obtain the results shown in Table 6.8. In all cases the fine-tuning process outperforms the DeepGCC model with a 29% of relative improvement. However, it is not enough to get a better location accuracy than

that obtained with the the synthetically training VNET model ($\texttt{DeepGCC}_{\texttt{VNET}_S}$).This is mainly due to the fact that the synthetic training approach fits best when addressing the single-speaker localization task. Fine-tuning with real data in the proposed form adds a complexity to the later analyzed problem that is useful in more complex multi-speaker scenarios.

	Speaker 1		
	DeepGCC	$\texttt{DeepGCC}_{\texttt{VNet}_S}$	$\texttt{DeepGCC}_{\texttt{VNet}_{FT}}$
C_{seq09}	728	**708**	814
C_{seq10}	806	**669**	774
C_{seq11}	1069	**486**	591
C_{seq12}	1025	**551**	667
C_{seq13}	798	**450**	565
Avg_{MOTP}	921	**540**	651
$\Delta^{MOTP}_{r_{DeepGCC}}$	—	**41.37**%	29.32%
$\Delta^{MOTP}_{r_{\texttt{DeepGCC}_{\texttt{VNet}_S}}}$	-70.55%	—	**-20.55**%

Table 6.8: Experiment 2.2 results comparing the performance of the $\texttt{DeepGCC}_{\texttt{VNET}+FT}$ model (DeepGCC maps refined with the fine-tuned VNET model) against the DeepGCC model and the synthetically trained VNET-based refinement model when facing single speaker sequences: MOTP localization error in mm.

In scenarios where two active acoustic sources are found simultaneously, a better average localization is obtained with $\texttt{DeepGCC}_{\texttt{VNET}+FT}$ than with $\texttt{DeepGCC}_{\texttt{VNET}_S}$. However, this method is not able to improve the DeepGCC proposal when localizing the second speaker. This is again due to the power difference between the two speakers. It should be noted that in sequence 22 this effect is overcome by obtaining a localization improvement for the second speaker of $20cm$ and $16cm$ on average for this sequence.

	Speaker 1			Speaker 2			Average		
	DeepGCC	$\texttt{DeepGCC}_{\texttt{VNet}_S}$	$\texttt{DeepGCC}_{\texttt{VNet}_{FT}}$	DeepGCC	$\texttt{DeepGCC}_{\texttt{VNet}_S}$	$\texttt{DeepGCC}_{\texttt{VNet}_{FT}}$	DeepGCC	$\texttt{DeepGCC}_{\texttt{VNet}_S}$	$\texttt{DeepGCC}_{\texttt{VNet}_{FT}}$
C_{seq22}	782	761	**661**	803	956	**600**	793	859	**631**
C_{seq23}	1000	1083	**911**	**641**	1172	1151	**821**	1128	1031
C_{seq24}	690	614	**601**	**830**	1231	1159	**760**	923	880
Avg_{MOTP}	827	823	**731**	**724**	1148	1032	**790**	986	882
$\Delta^{MOTP}_{r_{DeepGCC}}$	—	0.48%	**11.60**%	—	-58.56%	**-42.54**%	—	-24.81%	**-11.65**%
$\Delta^{MOTP}_{r_{\texttt{DeepGCC}_{\texttt{VNet}_S}}}$	-0.48%	—	**11.18**%	**36.93**%	—	10.10%	**19.88**%	—	10.55%

Table 6.9: Experiment 2.2 results comparing the performance of the $\texttt{DeepGCC}_{\texttt{VNET}+FT}$ model (DeepGCC maps refined with the fine-tuned VNET model) the against DeepGCC model and the synthetically trained VNET-based refinement model when facing two speaker sequences: MOTP localization error in mm.

Table 6.10 shows the results for scenarios where three sources are active at the same time. In this case it is noted that the fine tuned $\texttt{DeepGCC}_{\texttt{VNET}+FT}$ method significantly outperforms the synthetically trained method and get better results than the model without a refinement process in the two more energetic speakers. In addition, it can also be appreciated how this proposal improves on average both the $\texttt{DeepGCC}_{\texttt{VNET}_S}$ method by 14.63% and the $\texttt{DeepGCC}_{\texttt{VNET}_S}$ method by 26.55% in terms of relative improvement.

	Speaker 1			Speaker 2			Speaker 3			Average		
	DeepGCC	$\text{DeepGCC}_{\text{VNet}S}$	$\text{DeepGCC}_{\text{VNet}FT}$	DeepGCC	$\text{DeepGCC}_{\text{VNet}S}$	$\text{DeepGCC}_{\text{VNet}FT}$	DeepGCC	$\text{DeepGCC}_{\text{VNet}S}$	$\text{DeepGCC}_{\text{VNet}FT}$	DeepGCC	$\text{DeepGCC}_{\text{VNet}S}$	$\text{DeepGCC}_{\text{VNet}FT}$
C_{seq25}	633	692	**494**	557	623	**355**	**1312**	1971	1356	834	1095	**735**
C_{seq26}	860	600	**573**	692	**394**	410	1395	1454	**1346**	982	816	**776**
$\text{Avg}_{\text{EMOTP}}$	696	666	**516**	595	559	**370**	**1335**	1827	1353	875	1017	**747**
$\Delta^{\text{MOTP}}_{T_{\text{DeepGCC}}}$	—	4.31%	**25.86%**	—	6.05%	**37.82%**	—	-36.85%	**-1.35%**	—	-16.22%	**14.63%**
$\Delta^{\text{MOTP}}_{T_{\text{DeepGCC}_{\text{VNet}S}}}$	-4.50%	—	**22.52%**	-6.44%	—	**33.81%**	**26.93%**	—	25.94%	13.96%	—	**26.55%**

Table 6.10: Experiment 2.2 results comparing the performance of the $\text{DeepGCC}_{\text{VNET}+FT}$ model (DeepGCC maps refined with the fine-tuned VNET model) against the DeepGCC model and the synthetically trained VNET-based refinement model when facing three speaker sequences: MOTP localization error in *mm*.

Finally, in Figure 6.11 we show a graphical comparison of the average MOTP errors for different number of simultaneous active speakers. We see how for a single speaker the performance gets worse in terms of MOTP error, but when the number of active speakers increase the localization error decreases. This is because the information introduced by the real data used in the fine tuning process complements the neural network to correctly perform for both the single-speaker and multi-speaker cases.

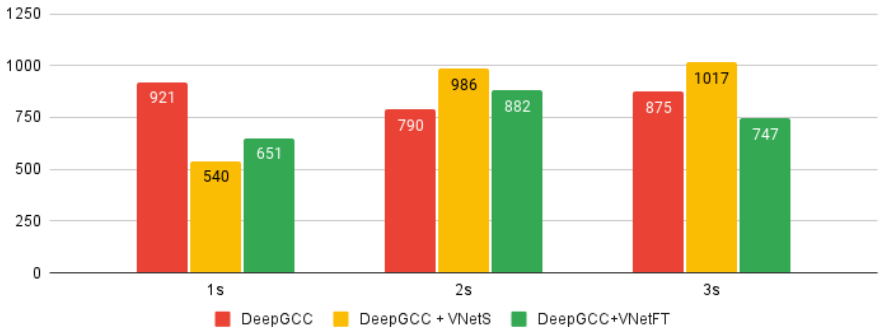


Figure 6.11: MOTP errors for the semi-synthetically trained VNet map refinement method according to the maximum number of simultaneous active speakers.

6.5.5.3 Experiment 2.3: Use of the Shrinkage Focal Loss function

In all the experiments with the VNET-based strategy, all the training and fine-tuning procedures have been carried out by using the standard MSE loss function to minimize the error between the estimated outputs and the ground truth values. Nevertheless, there exists more specific loss functions which, depending on the problem, they are capable of improving the network's gradient descense by paying attention to some aspects of the model estimation according to the desired output.

With this objective, we evaluated the *Shrinkage* focal loss function defined in Equation (6.13) and proposed in [136]. Since acoustic power maps retrieved by DeepGCC method are expected to have a few (or even a single) values where power becomes a maximum, *Shrinkage* focal function loss will make the model give more importance to values near to 1 than those close to 0. It is important to note that, in order to let the network to focus on learning the general aspects of the task, this loss function is only applied when fine-tuning the network.

As in previous experiments, we evaluate the proposed model (that will be referred to as $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$) in scenarios where one, two and three speakers are simultaneously active. We will compare these results against the DeepGCC model without refinement process, and against the fine-tuned $\texttt{DeepGCC}_{\texttt{VNET}}$ network with a MSE loss.

From the results of the experiment where there is only one active speaker, shown in Table 6.11, we can say that, in general terms, the fine tuning process with the shrinkage focal loss improves by 33.98% over the DeepGCC model without the map refinement block. Note that in sequence 9 as it is a simpler sequence, DeepGCC already provides an optimal solution and hence $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ is not capable of improving it, as previously mentioned. Regarding the $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ model, it can be seen how $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ obtains better results in all sequences with an average relative improvement of 6.61% over the $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ model.

	Speaker 1		
	DeepGCC	$\texttt{DeepGCC}_{\texttt{VNET}+FT}$	$\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$
C_{seq09}	**728**	814	769
C_{seq10}	806	774	**730**
C_{seq11}	1069	591	**552**
C_{seq12}	1025	667	**623**
C_{seq13}	798	565	**520**
Avg_{MOTP}	921	651	**608**
$\Delta^{\text{MOTP}}_{r_{DeepGCC}}$	—	29.32%	**33.98%**
$\Delta^{\text{MOTP}}_{r_{\texttt{DeepGCC}_{\texttt{VNET}+FT}}}$	-41.47%	—	**6.61%**

Table 6.11: Experiment 2.3 results comparing the performance of $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ model (DeepGCC maps refined with fine-tuned VNET model using a shrinkage focal loss) against the DeepGCC model and the fine-tuned VNET-based refinement model having used a MSE loss when facing single speaker sequences: MOTP localization error in *mm*.

The results for the two-speaker scenario are shown in Table 6.12. In this case, we can see that both the $\texttt{DeepGCC}_{\texttt{VNET}+FT}$ and $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ approaches have similar performance in terms of average MOTP relative improvement over the DeepGCC method in locating the most energetic speaker (Speaker 1). However, when it comes to finding the second most energetic region within the acoustic map, the shrinkage function loss based method $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ degrades due to the fact that these regions usually have values much lower than 1, which causes the focal loss to not focus on these regions when optimizing the network, thus resulting in a worse location result for the less energetic speaker in the environment (Speaker 2). This can be seen in the average results where $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ has a degradation of 8.73% over $\texttt{DeepGCC}_{\texttt{VNET}+FT}$ and 21.39% over the DeepGCC method.

	Speaker 1			Speaker 2			Average		
	DeepGCC	$\text{DeepGCC}_{\text{VNET}+FT}$	$\text{DeepGCC}_{\text{VNET}_{FT+SH}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}+FT}$	$\text{DeepGCC}_{\text{VNET}_{FT+SH}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}+FT}$	$\text{DeepGCC}_{\text{VNET}_{FT+SH}}$
C_{seq22}	782	**661**	683	803	**600**	631	793	**631**	657
C_{seq23}	1000	911	**802**	**641**	1151	1534	**821**	1031	1168
C_{seq24}	690	**601**	691	**830**	1159	1167	**760**	880	929
Avg_{MOTP}	827	**731**	**731**	**724**	1032	1187	**790**	882	959
$\Delta^{\text{MOTP}}_{\Gamma_{DeepGCC}}$	—	**11.6%**	**11.6%**	—	**-42.54%**	-63.95%	—	**-11.65%**	-21.39 %
$\Delta^{\text{MOTP}}_{\Gamma_{\text{DeepGCC}_{\text{VNET}+FT}}}$	-13.13%	—	**0%**	**29.84%**	—	-15.02%	**10.43%**	—	-8.73%

Table 6.12: Experiment 2.3 results comparing the performance of $\text{DeepGCC}_{\text{VNET}_{FT+SH}}$ model (DeepGCC maps refined with fine-tuned VNET model using a shrinkage focal loss) against the DeepGCC model and the fine-tuned VNET-based refinement model having used a MSE loss when facing two speaker sequences: MOTP localization error in *mm*.

As we can see in Table 6.13, the shrinkage-focal-loss based model did not perform as well as the $\text{DeepGCC}_{\text{VNET}+FT}$ proposal. On average, $\text{DeepGCC}_{\text{VNET}_{FT+SH}}$ has a relative improvement over DeepGCC of 4.09%, while $\text{DeepGCC}_{\text{VNET}+FT}$ has an improvement of 14.63% over the same method. This has the same explanation as the two-speaker experiment. The less energetic speakers (Speakers 2 and 3) contribute less to the acoustic power map, so their refinement is worse when using this method, and their localization results are not as good as those of $\text{DeepGCC}_{\text{VNET}+FT}$. Also, in these experiments, the localization of speaker 1 (the most energetic) does not get as good results as those obtained by $\text{DeepGCC}_{\text{VNET}+FT}$, especially on sequence 25. This could be due to the fact that in this particular sequence all the speakers are active in a reduced area, making it difficult to localize them correctly.

	Speaker 1			Speaker 2			Speaker 3			Average		
	DeepGCC	$\text{DeepGCC}_{\text{VNET}+FT}$	$\text{DeepGCC}_{\text{VNet}_{shrinkage}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}+FT}$	$\text{DeepGCC}_{\text{VNet}_{shrinkage}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}+FT}$	$\text{DeepGCC}_{\text{VNet}_{shrinkage}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}+FT}$	$\text{DeepGCC}_{\text{VNet}_{shrinkage}}$
C_{seq25}	633	**494**	568	557	**355**	506	**1312**	1356	1491	834	**735**	855
C_{seq26}	860	573	**515**	692	410	**403**	1395	**1346**	1421	982	**776**	780
Avg_{MOTP}	696	**516**	553	595	**370**	477	**1335**	1353	1471	875	**747**	834
$\Delta^{\text{MOTP}}_{\Gamma_{DeepGCC}}$	—	**25.86%**	20.55%	—	**37.82%**	19.83%	—	**-1.35%**	-10.19%	—	**14.63%**	4.69%
$\Delta^{\text{MOTP}}_{\Gamma_{\text{DeepGCC}_{\text{VNET}+FT}}}$	-34.88%	—	**-7.17%**	-60.81%	—	**-28.92%**	**1.33%**	—	-8.72%	-17.14%	—	**-11.65%**

Table 6.13: Experiment 2.3 results comparing the performance of the $\text{DeepGCC}_{\text{VNET}_{FT+SH}}$ model (DeepGCC maps refined with fine-tuned VNET model using a shrinkage focal loss) against the DeepGCC model and the fine-tuned VNET-based refinement model having used a MSE loss when facing three speaker sequences: MOTP localization error in *mm*.

Applying the *shrinkage* loss to the fine-tuning process does not improve the results in terms of source location accuracy for scenarios with multiple active speakers. However, the experimental work in this section has been limited and requires a deeper experimental work, at least to evaluate the influence of the a and c parameters of the shrinkage loss function defined in Equation (6.13). Nevertheless, Figure 6.12 summarizes the average MOTP errors for the different maximum number of simultaneous active sources.

Note that both proposed approaches outperform the DeepGCC method when locating only one active acoustic source, with $\text{DeepGCC}_{\text{VNET}_{FT+SH}}$ achieving an average improvement of 43cm over $\text{DeepGCC}_{\text{VNET}+FT}$. However, when faced with scenarios where there are multiple active speakers, using a focal loss does not produce better results than the previously proposed $\text{DeepGCC}_{\text{VNET}+FT}$ method. As explained before, this is due to the fact that less energetic loudspeakers are represented within the acoustic map with rays

whose maximum value is much lower than 1, and therefore using a focal loss does not seem to be appropriate in these cases.

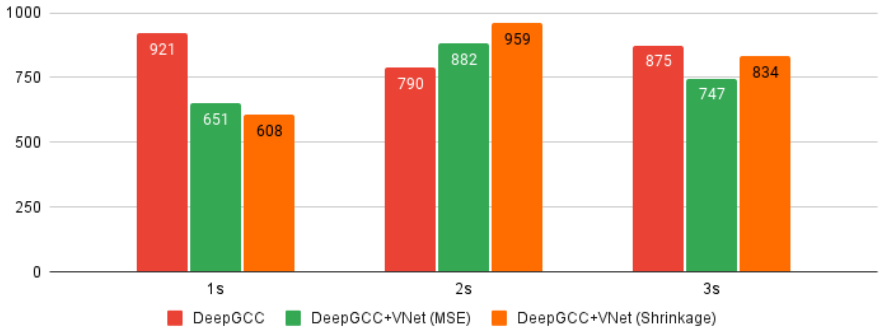


Figure 6.12: MOTP errors for the different used losses when fine-tuning the VNet map refinement method according to the maximum number of simultaneous active speakers.

6.5.5.4 Experiment 2.4: Discriminative Domain Adaptation

The last experiment carried out for the VNET-based map refinement strategies considers the application of domain adaptation state-of-the-art techniques. They consists of a more sophisticated procedure than that carried out in a standard fine-tuning process. Instead of fitting the network weights to the new (fine tuning) input data, these techniques try to adapt the new data to the data previously used for training.

In our case, we chose to use the ADDA technique proposed in [137]. The synthetically trained VNET model is split into a encoder side (referred to as $\texttt{VNET}_S^{enc}$) and a decoder side (referred to as $\texttt{VNET}_S^{dec}$), whose weights are always frozen. We also define a randomly initialized VNET encoder that will be used for the domain adaptation task (referred to as $\texttt{VNET}_{ADDA}^{enc}$).

The domain adaptation process is fitted by an adversarial training. This means that a discriminator network (D) tries to distinguish between the synthetic data encoded by $\texttt{VNET}_S^{enc}$ and the real data encoded by $\texttt{VNET}_{ADDA}^{enc}$ while $\texttt{VNET}_{ADDA}^{enc}$ along the $\texttt{VNET}_S^{dec}$ is fitted to get a proper map refinement result beside to give an encoded real data as close as possible to the encoded synthetic data, since the decoder $\texttt{VNET}_S^{dec}$ is always frozen, the only way to get an optimal result is to force the $\texttt{VNET}_{ADDA}^{enc}$ encoder to give an output similar to the output of the $\texttt{VNET}_S^{enc}$ encoder. Figure 6.13 summarizes this iterative adversarial training.

In our experiments, we train the DeepGCC model using the explained ADDA technique, proposal that will be referred to as $\texttt{DeepGCC}_{\texttt{VNET}_{ADDA}}$. For this training process we have iterated for 1000 epochs with a batch size of 10 samples. This iterative process is terminated when no better model is found in 20 consecutive epochs. We use the *ADAM* optimizer with a learning rate of 10^{-3}.

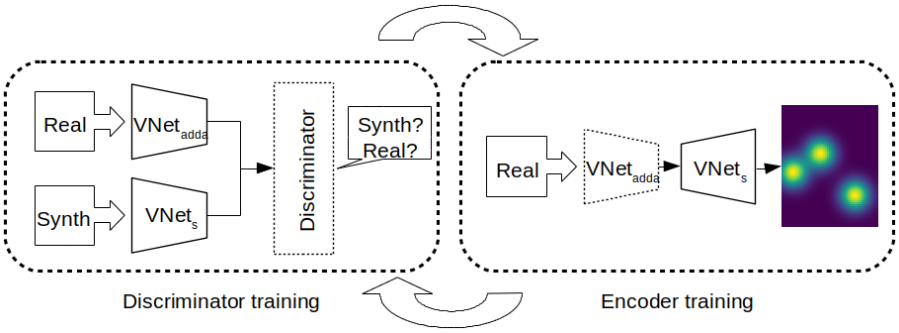


Figure 6.13: Graphical explanation of ADDA training.

The results obtained after training with the ADDA strategy are compared with those achieved by the DeepGCC model with the VNET-based refinement block trained on synthetically generated data ($\texttt{DeepGCC}_{\texttt{VNET}_S}$) and with the DeepGCC model without the refinement process.

Table 6.14 shows the results of the ADDA-based proposal when faced with single-speaker scenarios. We can clearly see that there is no consistent behavior when compared to the $\texttt{DeepGCC}_{\texttt{VNET}_S}$ and DeepGCC models, with sequences such as sequence 10, where 20cm from $\texttt{DeepGCC}_{\texttt{VNET}_S}$ and 40cm from DeepGCC of improvement are achieved, while in sequences like sequence 9, performance degrades with an average of 40 cm over the other two methods.

	Speaker 1		
	DeepGCC	$\texttt{DeepGCC}_{\texttt{VNET}_S}$	$\texttt{DeepGCC}_{\texttt{VNET}_{ADDA}}$
C_{seq09}	728	**708**	1132
C_{seq10}	806	669	**465**
C_{seq11}	1069	**486**	911
C_{seq12}	1025	551	**358**
C_{seq13}	798	**450**	880
Avg_{MOTP}	921	**540**	746
$\Delta^{MOTP}_{r_{DeepGCC}}$	—	**41.37**%	19.00%
$\Delta^{MOTP}_{r_{\texttt{DeepGCC}_{\texttt{VNET}_S}}}$	-70.55%	—	**-38.15**%

Table 6.14: Experiment 2.4 results comparing the performance of $\texttt{DeepGCC}_{\texttt{VNET}_{ADDA}}$ model (DeepGCC maps refined using an ADDA training methodology for the VNET model) against the DeepGCC model and the synthetically trained VNET-based refinement model when facing single speaker sequences: MOTP localization error in *mm*.

Table 6.15 summarizes the results when addressing the task of locating two simultaneous active speakers. In this case, the ADDA-based trained version of DeepGCC is, in average terms, better when locating the most energetic speaker. However, the second most energetic speaker location accuracy degrades significantly compared to the results of both DeepGCC and $\texttt{DeepGCC}_{\texttt{VNET}_S}$. This could be to the fact that the simplicity of the ADDA approach, since only a naive hyperparameter tuning have been performed.

	Speaker 1			Speaker 2			Average		
	DeepGCC	$\text{DeepGCC}_{\text{VNET}_S}$	$\text{DeepGCC}_{\text{VNET}_{ADDA}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}_S}$	$\text{DeepGCC}_{\text{VNET}_{ADDA}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}_S}$	$\text{DeepGCC}_{\text{VNET}_{ADDA}}$
C_{seq22}	782	761	**701**	**803**	956	859	793	859	**780**
C_{seq23}	1000	1083	**871**	**641**	1172	1859	**821**	1128	1365
C_{seq24}	690	**614**	712	**830**	1231	1396	**760**	923	1054
Avg_{MOTP}	827	823	**769**	**724**	1148	1452	**790**	986	1111
$\Delta^{\text{MOTP}}_{\Gamma_{DeepGCC}}$	—	0.48%	**7.01%**	—	**-58.56%**	-100.55%	—	**-24.81%**	-40.63%
$\Delta^{\text{MOTP}}_{\Gamma_{\text{DeepGCC}_{\text{VNET}_S}}}$	-0.49%	—	**6.56%**	**36.93%**	—	-26.84%	**19.88%**	—	-12.68%

Table 6.15: Experiment 2.4 results comparing the performance of $\text{DeepGCC}_{\text{VNET}_{ADDA}}$ model (DeepGCC maps refined using an ADDA training methodology for the VNET model) against the DeepGCC model and the synthetically trained VNET-based refinement model when facing two speaker sequences: MOTP localization error in *mm*.

The results when there are three simultaneous active sources inside the environment are shown in Table 6.16, where we can see that the ADDA-based proposal performs poorly, not being able to outperform neither the DeepGCC method or other $\text{DeepGCC}_{\text{VNET}_S}$ in any of the sequences. In average relative improvement terms, $\text{DeepGCC}_{\text{VNET}_{ADDA}}$ degrades in 4.92% over $\text{DeepGCC}_{\text{VNET}_S}$ and 21.94% over $\text{DeepGCC}_{\text{VNET}_{ADDA}}$.

	Speaker 1			Speaker 2			Speaker 3			Average		
	DeepGCC	$\text{DeepGCC}_{\text{VNET}_S}$	$\text{DeepGCC}_{\text{VNET}_{ADDA}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}_S}$	$\text{DeepGCC}_{\text{VNET}_{ADDA}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}_S}$	$\text{DeepGCC}_{\text{VNET}_{ADDA}}$	DeepGCC	$\text{DeepGCC}_{\text{VNET}_S}$	$\text{DeepGCC}_{\text{VNET}_{ADDA}}$
C_{seq25}	**633**	692	809	**557**	623	679	**1312**	1971	1762	**834**	1095	1083
C_{seq26}	860	600	**436**	692	394	**378**	**1395**	1454	2263	982	**816**	1026
Avg_{MOTP}	696	**666**	705	595	**559**	595	**1335**	1827	1902	**875**	1017	1067
$\Delta^{\text{MOTP}}_{\Gamma_{DeepGCC}}$	—	**4.31%**	-1.30%	—	**6.05%**	0.00%	—	**-36.85%**	-42.47%	—	**-16.22%**	-21.94%
$\Delta^{\text{MOTP}}_{\Gamma_{\text{DeepGCC}_{\text{VNET}_S}}}$	**-4.50%**	—	-5.86%	**-6.44%**	—	**-6.44%**	**26.93%**	—	-4.11%	**13.97%**	—	-4.92%

Table 6.16: Experiment 2.4 results comparing the performance of $\text{DeepGCC}_{\text{VNET}_{ADDA}}$ model (DeepGCC maps refined using an ADDA training methodology for the VNET model) against the DeepGCC model and the synthetically trained VNET-based refinement model when facing three speaker sequences: MOTP localization error in *mm*.

Finally, in Figure 6.14 we show the overall average results, from which we can conclude that in average terms the ADDA technique does not achieve any accurate enough results to outperform the DeepGCC or $\text{DeepGCC}_{\text{VNET}_S}$ methods in any of the proposed scenarios. This not promising results could be due to the fact that the hyperparameter tuning procedure when configuring the ADDA scheme has been naive. Further research and experimental procedures will be needed by using this approach of adapting the refinement model from synthetically generated data to a real environment with few labeled data.

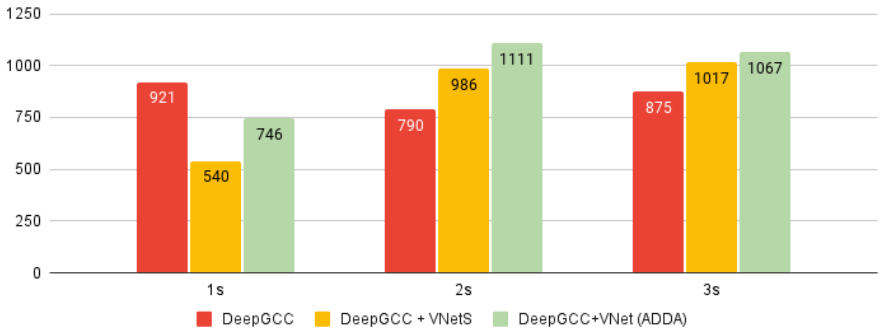


Figure 6.14: MOTP errors for the VNet map refinement method when training with ADDA methodology according to the maximum number of simultaneous active speakers.

6.5.6 Experiment 3: DeepGCC$_{\text{LASSO}}$ vs. DeepGCC$_{\text{VNET}}$

In this section we compare the performance of the two proposals described in this chapter to carry out an acoustic map refinement process, those based on the LASSO algorithm (DeepGCC$_{\text{LASSO}}$), and those based on the use of a VNET model (DeepGCC$_{\text{VNET}}$ variants).

6.5.6.1 Localization Performance Comparison

Figure 6.15 shows the overall results in terms of average MOTP for every experiment performed. We have proposed different acoustic power map refinement blocks and all have been evaluated against the SRP algorithm as well as the DeepGCC method.

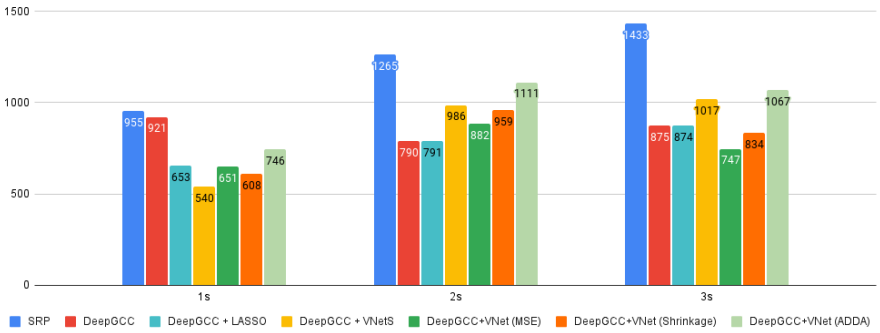


Figure 6.15: Summarized MOTP errors for all map refinement methods according to the maximum number of simultaneous active speakers.

The only non deep learning based method proposed (DeepGCC$_{\text{LASSO}}$) has achieved a good localization accuracy, always outperforming SRP for any proposed experiment (single-speaker, two-speaker and three-speaker), obtaining an improvement of 30cm, 50cm and 60cm approximately for each of the scenarios. Compared to DeepGCC performance, DeepGCC$_{\text{LASSO}}$ works better when facing single-speaker environments achieving 25cm of MOTP improvement. However, it cannot get a better localization estimation on multi-speaker sequences. This is due to the fact that maps retrieved by DeepGCC methods are clear enough and thus, DeepGCC$_{\text{LASSO}}$ model can only equals DeepGCC method performance.

As for the results obtained by the deep learning-based refinement block, synthetically trained DeepGCC$_{\text{VNET}_S}$, it can be observed how it obtains the best result in the single-speaker scenario, with an improvement of 42cm with respect to SRP and 38cm with respect to DeepGCC. However, when moving to environments where two or three speakers are active simultaneously, its performance degrades, failing to improve to the DeepGCC method without a refinement block. This is because the synthetic data with which DeepGCC$_{\text{VNET}_S}$ has been trained is not rich enough to describe the complexity of the problem.

To mitigate the above, a fine tuning process of the refinement block has been performed using the MSE and Shrinkage loss functions. In both cases it is observed that for the

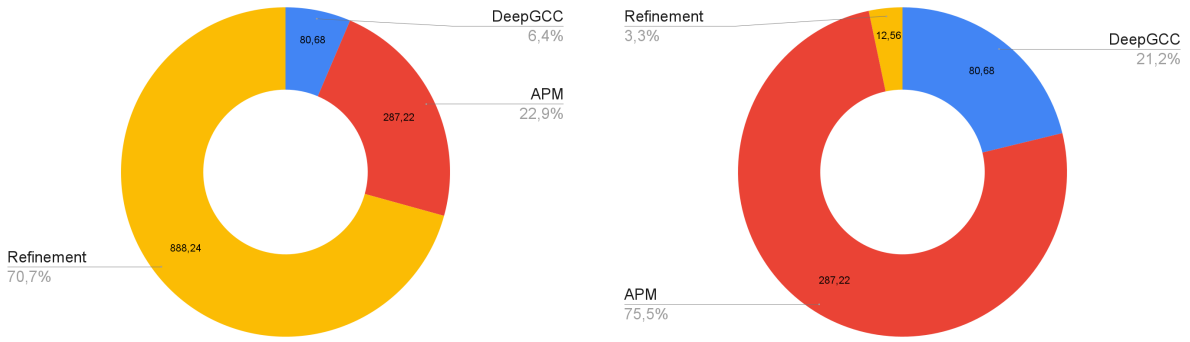
case of a single speaker the performance of both $\texttt{DeepGCC}_{\texttt{VNET}+FT}$ and $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ worsens with respect to the $\texttt{DeepGCC}_{\texttt{VNET}_S}$ method. However, for the two-speaker scenario an improvement is already noticed, but it is in the three-speaker scenario where the best results are obtained. In particular, in the case of $\texttt{DeepGCC}_{\texttt{VNET}+FT}$, an improvement of 67cm with respect to SRP and 13cm with respect to $\texttt{DeepGCC}_{\texttt{VNET}+FT}$ is obtained. As for $\texttt{DeepGCC}_{\texttt{VNET}_{FT+SH}}$ the results obtained are satisfactory, but a better performance is expected in subsequent experiments where the influence of each of the parameters of the loss function will be carefully evaluated.

Finally, the $\texttt{DeepGCC}_{\texttt{VNET}_{ADDA}}$ method always improves the SRP algorithm in each of the proposed experiments. However, this proposal only improves its version without the refinement block (DeepGCC) in the single-speaker scenario, being much worse for the two and three-speaker scenarios. In spite of this, this method shows very promising properties that would allow us to train in a first stage a generalist model, that will then be adapted to the environment where it will be assessed, with a very reduced number of data.

6.5.6.2 Computational Demands Comparison

In this section, we asses the computational requirements of the two map refinement proposals. The computer used for experiments described has an *Ubuntu* 20.04 *LTS* distro as *Operative System* and its hardware is composed of an $Intel^{TM}$ $i7-6700K$ 8 cores CPU, 32GiB of RAM and a $NVIDIA\ GeForce\ GTX$ 1080 GPU.

For this calculation we do not take into account the time needed to locate the different sources, so that we only consider the required time for the map refinement process. We also provide the time spent in the estimation of the DeepGCC signals, and the time devoted to building the acoustic power map to be refined as informative means. Note that these times are fixed regardless of the refining method applied, but they put in context the computational requirements for the map refining procedure.



(a) LASSO (b) *VNet*

Figure 6.16: Computational times (in ms) taken for getting the DeepGCC signals, building the acoustic power map and refining it whether by using *(a)* LASSO algorithm or *(b) VNet* model.

In Figure 6.16 we can see that the LASSO-based refinement process takes 1.25 seconds where 368ms are taken for building the power map and the 70.7% of the total time is dedicated to refine it, whereas in the *VNet*-based refinement process only a 12.56ms are added to the 368ms needed to construct the map, taken the refinement process only a 3.3% of the total time needed.

6.6 Conclusions

In this chapter we have proposed two approaches to refine the acoustic power maps obtained by using previously computed DeepGCC signals aimed at achieving better MOTP accuracy results in the ASL task. From the experimental work, we can draw the following main conclusions:

- The `LASSO`-based refinement strategy have proved to be, at least, as accurate as the DeepGCC model without applying any kind of refinement, outperforming it in several cases, specially when only focusing on those more energetic speakers. However, this method takes almost 1 second to generate a single refined acoustic map.
- the `VNET`-based refinement strategy has proved to be a quick and accurate method for general cases when trained using a semi-synthetic approach and the MSE loss function. Additionally, several variants have been evaluated, all of them performing better than the well-known SRP algorithm, but still with room for improvement.
- The refinement strategy carried out by the `VNET`-based model trained with synthetic samples has been evaluated with positive results in single-speaker environments, but its performance degrades as the number of speakers increases. Also when training the `VNET` model using the Shrinkage loss function, we have obtained good results. However, additional experimental work is required to evaluate further improvements in the MOTP results.
- The ADDA based training strategy achieved the worst overall results. However, this training schema is very sensitive in terms of learning parameters tuning, so that further experiments are needed in order to improve these results.

Finally, we have also compared the computational requirement for the proposed methods. We can conclude that the `VNET`-based models are much faster than those based on the `LASSO` optimization algorithm, allowing for near real time executions.

Chapter 7

Conclusions and Future Work

So comes snow after fire, and even dragons have their ending!

J. R. R. Tolkien

7.1 Conclusions

In this thesis, we have developed systems capable of addressing Acoustic Source Localization (ASL) tasks with state-of-the-art results. Our main goal was to present different approaches which could outperform the widely used Steered Response Power (SRP) algorithm as well as other state-of-the-art DNN-based proposals such as the well-known *SELDNet* model described in [88–91].

In order to do so, our first contribution was proposing `ASLNet`, a model topology designed to end-to-end estimate the (x, y, z) coordinates of the acoustic source from the raw audio captured by the microphones instead of the typical Direction of Arrival (DoA). In a first step, this method was trained with synthetically generated data and then fine tuned with real sequences according to three different signal window lengths.

Our experiments proved that the maximum average relative improvement over the SRP algorithm achieved in terms of MOTP metric was 31.3% and the relative improvement over the *SELDNet* model was 12.65%. However, this method has been also proved to be highly dependent of the data used for training. This contribution has been published in the *MDPI SENSORS Q2* journal with the title "*Towards End-to-End Acoustic Localization using Deep Learning: from Audio Signal to Source Position Coordinates*", and is fully detailed in Chapter 4 of this book.

In our second contribution, we addressed the problem of achieving geometrical domain independence in the context of ASL tasks. We proposed a method capable of performing an accurate localization process with environmental conditions that were different from those used for training the model while outperforming the robust SRP algorithm. We

proposed the definition of Deep Generalized Cross-Correlation (DeepGCC) signals, which are a deep learning based transform of the commonly used Generalized Cross-Correlation (GCC) signals. With these DeepGCC signals we were able to build acoustic power maps which were more robust to geometrical varying conditions to better find the location of the acoustic sources within those maps.

In the conducted experiments for our second contribution, we compared our proposed method with two existing models: `ASLNet` and *SELDNet*. The experiments aimed to assess the level of domain independence achieved by our approach. We began by evaluating positions within areas that matched the training dataset environment (room and microphone array conditions). Subsequently, we introduced variations in the room geometrical environment (room mismatched conditions) and further extended the changes to include both the room environment and the microphone array geometry (room and microphone array mismatched conditions). Moreover, we conducted tests under multi-speaker environments to demonstrate the capability of the `DeepGCC` model to handle this more complex scenario.

It is worth noting that the `DeepGCC` model consistently outperforms the other evaluated methods, both deep learning-based (`ASLNet` and *SELDNet*) and classical methods (SRP), in terms of mean MOTP. The exception occurs in the experiment with room and microphone array mismatched conditions, where the SRP algorithm achieves better performance when training the `DeepGCC` model using data from the UPC room and evaluating it in the CAV3D room. However, we demonstrate that our method is not reliant on the specific environment or sensor geometry. Moreover, it successfully handles the multi-speaker localization task, achieving higher accuracy in acoustic source localization compared to the same model trained for single-speaker localization.

Preliminary experiments of this work have been published in the *IEEE EUSIPCO* conference with the title "*Towards Domain Independence in CNN-based Acoustic Localization using Deep Cross Correlations*" and a much more detailed version was published in the *Elsevier Signal Processing Q1* journal with the title "*Acoustic Source Localization with Deep Generalized Cross Correlations*". A full description of the proposed method, the performed experiments and their results are included in Chapter 5 of this book.

Finally, in the third contribution of this thesis, two acoustic power map refinement methods have been proposed. The first one, based on the Least Absolute Shrinkage Selection Operator (LASSO) optimization technique and in the work presented in [25], in which we aim to estimate the coefficients to use in an environment based basis decomposition, minimized according to the Mean Squared Error (MSE) between it and the current acoustic map. In this optimization process we forced the coefficients to be sparse, which allowed us to denoise the acoustic power maps in a realistic way. With this approach we achieved an average MOTP relative improvement over the SRP algorithm of up to a 36.03%, and an average MOTP relative improvement over the DeepGCC model of 9.69%.

We have also tested the influence of the λ optimization hyperparameter used, concluding that values around $\lambda = 10^{-2}$ obtain the best results in practice.

The second method for refining acoustic power maps proposed in this thesis book utilized the well-known *VNet* model [102]. This learning-based approach was chosen to overcome the limitations of optimization-based techniques. We initially trained the model with synthetic data and then fine-tuned it using real sequences. By conducting the same experiments as with the LASSO refinement methods, we achieved an average MOTP relative improvement of 36.66% over the SRP algorithm and an average MOTP relative improvement of 10.76% over `DeepGCC`. Additionally, we explored alternative function losses and training schemes that showed promising results, surpassing the performance of the SRP algorithm.

We also evaluated the computational demands of both techniques: the LASSO-based refinement method and the *VNet*-based method. The LASSO-based technique took 888.24 ms to refine a map, accounting for 70.7% of the total execution time. In contrast, the VNet-based method only required 12.56 ms, representing just 3.3% of the total execution time. Detailed information on each method can be found in Chapter 6. Furthermore, we have published the LASSO-based proposal in the *Elsevier Signal Processing* journal under the title "*Acoustic Source Localization with Deep Generalized Cross Correlations*", while the contribution based on the *VNet* method will be included in an paper.

7.2 Future Research Lines

The completion of the thesis objectives has set the stage for further research and exploration in the field. In light of this, we propose the following future research directions:

- Development and implementation of novel deep learning layers that enable more generalized filtering of signal correlations. While PHAT filtering is highly effective on a per-signal basis, it poses challenges when integrated into learning models. We suggest designing a new layer that can generate filters heuristically, effectively weighting the given signal correlations and enhancing the localization of acoustic sources.
- Enhancement of multi-speaker localization methods through the integration of tracking algorithms. To address the issue of varying speaker intensities across acoustic frames, incorporating tracking techniques can lead to improved MOTP results.
- Extending the application of ASL techniques to Distributed Acoustic Sensing (DAS) environments. This involves developing novel methods to effectively utilize the abundant data collected by DAS sensors, such as fiber optic strain measurements [138,139]. Building upon the `DeepGCC`-based approaches presented in this thesis, we aim to design beamforming-based localization algorithms capable of handling a large number of sensors and their corresponding signals.

- Exploring model aggregation techniques as a means of domain adaptation. Inspired by recent advancements in federated learning approaches [140, 141], we intend to train a single model on multiple environments and aggregate the results into a global model. This enables the model to perform accurately in any of the trained environments, benefiting from the knowledge acquired across different settings. Importantly, this approach eliminates the need for retraining or fine-tuning.

- Another promising research direction is to explore the potential of utilizing the background information, or noise, captured by the sensors to estimate the geometry of the surrounding space where the sensors are deployed. This concept aligns with the work presented in [142]. The background noise recorded by the sensors contains valuable information about the multipath effects caused by signals reflecting within the rooms, allowing for the extraction of geometrical components of the environment. A possible system design involves an ASL system that first identifies the positions of the acoustic sources. It then subtracts the signals emitted by these sources from the recordings, considering only the direct path between each source and the sensor. The outcome of this process would be a signal containing only the multipath component, which can be utilized to estimate the dimensions and characteristics of the environment. This approach offers a straightforward and cost-effective means of generating an initial map of the environment, particularly beneficial for applications in virtual reality.

Bibliography

[1] S. E. Chazan, H. Hammer, G. Hazan, J. Goldberger, and S. Gannot, "Multi-microphone speaker separation based on deep doa estimation," in *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5.

[2] H.-Y. Lee, J.-W. Cho, M. Kim, and H.-M. Park, "Dnn-based feature enhancement using doa-constrained ica for robust speech recognition," *IEEE Signal Processing Letters*, vol. 23, no. 8, pp. 1091–1095, 2016.

[3] A. Xenaki, J. Bï¿½nsow Boldt, and M. Grï¿½sbï¿½ll Christensen, "Sound source localization and speech enhancement with sparse bayesian learning beamforming," *The Journal of the Acoustical Society of America*, vol. 143, no. 6, pp. 3912–3921, 2018. [Online]. Available: https://doi.org/10.1121/1.5042222

[4] X. Li, L. Girin, F. Badeig, and R. Horaud, "Reverberant sound localization with a robot head based on direct-path relative transfer function," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 2819–2826.

[5] J. H. DiBiase, H. F. Silverman, and M. S. Brandstein, "Robust localization in reverberant rooms," in *Microphone arrays*. Springer, 2001, pp. 157–180.

[6] S. Argentieri, P. Danï¿½s, and P. Souï¿½res, "A survey on sound source localization in robotics: From binaural to array processing methods," *Computer Speech & Language*, vol. 34, no. 1, pp. 87–112, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0885230815000236

[7] M. Cobos, F. Antonacci, A. Alexandridis, A. Mouchtaris, and B. Lee, "A survey of sound source localization methods in wireless acoustic sensor networks," *Wireless Communications and Mobile Computing*, vol. 2017, 2017.

[8] P.-A. Grumiaux, S. Kitić, L. Girin, and A. Guérin, "A survey of sound source localization with deep learning methods," 2021.

[9] J. Benesty, J. Chen, and Y. Huang, *Microphone array signal processing*. Springer Science & Business Media, 2008, vol. 1.

[10] B. Xu, G. Sun, R. Yu, and Z. Yang, "High-accuracy tdoa-based localization without time synchronization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 8, pp. 1567–1576, 2013.

[11] M. S. Brandstein and H. F. Silverman, "A practical methodology for speech source localization with microphone arrays," *Computer Speech & Language*, vol. 11, no. 2, pp. 91–126, 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0885230896900248

[12] J. DiBiase, *A high-accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays*, 2000.

[13] P. Stoica and J. Li, "Lecture notes - source localization from range-difference measurements," *IEEE Signal Processing Magazine*, vol. 23, no. 6, pp. 63–66, 2006.

[14] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 4, pp. 320–327, 1976.

[15] J. Velasco, D. Pizarro, J. Macias-Guarasa, and A. Asaei, "Tdoa matrices: Algebraic properties and their application to robust denoising with missing data," *IEEE Transactions on Signal Processing*, vol. 64, no. 20, pp. 5242–5254, 2016.

[16] M. Compagnoni, A. Pini, A. Canclini, P. Bestagini, F. Antonacci, S. Tubaro, and A. Sarti, "A geometrical-statistical approach to outlier removal for tdoa measurements," *IEEE Transactions on Signal Processing*, vol. 65, no. 15, pp. 3960–3975, 2017.

[17] J. P. Dmochowski and J. Benesty, "Steered beamforming approaches for acoustic source localization," in *Speech processing in modern communication*. Springer, 2010, pp. 307–337.

[18] J. P. Dmochowski, J. Benesty, and S. Affes, "A generalized steered response power method for computationally viable source localization," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2510–2526, 2007.

[19] X. Wan and Z. Wu, "Improved steered response power method for sound source localization based on principal eigenvector," *Applied Acoustics*, vol. 71, no. 12, pp. 1126–1131, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0003682X10001519

[20] H. Do and H. F. Silverman, "Srp-phat methods of locating simultaneous multiple talkers using a frame of microphone array data," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2010, pp. 125–128.

[21] M. Cobos, A. Marti, and J. J. Lopez, "A modified srp-phat functional for robust real-time sound source localization with scalable spatial sampling," *IEEE Signal Processing Letters*, vol. 18, no. 1, pp. 71–74, 2011.

[22] T. Butko, F. G. Pla, C. Segura, C. Nadeu, and J. Hernando, "Two-source acoustic event detection and localization: Online implementation in a smart-room," in *2011 19th European Signal Processing Conference*, 2011, pp. 1317–1321.

[23] A. Marti, M. Cobos, J. J. Lopez, and J. Escolano, "A steered response power iterative method for high-accuracy acoustic source localization," *The Journal of the Acoustical society of America*, vol. 134, no. 4, pp. 2627–2630, 2013.

[24] E. A. Habets, J. Benesty, S. Gannot, and I. Cohen, "The mvdr beamformer for speech enhancement," in *Speech Processing in Modern Communication*. Springer, 2010, pp. 225–254.

[25] J. Velasco, D. Pizarro, and J. Macias-Guarasa, "Source localization with acoustic sensor arrays using generative model based fitting with sparse constraints," *Sensors*, vol. 12, no. 10, pp. 13 781–13 812, 2012. [Online]. Available: https://www.mdpi.com/1424-8220/12/10/13781

[26] T. Padois, O. Doutres, A. Berry, and F. Sgard, "Comparison of acoustic source localization methods in time domain using sparsity constraints," 2015.

[27] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, pp. 276–280, 1986.

[28] 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*, vol. 21, no. 10, pp. 2193–2206, 2013.

[29] R. Roy, A. Paulraj, and T. Kailath, "Estimation of signal parameters via rotational invariance techniques - esprit," in *MILCOM 1986 - IEEE Military Communications Conference: Communications-Computers: Teamed for the 90's*, vol. 3, 1986, pp. 41.6.1–41.6.5.

[30] E. Mabande, H. Sun, K. Kowalczyk, and W. Kellermann, "Comparison of subspace-based and steered beamformer-based reflection localization methods," in *2011 19th European Signal Processing Conference*, 2011, pp. 146–150.

[31] 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*, 2007, pp. 18–21.

[32] A. Hogg, V. W. Neo, S. Weiss, C. Evers, and P. Naylor, "A polynomial eigenvalue decomposition music approach for broadband sound source localization," in *IEEE*

Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA). IEEE, July 2021. [Online]. Available: https://eprints.soton.ac.uk/450812/

[33] N. Roman and D. Wang, "Binaural tracking of multiple moving sources," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 4, pp. 728–739, 2008.

[34] M. I. Mandel, R. J. Weiss, and D. P. W. Ellis, "Model-based expectation-maximization source separation and localization," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 2, pp. 382–394, 2010.

[35] T. May, S. van de Par, and A. Kohlrausch, "A probabilistic model for robust localization based on a binaural auditory front-end," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 1, pp. 1–13, 2011.

[36] J. Woodruff and D. Wang, "Binaural localization of multiple sources in reverberant and noisy environments," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 5, pp. 1503–1512, 2012.

[37] Y. Dorfan and S. Gannot, "Tree-based recursive expectation-maximization algorithm for localization of acoustic sources," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 10, pp. 1692–1703, 2015.

[38] X. Li, L. Girin, R. Horaud, and S. Gannot, "Multiple-speaker localization based on direct-path features and likelihood maximization with spatial sparsity regularization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 10, pp. 1997–2012, 2017.

[39] S. Rickard and O. Yilmaz, "On the approximate w-disjoint orthogonality of speech," in *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, 2002, pp. I–529–I–532.

[40] A. Deleforge, F. Forbes, and R. Horaud, "Variational em for binaural sound-source separation and localization," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 76–80.

[41] A. Deleforge, R. Horaud, Y. Y. Schechner, and L. Girin, "Co-localization of audio sources in images using binaural features and locally-linear regression," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 4, pp. 718–731, 2015.

[42] H. Sawada, R. Mukai, and S. Makino, "Direction of arrival estimation for multiple source signals using independent component analysis," in *Seventh International Symposium on Signal Processing and Its Applications, 2003. Proceedings.*, vol. 2, 2003, pp. 411–414 vol.2.

[43] S. Gannot, E. Vincent, S. Markovich-Golan, and A. Ozerov, "A consolidated perspective on multimicrophone speech enhancement and source separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 4, pp. 692–730, 2017.

[44] E. Vincent, T. Virtanen, and S. Gannot, *Audio source separation and speech enhancement*. John Wiley & Sons, 2018.

[45] G. Hinton, Y. LeCun, and Y. Bengio, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[46] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[47] D. Salvati, C. Drioli, and G. L. Foresti, "On the use of machine learning in microphone array beamforming for far-field sound source localization," in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2016, pp. 1–6.

[48] A. Deleforge, "Acoustic space mapping: A machine learning approach to sound source separation and localization," Ph.D. dissertation, Université de Grenoble, 2013.

[49] Y. Kim and H. Ling, "Direction of arrival estimation of humans with a small sensor array using an artificial neural network," *Progress In Electromagnetics Research B*, vol. 27, 01 2011.

[50] H. Tsuzuki, M. Kugler, S. Kuroyanagi, and A. Iwata, "An approach for sound source localization by complex-valued neural network," *IEICE TRANSACTIONS on Information and Systems*, vol. 96, no. 10, pp. 2257–2265, 2013.

[51] N. Ma, G. Brown, and T. May, "Exploiting deep neural networks and head movements for binaural localisation of multiple speakers in reverberant conditions," in *Interspeech*, vol. 2015. International Speech Communication Association, 2015, pp. 160–164.

[52] M. Yiwere and E. Rhee, "Distance estimation and localization of sound sources in reverberant conditions using deep neural networks," *International Journal of Applied Engineering Research*, vol. 12, pp. 12 384–12 389, 01 2017.

[53] R. Takeda and K. Komatani, "Sound source localization based on deep neural networks with directional activate function exploiting phase information," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 405–409.

[54] ——, "Discriminative multiple sound source localization based on deep neural networks using independent location model," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, 2016, pp. 603–609.

[55] ——, "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)*, 2017, pp. 2217–2221.

[56] R. Takeda, Y. Kudo, K. Takashima, Y. Kitamura, and K. 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)*, 2018, pp. 3514–3518.

[57] 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)*, 2015, pp. 2814–2818.

[58] 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)*, 2016, pp. 1–6.

[59] F. B. Gelderblom, Y. Liu, J. Kvam, and T. A. 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)*, 2021, pp. 4390–4394.

[60] T. Hirvonen, "Classification of spatial audio location and content using convolutional neural networks," *Journal of the Audio Engineering Society*, may 2015.

[61] S. Chakrabarty and E. A. P. Habets, "Broadband doa estimation using convolutional neural networks trained with noise signals," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 136–140.

[62] ——, "Multi-speaker localization using convolutional neural network trained with noise," *CoRR*, vol. abs/1712.04276, 2017. [Online]. Available: http://arxiv.org/abs/1712.04276

[63] S. Chakrabarty and E. A. Habets, "Multi-scale aggregation of phase information for complexity reduction of cnn based doa estimation," in *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5.

[64] 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*, vol. 13, no. 1, pp. 8–21, 2019.

[65] W. He, P. Motlicek, and J.-M. Odobez, "Deep neural networks for multiple speaker detection and localization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 74–79.

[66] P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza, "Deep neural networks for joint voice activity detection and speaker localization," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 1567–1571.

[67] P. Vecchiotti, G. Pepe, E. Principi, and S. Squartini, "Detection of activity and position of speakers by using deep neural networks and acoustic data augmentation," *Expert Systems with Applications*, vol. 134, pp. 53–65, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417419303422

[68] J. M. Vera-Diaz, D. Pizarro, and J. Macias-Guarasa, "Towards end-to-end acoustic localization using deep learning: From audio signals to source position coordinates," *Sensors*, vol. 18, no. 10, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/10/3418

[69] S. Chytas and G. Potamianos, "Hierarchical detection of sound events and their localization using convolutional neural networks with adaptive thresholds," 01 2019, pp. 50–54.

[70] P. Vecchiotti, N. Ma, S. Squartini, and G. J. Brown, "End-to-end binaural sound localisation from the raw waveform," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 451–455.

[71] W. Zhang, Y. Zhou, and Y. Qian, "Robust doa estimation based on convolutional neural network and time-frequency masking," in *Proc. Interspeech 2019*, 2019, pp. 2703–2707.

[72] F. Hübner, W. Mack, and E. A. P. Habets, "Efficient training data generation for phase-based doa estimation," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 456–460.

[73] G. Bologni, R. Heusdens, and J. Martinez, "Acoustic reflectors localization from stereo recordings using neural networks," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 1–5.

[74] E. Vargas, J. R. Hopgood, K. Brown, and K. Subr, "On improved training of cnn for acoustic source localisation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 720–732, 2021.

[75] D. Diaz-Guerra, A. Miguel, and J. R. Beltran, "Robust sound source tracking using srp-phat and 3d convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 300–311, 2021.

[76] Y. He, N. Trigoni, and A. Markham, "Sounddet: Polyphonic sound event detection and localization from raw waveform," *CoRR*, vol. abs/2106.06969, 2021. [Online]. Available: https://arxiv.org/abs/2106.06969

[77] K. SongGong, W. Wang, and H. Chen, "Acoustic source localization in the circular harmonic domain using deep learning architecture," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 2475–2491, 2022.

[78] N. Yalta, K. Nakadai, , and T. Ogata, "Sound source localization using deep learning models," *Journal of Robotics and Mechatronics*, vol. 29, no. 1, pp. 37–48, 2017.

[79] D. Suvorov, G. Dong, and R. Zhukov, "Deep residual network for sound source localization in the time domain," *CoRR*, vol. abs/1808.06429, 2018. [Online]. Available: http://arxiv.org/abs/1808.06429

[80] W. He, P. Motlicek, and J.-M. Odobez, "Joint localization and classification of multiple sound sources using a multi-task neural network," in *Proc. Interspeech 2018*, 2018, pp. 312–316.

[81] ——, "Neural network adaptation and data augmentation for multi-speaker direction-of-arrival estimation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1303–1317, 2021.

[82] H. Pujol, E. Bavu, and A. Garcia, "Source localization in reverberant rooms using deep learning and microphone arrays," in *23rd International Congress on Acoustics (ICA 2019 Aachen)*, Aachen, Germany, Sep 2019. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02276941

[83] R. Ranjan, S. Jayabalan, T. N. T. Nguyen, and W. S. Gan, "Sound event detection and direction of arrival estimation using residual net and recurrent neural networks," 2019.

[84] K. Shimada, N. Takahashi, S. Takahashi, and Y. Mitsufuji, "Sound event localization and detection using activity-coupled cartesian doa vector and rd3net," 2020.

[85] H. Sundar, W. Wang, M. Sun, and C. Wang, "Raw waveform based end-to-end deep convolutional network for spatial localization of multiple acoustic sources," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 4642–4646.

[86] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[87] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[88] S. Adavanne, A. Politis, and T. Virtanen, "Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 1462–1466.

[89] 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*, vol. 13, no. 1, pp. 34–48, 2019.

[90] S. Adavanne, A. Politis, and T. Virtanen, "Localization, detection and tracking of multiple moving sound sources with a convolutional recurrent neural network," *CoRR*, vol. abs/1904.12769, 2019. [Online]. Available: http://arxiv.org/abs/1904.12769

[91] ——, "A multi-room reverberant dataset for sound event localization and detection," 2019.

[92] A. Politis, S. Adavanne, and T. Virtanen, "A dataset of reverberant spatial sound scenes with moving sources for sound event localization and detection," 2020.

[93] A. Politis, S. Adavanne, D. Krause, A. Deleforge, P. Srivastava, and T. Virtanen, "A dataset of dynamic reverberant sound scenes with directional interferers for sound event localization and detection," 2021.

[94] Q. Li, X. Zhang, and H. Li, "Online direction of arrival estimation based on deep learning," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 2616–2620.

[95] S. Kapka and M. Lewandowski, "Sound source detection, localization and classification using consecutive ensemble of crnn models," *arXiv preprint arXiv:1908.00766*, 2019.

[96] J. Zhang, W. Ding, and L. He, "Data augmentation and prior knowledge-based regularization for sound event localization and detection," *DCASE 2019 Detection and Classification of Acoustic Scenes and Events 2019 Challenge*, 2019.

[97] Y. Cao, T. Iqbal, Q. Kong, Y. Zhong, W. Wang, and M. D. Plumbley, "Event-independent network for polyphonic sound event localization and detection," *arXiv preprint arXiv:2010.00140*, 2020.

[98] A. Bohlender, A. Spriet, W. Tirry, and N. Madhu, "Exploiting temporal context in cnn based multisource doa estimation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1594–1608, 2021.

[99] A. S. Subramanian, C. Weng, S. Watanabe, M. Yu, and D. Yu, "Deep learning based multi-source localization with source splitting and its effectiveness in multi-talker speech recognition," 2021.

[100] K. Guirguis, C. Schorn, A. Guntoro, S. Abdulatif, and B. Yang, "Seld-tcn: Sound event localization amp; detection via temporal convolutional networks," in *2020 28th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 16–20.

[101] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[102] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 fourth international conference on 3D vision (3DV)*. IEEE, 2016, pp. 565–571.

[103] L. Comanducci, F. Borra, P. Bestagini, F. Antonacci, S. Tubaro, and A. 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*, vol. 28, pp. 2238–2251, 2020.

[104] S. Patel, M. Zawodniok, and J. Benesty, "Dcase 2020 task 3: A single stage fully convolutional neural network for sound source localization and detection," *DCASE2020 Challenge*, 2020.

[105] T. Jenrungrot, V. Jayaram, S. Seitz, and I. Kemelmacher-Shlizerman, "The cone of silence: Speech separation by localization," 2020.

[106] Y. Huang, X. Wu, and T. Qu, "A time-domain unsupervised learning based sound source localization method," in *2020 IEEE 3rd International Conference on Information Communication and Signal Processing (ICICSP)*, 2020, pp. 26–32.

[107] G. L. Moing, P. Vinayavekhin, D. J. Agravante, T. Inoue, J. Vongkulbhisal, A. Munawar, and R. Tachibana, "Data-efficient framework for real-world multiple sound source 2d localization," 2021.

[108] J. M. Vera-Diaz, D. Pizarro, and J. Macias-Guarasa, "Towards domain independence in cnn-based acoustic localization using deep cross correlations," in *2020 28th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 226–230.

[109] ——, "Acoustic source localization with deep generalized cross correlations," *Signal Processing*, vol. 187, p. 108169, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165168421002073

[110] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[111] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[112] Y. Cao, T. Iqbal, Q. Kong, F. An, W. Wang, and M. D. Plumbley, "An improved event-independent network for polyphonic sound event localization and detection," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 885–889.

[113] Q. Wang, J. Du, H.-X. Wu, J. Pan, F. Ma, and C.-H. Lee, "A four-stage data augmentation approach to resnet-conformer based acoustic modeling for sound event localization and detection," 2021.

[114] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," 2020.

[115] A. Moreno, A. Bonafonte, E. Lleida, J. Mariño, J. L. Boix, and D. P. Olive, "Albayzin speech database: design of the phonetic corpus: Eurospeech'93, 3rd european conference on speech communication and technology," 1993.

[116] G. Lathoud, J.-M. Odobez, and D. Gatica-Perez, "Av16. 3: An audio-visual corpus for speaker localization and tracking," in *International Workshop on Machine Learning for Multimodal Interaction*. Springer, 2004, pp. 182–195.

[117] X. Qian, A. Brutti, O. Lanz, M. Omologo, and A. Cavallaro, "Multi-speaker tracking from an audio-visual sensing device," *IEEE Transactions on Multimedia*, vol. 21, no. 10, pp. 2576–2588, 2019.

[118] A. Waibel, R. Stiefelhagen, R. Carlson, J. Casas, J. Kleindienst, L. Lamel, O. Lanz, D. Mostefa, M. Omologo, F. Pianesi *et al.*, "Computers in the human interaction loop," in *Handbook of Ambient Intelligence and Smart Environments*. Springer, 2010, pp. 1071–1116.

[119] R. Stiefelhagen, K. Bernardin, R. Bowers, R. T. Rose, M. Michel, and J. Garofolo, "The clear 2007 evaluation," in *Multimodal Technologies for Perception of Humans*. Springer, 2007, pp. 3–34.

[120] J. Torres-Solis, T. H. Falk, and T. Chau, *A review of indoor localization technologies: towards navigational assistance for topographical disorientation*. INTECH Open Access Publisher, 2010.

[121] T. Ruiz-López, J. L. Garrido, K. Benghazi, and L. Chung, "A survey on indoor positioning systems: foreseeing a quality design," in *Distributed Computing and Artificial Intelligence*. Springer, 2010, pp. 373–380.

[122] L. Mainetti, L. Patrono, and I. Sergi, "A survey on indoor positioning systems," in *2014 22nd international conference on software, telecommunications and computer networks (SoftCOM)*. IEEE, 2014, pp. 111–120.

[123] T. N. Sainath, R. J. Weiss, K. W. Wilson, A. Narayanan, M. Bacchiani, and A. Senior, "Speaker location and microphone spacing invariant acoustic modeling from raw multichannel waveforms," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 30–36.

[124] Y. Sun, J. Chen, C. Yuen, and S. Rahardja, "Indoor sound source localization with probabilistic neural network," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 8, pp. 6403–6413, 2017.

[125] 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)*. IEEE, 2018, pp. 2386–2390.

[126] N. Ma, T. May, and G. J. Brown, "Exploiting deep neural networks and head movements for robust binaural localization of multiple sources in reverberant environments," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 12, pp. 2444–2453, 2017.

[127] D. Salvati, C. Drioli, and G. L. Foresti, "Exploiting cnns for improving acoustic source localization in noisy and reverberant conditions," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 103–116, 2018.

[128] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *The Journal of the Acoustical Society of America*, vol. 65, no. 4, pp. 943–950, 1979.

[129] J. Velasco, C. J. Martín-Arguedas, J. Macias-Guarasa, D. Pizarro, and M. Mazo, "Proposal and validation of an analytical generative model of srp-phat power maps in reverberant scenarios," *Signal Processing*, vol. 119, pp. 209–228, 2016.

[130] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[131] 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*, vol. 18, no. 6, pp. 1429–1439, 2009.

[132] R. Scheibler, E. Bezzam, and I. Dokmanic, "Pyroomacoustics: A python package for audio room simulations and array processing algorithms," *CoRR*, vol. abs/1710.04196, 2017. [Online]. Available: http://arxiv.org/abs/1710.04196

[133] P. Pertil a and E. Cakir, "Robust direction estimation with convolutional neural networks based steered response power," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 6125–6129.

[134] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1996.tb02080.x

[135] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[136] X. Lu, C. Ma, B. Ni, X. Yang, I. Reid, and M.-H. Yang, "Deep regression tracking with shrinkage loss," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 353–369.

[137] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[138] L. Fu, W. Li, and Y. Ma, "Deep learning based das to geophone data transformation," *IEEE Sensors Journal*, pp. 1–1, 2023.

[139] S. Lapins, A. Butcher, J. M. Kendall, T. S. Hudson, A. L. Stork, M. J. Werner, J. Gunning, and A. M. Brisbourne, "Das-n2n: Machine learning distributed acoustic sensing (das) signal denoising without clean data," 2023.

[140] J. Wu, S. Drew, F. Dong, Z. Zhu, and J. Zhou, "Topology-aware federated learning in edge computing: A comprehensive survey," 2023.

[141] B. Li, M. N. Schmidt, T. S. Alstrøm, and S. U. Stich, "On the effectiveness of partial variance reduction in federated learning with heterogeneous data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 3964–3973.

[142] I. Cohen, O. Lindenbaum, and S. Gannot, "Unsupervised acoustic scene mapping based on acoustic features and dimensionality reduction," 2023.