Sequential Estimation Techniques and Application to Multiple Speaker Tracking and Language Modeling

Dissertation
zur Erlangung des Grades
des Doktors der Ingenieurwissenschaften
der Naturwissenschaftlich-Technischen Fakultät
der Universität des Saarlandes

von

Youssef Oualil

Saarbrücken
2017

Tag des Kolloquiums:	18.06.2018
Dekan:	Univ.-Prof. Dr. rer. nat. Guido Kickelbick
Berichterstatter:	Univ.-Prof. Dr. rer. nat. Dietrich Klakow,
	Univ.-Prof. Dr. Mikko Kurimo
Vorsitz:	Univ.-Prof. Dr. techn. Romanus Dyczij-Edlinger
Akad. Mitarbeiter:	Dr. Adam Yuile

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt.

Ort, Datum	(Unterschrift)

*I dedicate this thesis to my my mother Saedia and my father Abdelkader,
who crafted the person I became today, and to my wife Melanie and my children
Yassin and Mariam, who keep the light shining in my life.*

Abstract

For many real-word applications, the considered data is given as a time sequence that becomes available in an orderly fashion, where the order incorporates important information about the entities of interest. The work presented in this thesis deals with two such cases by introducing new sequential estimation solutions. More precisely, we introduce a:

I. Sequential Bayesian estimation framework to solve the multiple speaker localization, detection and tracking problem. This framework is a complete pipeline that includes 1) new observation estimators, which extract a fixed number of potential locations per time frame; 2) new unsupervised Bayesian detectors, which classify these estimates into noise/speaker classes and 3) new Bayesian filters, which use the speaker class estimates to track multiple speakers. This framework was developed to tackle the low overlap detection rate of multiple speakers and to reduce the number of constraints generally imposed in standard solutions.

II. Sequential neural estimation framework for language modeling, which overcomes some of the shortcomings of standard approaches through merging of different models in a hybrid architecture. That is, we introduce two solutions that tightly merge particular models and then show how a generalization can be achieved through a new mixture model. In order to speed-up the training of large vocabulary language models, we introduce a new extension of the noise contrastive estimation approach to batch training.

Deutsche Zusammenfassung

Bei vielen Anwendungen kommen Daten als zeitliche Sequenz vor, deren Reihenfolge wichtige Informationen über die betrachteten Entitäten enthält. In der vorliegenden Arbeit werden zwei derartige Fälle bearbeitet, indem neue sequenzielle Schätzverfahren eingeführt werden:

I. Ein Framework für ein sequenzielles bayessches Schätzverfahren zur Lokalisation, Erkennung und Verfolgung mehrerer Sprecher. Es besteht aus 1) neuen Beobachtungsschätzern, welche pro Zeitfenster eine bestimmte Anzahl möglicher Aufenthaltsorte bestimmen; 2) neuen, unüberwachten bayesschen Erkennern, die diese Abschätzungen nach Sprechern/Rauschen klassifizieren und 3) neuen bayesschen Filtern, die Schätzungen aus der Sprecher-Klasse zur Verfolgung mehrerer Sprecher verwenden. Dieses Framework wurde speziell zur Verbesserung der i.A. niedrigen Erkennungsrate bei gleichzeitig Sprechenden entwickelt und benötigt weniger Randbedingungen als Standardlösungen.

II. Ein sequenzielles neuronales Vorhersageframework für Sprachmodelle, das einige Nachteile von Standardansätzen durch das Zusammenführen verschiedener Modelle in einer Hybridarchitektur beseitigt. Konkret stellen wir zwei Lösungen vor, die bestimmte Modelle integrieren, und leiten dann eine Verallgemeinerung durch die Verwendung eines neuen Mischmodells her. Um das Trainieren von Sprachmodellen mit sehr großem Vokabular zu beschleunigen, wird eine Erweiterung des rauschkontrastiven Schätzverfahrens für Batch-Training vorgestellt.

Acknowledgements

During my long journey to this thesis, I had the pleasure to work, collaborate, discuss and meet many great people, who made this experience the most pleasant. Outside the office, the warm love and immense support of my wife Melanie, my son Yassin and my daughter Mariam turned my evenings and weekends into a source of energy to continue on this long road. After many years at the Spoken Language Systems (LSV) group of Saarland university, it became urgent to express an overdue note of gratitude and recognition to many colleagues who became friends, first and foremost, to my advisor Dietrich Klakow whose immense help and support brought the light needed to turn this experience into a success. Many thanks also go to my office mate, colleague and friend Rahil Mahdian Toroghi for the long fruitful discussions and collaboration and for making the office a pleasant place to work. On the same note, I send my gratitude to Friedrich Faubel for putting me on the right track to accomplish this journey. The office was a pleasant place to be and work due to the amazing colleagues, Andrea Fischer, Andrei Malchanau, Anna Schmidt, Clayton Greenberg, Dave Howcroft, Marc Schulder, Martin Gropp, Michael A. Hedderich, Michael Wiegand, Mittul Singh, Thomas Kleinbauer, Thomas Trost, Volha Petukhova and many others. They made my stay here an enjoyable experience and turned after many long years into a large family with diverse roots and experiences. I would like also to thank the many bachelor and master students that I supervised over the long years, and who helped me look at research and science from a different angle and question what many of us consider as "obvious". Apart from the above people, I am also in debt to Mathew Magimai-Doss, who helped me get the most of my one year research stay at Idiap research institute in Martigny, Switzerland, and whose comments helped me improve the quality of my publications and opened the gate to new research directions.

Contents

1 Introduction

Advanced technologies, such as smart phones and navigation systems, are continuously becoming unavoidable in our daily routines. The fast and increasing demand for such products opened the gate for countless business opportunities, as it drew new research horizons for the scientific community, which led to a large number of research projects and prototype development in academia and industry alike. Many research directions and product requirements focus on the most natural ways to reliably incorporate these technologies with a reduced cost and effort. Following the natural imitation process which constantly inspired human creativity, researchers tried to tackle this problem by building 1) machines that can *see*, leading to a great field of research commonly known as Computer Vision (CV) [1, 2], 2) machines that can *hear*, which gave birth to Automatic Speech Recognition (ASR) [3, 4], machines that are able to do the inverse process, namely, 3) machines that can *talk* [5], the work in this direction is generally referred to as Speech Synthesis (SS) or Text-To-Speech (TTS), and eventually, 4) machines that can *think and automatically learn* using sophisticated Artificial Intelligence (AI) [6] and machine learning approaches [7, 8].

These different research areas witnessed a great progress over the last two decades but at different rates. For many fields, the early introduction of theoretical frameworks that formulate and solve the tackled problems in a probabilistic manner including, but not limited to, learning, inference and estimation was the key to success. To name a few examples of approaches with a significant impact, Hidden Markov Model (HMM) [9], Conditional Random Fields (CRFs) [10] and Deep Learning (DL) [11, 12] are considered to be some of the most successful and largely adopted modeling and learning frameworks. In particular, the recent re-introduction of DL [11] was largely supported by the introduction of more powerful and super-fast machines, which allowed us to train large models leading to many great achievements and a significant jump in the performance of many applications, particularly, in the fields of speech recognition, machine translation and natural language processing. This thesis contributes to the research done in this direction by proposing new DL-based approaches to solve the language modeling task.

The continuous progress witnessed over the last two decades is also due to the increased quality and quantity of the data that is used to train the models. More particularly, large and diverse corpora dealing with different research aspects are being continuously collected and developed. Moreover, the introduction of multiple and diverse sensors allowed the investigation of complex multi-modal systems for a more natural human-machine interaction. Among these sensors, the usage of microphone arrays became essential to solve a large number of traditional speech and signal processing problems, such as speech separation and/or enhancement, acoustic source localization and tracking and also some multi-modal problems, such as camera steering for teleconferencing systems, audio-visual tracking and distant speech recognition. This thesis will introduce new solutions to some of the problems mentioned in this list.

1.1 Sequential Data Estimation

For many probabilistic estimation problems, the considered data is timely ordered as it becomes available in a time sequence. Thus, it is generally referred to as *sequential data*, whereas the approaches dealing with these problems are known as *sequential data estimation* [7, 13]. The work presented in this thesis investigates the theory and application of two sequential data estimation problems, namely:

1. *Sequential Bayesian Estimation*: The first part of this thesis will introduce a probabilistic framework, which uses the sequential audio data recorded by a microphone array to track multiple concurrent speakers. To do so, we will first introduce a multiple speaker localization and detection framework, i.e., a framework that estimates the number and location of multiple active speakers on each time frame, and then show how this approach can be used as a measurement detector in combination with a Kalman-based recursive Bayesian filter to track and, thereby, extract the short-term trajectories of multiple concurrent speakers.

2. *Sequential Neural Estimation*: The second part of this thesis will focus on textual data. In particular, we will present new neural solutions to the language modeling task, where the data is processed as an ordered sequence of words. Thus, language modeling is defined here as the task of predicting the next word given its history, i.e., the sequence of its preceding words. The new proposed solutions will mainly focus on improving short and long range dependencies, which are considered to be a bottleneck for many Natural Language Processing (NLP) applications.

1.2 Thesis Highlights and Objectives

The work presented in this thesis focuses on two sequential data problems as explained in the previous section. While these two problems both deal with sequential data, there are fundamental differences regarding their problem formulation. More precisely, the multiple speaker tracking problem can be formulated as the task of tracking a hidden state, i.e., the speaker(s) location, from a set of measurements that are extracted from the audio signal and can be of a different nature. Thus, the corresponding solutions, generally known as *sequential filtering* [14, 15], alternate between two steps, namely, 1) predicting the next state using a *state model*, followed by 2) an update of the state once a new observation becomes available using a *measurement model*. In classical language modeling, however, the measurement model is not used and the task is defined as *sequence prediction*, which can be reduced to the first step of *sequence filtering*. This is mainly due to the fact that the actual state of interest, i.e., next word, becomes available and is therefore directly used in the prediction. Thus, language modeling does not necessary need a measurement model to update its state. Note that even in the general case where the actual next word is not available, e.g., in speech recognition and machine translation tasks, the predicted word is used instead, which does not bring any additional information that would require a state update.

The second fundamental difference between these two problems lies in the nature of their state of interest. That is, the *location space* is continuous. Thus, the problem can be defined and dealt with using continuous probability distributions, whereas the *word space* is discrete and can be reduced to a pre-defined vocabulary set, which makes it difficult to define and use continuous probability distributions in this space. This difference forces us to use models that can learn how to predict each word directly from the training data. In fact, this aspect makes the introduction of few state-of-the-art generative models, such as Generative Adversarial Networks (GAN) [16], to solve different NLP tasks generally difficult, since one would need to accurately define how samples drawn from a continuous representation space map into the discrete linguistic space.

1.2.1 Sequential Bayesian Estimation: Multiple Speaker Tracking

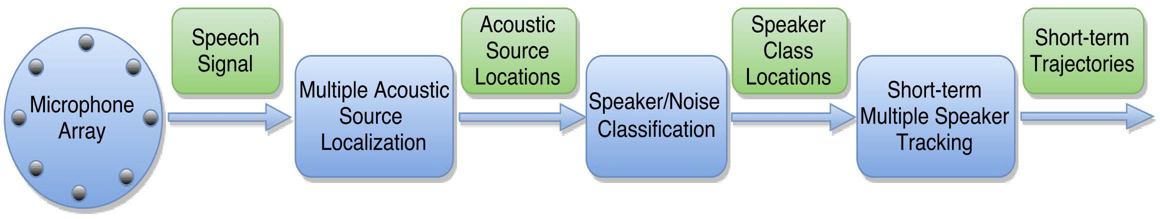

Microphone Array
Speech Signal
Multiple Acoustic Source Localization
Acoustic Source Locations
Speaker/Noise Classification
Speaker Class Locations
Short-term Multiple Speaker Tracking
Short-term Trajectories

Figure 1.1: *Block diagram illustrating the main components of the multiple speaker localization, detection and tracking solution proposed in the first part of this thesis. Each of these components will be discussed in details in the next chapters.*

The work detailed in the first part of this thesis comes as a novel solution to track multiple concurrent speakers. In particular, we propose a pipeline to solve the subsequent localization, detection and tracking of multiple overlapping speaker tasks. Figure 1.1 shows a block diagram of the different components of the proposed solution. The approaches considered here were mainly developed for wideband audio signals recorded with microphone arrays in a real in-door environment. The objective is to develop a probabilistic framework that sequentially, i.e., for each time frame, answers three main questions regarding the received audio signals:

1. **Where does the signal originate from?** The answer to this question aims at localizing the potential acoustic sources that emit the received signals. These acoustic sources can be human speakers producing signals of interest, i.e., speech, as they can be static noise sources, such as projectors and computers, or other occasional noise objects, such as doors and windows. The early multi-path reverberations are considered here as additional acoustic sources that can be detected as well. Throughout this thesis, we will refer to the task answering this question as *multiple acoustic source localization.*

2. **How many real simultaneous (overlapping) speakers?** The goal here is to classify the potential acoustic source locations answering the previous task into real speaker and noise sources. In the rest of this thesis, we will refer to this task as *multiple speaker detection.*

3. **Who are (locally) the speakers?** The approaches proposed to solve the previous two tasks are memoryless, i.e., they are performed on a frame-by-frame basis and do not include any localization or detection history to achieve the targeted tasks. Thus, this thesis also investigates a third task, which aims at filtering the history of the estimated speaker locations to track the different speech sources and estimate their trajectories as long as the speakers are active, i.e., speaking. These short-term location trajectories (over time) are considered to be unique local identifiers. We will refer to this task as *multiple concurrent speaker short-term tracking.*

The proposed probabilistic framework was developed to answer these three main questions while achieving the following objectives:

- Very little (to none) predefined assumptions and constraints, in particular, regarding the speakers location, their number and their movement.

- The approaches should be robust and stable across different indoor environments, in particular, in meeting rooms, offices and workplaces.

- Robustness to noise and reverberation. The type of noise we consider here is the typical in-door noise, such as projectors, door slams, etc. Although the proposed solutions can be further investigated and adapted to other noisy conditions, the work presented in this thesis was not tested or evaluated in such conditions.

- The developed solutions should be able to run online and (near) real-time.

1.2.2 Sequential Neural Estimation: Language Modeling

The second part of this thesis is focused on improving state-of-the-art Language Models (LMs). That is, we use model combination to tackle some of the shortcomings generally known to be present in standard neural architectures and to speed-up their training. we can divide the problems we are addressing in this thesis into the following categories:

- Loss of word position information in recurrent neural network models, which can model short patterns in language. These patters can be explicitly captured by N-gram and feed-forward models, whereas they are lost in recurrent models due to their usage of the single last seen word as an explicit history.
- Explicit and separate modeling of short and long range dependencies that most basic neural architectures do not address.
- Slow, sometimes impossible, training of large vocabulary neural language models. This problem is caused by the output layer normalization, which typically involves the evaluation of the softmax function over the complete vocabulary, for each training example.

In order to address these problems, we investigated model combination as a framework to develop new solutions. More precisely, we propose, in a first stage, new architectures that merge the learning capabilities of different basic models leading to hybrid neural network models. In a second stage, we investigate the generalization of this framework to be independent of the combined models and their number. The proposed solutions were designed to solve the problems mentioned above while targeting these objectives:

- No direct design or control over the linguistic features used to solve the language modeling task.
- Investigating the potential of neural networks for language modeling. Therefore, we will solely focus on neural network-based solutions.
- Avoid over-tuning and corpus-based models to solve the task. Thus, the new solutions should show their merits when they are trained and tested following comparable experimental setups to those used by the baseline models.
- Possibility to generalize these new solutions to other natural language processing tasks.

1.3 Chapter-wise Summary and Contributions

We present in this section a chapter-by-chapter review, in addition to the corresponding contributions and publications for each chapter.

Chapter 2 provides, in a first stage, a literature review of the multiple speaker localization, detection and tracking problem. That is, we define each of the main components needed to solve this problem and then highlight the main approaches proposed to achieve that. In a second stage, we present an overview of the AV16.3 corpus, which is extensively used in this thesis, in addition to the statistical evaluation framework that is used to assess the performance of the newly proposed solutions.

Chapter 3 introduces the basic statistical notions that are used throughout this thesis. In particular, we review here the expectation-maximization algorithm used to estimate the maximum likelihood solution of incomplete data, with an application to Gaussian mixture models. Then, we present the Bayesian estimation framework with its extension to sequential problems. Finally, we introduce few implementations of this framework, such as (unscented) Kalman filter and particle filters, that are generally used under different assumptions. These filters will be used as baseline models during the evaluation of the proposed solutions.

Chapter 4 presents the cornerstone model of the proposed multiple speaker localization, detection and tracking framework. That is, we explain in Section 4.2 that we consider the Time Difference Of Arrival (TDOA) of a given speaker as a random hidden variable, and then show how the Generalized Cross-Correlation (GCC) function, which is generally used to estimate the TDOA, can be interpreted as its probability density function (pdf). Based on this interpretation, Section 4.2.1 introduces two assumptions justifying the use of a Guassian Mixture Model (GMM) to approximate the TDOA pdf. The parameters of this GMM can be then estimated using two different approaches, namely, a Weighted Expectation-Maximization (WEM) algorithm that considers the GCC function as a set of weighted observations drawn from a hidden distribution [17, 18], this is explained in Section 4.2.2, and an acoustic-dominance based approximation that incorporates our knowledge into the estimation process, detailed in Section 4.2.3 [19]. Finally, we show how the GMM can be used to derive TDOA estimates in Section 4.2.4.

Chapter 5 introduces the proposed approaches to solve the first problem, namely, the multiple acoustic source localization task, which aims at extracting a fixed number of potential location estimates per time frame. These solutions are based on the assumption that each acoustic source in the room is generated, at most, by a single Gaussian component in each microphone pair GMM, estimated in Chapter 4. This chapter proceeds by reviewing the commonly used SRP solution and highlights its shortcomings in Section 5.1. Then, we introduce three new alternatives that deal with different conditions. That is, 1) Section 5.4 introduces a Probabilistic Steered Response Power (P-SRP) approach [18], which approximates the standard SRP solution based on the estimated TDOA-GMMs. Then, 2) Section 5.5 introduces a Cumulative Steered Response Power (C-SRP) approach [20], which improves the detection rate and speeds up P-SRP using a two stage localization process. That is, C-SRP reduces the search space into few regions with a high likelihood and then performs the standard SRP in the reduced space. Finally, Section 5.6 introduces an alternative Acoustic Event Joint Distribution (AEJD) [19], which solves the problem of local maxima that decreases the performance of SRP-based solutions, specially when only few microphones are available. This chapter also introduces an extension of each of the proposed solutions to the multiple speaker case.

Chapter 6 deals with the multiple speaker detection task. More precisely, the potential location estimates, obtained using one of the proposed approaches from Chapter 5, are not necessarily generated by an actual speaker but could also be generated by other acoustic sources, such as noise objects and reflections. Therefore, This chapter proposes to extract the actual speaker estimates by casting this problem as a binary classification task between a noise class and a speaker class, and then introduces in Section 6.3 an unsupervised Naive Bayesian Classifier (NBC) to solve it [21]. In order to achieve this goal, we also introduce here two new classification features: 1) Section 6.1 introduces the Dominance-based Cumulative Power (DCP) feature, which models the signal power emitted by an acoustic source, whereas 2) Section 6.2 introduces the Maximum Likelihood Error (MLE) feature, which describes the nature of the source. In order to be able to use this model in online systems, Section 6.5 introduces an online adaptation of the proposed classifier, in addition to the investigation of different additional features that can be used to enhance its performance [22].

Chapter 7 presents the proposed solutions for the sequential filtering problem. That is, we present here new Bayesian filters to track a single or multiple concurrent speakers. In a first stage, we consider the TDOA-based single speaker problem, which can be solved using simpler and much faster solutions compared to the multiple speaker case. To do so, we introduce two new solutions to deal with the problem of noisy/erroneous TDOA estimates, which drastically degrades the performance of standard approaches. The solution presented in Section 7.1.2 introduces a new Multiple Hypothesis Gaussian Mixture Filter (MH-GMF), which considers multiple TDOA estimates as observations instead of the most likely one, combined with a bank of parallel Kalman filters to process these observations [23]. As an alternative to increasing the robustness of the tracking filter, Section 7.1.3 introduces a new solution that improves the TDOA estimation itself using tracking information [17]. More precisely, this approach uses the predictive distribution of the tracking filter to enhance the TDOA-GMM, estimated in Chapter 4, before proceeding to the measurement detection stage. For the multiple speaker case, Section 7.2.2 introduces a new Short-Term Tracking (STT) approach [24], which considers the estimates classified as speaker location(s) (from Chapter 6) as noisy observations that are subsequently used to track multiple concurrent speakers. This new filter was particularly designed to overcome few problems that are related to the speech nature itself through the integration of an HMM, which models the different possible states of a speaker. Finally, we show in Section 7.2.3, how the overlap detection rate of multiple speakers can be improved by generalizing the approach introduced in Section 7.1.3 to the multiple speaker case [25].

Chapter 8 presents a literature review of statistical language modeling work in Section 8.1.2

with a particular focus on standard N-grams and their main shortcomings. Then, Section 9 discusses the basic neural network LMs and their underlying assumptions and modeling approaches. We also discuss in this chapter the difficulty of training large vocabulary neural LMs, and review some of the solutions proposed to overcome this problem in Section 8.2. Finally, Section 8.3 introduces the different corpora that are extensively used in this thesis, in addition to the different metrics used to assess the performance of the newly proposed solutions.

Chapter 9 introduces the basics and the mathematical formulation of the language modeling task, followed by a discussion of how short and long range context LMs approximate it, with a particular focus on neural network-based models. Finally, Section 9.2 briefly discusses the baseline neural architectures used in language modeling. Particularly, we review here the Feed-forward Neural Network (FNN), Elman Recurrent Neural Network (RNN) and the Long-Short Term Memory (LSTM) network.

Chapter 10 discusses some of the shortcomings of the baseline models, and then motivates the idea of using model combination as a solution that overcomes these problems. That is, we show here that merging multiple models in a single architecture can lead to a significant improvement when compared to the separate models. We introduce in this chapter two new examples of model combination followed by a generalization framework for neural model combination. More precisely, Section 10.1 introduces the first hybrid model; Sequential Recurrent Neural Network (SRNN), which proposes to enhance the standard RNN architecture using additional sequential connections that capture the word position information and any residual information, which may have been lost during the last update of the hidden state [26]. Then, Section 10.2 presents the second model; Long-Short Range Context (LSRC) network, which proposes to explicitly and separately model short and long range dependencies using two separate hidden states of an extended LSTM model. In doing so, this new model improves the context learning and representation and thereby the model performance [27]. Finally, Section 10.3 introduces a generalized architecture, which is independent of the type and number of models that can be combined [28]. In fact, this new model uses different neural architectures as kernels to capture different features, and then uses a mixing layer to combine these features.

Chapter 11 introduces our new approach; Batch Noise Contrastive Estimation (B-NCE), to train large vocabulary LMs [29]. More particularly, we briefly review in Section 11.1 the standard NCE approach and explain why it is not well-suited for batch training, and then introduce in Section 11.2 the proposed NCE extension to batch mode, which considers the words in the batch as target and noise samples at the same time (i.e., for each target word, the remainder of the batch is considered the be the noise samples). This chapter also introduces a new baseline and an experimental study of NNLMs on the one billion word benchmark.

Chapter 12 summarizes the work presented in this thesis, draws few conclusions and insights and introduces new research directions that can be explored in the future to improve some of the approaches proposed in this work.

Part I

Sequential Bayesian Estimation: Multiple Speaker Localization and Tracking

2 Background: Speaker Localization, Detection and Tracking

2.1 Speaker Localization and Detection

The design and performance of multiple object detection and tracking approaches are highly dependent on the model assumptions and constrains, in addition to the type of sensors that are considered and their performance. Among these sensors, microphone arrays have become essential to solve a large number of signal processing problems and to improve the single channel-based solutions [30]. This is mainly due to the spatial/temporal redundant information about the acoustic sources, which is carried by the audio signals and captured by the different channels. Their area of application includes speech separation [31], speech enhancement [32], acoustic source localization and tracking [33], but also more advanced approaches such as camera steering for teleconference systems and audio-visual tracking [34]. Among these applications, the detection and localization of multiple concurrent speakers i.e., estimating the number and the location of multiple speakers talking simultaneously, from a short speech segment remains a difficult and an open task. This chapter will review some of the research work proposed to tackle different aspects of this problem.

2.1.1 Single Speaker Localization

Acoustic source localization approaches can be divided into two main categories: single-step approaches, also known as direct approaches, and dual-step approaches, also known as TDOA-based methods or indirect approaches. In the later category, the TDOA introduced at each microphone pair is firstly estimated, and then the source location is extracted by virtue of geometrical intersections. One of the early solutions of this category was introduced by Schmidt [35], who proposed to extract the source location as the intersection of three planes. A similar spherical intersection method has been proposed by Schau et *al.* [36]. The spherical representation idea has also been used in [37] as an approximate linear Least Squares (LS) solution. The intersection based localization has been further developed to cover the Maximum Likelihood (ML) approximate solutions. This idea is well investigated in [38, 39], where a linear intersection method is proposed, and in [40], where the intersection of hyperbolic curves defined by the TDOA is considered. Although these approaches are generally fast and perform well in anechoic and noise-free environments, their performance drastically decreases in noisy/reverberant conditions. This, in fact, is due to the vulnerability of these approaches to noisy/erroneous TDOA measurements, which are produced by TDOA estimation methods whose performance decreases in these conditions.

The need for more robust source localization techniques regarding noise and/or reverberation led to a new category of direct approaches, which aim at inferring the source location directly from the signals using the redundant information about the acoustic sources, which is carried by the multi-channel signals. This is typically done by optimizing a criterion in a discrete grid over the 2-D/3-D space. This category can itself be divided into two main streams: 1) Spatial spectral estimation techniques, which mainly include Spatial Correlation Matrix (SCM)-based methods, and 2) Beamforming (BF)-based source localization [41]. The former stream provides a variety of approaches using different optimization criteria. The well-known approaches of this category are the Multi-Channel Cross-Correlation (MCCC)-based source localization [42] and the Adaptive Eigenvalue Decomposition (AED) [43]. While the first approach tends to search for the location which minimizes the determinant of SCM, the second approach aims at finding the location with the maximum eigenvalue. More details about these approaches can be found in [44, 45]. The most commonly used approach of the second stream is the Steered Response Power (SRP) technique [46]. This approach constructs a Delay-and-Sum Beamformer (DSB), which is steered at different locations. The source position is subsequently extracted as the one with the maximum power.

Although BF-based category of approaches is sufficiently robust to noise and reverberation, it is computationally burdensome as it requires a fine discretization of the 2-D/3-D space to achieve a good localization precision. Dmochowski et *al.* [47] proposed to reduce the search space to few regions with a high likelihood. This reduction is based on the inverse mapping of the TDOAs corresponding to the Generalized Cross Correlation (GCC) peaks. In the same direction, Do et *al.* [48, 49] proposed different space reduction strategies, which aim at reducing the search space at each iteration until it converges to the maximum point. Improving the SRP performance was further investigated using spatial averaging techniques. This idea was developed in [50], where the space is divided into sectors, e.g., vertical slices, and an SRP estimate is calculated for each sector. The source location is then obtained using a second, more refined, search step inside the sectors with maximum energy. This idea was further developed in [51], where an equivalent phase domain metric-based activity measure is investigated. A similar method is proposed in [52] using the fact that compact volumes in the location space; rectangular volumes in this case, map to closed intervals in the TDOA space.

2.1.2 Multiple Speaker Localization

Generalizing SRP techniques to the multiple speaker case drew much attention in the last decade. Lathoud et *al.* proposed in [53] an unsupervised threshold selection technique to estimate the number of speakers, the localization is then performed using the two-steps SRP method introduced briefly above [51]. Combining, respectively, Agglomerative Clustering (AC) and Gaussian Mixture (GM) modeling with the SRP was the subject of interest in multiple works; Nilesh et Raineh proposed in [54] a narrowband localization estimation technique combined with a GM model. This work is based on the disjointness of speech assumption in the frequency domain [55]. A different but similar work was proposed in [56], where the SRP is represented by a set of location samples selected according to their SRP outputs, and then approximated by a GM using the Expectation-Maximization (EM) algorithm. In the same line of thoughts, the authors of [57] proposed to cluster the set of samples using AC, which is expected to converge to the correct number and location of the speakers, whereas the authors of [58] proposed to divide the frequency domain to sub-bands, for which the SRP is calculated and then clustered using the AC technique.

2.1.3 Multiple Speaker Detection

A good multiple speaker localization performance cannot be achieved without a source detector, which classifies the obtained estimates into speaker/noise. This is mainly due to 1) the presence of noise and/or reverberation, which introduces secondary peaks, and to 2) the unknown time-varying number of sources per frame. Few attempts have been made to overcome this problem. The authors of [54] proposed to use the distance separating the estimates as a criterion to extract the number and location of the sources, whereas Do et *al.* [56, 57] proposed to combine the signal power with a double clustering technique to estimate the number of speakers. In a more advanced approach, Lathoud et *al.* [53] proposed an unsupervised threshold selection technique to control the false alarm rate.

2.2 Speaker Tracking

Tracking acoustic sources is becoming increasingly more important with its increasing number of applications, such as multi-party speech enhancement/separation, automatic camera steering, robotics, etc. Classical single acoustic source tracking approaches consist of two stages: 1) extracting the measurements, which can be either TDOAs at the sensor pairs [59, 17]; or noisy location estimates obtained with an SRP-based technique [46, 60, 61]. 2) These measurements are then processed by a filtering approach, such as Particle Filter (PF) [15, 62] or Kalman Filter (KF)-based approaches [63, 64], which extracts the actual location of the speaker(s). We discuss below the two main categories with a particular focus on their advantages and shortcomings.

2.2.1 TDOA-based Single Speaker Tracking

While SRP-based tracking is generally robust and reliable, it is very slow compared to TDOA-based tracking, which can easily lead to a real-time system. Unfortunately, the performance of the approaches that subscribe in this category of single speaker tracking drastically degrades in the presence of noise and multi-path effects. More precisely, under room acoustical conditions, early reflections and reverberation corrupt the GCC function, which is typically used to estimate the TDOA, through smearing and through introduction of secondary peaks [46, 62]. This in turn affects the tracking algorithms, which assume the error is a stationary Gaussian process, whereas the TDOA error in a multi-path environment is rather time-varying and multi-modal.

Few attempts have been made to solve this problem. Vermaak [62] proposed to use a multiple hypothesis particle filter, which considers all TDOA values as measurements associated to confidence weights. This approach has been further improved in [65], where an extended particle filter has been introduced. We will continue along these lines in this thesis by proposing two approaches to further improve the TDOA-based single speaker tracking performance.

2.2.2 SRP-based Multiple Speaker Tracking

Multiple object tracking is an open research topic that has a wide range of applications. More particularly, multiple speaker tracking using microphone arrays has become an essential tool to develop robust solutions to a large number of signal processing problems, such as multi-party distant speech recognition, speech separation/enhancement, speaker diarization, etc.

Extending the TDOA-based tracking to the multiple speaker case is very difficult given the data association problem it faces, i.e., how to assign peaks in the GCC functions to different speakers. Furthermore, these approaches are very sensitive to noise and reverberation, which makes them less attractive. Therefore, multiple speaker tracking approaches use SRP, instead of TDOA, to generate the measurements, as they extend the single speaker tracking solutions through a multi-modal estimation framework, which allows the tracking of multiple instantaneous speakers. Such approaches include the joint probabilistic data association filter [66], the multiple model particle filter [67] and the extended Kalman particle filter [68], to name a few.

Despite their relative success, these approaches were mainly designed to overcome few classical problems of multiple object tracking, such as the non-linearity of the state space model [60, 63, 66], the robustness to noise [68] and the correct estimation of the number of speakers [69]. These approaches, however, did not address two main problems related to the speech nature, namely, 1) the high discontinuity of spontaneous speech, where an active speaker becomes frequently inactive for a short time (100-300ms), and 2) the suppression problem, where the dominant speaker masks the remaining speakers. These two problems reduce the speaker detection rate, and thereby makes the tracking of acoustic sources possible *only in short-term*, i.e., while a speaker is talking without being suppressed by another source. Contrary to standard multiple object tracking problems, and as a first attempt to solve this short-term activity problem, Lathoud et al. [70] proposed a Short-Term Clustering (STC) approach, which extracts the speakers trajectories as short-term location clusters. The multiple speaker tracking work presented in this thesis follows a line of thought similar to [70] by proposing a multiple speaker short-term tracking framework, which consists of a bank of parallel KFs tracking multiple instantaneous speakers. The details of this approach are presented in Chapter 7.

2.3 Evaluation Framework of Speaker Localization and Tracking

This section introduces 1) the audio-visual AV16.3 corpus [71], which is used to evaluate the proposed speaker localization, detection and tracking approaches, and 2) the different metrics used throughout this thesis to report and evaluate the performance of the models.

2.3.1 Evaluation Corpus: AV16.3

The work conducted in this thesis was thoroughly evaluated on the AV16.3 corpus [71], which was particularity designed to cover different phenomena that occur in multi-party speech scenarios taking place in a standard meeting room. The abbreviation "AV16.3" means that the corpus is a collection of *Audio-Visual* data recorded in a fully synchronized manner with 16 microphones and 3 cameras. The microphones are distrusted on two 10 cm-radius, 8-microphone Uniform Circular Arrays (UCA), which are placed on a table with an inter-distance of 0.8 m. Figure 2.1 shows a top view of the physical setup used during the AV16.3 data collection, which took place in a smart meeting room of a size ≈ 30 cm^2 [72]. The *Speakers Area* marks the recording scene, which can be captured by all three cameras during the data collection.

The AV13.6 corpus provides very accurate annotation of the mouth location for 10 different audio sequences recorded under different scenarios. In particular, the 3-D trajectory of the mouth location of each speaker (ground truth), which is very crucial for the evaluation of speaker localization and tracking, was reconstructed using the cameras calibration parameters and the mouth tracking information provided by all cameras. Moreover, the visual tracking was facilitated using colorful balls that were placed on the head of each speaker. Figure 2.2 shows snapshots of a scene captured by the three cameras. We can see in this figure the small markers,

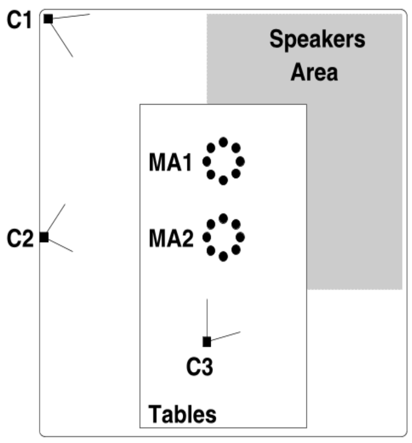
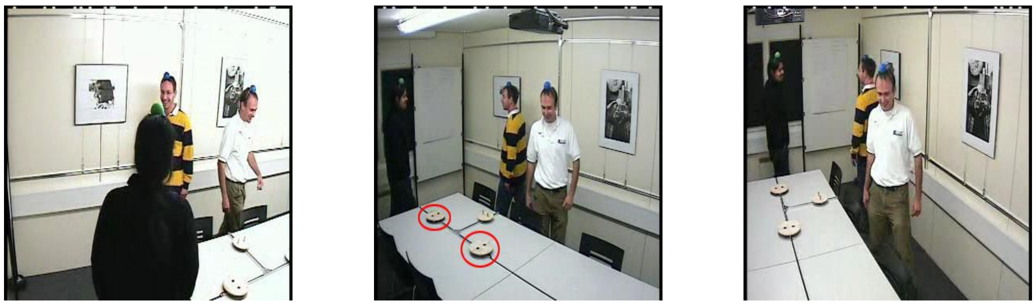


Figure 2.1: *Top view of the physical setup used in the AV16.3 recording: C1, C2 and C3 denote the cameras whereas MA1 and MA2 denote the two 8-microphone Uniform Circular Arrays (UCA). All recordings took place in the "Speakers Area" so that all participants can be captured by all three cameras.*

i.e., balls, placed on the head of each speaker, in addition to the location of the two microphone arrays on the table. This annotation setup led to a ground truth location of the mouth with an error margin of ≤ 5 cm. More details about the AV16.3 corpus can be found in [53, 73].



(a) Camera 1 (b) Camera 2 (c) Camera 3

Figure 2.2: *Snapshots from the sequence* `seq45-3p-1111` *taken by the three different cameras used to record the AV16.3 data. Camera 2 shows the location of the microphone arrays.*

The AV16.3 corpus contains 42 various audio recordings in WAV format and sampled at 16 kHz, with 10 sequences fully annotated using the setup described above. The sequence durations range from 14 seconds to 9 minutes and were recorded from 12 different speakers. AV16.3 offers various scenarios covering different interesting phenomena. In particular, from an audio speaker localization and tracking perspective, the corpus offers sequences modeling:

- Long audio segments with overlapping speech.
- Time-varying number of simultaneous speakers within a single sequence.
- Small, large and time-varying angular separation of the speakers.

- Close and far speaker locations from the microphone arrays.
- Static and dynamic speakers, including fast moving scenarios.

The work reported in this thesis will mainly focus on the 10 annotated sequences. Therefore, we introduce here more details about each recording in this set:

- `seq01-1p-0000`, `seq02-1p-0000` and `seq03-1p-0000`: These three sequences contain a single speaker counting from one to ten at 16 different locations. The speakers are static while speaking and silent while moving.
- `seq11-1p-0100`: A short sequence with one speaker who is mostly moving while speaking. The goal of this sequence is to test models on fast and unexpected motion scenarios.
- `seq15-1p-0100`: A single speaker alternating between silence and speech while moving. The evaluation on this sequence shows if tracking models can re-detect and resume tracking of the speaker after long segments of silence while moving.
- `seq18-2p-0101`: Sequence with a continuously overlapping speech from two seated speakers, who get slowly as close as possible from each other before getting back to their initial position.
- `seq24-2p-0111`: Sequence with two moving speakers crossing each other twice while talking most of the time.
- `seq37-3p-0001`: This is the longest sequence in the annotated set. It contains three speakers at static positions while speaking. Two speakers remain seated all the time while the third speaker is standing as he changes his location when he becomes silent. The number of overlapping speakers alternate between two and three.
- `seq40-3p-0111`: Sequence with three overlapping speakers for the complete duration of the recording. Two speakers are seated while the third one is standing as he walks back and forth once behind the seated speakers.
- `seq45-3p-1111`: Sequence with three overlapping and moving speakers who cross each other multiple times. The movement of the speakers in this scenario is not restricted to the *speakers area*. Figure 2.2 shows three snapshots of a scene from this scenario, taken by the three different cameras used in the AV16.3 recording.

2.3.2 Evaluation Framework and Metrics

Acoustic localization and tracking approaches are classically evaluated using restrictive metrics such as the *Anomaly Rate* (AR), which is defined as the percentage of estimates that vary from the true location by more than a given threshold, e.g., 5° [45]. While such metrics help us draw general conclusions about the overall performance, they fail in evaluating different aspects that can be very crucial to the model performance, such as the precision of the model, and may even change our conclusions. For instance, assuming a systematic error in the microphone locations; or the ground truth, will introduce a bias in the anomaly rate and thereby in our evaluation. To overcome this problem, Lathoud [73] proposed a statistical evaluation approach, which is more informative and robust to such errors. In this approach, a "Gaussian+Uniform" ($\mathcal{N}+\mathcal{U}$) distribution is used to classify the estimates into *speaker* and *noise* classes. More precisely, for each time frame, the error between each estimate and its closest ground truth location is calculated in a first step, followed by fitting a "Gaussian+Uniform" distribution to the data formed by the resulting set of errors using the EM algorithm. All different statistics needed to evaluate the localization and/or tracking performance can be then derived from the Gaussian distribution, which is expected to model the correct estimates, whereas the uniform distribution models the incorrect estimates.

While this approach is more informative and robust compared to the anomaly rate metric, its assumption that incorrect estimates are uniformly distributed, i.e., modeled with a uniform distribution, is valid only under few conditions. More particularly, the presence of non-negligible noise sources and/or strong reverberations emerging from a particular direction will lead to a bias in the incorrect estimates, causing the "$\mathcal{N}+\mathcal{U}$" approximation to fail in separating the two classes correctly. Therefore, we propose to replace the uniform distribution by a second Gaussian distribution. Thus, our new model is expressed as "$\mathcal{N}_s+\mathcal{N}_n$". In this case, $\mathcal{N}_n$ will be able to model any particular bias present in the incorrect (noise) estimates. If such bias is absent, $\mathcal{N}_n$ will simply approximate the uniform distribution through a large variance.

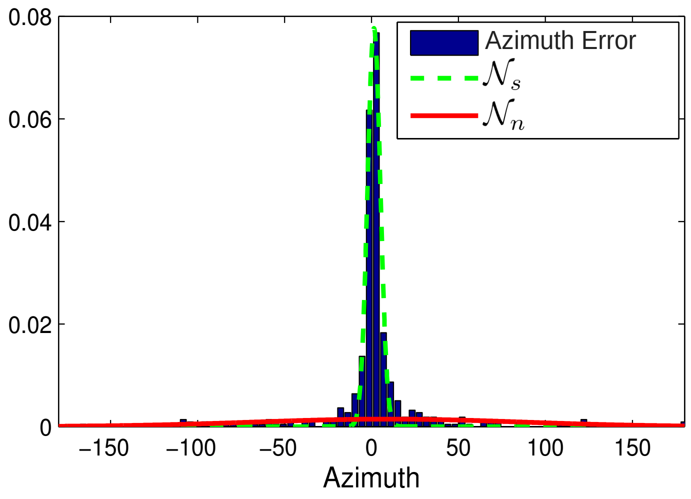


Figure 2.3: *Example illustrating the proposed "$\mathcal{N}_s+\mathcal{N}_n$" distribution used in the evaluation of a speaker localization and tracking approach on sequence* `seq11-1p-0100`.

Now, we will derive the different statistics needed for our evaluation and define the resulting metrics. To do so, let $\{\epsilon_\theta^i\}_{i=1}^N$ be a set of N errors calculated as the difference between each estimate and its closest ground truth location, and let $\mathcal{N}_x = \{p_x, \mu_x, \sigma_x\}$ with $x \in \{n, s\}$ be the speaker and noise Gaussian distributions obtained with the EM algorithm. p_x, μ_x and σ_x denote their weight, mean and covariance, respectively. Our first set of metrics is given by:

- The prior p_s of the speaker class: This is a soft approximation of the percentage of correct estimates. In fact, p_s is the opposite metric of the *anomaly rate* that takes into account all factors mentioned above.
- The covariance σ_s of the speaker class: This is a soft metric to evaluate the model precision. More precisely, a low covariance means that the Gaussian distribution approximating the speaker class is sharper. Thus, the location estimates are more accurate, and vice versa.
- The mean μ_s of the speaker class: This is a soft measure of any bias in the physical setup as discussed above. That is, a value different than 0 for μ_s means that either the setup information used in the localization is not accurate, such as the location of the microphones, or that the ground truth location has a uniform bias.

In addition to these metrics, which evaluate different aspects of the overall performance of the models, we will also report additional metrics, which particularly focus on the multiple speaker case. More precisely, we also report:

- The detection rate of each speaker (d_r): defined as the number of time frames, with a correct estimate for this given speaker, divided by the total number of frames which contain a ground truth location for this speaker.
- The overlap detection rate of order N (o_r): defined as the number of time frames with a correct estimation of N speakers divided by the total number of localization frames containing exactly N speakers.
- The tracking rate of each speaker (t_r): defined as the total tracking duration of a particular speaker divided by the total duration of tracking frames, which contain a ground truth location for this speaker. Note that $t_r \geq d_r$, this is due to the fact that tracking can also occur in frames where the speaker is not detected but considered as inactive.
- Run-time factor (t): defined as the processing time needed to run the model divided by the total duration of the localization/tracking frames.

In the case where a voice activity detection is used, the ground truth used to define these metrics will be restricted to the speech frames, whereas silence frames will be discarded from the evaluation.

3 Basics: Expectation Maximization and Bayesian Filtering

Probability is expectation founded upon partial knowledge. A perfect acquaintance with all the circumstances affecting the occurrence of an event would change expectation into certainty, and leave neither room nor demand for a theory of probabilities.

- George Boole (1815-1864), *An Investigation of the Law of Thought*

Probability theory is the branch of mathematics dealing with lack of perfect knowledge. Following the definition given by George Boole in the quote above, we can generally describe it as the scientific field which is concerned with the measure of uncertainty of events. Thus, it is a formalization of the principles and tools required to evaluate and understand the unknown through its available and partial knowledge. In formal terms, probability theory models events, also referred to as outcomes, as *random variables* to which it assigns probability distributions in the space formed by all possible events. These distributions can be then used to model and calculate the probability of occurrence of a given outcome under different circumstances, which are generally represented as dependencies on other observations [74].

Given that probability theory is a vast field to be fully covered in this thesis, we will introduce in this chapter the main aspects and notions that are needed to explain and understand the work conducted in this thesis. In particular, we will review the EM algorithm, which is an important class of ML solutions that are widely used to estimate the parameters of a hidden distribution from observed data, with an application to GM models. Then, we will introduce the Bayesian estimation approach of random states and its extension to sequential scenarios with few solutions. In the rest of this chapter, X and Y will denote a hidden and an observable random variable, respectively.

3.1 Expectation-Maximization Algorithm

The EM algorithm is a generalization of ML estimation to incomplete data. It is a solution that proposes to iteratively estimate the ML solution by re-constructing the missing information about the data [75, 76]. To illustrate this aspect, let us revisit the famous *coin flipping* experiment. Let us assume that we have N biased coins $c_1,\dots,c_N$ and that the probability of c_n landing on head is given by $P_n(h) = \theta_n$, whereas the probability of getting tail is $P_n(t) = 1-\theta_n$. $\theta_1,\dots,\theta_N$ are unknown parameters due to the biases of the coins. Our experiment consists of randomly choosing a coin from our set and flipping it. Repeating this experiment M times will lead to two sequences of observations, namely, the sequence of randomly chosen coins $\mathcal{X}_M = \{x_1,\dots,x_M\}$ with $x_m \in \{c_1,\dots,c_N\}$ for $m = 1,\dots,M$, and a sequence of outcomes $\mathcal{Y}_M = \{y_1,\dots,y_M\}$ with $y_m \in \{h,t\}$. The log-likelihood function for this problem is given by $L(\theta) = \log(p(\mathcal{Y}_M|\theta))$. Maximizing this function leads to the ML estimate of the parameter θ_n, which can be calculated as:

$$\hat{\theta}_n = \frac{\sum_{m=1}^{M} \delta_{x_m c_n} \cdot \delta_{y_m h}}{\sum_{m=1}^{M} \delta_{x_m c_n}}. \tag{3.1}$$

δ_{ab} is the Kronecker delta which is equal to one if $a = b$ and zero otherwise. In simple words, the ML estimate is the fraction of heads calculated on flipping experiments which were performed after randomly choosing coin number n.

Now, let us make this experiment a little bit more interesting and assume that we cannot observe the sequence of coins $\mathcal{X}_M$, which were randomly selected. We will refer to X as hidden or latent random variable. All what we can see in this case is the outcome of these tosses given by $\mathcal{Y}_M$. Obviously, the ML solution given by (3.1) can no longer be applied in this scenario. To overcome this problem, the EM approach proposes to softly reconstruct $\mathcal{X}_M$ by iteratively increasing the log-likelihood of the data, $L(\theta)$. This reconstruction, however, does not assume hard-known decisions about the chosen coins. Therefore, it does not search for the best sequence of coins which could have generated the outcomes. Instead, it expresses each step decision as a probability distribution over all coins. That is, we calculate the probability that each coin produced the outcome of each experiment, and then use the resulting probabilities to calculate a weighted ML solution. The derivation of the EM algorithm ensures the convergence of this iterative process towards a local maximum of the log-likelihood function $L(\theta)$.

Formally, the EM uses a given initialization of the parameters and the observed data to alternate between two steps until some convergence criteria are met. These two steps can be summarized as follows:

1. **Expectation Step (E-Step)**: Calculate the probability distribution over new guesses of the missing labels, e.g., the coin outcome in our example, using current model parameters.

2. **Maximization Step (M-Step)**: Re-estimate the model parameters by maximizing the log-likelihood of the data using the assigned soft labels from previous step.

The next section introduces a classical example of applying EM algorithm to estimate the parameters of the widely used GM distribution.

3.2 EM Example: Gaussian Mixture Model (GMM)

GMM is a family of probability distributions that are widely used to learn hidden structures present in the data through clustering, as they are also used to approximate probability density functions. While GMMs inherit the nice mathematical properties that the Gaussian distribution offers, they are more attractive due to their approximation power. That is, a GMM can approximate any multivariate pdf given enough components [77, 78]. GMMs are commonly used in combination with the EM algorithm to estimate their parameters.

A Gaussian mixture model is fully defined given the number of components in its mixture K and the mixture parameters $\Theta = \{p_k, \mu_k, \sigma_k\}_{k=1}^{K}$, where p_k, μ_k and σ_k denote the weight, mean and covariance of the k^{th} Gaussian component. Applying the EM algorithm, described in Section 3.1, to estimate the ML parameters of a GMM approximating some data $\mathcal{Y}_M = \{y_1, \ldots, y_M\}$ leads to the following E-Step and M-Step:

1. **E-Step**: Calculate the probabilities $p_{mk} = p(l_m = k|x, \Theta), m = 1, \ldots, M$ and $k = 1, \ldots, K$, which indicate how likely it is that the data sample y_m was generated by the k^{th} mixture component. Thus, $\{p_{mk}\}_{m,k}$ are soft assignments of the observations to the mixture components, whereas $\{l_m\}_m$ are the hidden observation labels.

2. **M-Step**: Re-estimate the parameters Θ using the new soft assignments according to:

$$C_k = \sum_{m=1}^{M} p_{mk}, \tag{3.2}$$

$$p_k = \frac{C_k}{\sum_{k=1}^{K} C_k}, \tag{3.3}$$

$$\mu_k = \frac{1}{C_k} \sum_{m=1}^{M} p_{mk} \cdot y_m, \tag{3.4}$$

$$\sigma_k^2 = \frac{1}{C_k} \sum_{m=1}^{M} p_{mk} \cdot (y_m - \mu_k)^2. \tag{3.5}$$

C_k can be interpreted as the soft fraction of samples that were assigned to the k^{th} component in the mixture.

3.3 Bayesian Estimation

Bayesian estimation considers the problem of inferring the state of a hidden variable from some related observation [74, 79]. This is generally done by integrating the available knowledge about the state, in the form of a *prior*, into the estimation process. Formally, if y denotes the available observation of the random variable Y, whereas X is the hidden target random variable. Then, the Bayesian estimation approach provides a way to calculate the distribution of interest $p_{X|y}(x)$ through integration of the state prior $p_X(x)$. This is generally done by considering the joint distribution of the state and observation $p_{X,Y}(x,y)$, which intrinsically integrates the statistical relationship between X and Y, and then conditioning it on the observation y.

Using Bayes theorem, we can formulate this estimation problem as:

$$p_{X|y}(x) = \frac{p_{X,Y}(x,y)}{p_Y(y)} = \frac{p_{Y|x}(y) \cdot p_X(x)}{p_Y(y)} = \frac{p_{Y|x}(y) \cdot p_X(x)}{\int p_{Y|x}(y) \cdot p_X(x) \, \mathrm{d}x}. \tag{3.6}$$

The approach considered here uses the first expression in (3.6). That is, we follow a two-step process to estimate $p_{X|y}(x)$ according to:

- **Step 1**: Calculate the joint distribution $p_{X,Y}$, which is defined over the augmented space formed by the state and observation. Thus, $p_{X,Y}$ summarizes the relationship between these two random variables. This calculation, however, requires the explicit expression of the observation model $y = h(x, w)$, which specifies the relationship between X and Y given some random noise W.
- **Step 2**: Calculate the posterior distribution $p_{X|y}$ by condition $p_{X,Y}$ on the available observation $Y = y$. This step basically returns a *plane* from the augmented space by restricting the joint distribution to the new information given by y.

Depending on the nature of h and the distributions of X, Y and W, different approaches can be used to calculate the transformation above. The next section will introduce the extension of this Bayesian estimation framework to the sequential case, and then discusses some particular models and implementations that are used as baseline filters throughout this thesis.

3.4 Sequential Bayesian Estimation

Now, let us consider the problem of estimating a time-varying system state x_t based on a sequence of corresponding observations $y_{1:t} = \{y_1, \ldots, y_t\}$. Solving this problem can be done by recursively applying the Bayesian estimation steps above to estimate the state x_t, at each time frame t. The resulting framework is generally referred to as sequential (also know as recursive)

Bayesian estimation. To do so, we need to introduce an additional transition model which, under Markov assumption of order one, specifies how the system state is expected to change from time step $t-1$ to time step t. In this case, the Bayesian estimation steps above can be extended to the sequential case according to:

- **Step 1**: Use the process model $x_t = f(x_{t-1}, v_t)$ to construct the state prior $p(x_t|y_{1:t-1})$ at time t, where v_t is the *process noise* with a random variable V_t.
- **Step 2**: Calculate the joint predictive distribution $p(x_t, y_t|y_{1:t-1})$ of the state and observation using the measurement model $y_t = h(x_t, w_t)$ with *measurement noise* w_t.
- **Step 3**: Calculate the posterior distribution $p(x_t|y_{1:t})$ by conditioning the joint distribution $p(x_t, y_t|y_{1:t-1})$ on the realized (actually measured) observation $Y_t = y_t$.

We can see here that the main difference between the two approaches is the additional first step, which iteratively redefines the state prior using the additional process model. Then, the estimation follows the standard Bayesian estimation steps from the previous section.

In practice, the estimation of the prior $p(x_t|y_{1:t-1})$ in the first step is accomplished by transforming the joint random variable of the last state X_{t-1} and the process noise V_t according to $f : X_{t-1} \longrightarrow X_t = f(X_{t-1}, V_t)$. In the second step, the joint distribution of X_t and Y_t is constructed by transforming (X_t, W_t) according to the augmented measurement function $\tilde{h}$ [80]:

$$\begin{bmatrix} X_t \\ Y_t \end{bmatrix} = \tilde{h}\left(\begin{bmatrix} X_t \\ W_t \end{bmatrix}\right) \text{ with } \tilde{h}\left(\begin{bmatrix} x_t \\ w_t \end{bmatrix}\right) \triangleq \begin{bmatrix} x_t \\ h(x_t, w_t) \end{bmatrix}. \tag{3.7}$$

Recursion of the above mentioned transformations forms the Bayesian tracking framework. The posterior filtering distribution $p(x_t|y_{1:t})$ constitutes the complete solution to the sequential probabilistic inference problem and allows us to calculate any optimal estimate of the state. Although this approach is simple, the optimal solution is usually tractable only for linear and Gaussian systems. In this case, all the involved random variables remain Gaussian at all times and the posterior can be obtained as a conditional Gaussian distribution [80]. This analytical closed form solution is generally known as *Kalman filter* [81, 82]. Most real-world systems, however, are nonlinear and/or non-Gaussian. Therefore, the optimal solution becomes intractable and approximate solutions must be used. These alternatives include well-known extensions of Kalman filter, such as the Unscented Kalman Filter (UKF) [83, 84], the Extended Kalman Filter (EKF) [85], sequential Monte-Carlo methods, also known as Particle Filters (PF) [14, 15] and Gaussian sum filters [86]. The next sections introduce the mathematical formulation of the extensions that are used as baseline models in the evaluation conducted throughout this thesis.

3.4.1 Kalman Filter

Kalman filter is the recursive estimation framework that we obtain when the process and measurement models are linear and when the initial posterior $p_{X_0|y}$, the process noise and the measurement noise distributions are Gaussian. In this case, the recursive Bayesian framework above has a closed form solution where the posterior distribution $p_{X_t|y_{1:t}}$ stays Gaussian at all times. Thus, estimating this distribution is reduced to calculating its new mean μ_{X_t} and covariance Σ_{X_t}, for each new time step t. Assuming these conditions are satisfied, let us express the linear transition and measurement models as:

$$x_t = f(x_{t-1}, v_t) = F x_{t-1} + v_t, \quad \text{and} \quad y_t = h(x_t, w_t) = H x_t + w_t. \tag{3.8}$$

where F and H are the matrix representations of f and h, respectively, and let us denote the Gaussian noise distributions according to:

$$p_{V_t}(v_t) = \mathcal{N}(v_t, \mu_{V_t}, \Sigma_{V_t}), \quad \text{and} \quad p_{W_t}(w_t) = \mathcal{N}(w_t, \mu_{W_t}, \Sigma_{W_t}). \tag{3.9}$$

The first step of the Bayesian framework, described in Section 3.4, requires the calculation of the predicted distribution of the state according to the transition model f. Given the Gaussian posterior distribution $p_{X_{t-1}|y_{1:t-1}}(x_{t-1}) = \mathcal{N}(x_{t-1}, \mu_{X_{t-1}}, \Sigma_{X_{t-1}})$ at time $t-1$, applying the linear model in (3.8) leads to a new Gaussian prior distribution $p_{X_t|y_{1:t-1}}$, where

$$p_{X_t|y_{1:t-1}}(x_t) = \mathcal{N}(x_t, \widehat{\mu}_{X_t}, \widehat{\Sigma}_{X_t}) \quad \text{with} \quad \begin{cases} \widehat{\mu}_{X_t} &= F\mu_{X_{t-1}} + \mu_{V_t}, \\ \widehat{\Sigma}_{X_t} &= F\Sigma_{X_{t-1}}F^T + \Sigma_{V_t}. \end{cases} \tag{3.10}$$

The notation " $\widehat{}$ " is used to highlight that the corresponding parameters are priors that were estimated using the prediction model.

Now, we need to construct the joint predictive distribution $p_{X_t,Y_t|y_{1:t-1}}$ using the augmented measurement model $\tilde{h}$, as explained in the second step of the Bayesian estimation framework. This would lead to a joint distribution of the form

$$p_{X_t,Y_t|y_{1:t-1}}(x_t, y_t) = \mathcal{N}\left(\begin{bmatrix} x_t \\ y_t \end{bmatrix}, \begin{bmatrix} \widehat{\mu}_{X_t} \\ \mu_{Y_t} \end{bmatrix}, \begin{bmatrix} \widehat{\Sigma}_{X_t} & \Sigma_{X_t Y_t} \\ \Sigma_{Y_t X_t} & \Sigma_{Y_t} \end{bmatrix}\right). \tag{3.11}$$

Similarly to the first step, the mean μ_{Y_t} and covariances Σ_{Y_t} and $\Sigma_{X_t Y_t}$ are given by:

$$\begin{cases} \mu_{Y_t} &= H\widehat{\mu}_{X_t} + \mu_{W_t}, \\ \Sigma_{Y_t} &= H\widehat{\Sigma}_{X_t}H^T + \Sigma_{W_t}, \\ \Sigma_{X_t Y_t} &= \Sigma_{Y_t X_t}^T = H\widehat{\Sigma}_{X_t} \quad \text{(the covariance matrix is symmetric).} \end{cases} \tag{3.12}$$

The final step in the estimation process is to condition the joint distribution on the new observation $Y_t = y_t$ to obtain the final (updated) conditional posterior distribution $p_{X_t|y_{1:t}}$. In this case, the parameters of this distribution are given by:

$$p_{X_t|y_{1:t}}(x_t) = \mathcal{N}(x_t, \mu_{X_t}, \Sigma_{X_t}) \quad \text{with} \quad \begin{cases} \mu_{X_t} = & \widehat{\mu}_{X_t} + \Sigma_{X_t Y_t} \Sigma_{Y_t}^{-1}(y_t - \mu_{Y_t}), \\ \Sigma_{X_t} = & \widehat{\Sigma}_{X_t} - \Sigma_{X_t Y_t} \Sigma_{Y_t}^{-1} \Sigma_{Y_t X_t}. \end{cases} \tag{3.13}$$

These equations form the Kalman update step, which is generally expressed using *Kalman gain* K [85, 87]. The latter is defined as $K = \Sigma_{X_t Y_t} \Sigma_{Y_t}^{-1}$. In this case, these equations become

$$\begin{cases} \mu_{X_t} = & \widehat{\mu}_{X_t} + K(y_t - \mu_{Y_t}), \\ \Sigma_{X_t} = & \widehat{\Sigma}_{X_t} - K \Sigma_{Y_t} K^T. \end{cases} \tag{3.14}$$

We can see that the update of the mean μ_{X_t} uses a *correction* term $K(y_t - \mu_{Y_t})$ to adjust the predicted mean $\widehat{\mu}_{X_t}$. This correction is based on how far the received observation y_t is from the predicted observation μ_{Y_t}. Similarly, the update of the covariance matrix generally decreases the uncertainty of the filter, which is estimated during the prediction stage. This is reflected by the integration of the "information" obtained from the new observation through the subtraction of the term $K\Sigma_{Y_t} K^T$ from the predicted covariance $\widehat{\Sigma}_{X_t}$. The detailed derivation of the different Kalman filter steps can be found in [88].

3.4.2 Unscented Kalman Filter

While KF is an attractive solution, real-word applications do not generally satisfy the conditions needed for it to be applied. That is, either the dynamic state space models are not linear or the involved distributions are not necessarily Gaussian. In this case, an alternative solution becomes necessary to solve the Bayesian estimation problem.

Traditionally, EKF has been used as the standard generalization of KF to *nonlinear* problems [85]. The main idea behind EKF consists of linearizing the transition and/or measurement models, f and h, at the last optimal state x_{t-1}, and then apply standard KF on the linearized system. This transformation, however, assumes the existence of the Jacobian matrices, which are not always easy, sometimes not possible, to calculate. As an alternative, the Unscented Transform (UT) was proposed to address the deficiencies of the linearization using a deterministic sampling approach [83, 84]. More precisely, UT represents each Gaussian distribution by a set of weighted samples, which are subsequently transformed through the nonlinear functions. The sample points are chosen such that they capture the mean and covariance of the transformed Gaussian distribution accurately up to the 2$^{\text{nd}}$ order in the Taylor series expansion, without any need to compute the Jacobian or Hessian matrices. Moreover, the computational complexity of the Unscented Kalman Filter (UKF) is of the same order as that of the EKF.

As we have seen in the sequential Bayesian estimation steps above, the main problem occurs in Step 1 and/or Step 2 due to the possible non-linearity of f and/or h, respectively. Both models, however, are expressed as a mapping of a joint distribution of two random variables into an augmented space. Thus, solving the non-linearity issue can be cast as the problem of accurately performing this transformation.

In order to show how UT solves the problem of transforming a joint distribution of two Gaussian random variables X and Y through a nonlinear mapping f, let n and m denote their respective dimensions, and let $p_X(x) = \mathcal{N}(x, \mu_X, \Sigma_X)$ and $p_Y(y) = \mathcal{N}(y, \mu_Y, \Sigma_Y)$ be their respective distributions. Furthermore, let AA^T and BB^T denote the Cholesky decompositions of Σ_X and Σ_Y, with A_i and B_i representing the rows of A and B, respectively. For an arbitrary $k \in \mathbb{R}$, we define $\lambda = n + m + k$. The resulting UT-based approximation of the joint distribution of X and Y is given by the $K = 2(n+m)+1$ sample points $\{[\mathcal{X}_k{}^T, \mathcal{Y}_k{}^T]\}_{k=0}^{K-1}$ with weights $\{\mathcal{W}_k^T\}_{k=0}^{K-1}$, which are calculated according to:

$$\begin{array}{lll|l}
\mathcal{X}_0 = \mu_X, & & \mathcal{Y}_0 = \mu_Y & \mathcal{W}_0 = k/\lambda \\
\mathcal{X}_{2i+1} = \mu_X + \sqrt{\lambda} A_i, & & \mathcal{Y}_{2i+1} = \mu_Y & \mathcal{W}_{2i+1} = 1/2\lambda \\
\mathcal{X}_{2i+2} = \mu_X - \sqrt{\lambda} A_i, & & \mathcal{Y}_{2i+2} = \mu_Y & \mathcal{W}_{2i+1} = 1/2\lambda \\
\mathcal{X}_{2(n+j)+1} = \mu_X, & & \mathcal{Y}_{2(n+j)+1} = \mu_Y + \sqrt{\lambda} B_i & \mathcal{W}_{2(n+j)+2} = 1/2\lambda \\
\mathcal{X}_{2(n+j)+2} = \mu_X, & & \mathcal{Y}_{2(n+j)+2} = \mu_Y - \sqrt{\lambda} B_i & \mathcal{W}_{2(n+j)+2} = 1/2\lambda
\end{array} \tag{3.15}$$

where $i = 0, ..., n-1$, $j = 0, ..., m-1$. k is a parameter specifying how much weight is placed on the mean $\mathcal{X}_0$. The mass-points representation above is known as the *augmented unscented transform* [83, 89]. Faubel et *al.* [90] proposed another representation, called the *extensive unscented transform*, which consists of generating the sample points for each Gaussian distribution, separately, and then considering their Cartesian product.

Once the weighted sample points are calculated, the Gaussian joint distribution $p_Z(z) = \mathcal{N}(z, \mu_Z, \Sigma_Z)$ approximating the nonlinear transformation of X and Y, i.e., $Z = f(X, Y)$, is obtained by transforming the sample points according to f:

$$\mathcal{Z}_k = f(\mathcal{X}_k, \mathcal{Y}_k), \quad k = 0, ..., K-1. \tag{3.16}$$

Then, the mean, covariance and cross-covariance matrices of p_Z are given by:

$$\mu_Z = \sum_{k=0}^{K-1} \mathcal{W}_k \mathcal{Z}_k, \tag{3.17}$$

$$\Sigma_Z = \sum_{k=0}^{K-1} \mathcal{W}_k (\mathcal{Z}_k - \mu_Z)(\mathcal{Z}_k - \mu_Z)^T, \tag{3.18}$$

$$\Sigma_{XZ} = \sum_{k=0}^{K-1} \mathcal{W}_k (\mathcal{X}_k - \mu_X)(\mathcal{Z}_k - \mu_Z)^T. \tag{3.19}$$

The UKF uses this weighted sample points-based transformation in the first and second steps of the Bayesian estimation framework, described in Section 3.4, to predict and then update the state X_t according to the augmented transition model $\tilde{f}$ and measurement model $\tilde{h}$.

Note that $\tilde{f}$ can be defined the same way we defined $\tilde{h}$. The third step in the estimation process follows the equations derived in the Kalman filter Section. In this case, Σ_{XZ} is the approximated cross-covariance matrix needed to calculate the Kalman gain.

3.4.3 Particle Filters (PF)

As mentioned above, the second case where KF cannot be applied concerns systems in which the involved random variables do not follow a Gaussian distribution. In such cases, EKF and UKF cannot be used as well, since the latter deal only with nonlinear systems but assume the Gaussianity of the distributions. As an alternative, particle filters have been proposed to solve this problem [14, 15].

PFs propose to represent the probability density function as an empirical distribution of N samples. On the contrary to UT, the PF samples are randomly drawn, and their number is generally much larger. In order to introduce the derivation of the Bayesian estimation framework using PFs, let $\{x_{t-1}^{(n)}\}_{n=1}^{N}$ be the set of the resulting random samples, we have in this case

$$p_{X_{t-1}|y_{1:t-1}}(x_{t-1}) = \frac{1}{N}\sum_{n=1}^{N}\delta_{x_{t-1}x_{t-1}^{(n)}}, \tag{3.20}$$

δ_{ab} is the Kronecker delta, which is equal to 1 if $a = b$ and 0 otherwise. The prediction step of the sequential Bayesian estimation framework is done by drawing N random samples, $\{v_t^{(n)}\}_{n=1}^{N}$, from the noise distribution p_{V_t} and then applying the transition model f sample-wise. That is, the prior distribution $p_{X_t|y_{1:t-1}}$ is given by:

$$p_{X_t|y_{1:t-1}}(x_t) = \frac{1}{N}\sum_{n=1}^{N}\delta_{x_t x_t^{(n)}}, \quad \text{with} \quad x_t^{(n)} = f(x_{t-1}^{(n)}, v_t^{(n)}). \tag{3.21}$$

In order to calculate the empirical distribution approximating the joint distribution of the state and observation, we first expand the latter according to:

$$\begin{aligned}
p_{X_t,Y_t|y_{1:t-1}}(x_t, y_t) &= p_{Y_t|x_t,y_{1:t-1}}(y_t)\cdot p_{X_t|y_{1:t-1}}(x_t) && (3.22)\\
&= p_{Y_t|x_t}(y_t)\cdot p_{X_t|y_{1:t-1}}(x_t) && (3.23)\\
&= p_{Y_t|x_t}(y_t)\cdot\left(\frac{1}{N}\sum_{n=1}^{N}\delta_{x_t x_t^{(n)}}\right) && (3.24)\\
&= \frac{1}{N}\sum_{n=1}^{N}\left(\delta_{x_t x_t^{(n)}}\cdot p_{Y_t|x_t^{(n)}}(y_t)\right). && (3.25)
\end{aligned}$$

The transition from (3.22) to (3.23) uses the *output independence assumption,* which states that, given the current state x_t, the current observation y_t is independent of all other states and observations. In order to condition the joint distribution on the received observation, we first express the posterior distribution in terms of the joint distribution and then replace the latter by its expression given by (3.25). That is,

$$p_{X_t|y_{1:t}}(x_t) = \frac{p_{X_t,Y_t|y_{1:t-1}}(x_t, y_t)}{p_{Y_t|y_{1:t-1}}(y_t)} = \frac{1}{N}\sum_{n=1}^{N}\left(\delta_{x_t x_t^{(n)}} \cdot \frac{p_{Y_t|x_t^{(n)}}(y_t)}{p_{Y_t|y_{1:t-1}}(y_t)}\right). \tag{3.26}$$

The normalization term $p_{Y_t|y_{1:t-1}}(y_t)$ can be calculated by marginalizing the joint distribution over X_t, using also its expression in (3.25), according to:

$$\begin{aligned} p_{Y_t|y_{1:t-1}}(y_t) = \int p_{X_t,Y_t|y_{1:t-1}}(x_t, y_t)\,\mathrm{d}x_t &= \int \frac{1}{N}\sum_{n=1}^{N}\left(\delta_{x_t x_t^{(n)}} \cdot p_{Y_t|x_t^{(n)}}(y_t)\right)\mathrm{d}x_t && (3.27)\\ &= \frac{1}{N}\sum_{n=1}^{N} p_{Y_t|x_t^{(n)}}(y_t). && (3.28)\end{aligned}$$

Finally, integrating this expression into (3.26) leads to the posterior distribution

$$p_{X_t|y_{1:t}}(x_t) = \sum_{n=1}^{N}\left(\frac{p_{Y_t|x_t^{(n)}}(y_t)}{\sum_{n=1}^{N} p_{Y_t|x_t^{(n)}}(y_t)}\right)\cdot \delta_{x_t x_t^{(n)}} = \sum_{n=1}^{N} w^{(n)} \cdot \delta_{x_t x_t^{(n)}}, \tag{3.29}$$

$\{w^{(n)}\}_{n=1}^{N}$ are the particle weights, which are given by:

$$w^{(n)} = \frac{p_{Y_t|x_t^{(n)}}(y_t)}{\sum_{n=1}^{N} p_{Y_t|x_t^{(n)}}(y_t)}. \tag{3.30}$$

PF algorithms generally use a resampling algorithm to avoid the degeneracy problem, which occurs when the weights of a few particles become significantly large, whereas the remaining particles have small weights. Resampling algorithms solve this problem by generating a new set of particles that approximate the same filtering density but have uniform weights [14, 15].

4

TDOA Modeling and Estimation

The arrival of sound waves at an M-microphone array introduces time differences between the individual sensor/microphone pairs. The time difference depends on the positions of the microphones $\mathbf{m}_i$, $i = 1,\ldots,M$, as well as the source location $\mathbf{l}$, which is typically given by its spherical coordinates, i.e., the radius r, azimuth θ and elevation ϕ. More precisely, the TDOA introduced at the microphone pair $q = \{m_i, m_k\} \in \mathcal{Q}$ is given by:

$$\tau^q(\mathbf{l}) = T_k(\mathbf{l}) - T_i(\mathbf{l}), \tag{4.1}$$

$T_{\{\cdot\}}(\mathbf{l})$ denotes the Time Of Flight (TOF), also known as *time of arrival*, from the location $\mathbf{l}$ to the microphone position $\mathbf{m}_{\{\cdot\}}$, whereas $\mathcal{Q}$ denotes the set of all microphone pairs.

Assuming a known source location $\mathbf{l}$ and microphone positions $\mathbf{m}_i$ and $\mathbf{m}_k$, and using c to denote the speed of sound, (4.1) can be then expressed as:

$$\begin{aligned}
\tau^q(\mathbf{l}) &= \frac{d_k(\mathbf{l})}{c} - \frac{d_i(\mathbf{l})}{c} \\
&= \frac{\|\mathbf{l}-\mathbf{m}_k\| - \|\mathbf{l}-\mathbf{m}_i\|}{c} && (4.2) \\
&\approx \frac{\mathrm{DOA}[\theta,\phi]^T \cdot (\mathbf{m}_k - \mathbf{m}_i)}{c} \quad \text{(under the far-filed assumption)}, && (4.3)
\end{aligned}$$

where $\|\cdot\|$ denotes the Euclidean norm and $d_{\{\cdot\}}(\mathbf{l})$ denotes the distance between the location $\mathbf{l}$ and the microphone position $\mathbf{m}_{\{\cdot\}}$. The far-field assumption allows us to ignore the distance between the array and the acoustic source. In this case, we can calculate the TDOA using the direction of arrival DOA$[\theta,\phi]$, which is given by the azimuth θ and elevation ϕ.

We discussed in Chapter 2 that acoustic source localization methods use the TDOA in two different ways to extract the source location. These two categories are:

1) Indirect approaches, also known as *dual-step* approaches or TDOA-based approaches, estimate the TDOA $\widehat{\tau}^q$ introduced by the acoustic source at each microphone pair $q = \{i,k\} \in \mathcal{Q}$, in a first step, and then infer the source location from the obtained TDOA estimates and the array geometry. More specifically, if $\{\widehat{\tau}^q\}_{q\in\mathcal{Q}}$ denotes the set of TDOA observations, the ML estimate of the source location can be obtained by minimizing the error function

$$\epsilon(\mathbf{l}) = \sum_{q\in\mathcal{Q}} \frac{1}{(\sigma^q)^2} \cdot \left[\tau^q(\mathbf{l}) - \widehat{\tau}^q\right]^2, \tag{4.4}$$

where $(\sigma^q)^2$ is the error variance associated to the TDOA measurement from microphone pair q. Given that the location-TDOA function given by (4.2) is a many-to-one mapping, the minimization problem given by equation (4.4) has no closed-form solution. The indirect approaches propose to approximate the solution by virtue of different geometrical intersection methods [36, 37, 38, 39, 40]. Chapter 2 discusses the details of some of these approaches.

2) Direct approaches, also known as *one-step* approaches, extract the source location directly from the audio signal [46]. In fact, these methods skip the TDOA estimation step which becomes intrinsic to the method. This is done by directly replacing the TDOA by its location-TDOA mapping, given by (4.2), into the problem formulation, and then solve the problem directly for the location $\mathbf{l}$. This category itself can be divided into two sub-categories, namely, 1) SCM-based methods [42, 43, 44, 45] and 2) SRP-based techniques [46, 47, 48, 52, 53]. Chapter 2 provides a detailed discussion of direct and indirect approaches.

The work presented in this thesis is a bridge between these two categories. More particularly, we propose an estimation approach that considers the TDOA, introduced by an acoustic source at a given microphone pair, as a random variable, and then introduce a probabilistic approach to approximate its pdf. The resulting pdfs are used in different (direct and indirect) methods to estimate and track single and multiple speakers.

This section will briefly review the TDOA estimation problem, and then introduce the proposed probabilistic approach to estimate the TDOA pdf at each microphone pair.

4.1 GCC-based TDOA Estimation

The most popular approach to estimate the TDOA introduced by an acoustic source at a microphone pair $q = \{i,k\}$ is the GCC function. This approach is based on calculating a weighted correlation between the signals $s_i(t)$ and $s_k(t)$, which have been received by the i$^{\text{th}}$ and k$^{\text{th}}$ microphones, receptively. Mathematically, the Fourier transform of the cross-correlation of two signals, i.e., *cross spectral density* or *cross spectrum*, is given by:

$$\mathcal{C}^q(\tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S_i(\omega) \cdot S_k{}^*(\omega) \cdot e^{j\omega\tau} \, \mathrm{d}\omega \,, \tag{4.5}$$

where $S_i(\omega)$ and $S_k(\omega)$ denote the short-time Fourier transforms of $s_i(t)$ and $s_k(t)$, respectively, and $\mathcal{C}^q(\tau)$ is their cross spectrum. The GCC function [59] uses an additional filter $\psi(\omega)$, carefully chosen based on a predefined criterion, to extended $\mathcal{C}^q(\tau)$ according to:

$$\mathcal{R}^q(\tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \psi_{i,k}(\omega) \cdot S_i(\omega) \cdot S_k{}^*(\omega) \cdot e^{j\omega\tau} \, \mathrm{d}\omega \,. \tag{4.6}$$

The most common choice of $\psi_{i,k}(\omega)$ is the Phase Transform (PHAT) filter [59], which assumes that the time delay between two signals is carried only by the phase. Thus, the signals magnitude is normalized on each frequency bin. Formally, PHAT weighting is given by:

$$\psi_{i,k}^{PHAT}(\omega) = \frac{1}{|S_i(\omega) \cdot S_k{}^*(\omega)|} \,. \tag{4.7}$$

In the classical GCC-based TDOA estimation approach, the most *likely* TDOA $\widehat{\tau}^q$, introduced by the acoustic source at the microphone pair $q = \{i, k\}$, is extracted as the time delay alignment with the maximum GCC value. That is,

$$\widehat{\tau}^q = \underset{\tau}{\operatorname{argmax}} \mathcal{R}^q(\tau) \,. \tag{4.8}$$

Note that all experiments conducted in this thesis and involving the GCC function use the PHAT weighting.

4.2 Estimation of TDOA Probability Density Function

The TDOA that an acoustic source introduces at a microphone pair is estimated as the time alignment which maximizes the GCC function of the two signals. Hence, the higher the GCC value is the more likely it is that the alignment is the *true* TDOA. From this point-of-view, the *normalized* GCC (nGCC) function of these two signals can be interpreted as a pdf of the TDOA; negative GCC values, if there are any, are set to 0. Alternatively, the nGCC function could be regarded as a set of weighted observations/particles drawn from a hidden distribution.

4.2.1 GM Approximation of TDOA Distribution: Assumptions

In the research conducted in this thesis, instead of considering the TDOA value corresponding to the maximum peak in the GCC as a single estimate, as it is typically done in acoustic source localization approaches, we propose to approximate its hidden distribution, and thereby estimate it, as a GM model. In doing so, this approach will allow us to evaluate the uncertainty about the TDOA estimates using their neighboring observations, and will lead to a novel probabilistic framework to solve the source localization, detection and tracking problem.

The GM choice is motivated by the multi-modality of the GCC function in noisy and/or reverberant environments and is based on two assumptions:

Assumption (A): Each peak in the nGCC function represents a *single* acoustic event in the room. This acoustic event is then assumed to be dominant in a closed interval of TDOA values, that we subsequently call the *interval of dominance*. Extracting these intervals, and associating the ones representing the same acoustic event but in different microphone pairs, allows us to extract and separate the different sources. Theoretically, this assumption does not always hold. To examine it more closely, let us consider an acoustic event corresponding to a given GCC peak. In this case, the set of locations having the same TDOA value can be approximated, under the far field assumption, by a cone [38]. This is due to the nonlinear location-TDOA function, given by (4.3), which is a many-to-one mapping. As a result, all the acoustic events lying on this cone will correspond to the same GCC peak and will subsequently have the same interval of dominance. Using more than one microphone pair, however, reduces the set of possible locations, i.e., corresponding to the same peak, to the intersection of the cones, this allows us to differentiate the sources. Hence, we can conclude that for any pair of locations, there might be (in the worst case) a few microphone pairs, for which the GCC peaks are the same, but for the other pairs, this assumption is still valid. Actually, this assumption is indirectly used in the intersection-based source localization as well as in speaker tracking algorithms which consider multiple GCC peaks as potential TDOA observations [60, 91].

Assumption (B): The error introduced in the TDOA detection, i.e., peaks of the GCC function, follows a Gaussian distribution. This assumption has been widely used and was repeatedly shown to achieve good results in practice, especially in acoustic source tracking applications [61, 62, 63, 91, 92].

The rest of this section introduces the mathematical formulation of the approaches adopted to estimate the hidden TDOA-GM distribution.

Estimating a good GM approximation from given data generally requires a sufficiently large number of observations. In order to increase the number of TDOA samples given by the GCC function, the later can be interpolated. The interpolation factor generally depends on the signal sampling rate and the dimensions of the microphone array. In the experiments conducted in this thesis, we use an interpolation factor between 5 and 20, depending on the approach.

Let s be the interpolation step and let $\{\tau_c^q\}_{c=1}^{C^q} = [-\tau_{max}^q, -\tau_{max}^q + s, \ldots, \tau_{max}^q - s, \tau_{max}^q]$ be the set of TDOA values (in samples) of the microphone pair $q = \{i, k\} \in \mathcal{Q}$ (C^q is the number of samples) after the interpolation. The maximum TDOA for this microphone pair is given by:

$$\tau_{max}^q = \frac{d^q}{c} \cdot f_s = \frac{||\mathbf{m}_k - \mathbf{m}_i||}{c} \cdot f_s \, , \tag{4.9}$$

where $||\cdot||$ is the Euclidean norm, d^q is the distance between the two microphones and f_s is the sampling rate. Moreover, let $\{w_c^q\}_{c=1}^{C^q} = \{\text{nGCC}(\tau_1^q), \ldots, \text{nGCC}(\tau_{C^q}^q)\}$ be the corresponding *normalized* GCC values. Negative GCC values, if there are any, are set to zero.

Our goal now is to estimate the GM distribution $\mathcal{M}^q$, given by the number of mixture components $K^q (K^q \le C^q)$ and the parameters $\Theta = \{\mu_k^q, \sigma_k^q, p_k^q\}_{k=1}^{K^q}$, which optimally approximates the normalized GCC function at $\{\tau_c^q\}_{c=1}^{C^q}$. That is,

$$\begin{aligned} \mathcal{M}^q(\tau^q) &= \sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}_k^q\big(\tau^q, \mu_k^q, (\sigma_k^q)^2\big), \quad \text{such that} \\ \mathcal{M}^q(\tau^q) &\approx nGCC(\tau^q) \quad \text{for} \quad \tau^q \in \{\tau_c^q\}_{c=1}^{C^q}. \end{aligned} \tag{4.10}$$

The parameters μ_k^q, σ_k^q and p_k^q denote the mean, standard deviation and mixture weight of the k^{th} Gaussian component in the mixture. We propose to estimate this GM distribution using two different approaches:

1. A Weighted Expectation-Maximization (WEM) algorithm.
2. An acoustic event-based approximation of the proposed WEM, which was mainly designed to overcome the significant estimation run-time required by the WEM.

The rest of this section introduces the proposed algorithms. For sake of readability, the microphone pair index q is dropped in the remaining part of this section.

4.2.2 The Weighted Expectation-Maximization (WEM) Algorithm

The estimation of the GM hidden parameters Θ can be achieved by, firstly, creating a larger TDOA set of samples with uniform weights. This can be done using the same approach used in particle filters. That is, we consider the samples and their weights as particles and resample them according to their weights, using for example importance sampling or any other sampling method. At the end of this step, we obtain a new set of samples with uniform weights but where the particles mass is dense at the original samples with larger weights. Then, the standard EM algorithm can be used to estimate the TDOA-based GM model. In this thesis, however, we propose a strictly equivalent approach, where the sample weights become part of the EM learning process but with a reduced computation cost. This approach is subsequently called the weighted EM algorithm and iterates between two steps:

1. **Expectation Step**: This step performs a soft assignment of the observations to the mixture components. This is given by the probabilities p_{ik}, $i = 1, \ldots, C$ and $k = 1, \ldots, K$, which indicate how likely it is that the observation τ_i was generated by the mixture component k. Formally: $p_{ik} = p(l_i = k | \tau_i, \Theta)$ where l_i is the observation label in the mixture.
2. **Weighted Maximization Step**: In this step, the mixture parameters Θ are re-estimated

using the new soft observation-label assignments according to:

$$C_k = \sum_{i=1}^{C} w_i \cdot p_{ik}, \tag{4.11}$$

$$p_k = \frac{C_k}{\sum_{k=1}^{K} C_k}, \tag{4.12}$$

$$\mu_k = \frac{1}{C_k} \sum_{i=1}^{C} w_i \cdot p_{ik} \cdot \tau_i, \tag{4.13}$$

$$\sigma_k^2 = \frac{1}{C_k} \sum_{i=1}^{C} w_i \cdot p_{ik} \cdot (\tau_i - \mu_k)^2. \tag{4.14}$$

C_k can be interpreted as the soft proportion of samples that were assigned to the k^{th} Gaussian. The observation weights $\{w_i\}_{i=1}^{C}$ incorporate the individual information about the peaks and valleys of the GCC function. As a result, the approximation relies more on the regions with a high likelihood, whereas it ignores the ones with low likelihood. This can be regarded as a *pseudo-regression* of the GCC function using a GM distribution (see example in Figure 4.1).

In practice, the WEM algorithm divides the TDOA space into TDOA-intervals $\{I_k\}_{k=1}^{K}$, where each mixture component is assumed to be dominant in a single interval. subsequently, these intervals are called *intervals of dominance*. The continuity of these intervals is a direct result of the convexity of the Gaussian distribution.Formally, the interval I_k associated with the k^{th} component is given by:

$$I_k = [\tau_k^{min}, \tau_k^{max}], \quad \text{where} \tag{4.15}$$

$$\tau_k^{min} = \underset{i}{\operatorname{argmin}}\{\tau_i \mid l_i = k\} \quad \text{and} \quad \tau_k^{max} = \underset{i}{\operatorname{argmax}}\{\tau_i \mid l_i = k\}. \tag{4.16}$$

4.2.3 Acoustic Event Dominance-based GM Approximation

Although the WEM algorithm is a reliable choice to estimate accurate hidden parameters of the GM models, applying it to each microphone pair q at each time frame t, however, would be computationally burdensome, which makes it unsuitable for real-time applications that require a fast performance. Thus, we use a computationally less expensive method that provides comparable results to those obtained with the WEM algorithm. This proposed approximation is a direct result of **assumption (A)**, which is detailed in Section 4.2.1. That is, each acoustic source in the room is dominant in its close “neighborhood”, which is defined as the area surrounding the acoustic source. In the GCC-based TDOA space, this assumption can be reformulated as: each peak in the GCC function is produced by, at most, one dominant acoustic event in the room within its “neighborhood” in the TDOA space. The “TDOA neighborhood” of an acoustic event is then defined as the interval between the left and right valleys surrounding

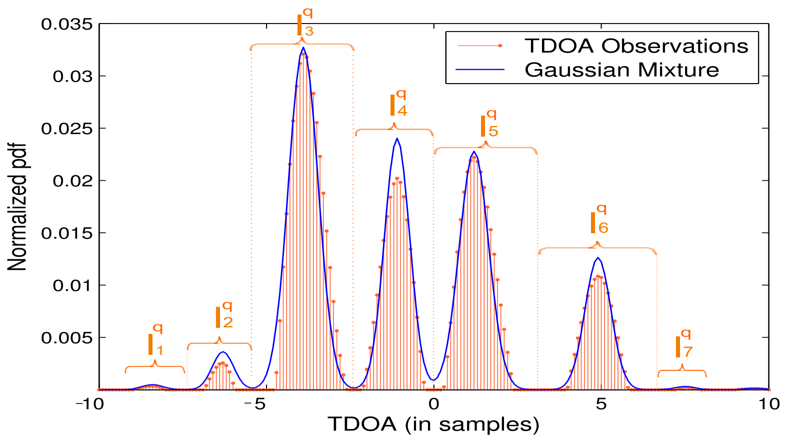


Figure 4.1: *Illustration of a TDOA Gaussian mixture distribution estimated with the proposed weighted expectation maximization algorithm ($K^q = 7$). The curly brackets (on top) indicate the intervals of dominance I_k.*

the GCC peak generated by this event. The resulting intervals are nothing else but an acoustic event-based approximation of the intervals of dominance produced by the WEM algorithm above. Furthermore, the resulting set of all intervals form an acoustic event-based partition of the TDOA space. The acoustic events we consider here are either produced by 1) a desired speaker, 2) a noise source or 3) reverberation, which is considered as an additional virtual noise source located within or at the environment borders, e.g., room walls, large objects, etc. It is also worth mentioning here that we do not impose any condition on the noise type or its duration. That is, a noise source can be temporary, e.g., door knock, or permanent, e.g., projector noise, as it can be produced by a point source, e.g., alarm, or a wide source, e.g., window slam.

Based on this GCC peak-acoustic event assumption, the number of components in each GM approximation is given by the number of peaks in the corresponding GCC function. We can also see that this assumption replaces the soft sample-component labels in the Expectation step of the WEM algorithm above by a hard one-to-one association. Formally, the parameters of the acoustic event-based GM approximation are estimated following these steps:

1. Extract the K peaks $\{\mu_1, ..., \mu_K\}$ of the GCC function.

2. Determine the intervals of dominance $\{I_1, ..., I_K\}$ corresponding to the different peaks. Each interval starts at the left foot of the corresponding peak and ends at the right foot. This step approximates the observation-component hard assignment, i.e. $p_{ik} \in \{0, 1\}$, see illustration in Figure 4.1.

3. Estimate the different Gaussian distributions associated to each interval of dominance using the weighted maximization step as described in the WEM algorithm above and the hard assignments p_{ik}.

More precisely, the parameters of the Gaussian distribution $\mathcal{N}(\tau, \mu_k, \sigma_k^2)$ and its mixture weight p_k, corresponding to the k^{th} acoustic event, are given by:

$$\mu_k = \arg\max_{\tau} \{k^{\text{th}} \text{ nGCC peak}\}, \tag{4.17}$$

$$\sigma_k^2 = \frac{\sum\limits_{i/\tau_i \in I_k} w_i \cdot (\tau_i - \mu_k)^2}{\sum\limits_{i/\tau_i \in I_k} w_i}, \tag{4.18}$$

$$p_k = \frac{\sum\limits_{i/\tau_i \in I_k} w_i}{\sum\limits_{i=1}^{C} w_i} \quad \text{or} \quad \frac{w_k}{\sum\limits_{i=1}^{K} w_i}. \tag{4.19}$$

Furthermore, the TDOA observation space $[-\tau_{max}, \tau_{max}]$ is expressed as the union of the intervals of dominance I_k, $k = 1, \ldots, K$. That is,

$$[-\tau_{max}, \tau_{max}] = \bigcup_{k=1}^{K} I_k. \tag{4.20}$$

The k^{th} interval of dominance I_k associated to the k^{th} acoustic event (GCC peak) is extracted according to:

$$I_k = [\tau_k^{min}, \tau_k^{max}], \quad \text{where} \tag{4.21}$$

$$\tau_k^{min} = \arg\max_{\tau} \{\tau \le \mu_k \mid \partial\mathcal{R}(\tau) = 0\} \quad \text{and} \tag{4.22}$$

$$\tau_k^{max} = \arg\min_{\tau} \{\tau \ge \mu_k \mid \partial\mathcal{R}(\tau) = 0\}. \tag{4.23}$$

$\partial\mathcal{R}$ denotes the first derivative of the GCC function $\mathcal{R}$. We can see that, in practice, τ_k^{min} and τ_k^{max} represent the left and right feet of the k^{th} peak μ_k in the GCC function, respectively.

Note that the open intervals of dominance, in both approaches, are mutually disjoint. This important property is very useful to efficiently separate and extract the different sources when they are mapped to the location space.

This acoustic event-based approximation proposes two alternatives to calculate the weights of the GM components. The first one is based on the hard assignments, i.e., $p_{ik} \in \{0, 1\}$, of the TDOA observations to each component, and assumes that the peak neighboring values were generated by the same acoustic event, hence they are assumed to contribute to the component weight. The second approximation, however, takes into account the nature of the source to relax the first assumption. That is, point sources are expected to have sharp peaks and fast decaying valleys while distributed sources, typically noise, may have wide peaks but slowly decaying valleys. Given this observation, a point source, e.g., an actual speaker, can get a lower weight compared to a distributed source, e.g., projector, under the first assumption.

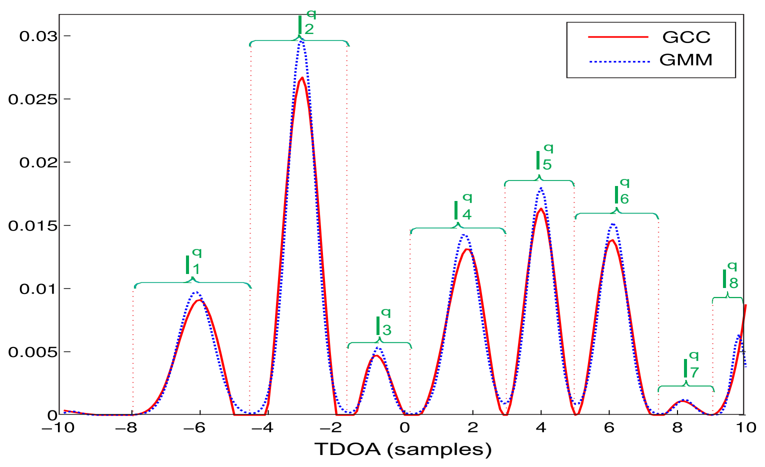


Figure 4.2: *Illustration of a TDOA Gaussian mixture distribution estimated using the acoustic event-based approximation ($K^q = 8$). The curly brackets I_k^q indicate the intervals of dominance (extracted as blocks) that were obtained under this assumption.*

4.2.4 TDOA Estimation

In the case where we are interested in the "best" TDOA estimate. The latter can be obtained in two different ways - the *maximum estimate* from (4.24) and the *mean estimate* from (4.25). That is,

$$\bar{\tau}_{max} = \arg\max_{\tau} \sum_{k=1}^{K} p_k \cdot \mathcal{N}(\tau; \mu_k, \sigma_k^2)\,, \tag{4.24}$$

$$\bar{\tau}_{mean} = \sum_{k=1}^{K} p_k \cdot \mu_k\,. \tag{4.25}$$

In both cases, the estimated TDOAs obtained from different microphone pairs can be stacked to form an observation vector, which can be then used as input in any indirect acoustic source localization or tracking approach. To construct this observation vector, we first estimate the TDOA $\bar{\tau}_t^q$ for each microphone pair $q \in \mathcal{Q}$, and then combine these individual estimates to form a joint measurement y_t, with $y_t = [\bar{\tau}_t^1, \ldots, \bar{\tau}_t^Q]$ for each time frame t.

The next chapter will show how the estimated TDOA probability density functions can be used to solve the acoustic source localization task following a new Bayesian framework.

5 Multiple Acoustic Source Localization

Beamforming-based acoustic source localization approaches, e.g., [46, 51, 48, 47, 52], can be used to solve different real word applications, such as in robotics [93, 94]. The success of this technology transfer, however, is highly dependent on the computation burden and performance of the adopted solution, which are generally limited due to the hardware and environment constraints that are imposed by the target application, e.g., the number of microphones that can be adjusted on a robot's head, the computation capabilities that it can achieve and the type/level of noise that the robot's motor produces. In particular, SRP-based techniques have a few drawbacks that can be summarized as follows:

- A higher localization precision needs a finer search grid over the 2-D or 3-D space, which can greatly increase the computation cost.
- In the case of multiple active speakers, detecting the number of speakers and estimating their locations is generally a difficult task.
- The localization becomes more difficult when a dominant source suppresses the rest of the speakers. This is generally the case when an acoustic source is much louder or closer to the microphone array compared to the rest of the speakers.

Depending on the target application, the required localization performance and resolution, different strategies can be combined to achieve the application goals. These solutions, however, mainly overcome the first issue but do not perform well regarding the second and third problems. More precisely, while the first problem can be addressed by reducing the search space through inverse mapping of relative delays [47] or through different methods for region contraction [48, 49, 95], the second and third issues remain difficult in practice. In [51], a sector-based approach was introduced to localize multiple speakers. Although this method has a low computation cost, it can only detect active regions, referred to as "sectors" in [51], which are more likely to contain the speakers. More accurate estimates require an additional search step inside each active sector. The authors of [56] proposed to combine agglomerative clustering with GMMs and region zeroing. A similar approach has been used in [54], where the GMM is obtained with the EM algorithm. The main difficulty of these approaches consists of determining the number of clusters or mixture components, which represent the active speakers.

We propose in this thesis an alternative solution based on the probabilistic interpretation and modeling of the TDOA that we presented in Chapter 4. The proposed solution overcomes the three drawbacks mentioned above in two steps:

1. Multiple acoustic source localization: We firstly extract all potential source locations. That is, we extract a static and predefined number of potential source locations for each time frame.

2. Multiple (overlapping) speaker detection: We classify, for each time frame, the previously extracted locations into speaker/noise classes using a set of newly introduced features and an unsupervised naive Bayesian classifier.

This chapter introduces the proposed approaches to achieve the first step while Chapter 6 presents the solutions developed to accomplish the second one. The proposed contributions to solve the multiple overlapping source localization problem (first step) can be summarized as follows:

1. A new probabilistic interpretation and approximation of the SRP function based on the TDOA pdfs estimated in Chapter 4.

2. An alternative joint probability distribution of multiple speakers in the location space, also based on the TDOA pdfs from Chapter 4. This solution is expected to perform better than the previous approach in the case where only few microphones are available.

3. A new cumulative SRP approach that benefits from the "intervals of dominance"-based partition of the TDOA space, discussed in Chapter 4, to speed-up and improve the detection rate of multiple speakers. This is done by mapping the notion of dominance from TDOA space to location space.

The probabilistic nature of these models helps us overcome the first and second drawbacks of SRP that were mentioned above. In particular, these models propose different multimodal probability distributions to approximate the multiple speaker pdf in the location space. In doing so, these models lead to the following advantages:

- The discretization of the 2-D/3-D space, required by classical SRP, is replaced by a numerical optimization approach, which can be much more accurate and faster compared to standard SRP approaches.
- The localization of multiple overlapping speakers is cast as an optimization problem, which aims at extracting multiple modes (maxima) of a multi-modal pdf.
- On the contrary to standard SRP variants, which can be used to only extract one point-location per speaker, these models can be additionally used to extract a location pdf for each speaker. This is particularly important in problems where the estimates variance/-confidence is required.

The rest of this chapter will briefly review the conventional SRP approach, and then introduce and discuss the details of the three proposed models.

5.1 Conventional Steered Response Power

The classical SRP approach [46] constructs a spatial filter using the location-TDOA mapping given by (4.2), i.e., a Delay-and-Sum Beamformer (DSB), which scans all possible source locations, given by a predefined grid over the 2-D or 3-D space. The speaker location is subsequently extracted as the grid position which maximizes the SRP. Formally, the DSB is given by:

$$b(t,\mathbf{l}) \overset{\text{def}}{=} \sum_{m=1}^{M} s_m(t - T_m(\mathbf{l})). \tag{5.1}$$

This formulation simply defines the signal at the output of the DSB, received from a source at location $\mathbf{l}$ at time t, as the accumulation of the different signals received by all microphones, subject to different time-shifts introduced by the TOF, which is needed to reach each microphone from location $\mathbf{l}$. In the frequency domain, (5.1) can be expressed as:

$$B(\omega,\mathbf{l}) = \sum_{m=1}^{M} S_m(\omega) \cdot e^{-j\omega T_m(\mathbf{l})}. \tag{5.2}$$

Similarly to the cross spectrum, the classical DSB can be extended to a Generalized Delay-and-Sum Beamformer (GDSB) by introducing a filter $\phi(\omega)$ according to:

$$B(\omega,\mathbf{l}) = \sum_{m=1}^{M} \phi(\omega) \cdot S_m(\omega) \cdot e^{-j\omega T_m(\mathbf{l})}. \tag{5.3}$$

The SRP is given by the spectral density of the DSB, which can be expressed as:

$$SRP(\mathbf{l}) = \int_{-\infty}^{\infty} |B(\omega,\mathbf{l})|^2 \cdot \mathrm{d}\omega = \int_{-\infty}^{\infty} B(\omega,\mathbf{l}) \cdot B^*(\omega,\mathbf{l})\, \mathrm{d}\omega, \tag{5.4}$$

$B^*(\omega, \mathbf{l})$ is the complex conjugate of $B(\omega, \mathbf{l})$. By substituting (5.2) in (5.4), we obtain

$$\begin{aligned} SRP(\mathbf{l}) &= \int_{-\infty}^{\infty} \Big(\sum_{i=1}^{M} S_i(\omega)\cdot e^{-j\omega T_i(\mathbf{l})}\Big)\cdot\Big(\sum_{k=1}^{M} S_k(\omega)\cdot e^{-j\omega T_k(\mathbf{l})}\Big)^* \,\mathrm{d}\omega \\ &= \int_{-\infty}^{\infty} \Big(\sum_{i=1}^{M} S_i(\omega)\cdot e^{-j\omega T_i(\mathbf{l})}\Big)\cdot\Big(\sum_{k=1}^{M} S_k{}^*(\omega)\cdot e^{j\omega T_k(\mathbf{l})}\Big) \,\mathrm{d}\omega \\ &= \int_{-\infty}^{\infty} \sum_{i=1}^{M}\sum_{k=1}^{M} \Big(S_i(\omega)\cdot e^{-j\omega T_i(\mathbf{l})}\Big)\cdot\Big(S_k{}^*(\omega)\cdot e^{j\omega T_k(\mathbf{l})}\Big) \,\mathrm{d}\omega \\ &= \int_{-\infty}^{\infty} \sum_{i=1}^{M}\sum_{k=1}^{M} S_i(\omega)\cdot S_k{}^*(\omega)\cdot e^{j\omega(T_k(\mathbf{l})-T_i(\mathbf{l}))} \,\mathrm{d}\omega. \end{aligned} \tag{5.5}$$

Using the expected TDOA given by (4.1) and the cross spectrum given by (4.5), (5.5) can be then expressed as:

$$\begin{aligned} SRP(\mathbf{l}) &= \sum_{i=1}^{M}\sum_{k=1}^{M} \int_{-\infty}^{\infty} S_i(\omega)\cdot S_k{}^*(\omega)\cdot e^{j\omega\tau_{i,k}(\mathbf{l})} \,\mathrm{d}\omega \\ &= \sum_{i=1}^{M}\sum_{k=1}^{M} 2\pi\cdot \mathcal{C}_{i,k}\big(\tau_{i,k}(\mathbf{l})\big). \end{aligned} \tag{5.6}$$

Thus, SRP is the spatial accumulation of all pair-wise cross-correlations, which are time-shifted to account for the steering delays introduced by the location **l** at each microphone pair. We can see here that SRP also includes the auto-correlation of each signal, which is independent of the space location **l**. Furthermore, by noticing that $\mathcal{C}_{i,k}(\tau_{i,k}(\mathbf{l})) = \mathcal{C}_{k,i}(\tau_{k,i}(\mathbf{l}))$, we can further expand (5.6) to become

$$SRP(\mathbf{l}) = 4\pi\cdot \sum_{\substack{i,k\\ i>k}}^{M} \mathcal{C}_{i,k}\big(\tau_{i,k}(\mathbf{l})\big) + \mathcal{K}, \tag{5.7}$$

$\mathcal{K} = \sum_{m=1}^{M} \mathcal{C}_{m,m}(0)$ is the accumulation of the auto-correlation of the signals received by all microphones. This term does not depend on the location **l** and can, therefore, be ignored during the source localization step.

Let $\mathcal{Q}$ be set of all microphone pairs i.e., $\mathcal{Q} = \{\{i,k\} \mid i,k = 1,\cdots,M \text{ and } i \neq k\}$. The cardinality Q of $\mathcal{Q}$, i.e., number of distinct microphone pairs, is given by $Q = |\mathcal{Q}| = \frac{(M-1)M}{2}$. Finally, the SRP function for a location $\mathbf{l}\big(r,\theta,\phi\big)$ can be defined as:

$$SRP\big(\mathbf{l}(r,\theta,\phi)\big) \propto \sum_{q\in\mathcal{Q}} \mathcal{C}^q(\tau^q(\mathbf{l})). \tag{5.8}$$

This definition can be further extended to the GDSB instead of the DSB. In particular, if we define the GDSB filter $\phi(\omega) = \dfrac{1}{S_m(\omega)}$ in (5.3) and replace the expression of the DSB by its GDSB counterpart in (5.4), the derivation above leads to the SRP-PHAT, which is given by:

$$SRP_{PHAT}\big(\mathbf{l}(r,\theta,\phi)\big) \propto \sum_{q\in\mathcal{Q}} \mathcal{R}^q(\tau^q(\mathbf{l})). \tag{5.9}$$

A detailed analysis and comparison of SRP and SRP-PHAT can be found in [46].

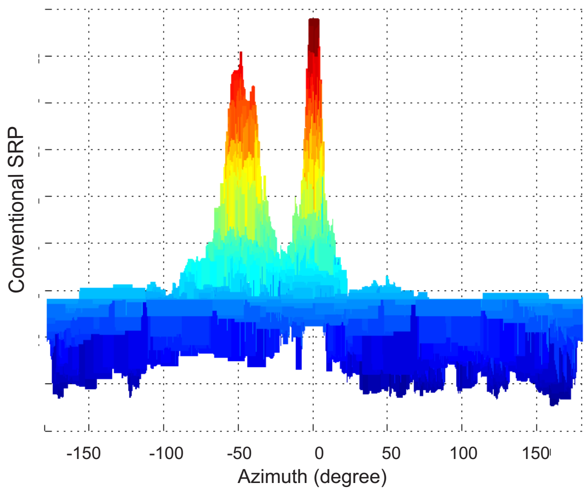


Figure 5.1: *A frontal view illustrating an SRP-PHAT example over a 2-D (azimuth [-180°,180°] and elevation [0°,90°]) grid. The example contains two distinct speakers, but the SRP shows that one of the peaks, representing one speaker, has two close but distinct local maxima. This phenomenon will generally cause standard SRP-based multiple speaker localization approaches to detect an additional speaker.*

Given SRP-PHAT, the source localization problem is solved by extracting the location $\hat{\mathbf{l}}$ which maximizes the SRP-PHAT output. That is,

$$\hat{\mathbf{l}} = \underset{\mathbf{l}}{\operatorname{argmax}}\, SRP_{PHAT}(\mathbf{l}). \tag{5.10}$$

Practically, solving the single source localization problem using SRP-PHAT can be implemented following the approach proposed in [46]. More precisely, we can summarize the SRP-based single source localization solution as shown in the following pseudo-code:

Algorithm 1 : SRP-PHAT-based Single Source Localization

```
1. Define a 2-D/3-D fine search grid over the space of interest.
for each time frame $t$ do
    2. Calculate GCC-PHAT function for all microphone pairs using (4.6).
    3. Calculate SRP-PHAT function for all locations on the grid using (4.2) and (5.9).
    4. Extract the speaker location using (5.10).
end for
```

Scanning all possible source locations is computationally expensive. Hence, a variety of approaches have been proposed to reduce the computation burden, such as splitting the 2-D or 3-D space into sectors [51] or adopting space reduction strategies in which only some regions of interest of the space are considered (see e.g., [47, 48, 49]). These approaches are, however, suffering from a poor resolution or from estimation errors due to the inherent discontinuity of the SRP (the latter can be partially alleviated by interpolating the GCC functions). Another problem is that conventional SRP-based acoustic source localization becomes more difficult in the dynamic multi-speaker case, which requires advanced techniques to jointly detect the time-varying number and location of the speakers. The next sections introduce three different solutions to overcome the shortcomings of the conventional SRP.

5.2 Mapping the TDOA Space Partition to the Location Space

Before we proceed to multiple source localization, we need to introduce an important new concept that will be used in all three proposed methods below, namely, mapping the GCC-based *partition* and *dominance* from the TDOA space to the location space.

The notion of *interval of dominance* in the TDOA space, introduced in Chapter 4, can be mapped to the *region of dominance* notion in the location space through the location-TDOA mapping given by (4.2). Formally, let $\mathcal{L}$ denote the location space, and let $\mathcal{I}^q = \{I_k^q\}_{k=1}^{K^q}$ be the TDOA space partition associated to the microphone pair $q \in \mathcal{Q}$. This partition can be obtained using either the WEM from Section 4.2.2 or its acoustic event dominance-based approximation, described in Section 4.2.3. Then, each interval of dominance I_k^q maps to a sub-space of locations $\mathcal{L}_k^q$, that we will refer to as *region of dominance*, according to:

$$\mathcal{L}_k^q = \{\mathbf{l} \in \mathcal{L} \mid \tau^q(\mathbf{l}) \in I_k^q\}. \tag{5.11}$$

Note that, through this mapping, $\mathcal{L}_k^q$ represents the space region where the associated acoustic event is dominant according to microphone pair q. Mapping all intervals of dominance $\{I_k^q\}_{k=1}^{K^q}$ leads to a q-partitioning $\mathcal{L}^q = \{\mathcal{L}_k^q\}_{k=1}^{K^q}$ of the location space $\mathcal{L}$, given by:

$$\mathcal{L} = \bigcup_{k=1}^{K^q} \mathcal{L}_k^q. \tag{5.12}$$

5.3 Mapping Acoustic Event Dominance to Location Space

The proposed multiple speaker localization approaches use the redundant information about the acoustic events, which is captured by different microphone pairs. More particularly, the localization of an acoustic event is mainly based on extracting the interval of dominance associated to it, for each microphone pair, and then map the resulting set of intervals to the location space to perform the localization.

Formally, Let $\mathcal{E}$ be an acoustic event and let us assume that we can extract, for each microphone pair $q \in \mathcal{Q}$, the interval of dominance $I_{\mathcal{E}}^{q}$ where $\mathcal{E}$ is dominant. Moreover, let $\mathcal{L}_{\mathcal{E}}^{q}$ be the corresponding region of dominance, which is given by (5.11). We define the joint TDOA space of dominance associated to $\mathcal{E}$ as:

$$I_{\mathcal{E}} = \bigtimes_{q \in \mathcal{Q}} I_{\mathcal{E}}^{q} = \left\{ [\tau_{\mathcal{E}}^{1}, \ldots, \tau_{\mathcal{E}}^{Q}] \mid \forall k \in \{1, \ldots, Q\} \tau_{\mathcal{E}}^{k} \in I_{\mathcal{E}}^{k} \right\}. \tag{5.13}$$

Using the location-TDOA function given by (4.2), we can map this joint TDOA space, associated to the acoustic event $\mathcal{E}$, to the location space according to:

$$\begin{aligned} \mathcal{L}_{\mathcal{E}} &= \left\{ \mathbf{l} \in \mathcal{L} \mid \forall k \in \{1, \ldots, Q\} \tau^{k}(\mathbf{l}) \in I_{\mathcal{E}}^{k} \right\} && (5.14) \\ &= \bigcap_{q \in \mathcal{Q}} \mathcal{L}_{\mathcal{E}}^{q} \quad \text{(using 5.11)}. && (5.15) \end{aligned}$$

We can conclude from (5.14) and (5.15) that localizing an acoustic event in the 2-D or 3-D space can be solved by calculating the intersection of the different regions of dominance associated to this acoustic event, which are estimated from all microphone pairs $q \in \mathcal{Q}$. Furthermore, the extraction of each region of dominance $\mathcal{L}_{\mathcal{E}}^{q}, q \in Q$, is fully defined through the extraction of its corresponding intervals of dominance in the TDOA space. Thus, the proposed solutions below can be seen as bridge between the TDOA-based approaches (dual-step) and beamforming-based approaches (one-step).

We finally define the indicator function $\mathbb{1}_{\mathcal{E}}$, associated to an acoustic event $\mathcal{E}$, according to:

$$\mathbb{1}_{\mathcal{E}}(\mathbf{l}) = \mathbb{1}_{\mathcal{L}_{\mathcal{E}}}(\mathbf{l}) = \prod_{q \in \mathcal{Q}} \mathbb{1}_{\mathcal{L}_{\mathcal{E}}^{q}}(\mathbf{l}) = \prod_{q \in \mathcal{Q}} \mathbb{1}_{I_{\mathcal{E}}^{q}}(\tau^{q}(\mathbf{l})) \quad \text{with} \quad \mathbb{1}_{I_{\mathcal{E}}^{q}}(\tau^{q}(\mathbf{l})) = \begin{cases} 1 & \text{if } \tau^{q}(\mathbf{l}) \in I_{\mathcal{E}}^{q} \\ 0, & \text{otherwise.} \end{cases} \tag{5.16}$$

The next sections will show how these acoustic dominance-related notions are used in practice to solve the multiple speaker localization problem.

5.4 Probabilistic Steered Response Power (P-SRP)

We present in this section our first probabilistic model to solve the acoustic source localization problem. This framework is further developed to cover the multiple speaker scenario.

Contrary to conventional SRP where the GCC functions are blindly accumulated, we propose a more advanced probabilistic approach, which uses the GCC functions efficiently to estimate more accurate source locations and extract more information about the speakers while being faster compared to the conventional SRP. We can summarize this approach as follows:

1. For a given microphone pair, characterize each acoustic event (given by a peak in the GCC function) by a Gaussian distribution in the TDOA space (as explained in Chapter 4).

2. Combine the different Gaussian distributions representing the *same* acoustic event but in different microphone pairs, and then map them to the location space. This leads to a GM distribution (of the TDOA) characterizing the acoustic event location.

3. In order to decide whether this acoustic event represents an actual speaker, we simply use the conventional SRP principle, which states that the acoustic source location maximizes the SRP output. More precisely, this is done by 1) estimating the optimal location of the acoustic event through use of numerical optimization algorithms, and then 2) comparing the realization of the pdf at this location to a probabilistic confidence threshold.

In doing so, this approach incorporates the information introduced by each GCC function. Thus, each speaker is characterized by a pdf rather than a point-based estimate. Moreover, this method does not require any discrete search method to estimate the optimal location, and thereby, can be expected to not suffer from the poor accuracy of the estimates. The probabilistic aspect of this approach allows us to efficiently estimate the number of speakers by defining a threshold, which characterizes the uncertainty introduced by noise. Contrary to previous methods that define environment dependent thresholds [51, 56], this probabilistic threshold is less dependent on the environment. This approach uses the SRP idea but in a probabilistic manner, therefore, we will refer to it in the rest of this thesis as Probabilistic SRP (P-SRP). The rest of this section introduces the mathematical formulation of P-SRP.

5.4.1 P-SRP Distribution

In order to define the P-SRP distribution, we first estimate the TDOA GM distribution underlying the GCC function, for each microphone pair $q \in \mathcal{Q}$. This is done using either the WEM approach from Section 4.2.2 or its acoustic event dominance-based approximation introduced in Section 4.2.3. Now, let $\{\mathcal{M}^q\}_{q\in\mathcal{Q}}$ be the set of the resulting GMs, with

$$\mathcal{M}^q(\tau^q) = \sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}_k^q\big(\tau^q, \mu_k^q, (\sigma_k^q)^2\big). \tag{5.17}$$

The probabilistic approximation of the SRP can be obtained by simply replacing the GCCs in the SRP formulation, given in (5.9), by their probabilistic GM approximations given by (5.17). Furthermore, each TDOA random variable is replaced by its deterministic location-TDOA mapping given by (4.2). This yields to

$$\begin{aligned} SRP_{prob}\big(\mathbf{l}(r,\theta,\phi)\big) &\propto SRP_{PHAT}\big(\mathbf{l}(r,\theta,\phi)\big) \\ &\propto \sum_{q\in\mathcal{Q}} \mathcal{R}^q\big(\tau^q(\mathbf{l})\big) \\ &\propto \sum_{q\in\mathcal{Q}} \mathcal{M}^q\big(\tau^q(\mathbf{l})\big) \\ &= \frac{1}{Z}\sum_{q\in\mathcal{Q}}\sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}_k^q\Big(\tau^q(\mathbf{l}), \mu_k^q, (\sigma_k^q)^2\Big), \end{aligned} \tag{5.18}$$

where Z is a normalization term, which we approximate in the rest of this thesis by:

$$Z \approx \sum_{q\in\mathcal{Q}}\sum_{k=1}^{K^q} p_k^q. \tag{5.19}$$

We should mention here that due to the many-to-one mapping of the location-TDOA function given by (4.2), we can show that

$$\begin{aligned} 1 = \int_{\tau^q} \mathcal{N}_k^q\big(\tau^q, \mu_k^q, (\sigma_k^q)^2\big)\,\mathrm{d}\tau^q &\neq \int_{\mathbf{l}} \mathcal{N}_k^q\Big(\tau^q(\mathbf{l}), \mu_k^q, (\sigma_k^q)^2\Big)\,\mathrm{d}\mathbf{l} \\ &\neq \int_r\int_\theta\int_\phi \mathcal{N}_k^q\Big(\tau^q\big(\mathbf{l}(r,\theta,\phi)\big), \mu_k^q, (\sigma_k^q)^2\Big)\,\mathrm{d}r\mathrm{d}\theta\mathrm{d}\phi. \end{aligned}$$

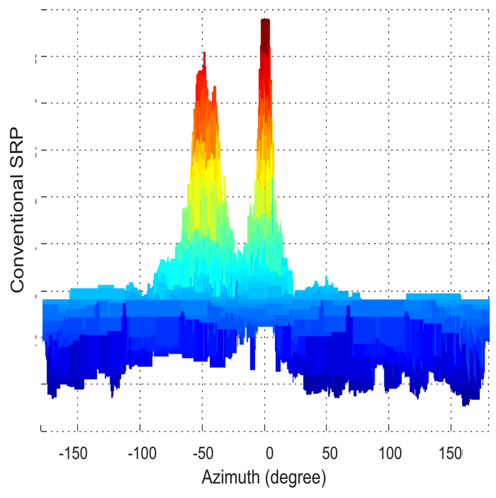
A Monte Carlo approximation of the integral on the right-hand side, however, has shown that it is very close from 1, which motivated the use of the approximation of Z given by (5.19). Therefore, the SRP-based pdf of multiple speakers, SRP_{prob}, is given by:

$$SRP_{prob}\big(\mathbf{l}(r,\theta,\phi)\big) \approx \sum_{q\in\mathcal{Q}}\sum_{k=1}^{K^q} \bar{p}_k^q \cdot \mathcal{N}_k^q\big(\tau^q(\mathbf{l}), \mu_k^q, (\sigma_k^q)^2\big), \quad \text{where} \quad \bar{p}_k^q = \frac{p_k^q}{\sum_{q\in\mathcal{Q}}\sum_{k=1}^{K^q} p_k^q}. \tag{5.20}$$

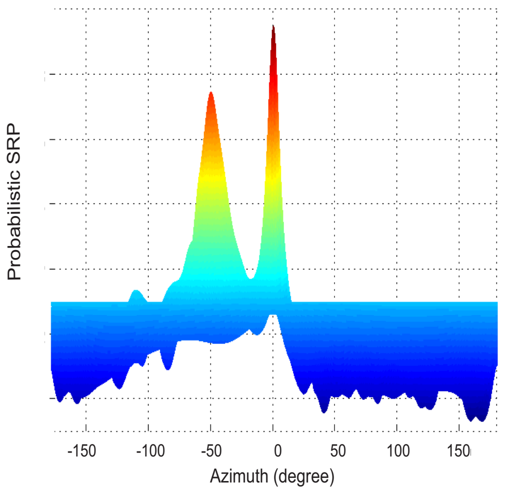
5.4.2 P-SRP-based Single Speaker Localization

Given SRP_{prob} and the notions of dominance introduced in Section 5.2 and Section 5.3, the acoustic source localization becomes relatively straightforward. Due to noise and reverberation, SRP_{prob} is more likely to be a *multi-modal* distribution, where each *mode* represents a potential acoustic event. The general idea here is to calculate the restriction of SRP_{prob} on the compact region of dominance, given by (5.14) and (5.15), associated to the acoustic event of interest. The restricted distribution, which is expected to be *unimodal*, is then used in combination with a numerical optimization algorithm to extract the optimal location.

Formally, let us assume that we can extract the region of dominance $\mathcal{L}_{\mathcal{E}}$, associated to an



(a) Conventional SRP



(b) Probabilistic SRP (P-SRP)

Figure 5.2: *A frontal view illustrating the probabilistic SRP in comparison to its conventional SRP counterpart. We can clearly see that P-SRP is a smoothed version of SRP-PHAT and that it does not suffer from secondary maxima problem.*

acoustic event $\mathcal{E}$, and its corresponding set of intervals of dominance $\{I_q^{\mathcal{E}}\}_{q\in\mathcal{Q}}$. The restriction $SRP_{prob}^{\mathcal{E}}$ of SRP_{prob} on the region of dominance $\mathcal{L}_{\mathcal{E}}$, associated to $\mathcal{E}$, is given by:

$$\begin{aligned} SRP_{prob}^{\mathcal{E}}(\mathbf{l}) &= \Big(\frac{1}{Z}\sum_{q\in\mathcal{Q}}\mathcal{M}^q(\tau^q(\mathbf{l}))\Big)\cdot \mathbb{1}_{\mathcal{E}}(\mathbf{l}) \\ &= \Big(\frac{1}{Z}\sum_{q\in\mathcal{Q}}\mathcal{M}^q(\tau^q(\mathbf{l}))\Big)\cdot \prod_{p\in\mathcal{Q}}\mathbb{1}_{\mathcal{L}_{\mathcal{E}}^p}(\mathbf{l}) \\ &= \frac{1}{Z}\sum_{q\in\mathcal{Q}}\Big(\mathcal{M}^q(\tau^q(\mathbf{l}))\cdot \prod_{p\in\mathcal{Q}}\mathbb{1}_{\mathcal{L}_{\mathcal{E}}^p}(\mathbf{l})\Big) \\ &\approx \frac{1}{Z}\sum_{q\in\mathcal{Q}}\Big(\mathcal{M}^q(\tau^q(\mathbf{l}))\cdot \mathbb{1}_{I_{\mathcal{E}}^q}(\tau^q(\mathbf{l}))\Big). \end{aligned} \tag{5.21}$$

Theoretically, the expression given by (5.21) is a relaxed approximation (up to a normalization constant) of the SRP-based pdf of the acoustic event location. In practice, however, the acoustic events are not known, therefore, estimating this pdf requires an additional step to detect each acoustic event $\mathcal{E}$ and extract its corresponding set of intervals of dominance $\{I_q^{\mathcal{E}}\}_{q\in\mathcal{Q}}$. As a simple solution, we propose here to accomplish this task by evaluating the SRP over a coarse grid with a resolution of 10° to 20° for azimuth and elevation. This grid will be subsequently called the *control grid*. Given that the probabilistic SRP uses the conventional SRP principle, we can conclude that both approaches can be used in this evaluation. The use of the control grid has the following motivations:

1. The grid locations with large SRP values carry the spatial information of where the potential

acoustic events are located, and therefore, it can be used as a reliable acoustic event detector, which provides good initial location estimates for the numerical optimization algorithms.

2. The control grid solves the problem of the high computation cost required during the fine discretization of the location space, without compromising the performance of the SRP. This is due to the partition of the location space into regions of dominance, where few locations per subspace are sufficient to ensure a good performance.

3. Each location in the control grid maps to a consistent vector of TDOA values, where each value lies in a single interval of dominance for a given microphone pair, i.e., the open intervals of dominance are mutually disjoint. Therefore, the extraction of the intervals of dominance representing the *same* acoustic event, but for different microphone pairs, is straightforward.

Moreover, the use of the control grid leads to a more practical approximation of $SRP^{\mathcal{E}}_{prob}$. To clarify that, let $\mathbf{l} = \mathbf{l}^{init}_{\mathcal{E}}$ be the initial location estimate of a potential acoustic event $\mathcal{E}$. $\mathbf{l}^{init}_{\mathcal{E}}$ is extracted as the control grid location which maximizes the SRP output. The first step then is to extract, for each microphone pair $q \in \mathcal{Q}$, the Gaussian distribution $\mathcal{N}^q_{k^q_{\mathcal{E}}}\left(\tau^q, \mu^q_{k^q_{\mathcal{E}}}, (\sigma^q_{k^q_{\mathcal{E}}})^2\right)$ and the corresponding intervals of dominance $\{I^q_{k^q_{\mathcal{E}}}\}_{q\in\mathcal{Q}}$, associated to the potential acoustic event $\mathcal{E}$, according to:

$$k^q_{\mathcal{E}} = \underset{k}{\operatorname{argmax}}\ \mathcal{N}^q_k\left(\tau^q(\mathbf{l}^{init}_{\mathcal{E}}), \mu^q_k, (\sigma^q_k)^2\right). \tag{5.22}$$

Furthermore, by neglecting the contribution of the rest of the components in the mixture within each interval of dominance $I^q_{k^q_{\mathcal{E}}}$, we obtain the following approximation:

$$\mathcal{M}^q\left(\tau^q(\mathbf{l})\right)\cdot \mathbb{1}_{I^q_{k^q_{\mathcal{E}}}}\left(\tau^q(\mathbf{l})\right) \approx \bar{p}^q_{k^q_{\mathcal{E}}} \cdot \mathcal{N}^q_k\left(\tau^q(\mathbf{l}), \mu^q_{k^q_{\mathcal{E}}}, (\sigma^q_{k^q_{\mathcal{E}}})^2\right). \tag{5.23}$$

Actually, this approximation is a direct result of **assumption (A)**, described in Section 4.2.1, which associates a single Gaussian distribution to each peak. Substituting (5.23) in (5.21) leads to a pdf approximating $SRP^{\mathcal{E}}_{prob}$ according to:

$$\widehat{SRP}^{\mathcal{E}}_{prob}(\mathbf{l}) = \frac{1}{Z_{\mathcal{E}}} \sum_{q\in\mathcal{Q}} \bar{p}^q_{k^q_{\mathcal{E}}} \cdot \mathcal{N}^q_k\left(\tau^q(\mathbf{l}), \mu^q_{k^q_{\mathcal{E}}}, (\sigma^q_{k^q_{\mathcal{E}}})^2\right). \tag{5.24}$$

$Z_{\mathcal{E}}$ is a normalization term resulting from the approximation. Similarly to Z, $Z_{\mathcal{E}}$ is approximated in the rest of this study by the sum of the mixture weights $\{\bar{p}^q_{k^q_{\mathcal{E}}}\}_{q\in\mathcal{Q}}$.

5.4.3 Estimation of the (Exact) Acoustic Event Location

$\widehat{SRP}^{\mathcal{E}}_{prob}$ is an infinitely differentiable function, i.e., composition of infinitely differentiable functions, therefore, the estimation of the optimal (exact) location of the potential acoustic event $\mathcal{E}$ can be performed using any nonlinear optimization algorithm on the pdf given by (5.24), and using $\mathbf{l}_{\mathcal{E}}^{init}$ for initialization. The corresponding literature (e.g., [96, 97, 98]) proposes a large number of methods which can be used for this purpose. The choice generally depends on the speed of the algorithm in addition to its precision. We have tried in the framework of this study two different iterative algorithms for nonlinear optimization:

1. *Gradient descent* which is a first-order optimization algorithm.
2. *Broyden-Fletcher-Goldfarb-Shanno* known as BFGS, which is a quasi-Newton algorithm.

Both methods achieve a good performance in few iterations (between 10 and 20 iterations).

Theoretically, although the different Gaussian components in the restricted distribution given by (5.24) represent the same acoustic event, $\widehat{SRP}_{prob}^{\mathcal{E}}$ is not necessarily a convex function. this is a direct result of the fact that the location-TDOA mapping is a many-to-one function. In practice, however, the above approximation has in most cases a single sharp peak within a large region of dominance with non-decreasing high SRP values. Therefore, any initial guess within this sub-space would lead to the optimal estimate. Actually, this is another reason behind the use of the control grid, which provides initial estimates lying within these sub-spaces. Furthermore, the “convexity” of $\widehat{SRP}_{prob}^{\mathcal{E}}$ increases with an increasing number of microphone pairs, where the peak becomes sharper and the rest of the function becomes flatter.

5.4.4 From Intervals of Dominance to Intervals of Confidence

In some particular scenarios, the error in the TDOA detection may be large enough to cause the components in the GM to map to distinct local maxima within the same region of dominance. This is generally the case when a single acoustic event has secondary local peaks in SRP_{prob}. This problem can be solved by remarking that using only *some* components from the GM, in such cases, can lead to a more accurate estimate comparable to the one obtained when using *all* components. Therefore, we decided to run the optimization on a sub-GM containing only a subset of consistent Gaussian components.

Formally, let $T \in [0,1]$ denote a probabilistic confidence threshold. The Gaussian component $k_{\mathcal{E}}^{q}$ belongs to the sub-GM if

$$\sqrt{2\pi}\cdot\sigma_{k_{\mathcal{E}}^{q}}^{q}\cdot\mathcal{N}_{k}^{q}\left(\tau^{q}(\mathbf{l}_{\mathcal{E}}^{init}),\mu_{k_{\mathcal{E}}^{q}}^{q},(\sigma_{k_{\mathcal{E}}^{q}}^{q})^{2}\right)\geq T. \tag{5.25}$$

Given T, we can define an interval of confidence $C_{k_{\mathcal{E}}^{q}}^{q}$, for each component $k_{\mathcal{E}}^{q}$ in the GM, according to:

$$C_{k_{\mathcal{E}}^{q}}^{q}=\left[-\sigma_{k_{\mathcal{E}}^{q}}^{q}\sqrt{2\ln\frac{1}{T}}+\mu_{k_{\mathcal{E}}^{q}}^{q}\ ,\ \sigma_{k_{\mathcal{E}}^{q}}^{q}\sqrt{2\ln\frac{1}{T}}+\mu_{k_{\mathcal{E}}^{q}}^{q}\right]. \tag{5.26}$$

In this case, the Gaussian component $k_{\mathcal{E}}^{q}$ belongs to the sub-GM if

$$\tau^{q}\left(\mathbf{l}_{\mathcal{E}}^{init}\right)\in C_{k_{\mathcal{E}}^{q}}^{q}. \tag{5.27}$$

The choice of the threshold T depends on the resolution of the control grid, the desirable precision and the number of microphones.

5.4.5 PSRP as Linearized Weighted ML Estimator

The maximization of $\widehat{SRP}^{\mathcal{E}}_{prob}$ can be seen, from a TDOA point-of-view, as the optimal linearized solution to a weighted ML problem, which is a generalization of the ML problem, given by (4.4), that takes into account the confidence of each TDOA estimate. To prove that, let us linearize $\widehat{SRP}^{\mathcal{E}}_{prob}$ in the proximity of the optimal location $\mathbf{l}^{opt}_{\mathcal{E}}$.

Assuming that $\tau^q(\mathbf{l}) \to \mu_{k^q_{\mathcal{E}}}$ when $\mathbf{l} \to \mathbf{l}^{opt}_{\mathcal{E}}$, and using the first-order approximation $\exp(x) \underset{0}{\approx} 1+x$, we can linearize the component $k^q_{\mathcal{E}}$ according to:

$$\mathcal{N}_{k^q_{\mathcal{E}}}\left(\tau^q(\mathbf{l}), \mu_{k^q_{\mathcal{E}}}, (\sigma_{k^q_{\mathcal{E}}})^2\right) \underset{\tau^q(\mathbf{l}) \to \mu_{k^q_{\mathcal{E}}}}{\approx} \frac{1}{\sqrt{2\pi}\sigma_{k^q_{\mathcal{E}}}} \left(1 - \frac{\left(\tau^q(\mathbf{l}) - \mu_{k^q_{\mathcal{E}}}\right)^2}{2(\sigma_{k^q_{\mathcal{E}}})^2}\right). \tag{5.28}$$

Substituting this approximation in $\widehat{SRP}^{\mathcal{E}}_{prob}$ leads to

$$\begin{aligned}
\widehat{SRP}^{\mathcal{E}}_{prob}(\mathbf{l}) &\approx \sum_{q\in\mathcal{Q}} \bar{p}^q_{k^q_{\mathcal{E}}} \cdot \mathcal{N}^q_k\left(\tau^q(\mathbf{l}), \mu^q_{k^q_{\mathcal{E}}}, (\sigma^q_{k^q_{\mathcal{E}}})^2\right) \quad (\bar{p}^q_{k^q_{\mathcal{E}}} \text{ are normalized using } Z_{\mathcal{E}}) \\
&\approx \sum_{q\in\mathcal{Q}} \hat{p}^q_{k^q_{\mathcal{E}}} - \frac{1}{2}\sum_{q\in\mathcal{Q}} \hat{p}^q_{k^q_{\mathcal{E}}} \cdot \frac{\left(\tau^q(\mathbf{l}) \;-\; \mu^q_{k^q_{\mathcal{E}}}\right)^2}{(\sigma^q_{k^q_{\mathcal{E}}})^2} \quad \text{with} \quad \hat{p}^q_{k^q_{\mathcal{E}}} = \frac{\bar{p}^q_{k^q_{\mathcal{E}}}}{\sqrt{2\pi}\sigma^q_{k^q_{\mathcal{E}}}} \\
&\approx \text{constant}_{/\mathbf{l}} \;-\; \frac{1}{2}\sum_{q\in\mathcal{Q}} \frac{\hat{p}^q_{k^q_{\mathcal{E}}}}{(\sigma^q_{k^q_{\mathcal{E}}})^2} \cdot \left[\tau^q(\mathbf{l}) - \mu^q_{k^q_{\mathcal{E}}}\right]^2.
\end{aligned} \tag{5.29}$$

We can clearly see that the maximization of $\widehat{SRP}^{\mathcal{E}}_{prob}$ is strictly equivalent to the minimization of the weighted ML estimator given by the right-hand side of (5.29). This weighted ML is simply a generalization of the ML estimator given by (4.4). In particular, optimizing $\widehat{SRP}^{\mathcal{E}}_{prob}$ instead of (4.4) has the advantages of being more reliable. This is a direct result of the estimation method, which extracts each error covariance $(\sigma^q_{k^q_{\mathcal{E}}})^2$ and weight $\hat{p}^q_{k^q_{\mathcal{E}}}$ directly from the GCC function itself. Moreover, on the contrary to dual-step approaches which approximate (4.4) and extract TDOA estimates from the main GCC peaks without consistency check, using the intervals of dominance/confidence is expected to extract consistent TDOA means $\{\mu^q_{k^q_{\mathcal{E}}}\}_{q\in\mathcal{Q}}$, i.e., map to the same region in the location space, leading to more accurate location estimates.

5.4.6 P-SRP-based Multiple Speaker Localization

We can observe that the approximation in (5.29) does not require a search grid to perform the localization as it can be done analytically. Furthermore, the proposed P-SRP-based approach can be easily extended to the multiple speaker case, by iteratively detecting and extracting new acoustic events. While the next section introduces a more advanced approach to decide whether the extracted acoustic events are actual speakers or noise sources, we use in this section a probabilistic threshold as stopping criterion, and we assume that all the extracted acoustic events are actual speakers. The pseudo-code detailed in Algorithm 2 presents an iterative approach to detect and localize multiple speakers.

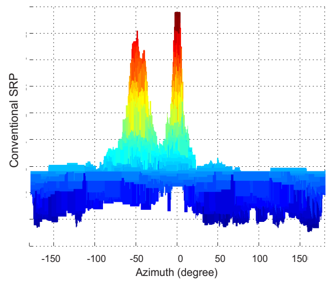
Algorithm 2 : P-SRP-based Multiple Speakers Localization

```
Let $N_{max}$ be the maximum number of speakers
for $i = 1 : N_{max}$ do
    1. Calculate the $SRP_{prob}$ for a coarse grid (10° to 20° resolution)
    2. Use the location with the maximum $SRP_{prob}$ output as initialization $\mathbf{l}^{init}$
    for all microphone pairs $q \in \mathcal{Q}$ do
        3. Determine the Gaussian components $\{k_s^q\}_{q\in\mathcal{Q}}$ which generated $\mathbf{l}^{init}$ (using 5.22)
    end for
    4. Define $SRP_{prob}^{s}(\mathbf{l})$ for the current potential speaker $s$ (using 5.24)
    5. Run numerical optimization on $SRP_{prob}^{s}(\mathbf{l})$ to estimate the optimal location $\mathbf{l}_s^{opt}$
    if $SRP_{prob}(\mathbf{l}_s^{opt}) > P_{noise}$ then
        6. Add $\mathbf{l}_s^{opt}$ to the set of speakers $\mathcal{S}$
        7. Discard the Gaussian components $\{k_s^q\}_{q\in\mathcal{Q}}$ of speaker $s$ from $SRP_{prob}$
        8. Re-normalize the remaining weights
        9. Go to step 1
    else
        10. BREAK
    end if
end for
11. Return the set of speakers $\mathcal{S}$
```

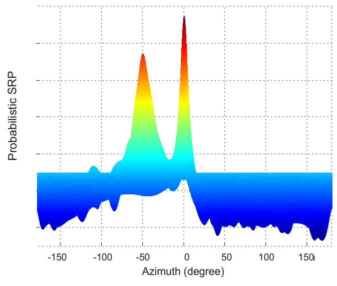
Note that the number of speakers may be overestimated in the case where the maximum number of concurrent speakers N_{max} is not known. This may increase the computation complexity, but it does not affect the localization performance. In order to ensure that we do not miss any speaker, the **BREAK** instruction in Step 10 of Algorithm 2 can be removed and the speaker/non-speaker classification approach, described in the next chapter, can be used instead after extracting N_{max} potential acoustic events.

It is also worth mentioning that the speaker locations from the previous time frame can be used as initialization for the current time frame. Thus, the coarse grid in Step 1 of Algorithm 2 will be required only if the maximum number of speakers is not reached, and if the current location has a probability SRP_{prob} higher than the confidence threshold (i.e., P_{noise}). However, as the energy of each speaker spans over a sector [51], a coarse grid (10° to 20°) can be used to initialize the process.

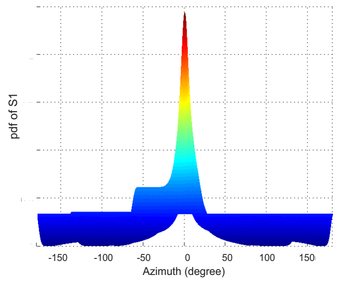
The probabilistic threshold P_{noise} characterizes the regions of confidence. In fact, assuming that the source location can be approximated by a Gaussian distribution, the region of confidence is represented then by an ellipse with equal likelihood. Therefore, defining a region of confidence is equivalent to defining a threshold P_{noise} over SRP_{prob}.



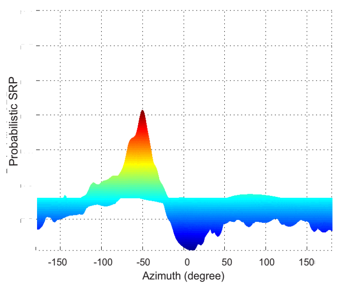
(a) Conventional SRP



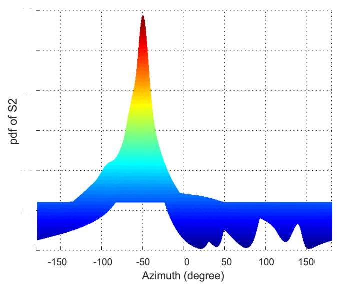
(b) Probabilistic SRP



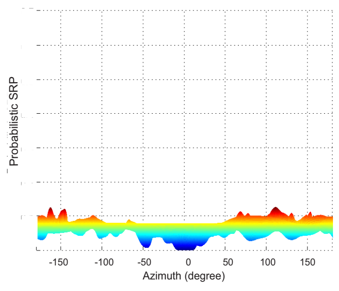
(c) Extraction of 1st speaker (S1)



(d) P-SRP after removing S1



(e) Extraction of 2nd speaker (S2)



(f) P-SRP after removing S2

Figure 5.3: *A frontal view illustrating P-SRP and the multiple speaker localization algorithm. The horizontal coordinates are azimuth [-180°,180°] and elevation [0°,90°].*

Figure 5.3 shows a localization example of two overlapping speakers using Algorithm 2. The confidence threshold is $P_{noise} = 0.3$. The proposed algorithm does not only localize the speakers, but it also provides an approximation of the pdf of each speaker. This approximation can be efficiently used to obtain more information about the sources, such as the uncertainty of the estimates given by the variance as well as the higher order moments. More precisely, this can be done using importance/rejection sampling techniques to approximate this pdf by a single Gaussian distribution as suggested by Figure. 5.3c and Figure. 5.3e.

5.4.7 Evaluation of P-SRP-based Speaker Localization

5.4.7.1 Experimental Setup

We evaluated the proposed approach using the AV16.3 corpus [71], where human speakers have been recorded in a smart meeting room (approximately 30m^2 in size) with a 20cm 8-channel UCA. The sampling rate is 16 kHz and the real mouth position is known with an error $\leq$ 5cm [71]. The AV16.3 corpus has a variety of scenarios, such as stationary or quickly moving speakers, varying number of simultaneous speakers, etc. Section 2.3.1 provides a detailed introduction to this corpus, including the different scenarios it offers.

In the experiments reported below, the signal was divided into frames of 512 samples (32ms); the GCCs were calculated using PHAT weighting [59], as explained in Section 4.1, where each GCC is interpolated by a factor of 10. Since there is no benefit in localizing acoustic sources in silence frames, we use an energy-based voice activity detector to discard these frames [99].

The localization task is performed in the entire 3-D space but, due to the far-field assumption in which the range is ignored, the results are limited to the Direction Of Arrival (DOA). More precisely, the results are reported in terms of the detection rate d_r and the standard deviations of the azimuth $\sigma_{s,\theta}$, and elevation $\sigma_{s,\phi}$. These measures are obtained by fitting a 2-component Gaussian mixture to the estimates error. Section 2.3.2 provides more details about the evaluation approach and the metrics that are used to evaluate the localization performance. We also report the real-time factor t on a standard Pentium(R) Dual-Core CPU clocked at 2.50GHz.

5.4.7.2 Single Speaker Results

Although the P-SRP approach was mainly introduced to solve the multiple speaker localization problem, we ran a set of experiments on the single speaker sequences from the AV16.3 corpus. This allowed us to compare its performance to that of the classical solutions adopted in this case. In particular, the comparison includes the conventional SRP and the MCCC approaches. The conventional SRP uses a 3-D grid with an angular and radial resolutions of 0.5° and 10cm, respectively, whereas MCCC uses only a 2-D grid with the same angular resolution. This choice was mainly imposed by the significant processing time needed by MCCC. Moreover, the 3-D coarse grid used by P-SRP has an angular and radial resolutions of 7° and 30 cm, respectively.

Approaches	seq01-1p-0000				seq02-1p-0100				seq03-1p-0100			
	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t
MCCC	52.5	1.86	8.16	125	79.1	1.71	10.9	117	69.0	1.59	7.61	108
Class-SRP	52.6	**1.79**	7.55	30.4	80.1	1.63	**7.77**	30.4	70.5	1.32	**5.50**	30.1
P-SRP-WEM	**53.7**	2.15	7.63	63.7	**81.5**	1.97	8.62	60.3	**71.8**	1.72	6.82	54.8
P-SRP-ADA	52.6	1.83	**7.50**	**1.49**	79.1	**1.51**	8.15	**1.50**	70.3	**1.31**	5.79	**1.51**

(a) Single speaker localization performance on audio sequences with a *static* speaker.

Approaches	seq11-1p-0100				seq15-1p-0100			
	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t
MCCC	**76.7**	2.65	11.5	136	74.8	1.73	7.81	108
Class-SRP	76.1	**2.19**	**7.72**	26.6	**79.2**	**1.60**	**5.27**	30.4
P-SRP-WEM	76.2	2.44	8.38	64.3	78.2	1.73	7.49	61.5
P-SRP-ADA	76.0	2.38	8.25	**1.52**	77.7	1.68	6.15	**1.49**

(b) Single speaker localization performance on audio sequences with a *moving* speaker.

Table 5.1: P-SRP-based single speaker localization performance on the AV16.3 corpus.

Table 5.1 reports the single speaker localization results. We can conclude that, overall, the different models lead to a comparable performance with no particular winner regarding the precision and accuracy. We can also conclude that SRP-based approaches lead to more accurate estimates compared to the MCCC approach, in particular, concerning the estimation of the elevation. Regarding the P-SRP approaches, Table 5.1 shows that the acoustic dominance-based approximation, P-SRP$_{ADA}$, leads to a significant speed-up of the localization time with a negligible degradation of the achieved accuracy of correct estimates when compared to the WEM-based P-SRP, i.e., P-SRP$_{WEM}$, with an improved azimuth and elevation precision. In fact, these results actually confirm the assumption that the peaks of the GCC function are actually generated by the acoustic events in the room. The acoustic dominance-based approximation uses this assumption to define the means of the GM components based on the GCC peaks, whereas the WEM algorithm estimates the means based on the whole GCC distribution. When comparing the real-time factor t, we can see that P-SRP$_{ADA}$ achieves a near real-time performance while being, approximately, 25 and 76 times faster compared to the classical SRP and MCCC, respectively. In addition to this significant speed-up, P-SRP is very easy to extend to the multiple speaker case, which is much more difficult for the classical approaches. This is evaluated and analyzed in the next section.

5.4.7.3 Multiple Speaker Results

In the multiple speaker scenario, the threshold P_{noise} is set to 0.3 and the maximum number of speakers N_{max} is given by the maximum number of speakers in each audio sequence.

Table 5.2 shows the multiple speaker localization results on the multi-party recordings from the AV16.3 corpus. These results show that, overall, P-SRP extracts the correct estimates. In particular, P-SRP$_{WEM}$ leads to an average precision rate of $\approx 81\%$, whereas P-SRP$_{ADA}$ results in an average precision rate of $\approx 72\%$. That is, P-SRP$_{WEM}$ is more restrictive compared P-SRP$_{ADA}$. Thus, the rejection and False Alarm Rate (FAR) are lower for the latter approach. This in fact affects the detection rate of the different speakers, which is inversely correlated to the precision rate. In particular, P-SRP$_{ADA}$ achieves a higher detection rate but with a lower precision rate.

A further analysis of these results showed that the FAR of the different experiments varies between 8% to 30%. This significant variation is mainly caused by the use of a fixed confidence threshold P_{noise} for all experiments. The AV16.3 corpus design, however, focused on covering as many scenarios and conditions as possible that can occur in spontaneous speech, including long silences, varying distance to the microphone, different number of speakers, etc. Therefore, an optimal speaker localization approach should be able to adapt itself to new conditions, which is not the case for a static confidence threshold P_{noise}. Chapter 6 will introduce a new approach to solve this problem.

It is also worth mentioning here that using different P_{noise} for P-SRP$_{WEM}$ and P-SRP$_{ADA}$, such that they achieve the same FAR, shows that P-SRP$_{WEM}$ achieves a slightly better performance compared to P-SRP$_{ADA}$.

Approaches	seq18-2p-0101		seq24-2p-0111	
	P-SRP$_{WEM}$	P-SRP$_{ADA}$	P-SRP$_{WEM}$	P-SRP$_{ADA}$
p_s	**90.1**	82.1	**92.5**	78.0
$\sigma_{s,\theta}$	2.13	**1.74**	2.69	**2.05**
$\sigma_{s,\phi}$	**8.51**	9.67	7.55	**6.98**
Detection rate				
of Speaker 1	61.1	**63.1**	57.7	**60.1**
of Speaker 2	47.6	**56.6**	38.0	**39.1**

(a) Multiple speaker localization performance on audio sequences with *two* simultaneous speakers.

Approaches	seq40-3p-0111		seq45-3p-1111		seq37-3p-0001	
	P-SRP$_{WEM}$	P-SRP$_{ADA}$	P-SRP$_{WEM}$	P-SRP$_{ADA}$	P-SRP$_{WEM}$	P-SRP$_{ADA}$
p_s	79.1	**90.6**	**61.1**	52.5	**80.0**	56.6
$\sigma_{s,\theta}$	**2.87**	6.72	2.18	**2.05**	2.54	**1.31**
$\sigma_{s,\phi}$	**7.95**	11.1	**7.28**	8.09	7.39	**7.33**
Detection rate						
of Speaker 1	32.0	**49.4**	20.6	**52.5**	32.8	**36.2**
of Speaker 2	34.6	**48.7**	43.6	**49.5**	**64.3**	60.2
of Speaker 3	52.5	**70.5**	45.2	**52.4**	**42.6**	37.7

(b) Multiple speaker localization performance on audio sequences with *three* simultaneous speakers.

Table 5.2: P-SRP-based multiple speaker localization performance on the AV16.3 corpus.

5.5 Cumulative Steered Response Power (C-SRP)

Although the probabilistic SRP approach is reliable and faster compared to conventional SRP, it may be difficult to run it in real-time on low computation devices, e.g., small robots. Especially, if the WEM approach is used to estimate the GM approximating each TDOA pdf. The related literature proposes two different classes of approaches to overcome the run-time problem when using the conventional SRP: 1) *space reduction* strategies which focus the search effort on few regions of interest. For instance, Dmochowski et *al.* [47] proposed to overcome this issue by reducing the search space through inverse mapping of the TDOA, whereas Do et *al.* [48, 49] used an iterative reduction search strategy to estimate the optimal source location. The second class of approaches is 2) *spatial averaging* techniques, which evaluate SRP on compact volumes instead of point-based evaluation. This idea was investigated in [50] using a sector-based approach, whereas the method proposed in [52] is based on mapping compact volumes in the location space to closed intervals in the TDOA space.

Motivated by the work done in [47, 50, 52], we propose a novel framework which combines the advantages of search space reduction strategies [47, 48] and spatial averaging techniques [50]. This approach improves the detection rate of P-SRP and can be much faster. Thus, it can be used as a reliable alternative in low-resource applications. We can summarize this approach in these steps:

1. Use the acoustic event dominance-based GM approximation, described in Section 4.2.3, to extract the TDOA space partitions given by (4.20).
2. Use the resulting partitions to reduce the location search space to few regions, which are likely to contain an acoustic source.
3. Extract the speaker sub-space as the region which maximizes a Cumulative Steered Response Power (C-SRP).
4. Perform the classical SRP search in the speaker sub-space.

In doing so, this approach drastically decreases the computation cost by reducing the search space. Moreover, it improves the multiple speaker localization performance through use of the cumulative SRP. The extension of C-SRP to multiple speakers is straightforward.

5.5.1 Dominance-based Space Reduction

As we have seen in Section 5.3, the region of dominance associated to an acoustic event $\mathcal{E}$ is expressed as the intersection of sub-spaces of dominance, which are the direct mapping of the intervals of dominance into the location space. To recapitulate, the region of dominance $\mathcal{L}_{\mathcal{E}}$, associated to an acoustic event $\mathcal{E}$, is given by:

$$\mathcal{L}_{\mathcal{E}} = \bigcap_{q \in \mathcal{Q}} \mathcal{L}_{\mathcal{E}}^{q} \quad \text{with} \quad \mathcal{L}_{\mathcal{E}}^{q} = \{\mathbf{l} \in \mathcal{L} \mid \tau^{q}(\mathbf{l}) \in I_{\mathcal{E}}^{q}\}. \tag{5.30}$$

Given (5.30), we can conclude that the acoustic source localization problem can be cast as the problem of extracting all regions in the location space that are expressed as intersections of $\{\mathcal{L}_k^q\}_k$, resulting from different microphone pairs $q \in \mathcal{Q}$. Theoretically, the number of **all** possible intersections is large and equal to $\prod_{q=1}^{Q} K^q$. In practice, however, most of these intersections are empty. This is due to the physical constraints introduced by the microphone pairs. More precisely, let $\mathcal{P} \subset \mathcal{Q}$ be a subset of microphone pairs, and let $\mathcal{L}_{\mathcal{E}}^{\mathcal{P}}$ be the sub-intersection of their regions of dominance associated to $\mathcal{E}$. By definition of intersection, the volume of $\mathcal{L}_{\mathcal{E}}^{\mathcal{P}}$ continuously decreases when the number of microphone pairs in $\mathcal{P}$ increases. For all true sources, we can expect that when the number of microphone pairs in $\mathcal{P}$ is sufficiently large, we get

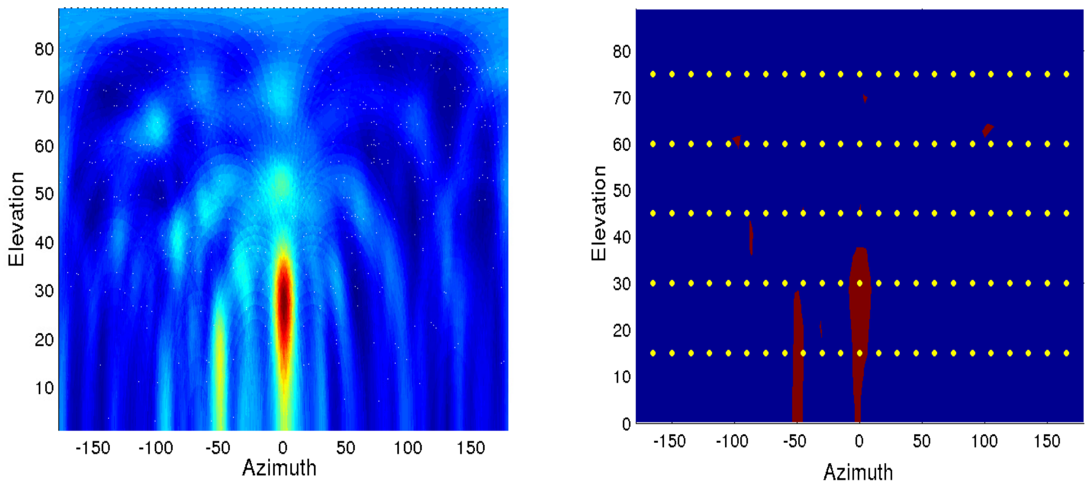
$$\forall q \in \mathcal{Q}/\mathcal{P},\ \exists k^q \in \{1,\cdots,K^q\} \begin{cases} 1)\ \mathcal{L}_{\mathcal{E}}^{\mathcal{P}} \subset \mathcal{L}_{k^q}^q, \quad and \\ 2)\ \forall k \in \{1,\cdots,k^q-1,k^q+1,\cdots,K^q\} : \mathcal{L}_{\mathcal{E}}^{\mathcal{P}} \cap \mathcal{L}_k^q = \emptyset. \end{cases} \tag{5.31}$$

In other terms, when this intersection becomes small enough, we can expect that for each new space partition, given by a microphone pair that is not in $\mathcal{P}$, the intersection $\mathcal{L}_{\mathcal{E}}^{\mathcal{P}}$ maps into a single sub-space of dominance in this partition, and therefore, its intersections with the remaining sub-spaces is mostly empty. This observation drastically decreases the number of intersections that need to be extracted. The experiments conducted in this thesis have shown that such property occurs when the number of microphone pairs in the intersection is ≥ 4.

The extraction of all intersections is analytically intractable. Hence, we propose an alternative iterative solution, introduced in Algorithm 3. This is done using (5.30), which shows that each region of dominance is defined by the set of intervals of dominance which map to it. Therefore, the space reduction, i.e., extraction of few sub-spaces of dominance, is reduced to finding all physically possible combinations of the intervals of dominance. Formally, this can be done using the *coarse grid* (15° to 30° or 50 to 100 cm). The grid resolution is chosen such that at least one location falls into each $\mathcal{L}_{\mathcal{E}}^q$. Then, for each location $\mathbf{l}^{init}$ in this grid (dots in Figure 5.4b), the associated intervals of dominance $I_{\mathbf{l}^{init}}^q$ are extracted such that $\tau^q(\mathbf{l}^{init}) \in I_{\mathbf{l}^{init}}^q$.

Algorithm 3 : Extraction of the Sub-spaces of Dominance

```
Let $\mathcal{G}$ be the coarse grid (on the 3-D or 2-D space)
Let $\mathcal{D}_{\mathcal{L}}$ be the set of sub-spaces of dominance
$\forall q \in \mathcal{Q}$ calculate the TDOA partition $\{I_k^q\}_{k=1}^{K^q}$
for each $\mathbf{l}^{init} \in \mathcal{G}$ do
    $\forall q \in \mathcal{Q}$ find $k_{\mathbf{l}^{init}}^q$ such that $\tau^q(\mathbf{l}^{init}) \in I_{k_{\mathbf{l}^{init}}^q}^q$
    if $\mathcal{L}_{\mathbf{l}^{init}} = \bigcap_{q \in \mathcal{Q}} \mathcal{L}_{k_{\mathbf{l}^{init}}}^q \notin \mathcal{D}_{\mathcal{L}}$ then
        Add $\mathcal{L}_{\mathbf{l}^{init}}$ to $\mathcal{D}_{\mathcal{L}}$
    end if
end for
```



(a) Conventional SRP (top view) (b) Search space reduction (top view)

Figure 5.4: *Figure (a) exemplifies the SRP approach for a frame with two speakers, whereas Figure (b) presents the sub-spaces of dominance resulting from mapping all TDOA space partitions into the location space.*

5.5.2 C-SRP Formulation

The space reduction approach is based on extracting those sub-spaces where each acoustic event is dominant. Hence, we can assume that the contribution of the other sources is negligible in each sub-space. As a consequence, all the signal power returned from a given region is assumed to be generated by the acoustic source which is dominant in that region. Therefore, similarly to P-SRP, we define $SRP^{\mathcal{E}}$ associated to $\mathcal{E}$ as the restriction of SRP on the sub-space of dominance $\mathcal{L}_{\mathcal{E}}$. That is,

$$SRP^{\mathcal{E}}(\mathbf{l}) = SRP(\mathbf{l}) \cdot \mathbb{1}_{\mathcal{L}_{\mathcal{E}}}(\mathbf{l}), \tag{5.32}$$

where $\mathbb{1}_{\mathcal{L}_{\mathcal{E}}}(\mathbf{l})$ is the indicator function given by (5.16). Given the definition of SRP in (5.9), we can further simplify (5.32) to

$$SRP^{\mathcal{E}}(\mathbf{l}) \quad \propto \quad \sum_{q\in\mathcal{Q}} \mathcal{R}^q(\tau^q(\mathbf{l})) \cdot \prod_{q\in\mathcal{Q}} \mathbb{1}_{I^q_{\mathcal{E}}}(\mathbf{l}). \tag{5.33}$$

Now, we define the Cumulative SRP (C-SRP) of the source $\mathcal{E}$, denoted by $SRP_c(\mathcal{E})$, as the sum of the steered power returned from all locations $\mathbf{l}$ in the region of dominance $\mathcal{L}_{\mathcal{E}}$. More precisely, $SRP_c(\mathcal{E})$ is calculated according to:

$$\begin{aligned} SRP_c(\mathcal{E}) &= \int_{\mathcal{L}} SRP^{\mathcal{E}}(\mathbf{l})\,\mathrm{d}\mathbf{l} \\ &= \int_{\mathcal{L}_{\mathcal{E}}} SRP(\mathbf{l})\,\mathrm{d}\mathbf{l} \\ &\approx \int_{\mathcal{L}_{\mathcal{E}}} \left(\sum_{q \in \mathcal{Q}} \mathcal{R}^q(\tau^q(\mathbf{l})) \right) \mathrm{d}\mathbf{l} \\ &\approx \sum_{q \in \mathcal{Q}} \int_{I_{\mathcal{E}}^q} \mathcal{R}^q(\tau^q)\,\mathrm{d}\tau^q \\ &\approx \sum_{q \in \mathcal{Q}} \sum_{\tau^q \in I_{\mathcal{E}}^q} \mathcal{R}^q(\tau^q). \end{aligned} \tag{5.34}$$

The region of dominance $\mathcal{L}_{\mathcal{E}}$ is extracted as the one with the highest cumulative SRP. Then, the optimal location estimate $\mathbf{l}_{\mathcal{E}}^{opt}$ is obtained using the classical approach in the reduced space $\mathcal{L}_{\mathcal{E}}$. This is done by maximizing the SRP output on a sub-grid of locations, centered on the initial location $\mathbf{l}^{init} \in \mathcal{L}_{\mathcal{E}}$, which is given by the coarse grid in Algorithm 3. In order to further speed-up this search step, all the sub-grids are calculated and stored offline.

5.5.3 C-SRP-based Multiple Speaker Localization Algorithm

The proposed acoustic source localization approach can be easily extended to the multiple speaker case. Algorithm 4 presents one possible extension using an iterative approach. The algorithm is iterative in order to overcome the many-to-one aspect of the location-TDOA mapping, which causes each interval $\mathcal{I}_k^q$ to map to more than one sub-space of dominance in Algorithm 3. This idea is implemented by successively zeroing the restriction of the GCC function on $\mathcal{I}_{\mathbf{l}_n^{opt}}^q$ in step 6. The sub-grid used in the second search step (step 4) is calculated offline by associating each location $\mathbf{l}^{init}$ in the coarse grid $\mathcal{G}$ to a small grid centered on $\mathbf{l}^{init}$. In the case where N_{max} is unknown, it can be simply overestimated.

5.5.4 CSRP-based Noise/Speaker Classification

The proposed method extracts the source location as the one with the highest cumulative SRP, but it does not consider whether this location has been generated by an actual source or by secondary peaks. This problem becomes more difficult in the multiple speaker scenario, where the secondary peaks, resulting from the many-to-one mapping of the location-TDOA relationship, become comparable to the low-energy speakers. We propose to separate noise estimates from actual speakers using an unsupervised classifier. The proposed approach uses the cumulative SRP values $\mathcal{C}_n^{opt}$, $n = 1, \ldots, N_e$ ($N_e = N_{max} \times$ *number of frames*), as a classification feature. Then, a two-component Gaussian mixture fit is calculated using the EM algorithm. More precisely,

Algorithm 4 : C-SRP-based Multiple Speaker Localization

```
Let $N_{max}$ be the maximum number of speakers
Extract the sub-spaces of dominance $\mathcal{D}_{\mathcal{L}}$ (Algorithm 3)
**for** $n = 1 : N_{max}$ **do**
  1. $\forall \mathcal{L} \in \mathcal{D}_{\mathcal{L}}$ : calculate $\mathcal{C}(\mathcal{L}) = SRP_c(\mathcal{L})$
  2. Find $\mathcal{L}_n^{max} = \underset{\mathcal{L}}{\text{argmax}}\ \mathcal{C}(\mathcal{L})$
  3. Define $\mathcal{C}_n^{opt} = \mathcal{C}(\mathcal{L}_n^{max})$
  4. Find $\mathbf{l}_n^{opt} = \underset{\mathbf{l}}{\text{argmax}} SRP_{\mathcal{L}_n^{max}}(\mathbf{s})$ on the sub-grid
  5. Add $\left(\mathbf{l}_n^{opt}, \mathcal{C}_n^{opt}\right)$ to the set of *potential* speakers
  6. Set the restriction of $\mathcal{R}^q$ on $I^q_{\mathbf{l}_n^{opt}}$ to 0
**end for**
```

the Gaussian mixture distribution is given by:

$$f(\mathcal{C}) \propto w_n \cdot f_n(\mathcal{C} \mid \text{noise}) + w_s \cdot f_s(\mathcal{C} \mid \text{speaker}), \tag{5.35}$$

where $f_n(.)$ and $f_s(.)$ represent the likelihood distributions of the noise and speaker estimates, respectively, whereas w_n and w_s denote the corresponding priors. The posterior probability of the noise/speaker class, given a location estimate $\mathbf{l}$ with a cumulative SRP $\mathcal{C}$, is calculated according to:

$$p(\text{speaker} \mid \mathbf{l}) = \frac{w_s \cdot f_s(\mathcal{C} \mid \text{speaker})}{w_n \cdot f_n(\mathcal{C} \mid \text{noise}) + w_s \cdot f_s(\mathcal{C} \mid \text{speaker})}, \tag{5.36}$$

$$p(\text{noise} \mid \mathbf{l}) = 1 - p(\text{speaker} \mid \mathbf{l}). \tag{5.37}$$

The location estimate $\mathbf{l}$ is considered to be an actual source if $p(\text{speaker} \mid \mathbf{l}) > p(\text{noise} \mid \mathbf{l})$. In fact, this approach is the first building block of the multiple speaker detection approach that we introduce in Chapter 6.

5.5.5 Evaluation of C-SRP-based Speaker Localization

We evaluated the proposed approach on the AV16.3 corpus using the same experimental setup used in the P-SRP evaluation and described in Section 5.4.7. Similarly to P-SRP, the localization task is performed in the entire 3-D space but, due to the far-field assumption in which the range is ignored, the results are limited to the direction of arrival. The results are reported in terms of the detection rate d_r and the standard deviations of the azimuth $\sigma_{s,\theta}$, and elevation $\sigma_{s,\phi}$.

Table 5.3 presents the performance of the C-SRP approach on single source sequences, and compares it to the classical SRP [46] and MCCC [42] approaches. Note that in these experiments the detection approach from Section 5.5.4 was not used and N_{max} was set to 1. The coarse grid resolution used in P-SRP and C-SRP is $20° \times 20° \times 30$cm for the azimuth, elevation and range, respectively, whereas the resolution of SRP, MCCC and the second-step search grid for C-SRP is

$0.5^\circ \times 0.5^\circ \times 10$cm. The reduced search grid, which is used in the second search step, has a size of $30^\circ \times 40^\circ \times 4$m. The choice to use a coarse grid with an angular resolution of 20° instead of 7°, which was used in the P-SRP experiments, is motivated by two reasons: 1) show the impact of the coarse grid resolution on the P-SRP performance, and 2) highlight the importance of using the cumulative SRP to detect active speakers instead of the simple evaluation on the coarse grid, specially when the angular resolution of the latter is high. This comparison is carried, however, only for the multiple speaker case.

Approaches	seq01-1p-0000				seq02-1p-0100				seq03-1p-0100			
	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t
MCCC	52.5	1.86	8.16	125	79.1	1.71	10.9	117	69.0	1.59	7.61	108
SRP	**52.6**	**1.79**	7.55	30.4	**80.1**	**1.63**	7.77	30.4	**70.5**	**1.32**	5.50	30.1
C-SRP	49.3	**1.79**	**7.44**	**1.10**	77.0	1.73	**7.44**	**1.03**	68.9	1.42	**5.42**	**1.05**

(a) Single speaker localization performance on audio sequences with a static speaker.

Approaches	seq11-1p-0100				seq15-1p-0100			
	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t
MCCC	**76.7**	2.65	11.5	136	74.8	1.73	7.81	108
SRP	76.1	**2.19**	**7.72**	26.6	**79.2**	**1.60**	**5.27**	30.4
C-SRP	71.6	2.23	7.80	**1.03**	72.6	1.71	7.63	**1.04**

(b) Single speaker localization performance on audio sequences with a moving speaker.

Table 5.3: C-SRP-based localization performance on audio recordings, from the AV16.3 corpus, with a single speaker.

The results reported in Table 5.3 show that the performance of the C-SRP approach is comparable to the other methods. More precisely, the standard deviations of the azimuth $\sigma_{s,\theta}$ and the elevation $\sigma_{s,\phi}$ are comparable, with a slight degradation of the C-SRP precision rate p_s. This negligible degradation is mainly due to the large coarse grid ($20^\circ \times 20^\circ \times 30$cm) used in the first search stage needed by the C-SRP algorithm. Moreover, we can see that C-SRP is approximately 48 and 100 times faster than SRP and MCCC, respectively, with an almost-real time performance on a standard machine. That is with a negligible degradation of the performance. These results illustrates the efficiency of the C-SRP approach. The MCCC approach, however, is very slow due to the calculation of the correlation matrix determinant for all locations at each time frame. We should also mention here that reducing the resolution of the coarse grid leads to a better performance while making the algorithm slightly slower. This is in fact a trade-off that can be tuned based on the available resources and expected precision.

The merits of applying C-SRP to solve the multiple speaker localization task are shown in Table 5.4, which presents results for sequences with a varying number of simultaneous speakers (between zero and three). In these experiments, $N_{max} = 4$, whereas the detection threshold of the P-SRP is chosen such that the resulting FAR is equal to that of the C-SRP approach, which uses an unsupervised detection approach. Note that P-SRP uses the same coarse grid used by C-SRP. As mentioned above, the motivation behind this choice is to highlight the impact of the additional C-SRP step, which extracts the regions with high likelihood before proceeding to multiple speaker localization.

Approaches	seq18-2p-0101		seq24-2p-0111	
	C-SRP	P-SRP	C-SRP	P-SRP
p_s	**85.7**	82.8	**84.2**	82.9
$\sigma_{s,\theta}$	**1.78**	2.22	**2.53**	3.85
$\sigma_{s,\phi}$	**4.51**	8.93	**6.79**	9.45
Detection rate (d_r)				
of Speaker 1	**54.2**	51.7	**62.0**	55.6
of Speaker 2	45.8	**45.9**	**37.3**	34.9

(a) Multiple speaker localization performance on audio sequences with *two* simultaneous speakers.

Approaches	seq40-3p-0111		seq45-3p-1111		seq37-3p-0001	
	C-SRP	P-SRP	C-SRP	P-SRP	C-SRP	P-SRP
p_s	74.2	**77.5**	**52.6**	50.1	77.1	**90.1**
$\sigma_{s,\theta}$	2.67	**1.95**	**1.92**	2.13	**2.44**	7.65
$\sigma_{s,\phi}$	8.91	**6.40**	**7.74**	8.48	**8.25**	11.4
Detection rate (d_r)						
of Speaker 1	**27.3**	22.2	**17.4**	15.2	31.2	**51.4**
of Speaker 2	**32.2**	23.2	**45.3**	39.0	59.6	**64.6**
of Speaker 3	47.4	**55.5**	41.3	**44.0**	40.3	**42.1**

(b) Multiple speaker localization performance on audio sequences with *three* simultaneous speakers.

Table 5.4: C-SRP-based multiple speaker localization performance on the AV16.3 corpus.

Table 5.4 shows clearly that C-SRP outperforms the P-SRP approach. This improvement appears clearly in the increased percentage of correct estimates p_s and the average detection rate d_r of each speaker. This improvement is due to the additional C-SRP step, which locates the most likely regions to contain the speakers. This observation is true except for the audio sequence *seq37-3p-0001*, where P-SRP performs better than C-SRP. The main reason behind this exception is the fact that this sequence is very long, i.e., approximately 9 min, and includes many conditions, such as changing number of simultaneous speakers, moving speakers, changing distance between speakers and microphones, etc. As a result, the two-component Gaussian mixture model used by the detection approach does no converge properly. Thus, the noise and speaker classes are not optimally trained for this sequence. This observation highlights the importance of a detection model that is able to adapt to its environment. It is also worth mentioning that the proposed unsupervised classification approach leads to a FAR

$\approx$ 10% for all experiments, whereas the threshold-based detection model used in the P-SRP approach leads to different FARs when the threshold is fixed. This result makes the proposed unsupervised classification technique more attractive. This is actually one of the main motivations to extend it to a full multiple speaker detection approach in Chapter 6.

5.6 Acoustic Event Joint Space

Although the probabilistic interpretation and approximation of SRP, i.e., SRP_{prob}, leads to a good multiple speaker localization performance, SRP_{prob} does not address the problem of local maxima resulting from the many-to-one location-TDOA mapping, which can strongly affect multiple speaker detection, especially when the number of microphones is low. In fact, most SRP-based speaker localization approaches suffer from this problem.

We introduce in this section a purely probabilistic framework to overcome the local maxima problem. This is done by estimating the join probability distribution over the joint space of the acoustic events. The resulting pdf is expected to be multi-modal, where each *mode* corresponds to an acoustic event it the room. We also show in this section how multiple speaker localization can be achieved using the joint pdf. The proposed solution is based on the same interpretation of GCC as a pdf of the TDOA random variable and its GM approximations, introduced in Chapter 4. We can summarize this approach as follows:

1. Estimate the GM distribution approximating the TDOA pdf.
2. Use the resulting GMMs to construct a joint pdf over the entire (joint) multi-dimensional TDOA space.
3. Map the TDOA joint distribution to the location space.
4. Use the spatial averaging idea used in C-SRP to extract space regions with high likelihood.
5. Use a numerical optimization, similarly to P-SRP, to estimate the optimal location.

In doing so, this new approach preserves the computational efficiency and accuracy of P-SRP, while additionally solving the problem of multiple local maxima. Similarly to P-SRP and C-SRP, the extension of this approach to the multiple speaker case is straightforward.

5.6.1 Acoustic Event Joint Distribution

The idea here is to approximate the pdf over the joint TDOA space before mapping it to the location space. To do so, we first estimate the TDOA GM distribution underlying the GCC function, for each microphone pair $q \in \mathcal{Q}$. This is done using either the WEM approach from Section 4.2.2 or its acoustic event dominance-based approximation introduced in Section 4.2.3.

Now, let $\{\mathcal{M}^q\}_{q\in\mathcal{Q}}$ be the set of the resulting GMs, where

$$\mathcal{M}^q(\tau^q) \;=\; \sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}_k^q\big(\tau^q, \mu_k^q, (\sigma_k^q)^2\big). \tag{5.38}$$

Assuming that all TDOAs $\{\tau^q\}_{q\in\mathcal{Q}}$ are independent random variables, we can then define the joint pdf of the TDOAs, $p(\tau^1,\ldots,\tau^Q)$, according to:

$$p(\tau^1,\ldots,\tau^Q) = \prod_{q\in\mathcal{Q}} p(\tau^q) = \prod_{q\in\mathcal{Q}} \sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}^q\big(\tau^q, \mu_k^q, (\sigma_k^q)^2\big). \tag{5.39}$$

The independence assumption of the TDOA random variables, observed at the microphone pairs, is actually not valid due to the physical dependencies of the microphones in the array. Formulating these dependencies, however, is very difficult and can lead to a problem formulation with no closed-form solution. In practice, assuming the independence of these random variables simplifies this issue and leads to a good performance.

Theoretically, this joint pdf describes the entire multi-dimensional joint TDOA space. In practice, however, given that the TDOA distributions were generated by the same mixture of acoustic events in the room and captured by the same set of microphones, we can conclude that not every TDOA combination is physically possible. Note that this is the same observation that facilitated the extraction of sub-spaces of dominance in the C-SRP approach, introduced in Section 5.5. The joint TDOA sub-space of physically possible combinations can be obtained by simply mapping the joint pdf, given by (5.39), to the location space using the location-TDOA mapping given by (4.2), which leads to the Acoustic Event Joint Distribution (AEJD) $p_{\mathcal{E}\mathcal{M}}(\mathbf{l})$ describing the mixture of acoustic events in the location space, defined as

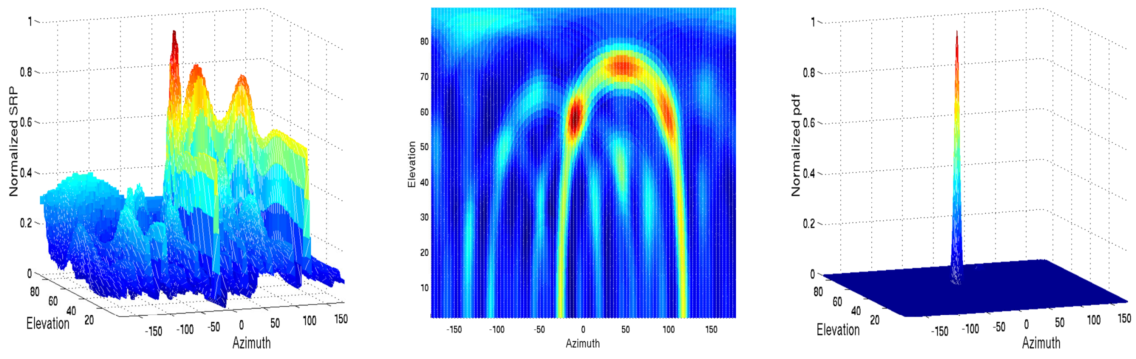
$$p_{\mathcal{E}\mathcal{M}}(\mathbf{l}) \propto \prod_{q\in\mathcal{Q}} \sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}^q\big(\tau^q(\mathbf{l}), \mu_k^q, (\sigma_k^q)^2\big), \tag{5.40}$$

where $\mathcal{E}\mathcal{M}$ denotes the mixture of acoustic events.

5.6.2 Localization of an Acoustic Event

The pdf $p_{\mathcal{E}\mathcal{M}}(\mathbf{l})$ is a multi-modal prior distribution, where each *mode* represents a potential acoustic source. Following the dominance-based localization that we previously used in P-SRP and C-SRP, the extraction of each potential acoustic source consists of calculating the restriction of $p_{\mathcal{E}\mathcal{M}}$ on the region of dominance associated to it.

Formally, the restriction, $p_{\mathcal{E}}(\mathbf{l})$, of $p_{\mathcal{E}\mathcal{M}}(\mathbf{l})$ on the region of dominance $\mathcal{L}_{\mathcal{E}}$, associated to an



(a) Conventional SRP: Side view (b) Conventional SRP: Top view (c) AEJD approach: Side view

Figure 5.5: *The graphs in (a) and (b) exemplify the SRP approach for a frame with a single speaker. We can see that the dominant GCC peak generates multiple local maxima. The graph in (c) shows that the AEJD-based approach cancels these secondary local maxima using the information provided by other GCC functions.*

acoustic event $\mathcal{E}$, is given by:

$$\begin{aligned}
p_{\mathcal{E}}(\mathbf{l}) &\propto \mathbb{1}_{\mathcal{L}_{\mathcal{E}}}(\mathbf{l}) \cdot p_{\mathcal{E}\mathcal{M}}(\mathbf{l}) \\
&\propto \mathbb{1}_{\mathcal{L}_{\mathcal{E}}}(\mathbf{l}) \cdot \left(\prod_{q\in\mathcal{Q}} \sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}^q \left(\tau^q(\mathbf{l}), \mu_k^q, (\sigma_k^q)^2\right) \right) \\
&\propto \prod_{q\in\mathcal{Q}} \mathbb{1}_{I_{\mathcal{E}}^q}\left(\tau^q(\mathbf{l})\right) \cdot \left(\prod_{q\in\mathcal{Q}} \sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}^q \left(\tau^q(\mathbf{l}), \mu_k^q, (\sigma_k^q)^2\right) \right) \\
&\propto \prod_{q\in\mathcal{Q}} \left(\mathbb{1}_{I_{\mathcal{E}}^q}\left(\tau^q(\mathbf{l})\right) \cdot \sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}^q \left(\tau^q(\mathbf{l}), \mu_k^q, (\sigma_k^q)^2\right) \right). \qquad (5.41)
\end{aligned}$$

Using **assumption (A)**, introduced in Section 4.2.1 and used also in P-SRP and C-SRP, we can assume that the GM contribution on each interval of dominance $I_{\mathcal{E}}^q$ mainly comes from the Gaussian component $k_{\mathcal{E}}^q$ which is associated to this interval. Therefore, we can further simplify $p_{\mathcal{E}}(\mathbf{l})$ to obtain a more practical approximation, given by:

$$p_{\mathcal{E}}(\mathbf{l}) \approx \prod_{q\in\mathcal{Q}} p_{k_{\mathcal{E}}^q}^q(\mathbf{l}) \cdot \mathcal{N}^q \left(\tau^q(\mathbf{l}), \mu_{k_{\mathcal{E}}^q}^q, (\sigma_{k_{\mathcal{E}}^q}^q)^2 \right). \qquad (5.42)$$

Similarly to P-SRP, acoustic source location is then obtained using numerical optimization algorithms on $p_{\mathcal{E}}(\mathbf{l})$ [97]. Theoretically, $p_{\mathcal{E}}(\mathbf{l})$ is not a convex function. In practice, however, this function has a sharp peak and very flat tails (see illustration in Figure 5.5c). Furthermore, any initial guess from $\mathcal{L}_{\mathcal{E}}$ ensures the convergence to the optimal location. In the following section, we will show how regions with high likelihood can be detected before optimizing this distribution on each of these regions.

5.6.3 Detection of Space Regions with High Likelihood

We use here the same accumulation approach used in C-SRP. More precisely, given the formal definition of the region of dominance $\mathcal{L}_{\mathcal{E}}$, associated to an acoustic event $\mathcal{E}$, given by (5.11), we can conclude that all locations in $\mathcal{L}_{\mathcal{E}}$ will map to the same combination of intervals of dominance $\{I_{\mathcal{E}}^{q}\}_{q}$. Hence, the extraction of $\mathcal{L}_{\mathcal{E}}$ can be reduced to finding a single location from it. Similarly to P-SRP and C-SRP, we can use here a *coarse grid* (15° to 30° or 50 to 150 cm). The grid resolution is chosen such that at least one location falls into $\mathcal{L}_{\mathcal{E}}$. Then, for each location $\mathbf{l}^{init}$ in this grid, and for each microphone pair q, 1) the associated interval of dominance $I_{\mathbf{l}^{init}}^{q}$ is extracted such that the corresponding Gaussian component $\mathcal{N}^{q}(\tau^{q}(\mathbf{l}), \mu_{k}^{q}, (\sigma_{k}^{q})^{2})$ maximizes $\tau^{q}(\mathbf{l}^{init})$, and 2) the cumulative density function (cdf) $\mathcal{C}(\mathcal{L}_{\mathbf{l}^{init}})$ of $p_{\mathcal{E}\mathcal{M}}(\mathbf{l})$ over the associated region of dominance $\mathcal{L}_{\mathbf{l}^{init}}$, given by (5.11), is calculated according to:

$$\begin{aligned}
\mathcal{C}(\mathcal{L}_{\mathbf{l}^{init}}) &= \int_{\mathcal{L}_{\mathbf{l}^{init}}} p_{\mathcal{E}\mathcal{M}}(\mathbf{l})\, d\mathbf{l} \\
&= \int_{\mathcal{L}_{\mathbf{l}^{init}}} \left(\prod_{q \in \mathcal{Q}} p(\tau^{q}(\mathbf{l})) \right) d\mathbf{l} \\
&\approx \prod_{q \in \mathcal{Q}} \left(\int_{I^{q}_{k^{q}_{\mathbf{l}^{init}}}} p(\tau^{q})\, d\tau^{q} \right) \\
&\approx \prod_{q \in \mathcal{Q}} p^{q}_{k^{q}_{\mathbf{l}^{init}}}.
\end{aligned} \tag{5.43}$$

Note that the dominance notion and its mapping from the location space to the TDOA space are used to calculate this expression. That is, the first approximation maps the region of dominance $\mathcal{L}_{\mathbf{l}^{init}}$ to the corresponding set of intervals of dominance $\{I^{q}_{k^{q}_{\mathbf{l}^{init}}}\}_{q \in \mathcal{Q}}$ to simplify the calculation, whereas the second approximation ignores the contribution of the other components in the GM on each of these intervals.

The region of dominance $\mathcal{L}_{\mathcal{E}}$, to which the initial location $\mathbf{l}^{init}$ belongs, is extracted as the one with the highest cumulative distribution. Thus, $\mathbf{l}^{init}$ can be used as an initial guess to run the numerical optimization. The experiments reported below use the gradient descent algorithm to perform this task [97]. Note that we can also use a second sub-grid centered at $\mathbf{l}^{init}$, similarly to C-SRP, to obtain a better initial guess. The resolution of this sub-grid, however, would be much larger compared to the one used in C-SRP.

5.6.4 Acoustic Source Detection

The proposed method extracts the source location as the one with the highest likelihood, but it does not consider whether this location is generated by a dominant peak in the GCC function or by a consistent set of peaks from different GCC functions. This problem becomes more difficult in the multiple speaker scenario, especially, when the number of overlapping sources is unknown. Furthermore, when only few microphones are available, a dominant GCC peak may generate many local maxima, which makes the problem more complex. We present in this section a threshold-based detection method to solve the speaker/non-speaker classification task.

Finding the optimal location requires the optimization of $p_{\mathcal{E}}(\mathbf{l})$, which is equivalent to the minimization of the ML criterion [39, 64], given by:

$$\mathbf{l}_{\mathcal{E}}^{opt} = \underset{\mathbf{l}}{\operatorname{argmax}}\ \epsilon(\mathbf{l}) \quad \text{with} \quad \epsilon(\mathbf{l}) = \sum_{q\in\mathcal{Q}} \frac{1}{(\sigma^q_{k^q_{\mathcal{E}}})^2} \cdot \left[\tau^q(\mathbf{l}) - \mu^q_{k^q_{\mathcal{E}}}\right]^2. \tag{5.44}$$

The error function $\epsilon(\mathbf{l})$ characterizes the *consistency* of the optimal location $\mathbf{l}^{opt}$. More precisely, a high value of $\epsilon\left(\mathbf{l}^{opt}\right)$ means that $\mathbf{l}^{opt}$ was generated by a dominant peak rather than a combination of consistent GCC peaks and vice versa. Hence, $\epsilon(\mathbf{l})$ can be used as a validation criterion. Formally, $\mathbf{l}^{opt}$ is assumed to be generated by an actual source if $\epsilon(\mathbf{l}^{opt}) \leq \Gamma$, where Γ is a predefined threshold. This error will be used as a feature in the next chapter to introduce a more advanced Bayesian framework for the speaker/noise classification problem.

5.6.5 AEJD-based Multiple Speaker Localization

The proposed acoustic source localization approach can be easily extended to the multiple speaker case. Algorithm 5 presents one possible extension using an iterative approach. The algorithm is iterative in order to avoid re-detecting the same region of dominance in the case where more than one location $\mathbf{l}^{init}$ belongs to it. This idea is implemented by successively discarding all the regions $\mathcal{L}_i$ that contain the optimal location $\mathbf{l}_n^{opt}$ from the previous iteration in step 7. In the case where N_{max} is unknown, it can be simply overestimated.

Algorithm 5 : AEJD-based Multiple Speaker Localization

```
Let $N_{max}$ be the maximum number of speakers
Let $\mathcal{G}$ be the coarse grid
1. Estimate the GM approximations and define $p_{\mathcal{E}\mathcal{M}}(\mathbf{l})$
2. Calculate $\mathcal{C}(\mathcal{L}_i)$ for each location $\mathbf{l}^i \in \mathcal{G}$
for $n = 1 : N_{max}$ do
    3. Find $\mathcal{L}_n^{max}$ which maximizes $\mathcal{C}(\mathcal{L}_i)$
    4. Calculate $p_{\mathcal{L}_n^{max}}(\mathbf{l})$
    5. Run numerical optimization on $p_{\mathcal{L}_n^{max}}(\mathbf{l})$ to estimate $\mathbf{l}_n^{opt}$
    if $\epsilon(\mathbf{l}_n^{opt}) \leq \Gamma$ then
        6. Add $\mathbf{l}_n^{opt}$ to the set of speakers $\mathcal{S}$
        7. Remove all $\mathcal{L}_i$, for which $\mathbf{l}_n^{opt} \in \mathcal{L}_i$, from $\mathcal{G}$
    else
        9. BREAK
    end if
end for
10. Return the set of speakers $\mathcal{S}$
```

5.6.6 Evaluation of AEJD-based Speaker Localization

Similarly to P-SRP and C-SRP, we evaluated the proposed AEJD-based speaker localization on the AV16.3 corpus [71], Section 2.3.1 gives a more detailed introduction to this corpus, including the different scenarios it offers.

The evaluation reported below was conducted following the same experimental setup used to evaluate P-SRP and C-SRP with a crucial modification, namely, we use only 4 microphones in this evaluation instead of 8 to highlight the ability of the proposed approach to cancel local maxima. The results are reported in terms of the same metrics that were previously used to evaluate the performance in Section 5.4.7 and Section 5.5.5.

Table 5.5 presents the performance of the proposed AEJD-based single speaker localization, and compares it to the classical SRP [46] and MCCC [42] approaches. In addition to this, we also report results of the P-SRP approach to highlight its degradation when only few microphones are used. Note that, in these experiments, the detection approach from Section 5.6.4 was not used. Hence, N_{max} was set to 1. The coarse grid resolution is $20° \times 30° \times 50$cm for the azimuth, elevation and range, respectively, whereas the resolution of the SRP and MCCC grid is $0.5° \times 0.5° \times 10$cm.

Approaches	seq01-1p-0000				seq02-1p-0100				seq03-1p-0100			
	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t
MCCC	46.4	3.39	9.56	105	70.6	3.05	12.0	83.3	**67.4**	3.34	10.4	83.1
SRP	**46.7**	**3.24**	**8.34**	12.4	**71.0**	**2.85**	9.15	12.0	67.0	**2.91**	**8.95**	10.5
P-SRP	37.0	3.80	9.71	**0.27**	60.1	3.69	10.8	**0.26**	56.1	4.46	10.4	**0.26**
AEJD	45.1	3.78	9.30	0.43	68.1	2.99	**8.77**	0.37	64.1	3.06	9.03	0.37

(a) Single speaker localization performance on audio sequences with a static speaker.

Approaches	seq11-1p-0100				seq15-1p-0100			
	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t	p_s	$\sigma_{s,\theta}$	$\sigma_{s,\phi}$	t
MCCC	68.5	3.17	9.65	83.5	64.6	2.64	8.88	83.2
SRP	**70.7**	**3.09**	9.95	10.6	**65.9**	2.41	7.08	10.5
P-SRP	67.8	4.46	10.7	**0.24**	51.6	**2.24**	7.27	**0.23**
AEJD	68.8	3.63	**9.39**	0.35	60.4	2.81	**6.81**	0.38

(b) Single speaker localization performance on audio sequences with a moving speaker.

Table 5.5: AEJD-based localization performance on audio recordings, from the AV16.3 corpus, with a single speaker.

The results reported in Table 5.5 show that AEJD-based single speaker localization performs comparably to SRP and MCCC while it clearly outperforms P-SRP. The latter suffers from the coarse resolution of the initialization grid, which is highly affected by local maxima that become comparable to the main peak when only few microphones are used (see example in Figure 5.5a). SRP and MCCC, however, suffer less from this problem given the fine resolution of their search grid. Regarding the computation cost, we can conclude that AEJD-based single speaker localization and P-SRP can easily run in real-time on a low computation machine, which is not the case for the classical SRP and MCCC approaches. The latter is very slow due to the calculation of the correlation matrix determinant for all locations, at each time frame.

Table 5.6 reports the multiple speaker localization results for a 4-microphone array. In addition to the metrics above, we also report the percentage of frames with N correct, simultaneous estimates. p_s in this case represents the percentage of correct estimates for all speakers. The detection thresholds, i.e., Γ for AEJD and P_{noise} for P-SRP, are chosen such that the resulting FAR is the same for both methods and equal to 30%.

Approaches	seq18-2p-0101		seq24-2p-0111	
	AEJD	P-SRP	AEJD	P-SRP
p_s	**60.1**	45.2	**48.1**	45.0
$\sigma_{s,\theta}$	**2.00**	2.79	**2.53**	3.23
$\sigma_{s,\phi}$	**4.51**	9.76	**5.54**	10.6
% of frames with	Overlap detection rate o_r (%)			
1 Speaker	23.5	**66.7**	40.5	**63.9**
2 Speakers	**63.1**	23.6	**36.0**	18.5

(a) Multiple speaker localization performance on audio sequences with *two* simultaneous speakers.

Approaches	seq40-3p-0111		seq45-3p-1111		seq37-3p-0001	
	AEJD	P-SRP	AEJD	P-SRP	AEJD	P-SRP
p_s	**50.3**	45.6	**38.3**	29.0	**54.5**	46.2
$\sigma_{s,\theta}$	**1.90**	4.75	**2.16**	3.19	**2.66**	5.0
$\sigma_{s,\phi}$	**7.93**	13.4	**5.88**	10.8	**9.47**	13.4
% of frames with	Overlap detection rate o_r (%)					
1 Speaker	21.4	**59.5**	31.3	**51.1**	25.2	**65.0**
2 Speakers	**45.0**	23.0	**25.0**	10.0	**41.0**	15.4
3 Speakers	**7.47**	1.72	**4.07**	0.77	**7.51**	0.96

(b) Multiple speaker localization performance on audio sequences with *three* simultaneous speakers.

Table 5.6: AEJD-based multiple speaker localization performance on the AV16.3 corpus.

Regarding the multiple speaker results reported in Table 5.6, we can see that AEJD-based multiple speaker localization approach deals better with the problem of local maxima resulting from the multivalued TDOA-location function, where a dominant GCC peak may generate many peaks in the location space (see example in Figure 5.5). This improvement appears in the increased number of correct estimates p_s, and the percentage of frames with correct simultaneous estimates. The latter shows that the proposed approach leads to a lower number of frames with one correct estimate, compared to P-SRP, whereas the percentage of frames with two or more correct estimates increases. This shows the ability of AEJD to detect the sources in the location space. Regarding the localization precision, we can once again conclude that the proposed approach leads to a more accurate azimuth and elevation estimates compared to P-SRP. This is mainly due to the nature of the maxima in each distribution. More particularly, AEJD has sharp peaks, whereas the peaks of P-SRP are flat and span over large regions.

To conclude, we can say that AEJD proposes a different method to combine the GCC functions in order to increase the localization precision when only few microphones are available. This approach was also shown to be more effective in canceling the local maxima resulting from the multivalued TDOA-location function, which highly affects the multiple speaker detection performance.

6 Multiple Speaker Detection: Speaker/Noise Classification

A good multiple speaker localization performance cannot be achieved without a speech detector, which classifies the estimated acoustic events, obtained with one of the multiple speaker localization approaches introduced in Chapter 5, to speaker/noise. This is mainly due to:

- The presence of noise and/or reverberation which introduces secondary peaks.
- The unknown and time-varying number of speech and noise sources per time frame.

Few attempts have been made in the past to solve this problem, the authors of [54] proposed to use the distance separating the location estimates as a criterion to extract the number and location of the speakers, whereas Do et *al.* [56, 57] proposed to combine the signal power with a double clustering technique to estimate the number of speakers. In a more advanced approach, Lathoud et *al.* [53] proposed an unsupervised threshold selection technique to control the false alarm rate.

We have introduced in Chapter 5 different iterative approaches to localize multiple speakers using P-SRP (Algorithm 2), C-SRP (Algorithm 4) and AEJD-based localization (Algorithm 5), as we have also seen that C-SRP used an unsupervised classifier to detect real speaker estimates, whereas P-SRP and AEJD used threshold-based detection methods to solve this task. More precisely, P-SRP uses a probabilistic threshold, whereas AEJD uses a TDOA consistency-based threshold. While these thresholds could be selected carefully to achieve a good performance, these models are highly dependent on different factors that can cause the optimal values to change drastically, which makes the usage of a fixed threshold far from being optimal and difficult to adapt online. The factors that could cause this problem include:

- The location of the speakers and their distance to the microphone array.
- The number of microphones in the array.
- The Signal-to-Noise Ratio (SNR) and the noise type.
- The number of speakers in the room.
- The dimensions, reverberation time and Signal-to-Reverberation Ratio (SRR).

To overcome this problem, we follow a line of thought similar to [53] and propose to estimate the optimal boundary between noise and speaker classes by extending the detection approach used in C-SRP using a Naive Bayesian Classifier (NBC). Contrary to the approaches taken in [53, 56], where a *single* power-based feature is used, we propose in this work a Bayesian framework which combines two types of features, namely,

1. The Dominance-based Cumulative Power (DCP) feature, which characterizes the steered power/probability mass of the sub-space of dominance associated to an acoustic event. This is similar to the one used to define C-SRP in Section 5.5.2 and to the feature used to extract regions with high likelihood when using AEJD (Section 5.6.3).
2. The Maximum Likelihood Error (MLE) feature at the location estimate. This is similar to the one used in combination with AEJD to detect speakers (Section 5.6.4).

In doing so, the speaker/noise classes become more separable. This property is of most interest in low SNR/SRR environments, as well as in the multiple speaker case, where the signal power of secondary speakers becomes comparable to the noise/reverberation power.

In this framework, we first estimate the likelihood distribution and the prior of each class. This is done by fitting a 3-component mixture to each feature space. Then, the posterior distribution of each class is obtained using an NBC, which combines these two features. The choice of the mixtures is dependent on the feature.

6.1 Dominance-based Cumulative Power Classifier

Similarly to the approach taken in [53], we propose to use the steered power as the first classification feature. This approach, however, does not simply consider the power coming from the location of the acoustic event, it rather considers the cumulative power emerging from the sub-space of dominance associated to it. Let $\mathcal{E}$, $\mathcal{L}_{\mathcal{E}}$ and $\mathbf{l}_{\mathcal{E}}$ be an acoustic event, its sub-space of dominance and its estimated location, respectively. The next section introduces the DCP feature for P-SRP, C-SRP and the acoustic event joint distribution.

6.1.1 DCP Feature

The DCP feature is based on **assumption (A)**, described in Section 4.2.1. That is, we can assume that all the probabilistic steered response power over a sub-space of dominance is fully produced by the corresponding acoustic event. Therefore, we can define the P-SRP-based DCP feature for an acoustic event $\mathcal{E}$ according to:

$$\begin{aligned} \mathcal{C}_{prob}(\mathcal{L}_{\mathcal{E}}) &= \int_{\mathcal{L}_{\mathcal{E}}} SRP_{prob}(\mathbf{l})\, d\mathbf{l} \\ &= \int_{\mathcal{L}_{\mathcal{E}}} \left(\sum_{q\in\mathcal{Q}} p(\tau^q(\mathbf{l})) \right) d\mathbf{l} \\ &\approx \sum_{q\in\mathcal{Q}} \left(\int_{I^q_{k^q_{\mathcal{E}}}} p(\tau^q)\, d\tau^q \right) \\ &\approx \sum_{q\in\mathcal{Q}} p^q_{k^q_{\mathcal{E}}}. \end{aligned} \tag{6.1}$$

Similarly, we define the AEJD-based DCP feature for an acoustic event, extracted using the AEJD-based multiple speaker localization approach (Section 5.6.5), according to:

$$\mathcal{C}_{joint}(\mathcal{L}_{\mathcal{E}}) \approx \prod_{q\in\mathcal{Q}} p^q_{k^q_{\mathcal{E}}}. \tag{6.2}$$

More details about this approximation are given in Section 5.6.3.

For low resource applications where the C-SRP becomes a more attractive choice, we also define the C-SRP-based DCP feature for an acoustic event $\mathcal{E}$ as:

$$\mathcal{C}_{csrp}(\mathcal{L}_{\mathcal{E}}) \approx \sum_{q\in\mathcal{Q}} \sum_{\tau^q \in I^q_{\mathcal{E}}} \mathcal{R}^q(\tau^q). \tag{6.3}$$

More details about this approximation are given in Section 5.5.2.

6.1.2 DCP Mixture Distribution

Let $\{(\mathbf{l}^i, \mathcal{C}_{\mathbf{l}^i})\}_{i=1}^{N_T}$ denote the set of N_T location estimates $\mathbf{l}^i$ and their corresponding dominance-based cumulative power features $\mathcal{C}_{\mathbf{l}^i}$, obtained in T frames ($N_T = T \times N_{max}$). $\mathcal{C}_{\mathbf{l}^i}$ are calculated using (6.1) when using P-SRP for multiple speaker localization (Algorithm 2), according to (6.2) when using AEJD-based solution (Algorithm 5) and according to (6.3) when using C-SRP (Algorithm 4). We propose to separate real speakers from noise by fitting a 3-component mixture distribution to the data in the DCP space. This mixture is obtained by maximizing the likelihood of the DCP estimates $\{\mathcal{C}_{\mathbf{l}^i}\}_{i=1}^{N_T}$ using the EM algorithm [100].

Formally, the EM algorithm estimates a mixture distribution of the form:

$$f^{dcp}(\mathbf{l}) = w_n^{dcp} \cdot \mathcal{G}_n^{dcp}(\mathcal{C}_\mathbf{l}) + w_s^{dcp} \cdot f_s^{dcp}(\mathcal{C}_\mathbf{l}), \tag{6.4}$$

$\mathcal{G}_n^{dcp}(.)$ is a Gaussian distribution approximating the likelihood distribution of the noise, whereas $f_s^{dcp}(.)$ is a "Gaussian+Uniform" mixture distribution approximating the likelihood distribution of the speakers (see example Figure 6.1). Thus, $f_s^{dcp}(.)$ is given by:

$$f_s^{dcp}(\mathcal{C}) \propto \mathcal{G}_s^{dcp}(\mathcal{C}) + \mathcal{U}_s^{dcp}(\mathcal{C}), \tag{6.5}$$

where $\mathcal{G}_s^{dcp}(.)$ is a Gaussian distribution and $\mathcal{U}_s^{dcp}$ is a uniform distribution. w_n^{dcp} and w_s^{dcp} denote the noise and speaker priors, respectively. The uniform distribution $\mathcal{U}_s^{dcp}$ is introduced to model the high C-SRP values, which are poorly modeled by $\mathcal{G}_s^{dcp}$.

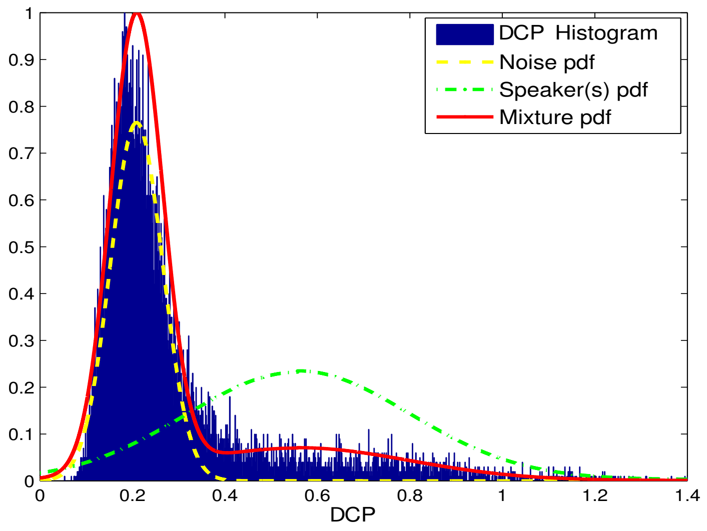


Figure 6.1: *Mixture distribution approximating the pdf of the C-SRP-based DCP feature.*

6.2 Maximum Likelihood Error Classifier

The second classifier is based on the MLE introduced at each location estimate. We explain here the idea behind the MLE feature and show how its pdf can be approximated.

6.2.1 MLE Feature

The MLE feature given by (4.4) is used under the assumption that it highly depends on the nature of the acoustic event. More precisely, we expect MLE to be large for *distributed* acoustic events and diffused noise, whereas it is expected to be low for *point* events. In order to evaluate this assumption, we carefully investigated the correlation between the variance of the C-SRP-based location estimates and their corresponding DCP feature. This investigation has shown that the distributed acoustic events such as door slams, projector noise, diffused noise, etc. are characterized by flat peaks, whereas the point sources map to sharp peaks. This property is mainly due to the nature of the GCC peaks representing the same source but in different microphone pairs. For a distributed event, the peaks are generally flat, and might map to different peaks in the location space. As a result, the variance of the estimates is expected to be large and the MLE tends to increase, and vice versa. Figure 6.2 shows an illustration of this investigation. We can clearly distinguish two separate classes, where the location estimates of distributed events are more likely to fall in the space region with a low C-SRP value and high estimate variance and vice versa.

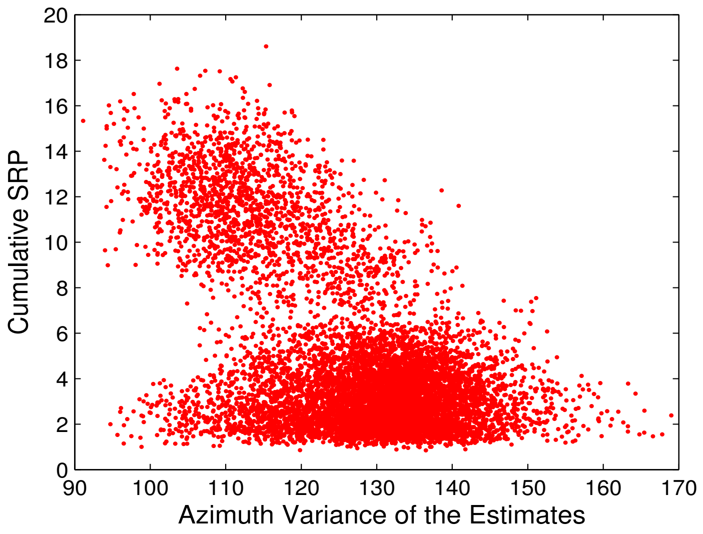


Figure 6.2: *Illustration of the high correlation between the variance of the C-SRP peaks generating the acoustic event location estimates and the C-SRP-based DCP feature. This figure shows clearly that the estimates map to two distinct classes, where distributed events are mostly characterized by a high variance but a lower DCP.*

6.2.2 MLE Mixture Distribution

We propose to use the approach presented in Section 6.1.2 to estimate the noise and speaker likelihoods with the exception of using different distributions. Let $\{(\mathbf{l}^i, \epsilon_{\mathbf{l}^i})\}_{i=1}^{N_T}$ denote the set of N_T location estimates $\mathbf{l}^i$ and their corresponding MLE values $\epsilon_{\mathbf{l}^i}$, obtained in T frames.

We propose to use a 3-component mixture distribution given by:

$$f^{mle}(\mathbf{l}) = w_s^{mle} \cdot \Gamma_s^{mle}(\epsilon_{\mathbf{l}}) + w_n^{mle} \cdot f_n^{mle}(\epsilon_{\mathbf{l}}), \tag{6.6}$$

where $\Gamma_s^{mle}(.)$ is a Gamma distribution approximating the likelihood distribution of the MLE feature for the speaker class, whereas $f_n^{mle}(.)$ is a "Gaussian+Uniform" mixture distribution approximating the likelihood distribution of the noise. $f_n^{mle}(.)$ is given by:

$$f_n^{mle}(\epsilon) \propto \mathcal{G}_n^{mle}(\epsilon) + \mathcal{U}_n^{mle}(\epsilon), \tag{6.7}$$

Similarly to (6.5), $\mathcal{U}_n^{mle}$ is introduced to model the high MLE values, which are poorly modeled by $\mathcal{G}_n^{mle}$. The DCP and the MLE features are combined in Section 6.3 to improve multiple speaker detection performance.

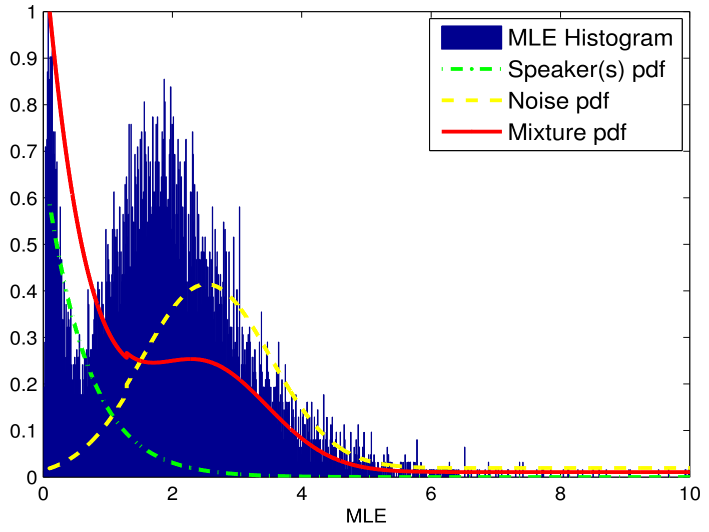


Figure 6.3: *Mixture distribution approximating the pdf of the MLE feature.*

6.3 Speaker/Noise Detection: A Naive Bayesian Classifier

The detection task can be improved by fitting a mixture distribution to the joint 2-D feature space formed by the DCP and the MLE features. Such approach would be very beneficial because it will incorporate the correlation between the two features, which would lead to a more realistic model. The distribution of the 2-D data formed by the concatenation of these two features, however, narrows the possible choices of a mixture distribution (see illustration in Figure 6.4a), which can efficiently maximize the likelihood, and thereby accurately models the joint feature distribution. This problem can be solved by maximizing the likelihood of the data in each feature space (Section 6.1.1 and Section 6.2.1) and then combining the resulting distributions using a naive Bayesian classifier [101].

Formally, let $\{X_i = (\mathbf{l}^i, \mathcal{C}_{\mathbf{l}^i}, \epsilon_{\mathbf{l}^i})\}_{i=1}^{N_T}$ be the set of augmented estimates, and let α be the classifier decision, $\alpha \in \{\text{speaker, noise}\}$. The posterior probability of the decision α given an estimate $X = (\mathbf{l}, \mathcal{C}_{\mathbf{l}}, \epsilon_{\mathbf{l}})$ is given by:

$$p(\alpha \mid X) = \frac{p(X \mid \alpha) \cdot p(\alpha)}{p(X)}. \tag{6.8}$$

NBC assumes the independence of the features [101] and expresses the joint likelihood distribution according to:

$$p(X \mid \alpha) = \prod_{k=1}^{2} p(X_k \mid \alpha) = p(\mathcal{C}_1 \mid \alpha) \times p(\epsilon_1 \mid \alpha). \tag{6.9}$$

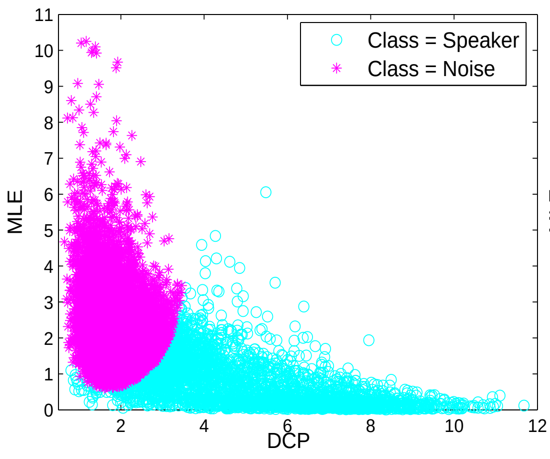
Replacing the terms in (6.8) and (6.9) by their expressions from (6.4), (6.5), (6.6) and (6.7) leads to the following unsupervised classifier:

$$p(\text{speaker} \mid X) \propto f_s^{dcp}(\mathcal{C}_1) \cdot \Gamma_s^{mle}(\epsilon_1) \cdot w_s^{dcp} \cdot w_s^{mle}, \tag{6.10}$$

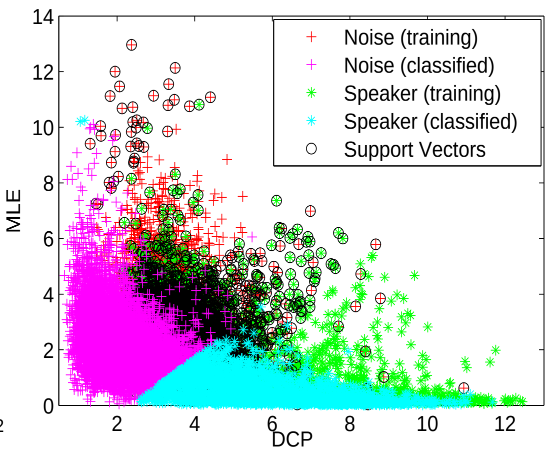
$$p(\text{noise} \mid X) \propto \mathcal{G}_n^{dcp}(\mathcal{C}_1) \cdot f_n^{mle}(\epsilon_1) \cdot w_n^{dcp} \cdot w_n^{mle}. \tag{6.11}$$

The decision α is independent of the probability of the estimate X. Therefore, $p(X)$ is ignored in (6.10) and (6.11). X is considered to be generated by an actual source if

$$p(\text{speaker} \mid X) \geq p(\text{noise} \mid X). \tag{6.12}$$



(a) Example of classification using NBC



(b) Example of classification using SVM

Figure 6.4: *The graph in (a) shows an example of a classification boundary obtained using the proposed unsupervised classifier (NBC), whereas (b) shows the classification boundary obtained using the SVM classifier on the same data.*

6.3.1 Loss Function For Noise Control

Acoustic source localization approaches can be generally integrated and used by a large number of applications, some of which may require a reduced noise rate, such as beamforming techniques [102], whereas others, such as audio-visual tracking approaches [23, 60], are more robust against noise and expect a higher frequency of location estimates, even if that leads to an increased noise rate. The variety of these approaches requires more flexibility in the acoustic event classification. This idea is successfully implemented using the loss function [7, 101].

Formally, let $\lambda(\alpha \mid g)$, be the loss incurred for deciding α knowing that g is the true class, with $\alpha, g \in \{\text{speaker, noise}\} = \{\mathcal{S}, \mathcal{N}\}$. The risk associated with taking the decision α given the estimate X is calculated according to:

$$\text{Risk}\,(\alpha \mid X) = \lambda(\alpha \mid \mathcal{S}) \cdot p(\mathcal{S} \mid X) + \lambda(\alpha \mid \mathcal{N}) \cdot p(\mathcal{N} \mid X). \tag{6.13}$$

The classification according to the minimum-risk decision rule is obtained by deciding $\mathcal{S}$ when Risk $(\mathcal{S} \mid X) \leq$ Risk $(\mathcal{N} \mid X)$ and vise versa. This rule is equivalent to:

$$\frac{\lambda(\mathcal{N} \mid \mathcal{S}) - \lambda(\mathcal{S} \mid \mathcal{S})}{\lambda(\mathcal{S} \mid \mathcal{N}) - \lambda(\mathcal{N} \mid \mathcal{N})} \geq \frac{p(\mathcal{N} \mid X)}{p(\mathcal{S} \mid X)}, \tag{6.14}$$

$\lambda(\mathcal{S} \mid \mathcal{S})$ and $\lambda(\mathcal{N} \mid \mathcal{N})$ represent the loss incurred for making the right decision. Therefore, these two parameters are generally set to 0. On the other hand, setting $\lambda(\mathcal{N} \mid \mathcal{S}) = \lambda(\mathcal{S} \mid \mathcal{N}) = 1$ leads to NBC (Section 6.3). Thus, we can conclude that (6.14) is a generalization of the proposed unsupervised classifier, where the noise rate can be controlled by adapting the ratio of $\lambda(\mathcal{N} \mid \mathcal{S})$ and $\lambda(\mathcal{S} \mid \mathcal{N})$, which are generally fixed beforehand.

6.4 Evaluation of NBC-based Speaker Detection

6.4.1 Experimental Setup

We evaluate the proposed approach using the AV16.3 corpus, which has been introduced in Section 2.3.1. More details about this corpus can be found in [71].

The acoustic source localization experimental setup used in this section is the same as one used in the evaluation of the multiple speaker localization approaches conducted in Chapter 5. More precisely, we use the P-SRP-based multiple speaker localization approach, detailed in Algorithm 2, to extract the potential location estimates following the configuration described in Section 5.4.7. The main difference concerns the fixed number of potential location estimates, which is fixed at $N_{max} = 6$ potential locations per time frame. Given that the maximum number of simultaneous (overlapping) speakers in AV16.3 recordings varies between 0 and 3, we can see that using $N_{max} = 6$ will lead to a noise rate of $\geq 50\%$. In the results reported below, the proposed speaker detection approach is compared to the classical Support Vector Machine (SVM) [7, 101] approach with a quadratic kernel (see illustration Figure 6.4b). Using different kernels did not improve the results. The SVM training data is obtained by calculating the MLE and DCP features for all locations given by the ground truth to form the *speaker* class, and for noise locations selected randomly to form the *noise* class. The reported results were obtained with a training on the audio sequence *seq02-1p-0000*, and then testing on the remaining multiple speaker sequences from the AV16.3 corpus. Using other sequences for training led to comparable results. The latter are reported in terms of Recall (R), Precision (P) and F-measure (F).

These measures are given by:

$$
\begin{aligned}
P &= \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}, && (6.15)\\
R &= \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}, && (6.16)\\
F &= 2 \cdot \frac{R \cdot P}{R + P}. && (6.17)
\end{aligned}
$$

The values of these measures are between 0 and 1. The higher they are, the better the classification is. The recall represents the fraction of *actual* speaker estimates that are correctly classified, whereas the precision reports the overall fraction of estimates which are correctly classified. Finally, the F-measure is a weighted harmonic mean of precision and recall. This measure is very relevant in assessing the overall performance.

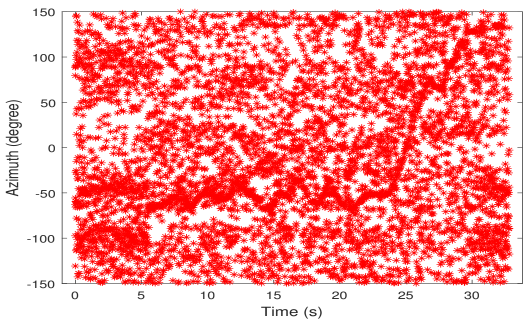
6.4.2 Results and Analysis

Sequences	SVM+DCP			SVM+MLE			SVM+DCP+MLE		
	R	*P*	*F*	*R*	*P*	*F*	*R*	*P*	*F*
seq18-2p-0101	0.46	**0.85**	0.60	**0.94**	0.33	0.49	0.83	0.67	**0.74**
seq24-2p-0111	0.42	**0.80**	0.55	**0.94**	0.27	0.45	0.83	0.56	**0.66**
seq40-3p-0111	0.26	**0.92**	0.41	**0.81**	0.56	0.67	0.58	0.82	**0.68**
seq45-3p-1111	0.30	**0.55**	0.40	**0.89**	0.26	0.40	0.70	0.42	**0.52**
seq37-3p-0001	0.10	**0.91**	0.17	**0.77**	0.49	0.60	0.72	0.61	**0.66**

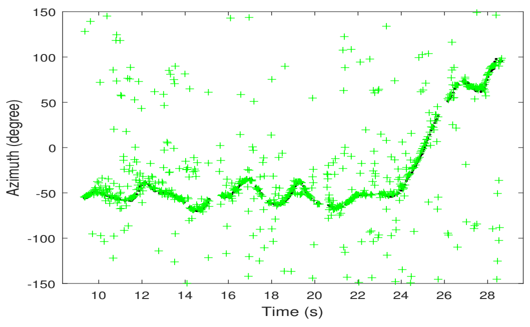
Table 6.1: Speaker/noise classification results using the proposed DCP and MLE features with an SVM classifier.

Table 6.1 presents the results of the multiple source detection task using the SVM classifier, when it is combined with each feature separately, as well as when the features are jointly used. These results show that combining the MLE and DCP features leads to better classification results. More precisely, we can see that using the MLE feature alone leads to a good recall performance but very poor precision. On the other hand, using the DCP feature alone results in a good precision performance but a poor recall. Combining these two features, however, provides more information to the SVM classifier, which successfully increases the F-measure of all sequences. We can also see that on the contrary to the *MLE only* and *DCP only* results, the recall and precision performance on experiments with the joint features are balanced. We can conclude from these results that combining the MLE and DCP features increases the detection performance.

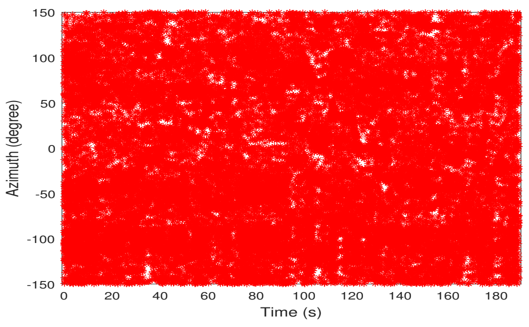
Table 6.2 reports the results of the proposed unsupervised NBC. These results show clearly that, overall, the proposed classifier performs slightly better than SVM. This is mainly due to the dependency of the features on the source location and the number of speakers. These two



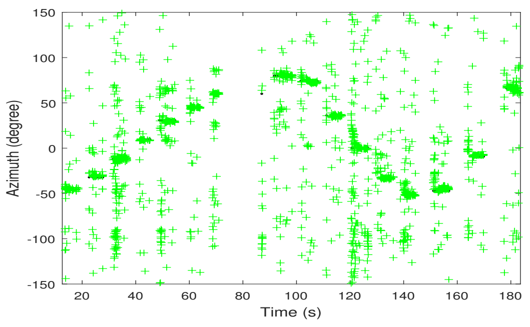
(a) *seq11-1p-0100*: Before speaker detection



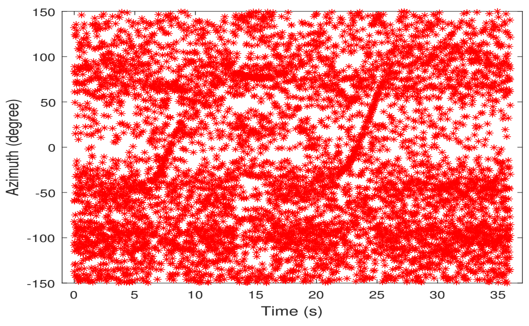
(b) *seq11-1p-0100*: After speaker detection



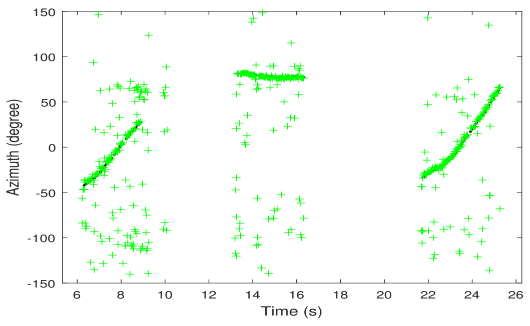
(c) *seq02-1p-0000*: Before speaker detection



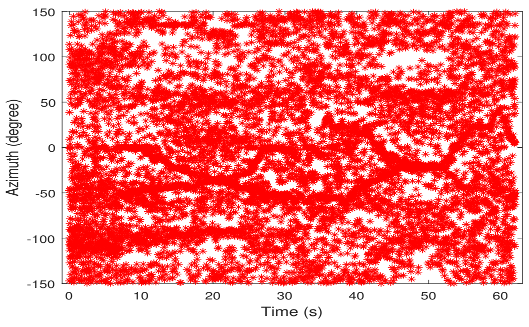
(d) *seq02-1p-0000*: After speaker detection



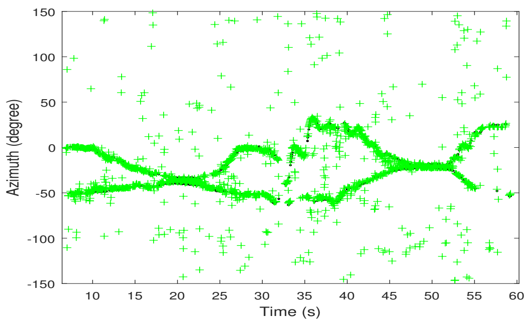
(e) *seq15-1p-0100*: Before speaker detection



(f) *seq15-1p-0100*: After speaker detection

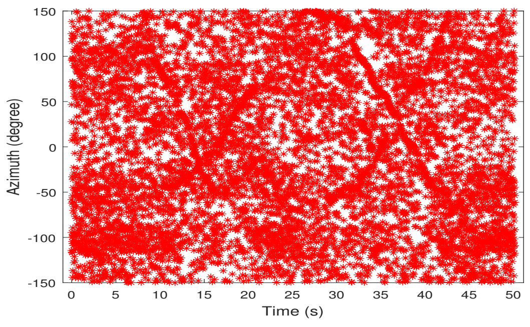


(g) *seq18-2p-0101*: Before speaker detection

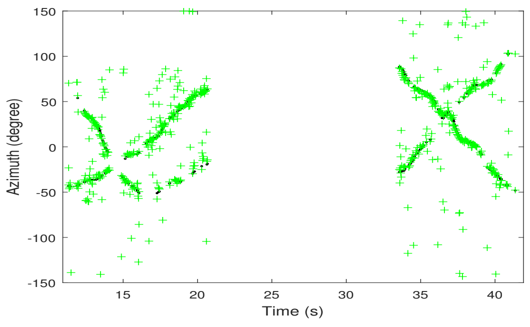


(h) *seq18-2p-0101*: After speaker detection

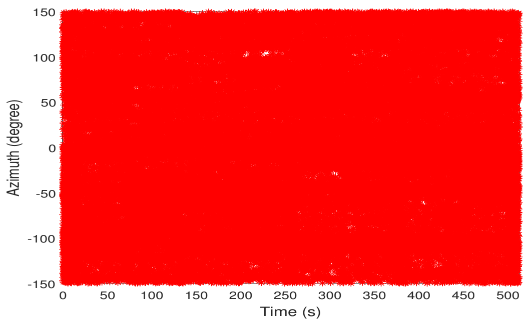
Figure 6.5: *Illustration of the detection performance of the proposed NBC, which uses DCP and MLE as classification features, on different sequences from the AV16.3 corpus. The left column shows the P-SRP-based localization (before detection), which extracts a fixed number ($N = 6$) of location estimates per time frame, whereas the right column shows the extracted speaker class after applying the NBC-based (multiple) speaker detection.*



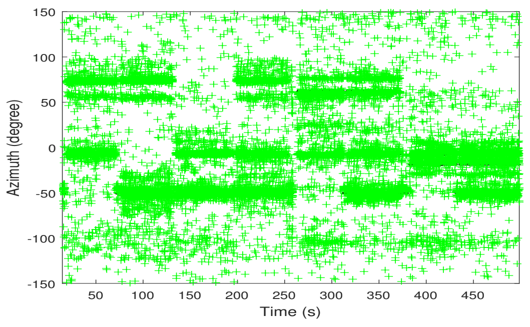
(i) *seq24-2p-0111*: Before speaker detection



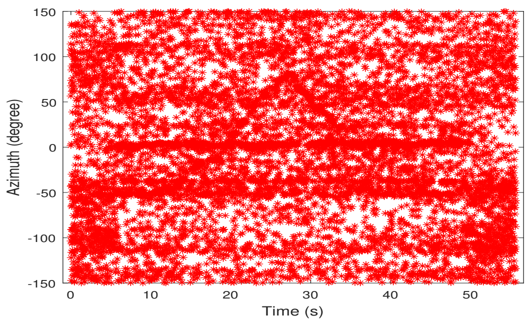
(j) *seq24-2p-0111*: After speaker detection



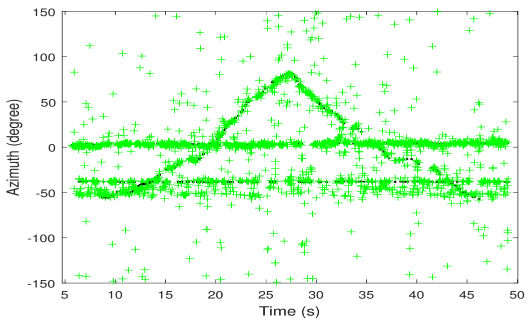
(k) *seq37-3p-0001*: Before speaker detection



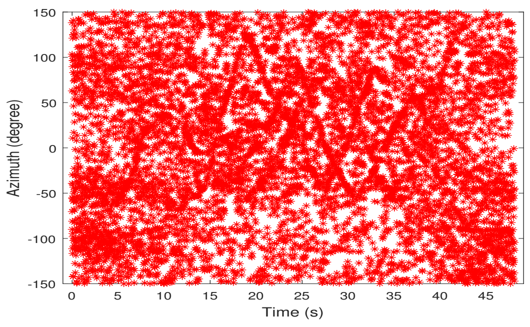
(l) *seq37-3p-0001*: After speaker detection



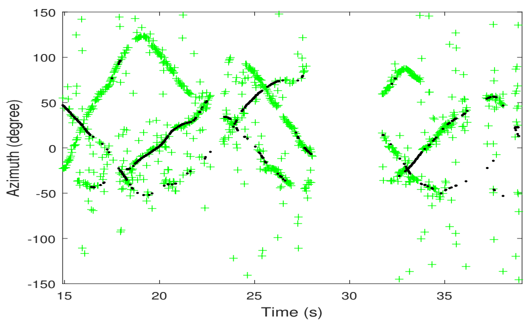
(m) *seq40-3p-0111*: Before speaker detection



(n) *seq40-3p-0111*: After speaker detection



(o) *seq45-3p-1111*: Before speaker detection



(p) *seq45-3p-1111*: After speaker detection

Figure 6.5: *Illustration of the detection performance of the proposed NBC, which uses DCP and MLE as classification features, on different sequences from the AV16.3 corpus. The left column shows the P-SRP-based localization (before detection), which extracts a fixed number ($N = 6$) of location estimates per time frame, whereas the right column shows the extracted speaker class after applying the NBC-based (multiple) speaker detection.*

	SVM+DCP+MLE			NBC+DCP+MLE		
Sequences	*R*	*P*	*F*	*R*	*P*	*F*
seq18-2p-0101	**0.83**	0.67	0.74	0.81	**0.72**	**0.76**
seq24-2p-0111	**0.83**	0.56	0.66	0.75	**0.71**	**0.73**
seq40-3p-0111	**0.58**	0.82	**0.68**	0.56	**0.85**	0.67
seq45-3p-1111	**0.70**	0.42	0.52	0.65	**0.48**	**0.55**
seq37-3p-0001	**0.72**	0.61	0.66	0.70	**0.65**	**0.67**

Table 6.2: Performance of the proposed NBC-based speaker detection in comparison to SVM when using the joint DCP and MLE features.

factors highly affect the level of the signal power and the SNR. Therefore, using a single training sequence to classify the different scenarios proposed by the AV16.3 corpus leads to a sub-optimal performance. The proposed classifier, however, adapts easily to these changes. This is due to the self-learning approach, which uses the data itself to infer the best boundary that explains the two classes.

6.5 Online Estimation of the Feature Distributions

Acoustic source localization applications, such as camera steering and audio-visual tracking, often require an online localization performance. Therefore, the speaker/noise classification should be also performed online. Algorithm 6 proposes an approach that accomplishes an online estimation of the feature distribution parameters from Section 6.1.2 and Section 6.2.2.

Algorithm 6 : Online Parameter Estimation

```
1. Initialize the distribution parameters randomly
2. Let $T$ be the re-estimation period
for each time $t$ multiple of $T$ do
    3. Set the initial parameters to the current parameters
    4. Keep the estimates from the last $N$ frames
    5. Re-estimate the parameters using the EM algorithm
end for
```

The proposed algorithm takes into account any possible changes in the distance between the acoustic sources and the microphone array, the number of speakers and the noise conditions, which might affect the detection performance. Therefore, only the last N frames are used to re-estimate the parameters. It is worth mentioning that N should not be too small as well to allow for an accurate update of the parameters.

6.6 History-based Classification Features

We have introduced in the previous section an online version of the naive Bayesian classifier. This approach considers only the classification features that are extracted from the last N processed frames. This section investigates an extension of the feature space by training new classification models based on the recent classification history. That is, we show here how the classification history itself can be used as labeled data to augment the feature space and thereby improve the detection performance. In particular, we consider three additional potential features, namely, 1) the classified location estimates and their corresponding 2) P-SRP and 3) kurtosis values are investigated. The idea here is to use the classified estimates into noise/speaker classes as labeled data to train, separately, new speaker and noise models for the different features. These models are then incorporated into the online NBC.

6.6.1 Location Feature

The location estimates are widely used as a main feature in speaker clustering and classification approaches. This is mainly due to the high density of the estimates originated from the same speaker in the location space, whereas the noise estimates are assumed to be randomly distributed. The main problem, however, is to identify which clusters of estimates represent actual speakers. This is mainly solved using speech cues or cluster variance-based discrimination [73]. We propose to overcome this identification problem here by training separately speaker(s) and noise models using the late classification history. Each model has the form of a GM with a number of components given by the minimum Bayesian Information Criterion (BIC). Algorithm 7 shows the proposed online training approach.

Algorithm 7 : Training of the location-based classification models

```
1. Let $T_{loc}$ be the re-estimation period.
for $t$ multiple of $T_{loc}$ do
    2. Use the last $N_{loc}$ speaker/noise estimates as two separate training sets.
    for $k = 1 \dots K$ do
        3. Train a speaker GM model $\mathcal{M}^{k}_{s,loc}$ ($k$ components)
        4. Train a noise GM model $\mathcal{M}^{k}_{n,loc}$ ($k$ components)
    end for
    5. Return the speaker and noise models $\mathcal{M}^{k_s}_{s,loc}$ and $\mathcal{M}^{k_n}_{n,loc}$, with
    $k_s = \underset{k}{\mathrm{argmin}}\ \mathrm{BIC}(\mathcal{M}^{k}_{s,loc})$
    $k_n = \underset{k}{\mathrm{argmin}}\ \mathrm{BIC}(\mathcal{M}^{k}_{n,loc})$
end for
```

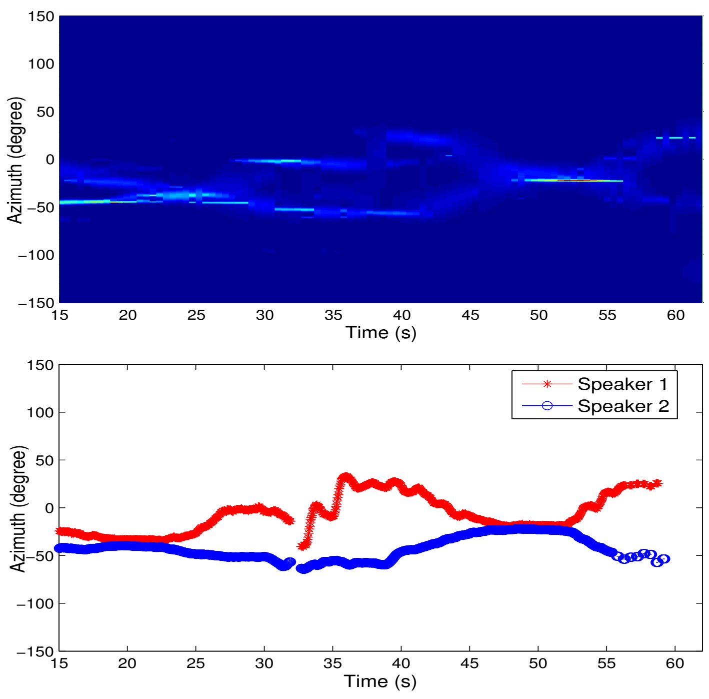


Figure 6.6: *Illustration of a history-based classification model: top figure illustrates the time evolution of the GM speaker model for the location feature. The figure in the bottom shows the location ground truth of the corresponding (two) overlapping speakers.*

6.6.2 Kurtosis Feature

High order statistics of signals have been widely investigated to solve speech and signal processing problems. More particularly, the kurtosis of a signal is typically used in Blind Source Separation (BSS) to separate speech sources [103, 104]. The kurtosis of a zero mean random variable x is calculated according to:

$$\text{kurt}(x) = \frac{\mathcal{E}\{x^4\}}{\{\mathcal{E}\{x^2\}\}^2}, \tag{6.18}$$

where $\mathcal{E}\{.\}$ is the expectation operator.

Speech signals are generally assumed to follow a super-Gaussian distribution, whereas noise signals are mostly modeled as a Gaussian distribution or as a mixture of Gaussians. Therefore, speech and noise signals are expected to have different kurtosis. Based on this difference, we propose to use the kurtosis as a new detection feature. More precisely, the signal coming for each location estimate is calculated using a superdirective beamformer, followed by the calculation of the kurtosis according to (6.18) as shown in Figure 6.7. The likelihood of each kurtosis model is approximated by a Gaussian distribution.

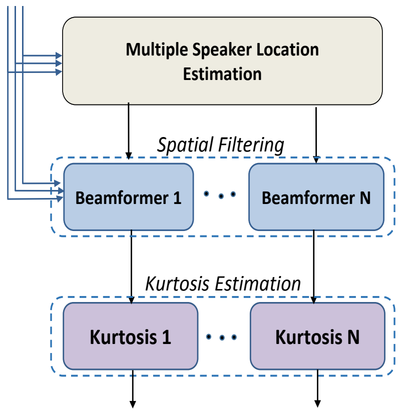


Figure 6.7: *Pipeline for calculating the kurtosis at each location estimate.*

6.6.3 P-SRP Feature

The P-SRP feature $SRP_{prob}(\mathbf{l})$ of a location estimate $\mathbf{l}$ is a probabilistic interpretation of the signal power at that particular location. We have shown in Section 6.2 that the DCP feature is highly correlated with the variance of the location estimates. More precisely, sharp SRP peaks representing point speech sources (mouth) are more likely to have a small variance, and therefore small MLE, contrary to noise sources, which are expected to span over wider space regions resulting in a larger variance and flat SRP peaks. This correlation is extended here to include the P-SRP feature. More particularly, we expect the P-SRP feature to help the classifier discriminating between estimates based on the signal power at these locations, whereas the DCP classifies the estimates based on the cumulative signal power coming from the region surrounding the location. Therefore, we expect the classifier to be able to correctly classify distributed noise source, such as projector, which generally have a high DCP value but low P-SRP. The main goal here is to use this complimentary and redundant information provided by MLE, DCP and P-SRP features to increase the robustness of the classifier. Similarly to the location and kurtosis features, the speaker and noise P-SRP classification models are trained separately using the recently classified speech and noise estimates, with the only exception of using a 3-component mixture similar to the one used to train the DCP classifier.

6.7 Evaluation of the Online History-based NBC

Following the evaluation of the offline NBC-based multiple speaker detection approach conducted in Section 6.4, we evaluated the proposed online approach on the AV16.3 corpus using the same experimental setup described in Section 6.4.1. Regarding the online NBC setup, the re-estimation period T of DCP, MLE, P-SRP and kurtosis is set to $T = 3$s, whereas $T_{loc} = 1$s. This differences aims at modeling any possible fast changes in the speaker(s) location. Moreover, the number of history-based classification estimates is fixed at $N = 1000$, whereas $N_{loc} = 300$. The first 20s of each recording were used to initialize the models.

We have shown in Section 6.4 that the offline NBC outperforms the classical SVM classifier [7, 101] when they are applied to the DCP and MLE features. Therefore, we will focus in this section on 1) the experimental comparison of the online NBC to its offline counterpart when they are applied to the same features, and then 2) evaluate the performance of the online classifier when the feature space is augmented with the proposed history-based classification models. The results are reported in terms of the Recall (R), Precision (P) and F-measure (F), which have been introduced in Section 6.4.1.

	seq18-2p-0101			seq24-2p-0111		
	R	*P*	*F*	*R*	*P*	*F*
Offline MLE+DCP	0.81	**0.72**	0.76	0.75	**0.71**	**0.73**
Online MLE+DCP	0.84	0.68	0.75	0.76	0.68	0.72
+P-SRP (only)	0.84	**0.72**	**0.78**	0.77	0.70	**0.73**
+LOC (only)	**0.88**	0.67	0.76	**0.80**	0.57	0.67
+KURT (only)	0.83	0.65	0.73	**0.80**	0.60	0.68
+P-SRP+LOC	0.87	0.71	**0.78**	0.78	0.68	**0.73**
+P-SRP+LOC+KURT	0.87	0.71	**0.78**	0.78	0.66	0.72

(a) Noise/speaker classification performance on recordings with *two* simultaneous speakers.

	seq40-3p-0111			seq45-3p-1111			seq37-3p-0001		
	R	*P*	*F*	*R*	*P*	*F*	*R*	*P*	*F*
Offline MLE+DCP	0.56	0.85	0.67	0.65	0.48	0.55	0.70	0.65	0.67
Online MLE+DCP	0.63	0.81	0.70	0.68	0.49	0.57	0.76	**0.71**	0.73
+P-SRP (only)	0.60	0.86	0.71	0.65	**0.51**	**0.58**	0.75	0.67	0.70
+LOC (only)	**0.65**	0.86	**0.74**	**0.73**	0.41	0.53	**0.80**	0.67	0.73
+KURT (only)	0.60	0.81	0.69	0.70	0.44	0.54	0.73	0.59	0.65
+P-SRP+LOC	0.64	**0.89**	**0.74**	0.68	0.50	**0.58**	0.79	0.70	**0.74**
+P-SRP+LOC+KURT	**0.65**	0.87	**0.74**	0.68	0.50	**0.58**	0.79	0.70	**0.74**

(b) Noise/speaker calssification performance on recordings with *three* simultaneous speakers.

Table 6.3: Speaker/noise classification performance of the proposed online NBC in comparison to its offline counterpart, in addition to the impact of iteratively adding different history-based features.

The results reported in Table 6.3 show that the proposed online classifier combined only with the MLE and DCP features perform better than its offline counterpart. This improvement is mainly due to the online and iterative adaptation of the model parameters, which is necessary in the case of changes in the speaker(s) environment. This improvement appears clearly in the sequences *seq40-3p-0111* and *seq37-3p-0001*, where the number of simultaneous speakers and distance to the array change over time. We can also conclude that the online classifier adapts quickly to the environment, as it only needs 20s to initialize and start using the models. We can also see that adding more features increases the robustness of the online classifier. More precisely, we can see that augmenting the online classifier with the P-SRP or the location (LOC) features alone leads to an *unstable* improvement of the performance. This is mainly due to the non-convergence of the classification models in few parameter re-estimation steps or due to long segments of intended silence. This instability of the performance appears also in the unbalanced precision (P) and recall (R) results. Combining more features, however, provides more information to the classifier, which successfully increases the F-measure of all sequences using the feature combination MLE+DCP+P-SRP+LOC. We can also conclude that adding the kurtosis feature to this mixture did not improve the performance. This is mainly due to the BF step which introduces distortions in the speech signal, leading to useless classification models in many of the re-estimation steps. This confirms a similar conclusion regarding the Mel Frequency Cepstral Coefficents (MFCC) features which was reported in [73].

7 Multiple Concurrent Speaker Tracking

Classical acoustic source tracking approaches consist of two stages: 1) extracting the measurements, which can be either TDOA at the microphone pairs [59], or noisy location estimates obtained with an SRP-based technique [46]. Then, 2) processing these measurements using a filtering approach such as PF [15, 62] or KF-based approaches [17, 64]. These two steps are generally combined with a multi-modal estimation framework, which allows the tracking of multiple instantaneous speakers, such approaches include the joint Probabilistic Data Association Filter (PDAF) [66], the multiple model particle filter [67] and the extended Kalman PF [68].

While SRP-based tracking is more robust and reliable, it is very slow compared to TDOA-based tracking, which can be easily integrated into a low-resource, real-time system. This category of approaches is, however, very difficult to extend to the multiple speaker scenario, and degrades drastically in the presence of noise and reverberation. In this case, using an SRP-based tracking becomes more crucial to solve the multiple speaker problem.

This chapter introduces new robust and fast approaches to track acoustic sources for different scenarios. In particular,

- Section 7.1 will present the single speaker tracking problem and then introduce two new solutions to increase the robustness of TDOA-based approaches.
- Section 7.2 will present the more general multiple speaker case and then propose a novel Kalman-based approach, which uses the location estimates extracted using one of the multiple speaker localization and detection approaches introduced in Chapter 5 and

Chapter 6, to track multiple concurrent speakers. This approach was specifically designed to overcome tracking problems that are related to the nature of spontaneous/-conversational speech.

7.1 TDOA-based Single Speaker Tracking

The performance of TDOA-based single speaker tracking approaches drastically degrades in the presence of noise and multi-path effects, especially under room acoustical conditions where early reflections and reverberation corrupt the GCC function through smearing as well as through the introduction of secondary peaks [46, 62]. This in turn affects the tracking filter, which assumes the error to be a stationary Gaussian process, whereas the TDOA error in a multi-path environment is rather time-varying and multi-modal. Classical approaches propose to overcome this problem by increasing the robustness of the adopted tracking filter to noisy TDOA estimates. An early work in this direction goes back to Vermaak [62] who proposed to use a multiple hypothesis particle filter. This approach has been further improved in [65], where an extended particle filter was combined with an SRP-based measurement estimator.

This section will introduce two solutions to improve the tracking performance, and increase the filter robustness to noisy TDOA measurements in the single speaker case. These two contributions can be summarized as follows:

- The first solution continues along the lines of [62, 65] by proposing a new Multiple Hypothesis Gaussian Mixture Filter (MH-GMF), which propagates the uncertainty of the TDOA estimates to the tracking stage. That is, this approach focuses on increasing the robustness of the filter by allowing it to handle multiple observations instead of one [23].
- Instead of focusing on increasing the robustness of the filter to noise, as it is done in classical approaches, the second solution proposes to improve the TDOA measurement quality itself using the tracking information available at each time step [17].

7.1.1 TDOA-based DSSM

As we have shown in Section 3.4, Bayesian filtering generally requires a Dynamic State Space Model (DSSM), which analytically determines the dynamics of the tracking system. In our case, where the estimated TDOA vectors play the role of observations and the hidden state is the speaker location (in spherical coordinates), the state space model should provide a *process model*, which predicts the transition, i.e., possible changes in the speaker location from previous to the current time steps, and a *measurement model*, which describes the physical/-geometrical relationship between the TDOA observations and the speaker location.

To express the proposed TDOA-based DSSM, let Q be the number of microphone pairs, i.e., Q is the cardinality of $\mathcal{Q}$, and let θ_{t-1} and ϕ_{t-1} be the azimuth and elevation of the speaker at time $t-1$. Our goal here is to propagate the filter to time t.

Single speaker tracking can be performed using Kalman filter [63, 64]. In order to do so, we use the following process model:

$$\begin{bmatrix}\theta_t\\ \phi_t\end{bmatrix} = f\left(\begin{bmatrix}\theta_{t-1}\\ \phi_{t-1}\end{bmatrix}, \mathbf{v}_t\right) = \begin{bmatrix}\theta_{t-1} + v_{t,\theta}\\ \phi_{t-1} + v_{t,\phi}\end{bmatrix}, \tag{7.1}$$

where $v_{t,\theta}$ and $v_{t,\phi}$ denote the zero-mean Gaussian process noise with a variance of σ_θ^2 and σ_ϕ^2, respectively. The choice to use a random walk as a process model is motivated by two main reasons: 1) The signal window used in this thesis is very short, i.e., 32ms, therefore, we can assume that this duration is very short for a human speaker to significantly change his/her location. 2) The second motivation is based on the fact that we would like to avoid imposing any constraints or pre-assumptions regarding the movement of the speaker and its direction. Furthermore, this choice has been shown to be reliable for tracking acoustic sources [62, 63, 64]. This model can be, however, extended to include the speed of the speaker as an additional process state to track.

The measurement model we use in our work is given by:

$$y_t = h\left(\begin{bmatrix}\theta_t\\ \phi_t\end{bmatrix}, \mathbf{w}_t\right) = \begin{bmatrix}\tau^1\left(\mathrm{DOA}[\theta_t, \phi_t]\right) + w_t^1\\ \vdots\\ \tau^Q\left(\mathrm{DOA}[\theta_t, \phi_t]\right) + w_t^Q\end{bmatrix}. \tag{7.2}$$

In this equation, $\tau^q\left(\mathrm{DOA}[\theta_t, \phi_t]\right)$ denotes the predicted TDOA of microphone pair $q \in \mathcal{Q}$, whereas w_t^q is a zero-mean Gaussian measurement noise with a variance of $\sigma_{\tau^q}^2$. This measurement model is nonlinear since the calculation of the predicted TDOAs according to (4.3) involves the evaluation of sines and cosines for the direction of arrival DOA$[\theta_t, \phi_t]$. Thus, the use of an extension of KF is required. We propose to use the UKF in the MH-GMF-based Bayesian estimation framework. This choice is motivated by, and can be seen as an extension of, the work presented in [63], which used an UKF to track speakers in a single observation model.

Note that our choice to track the Direction of Arrival (DOA) instead of the 3-D spherical coordinates is mainly due to the small and planar geometry of the microphone array, which makes tracking of the distance to the speaker unreliable. This problem can be dealt with, however, using two separate arrays and then tracking the distance to the speaker as the intersection of their respective DOAs. The rest of this section introduces the new proposed solutions.

7.1.2 Multiple Hypothesis Gaussian Mixture Filter (MH-GMF)

The MH-GMF proposes to solve the erroneous TDOA problem by propagating the measurements uncertainty to the tracking stage. Contrary to previous multiple hypothesis filters, our approach treats each observation individually by running a bank of UKFs in parallel. In doing so, the proposed approach incorporates the individual information introduced by each hypothesis. The main problem, however, is how to estimate the multiple observations. This is mainly due to the high dimensionality of the joint TDOA observation space corresponding to Q microphone pairs.

Ideally, we would like to use all possible TDOA combinations from different microphone pairs, weighted with their respective GCC values. As this Cartesian product is computationally intractable, we propose to reduce this set using a Monte Carlo (MC) scheme, which first, draws TDOA candidates from the individual GCCs and then combines these TDOAs in a *proximately consistent* fashion. In doing so, we statistically focus on TDOA combinations with high likelihood. In fact, this approach is based on the interpretation of the normalized GCC as a pdf of the TDOA, which we introduced in Section 4.2. It is worth mentioning here that a similar idea was originally proposed in [62] in combination with PF, and was later on applied in an SRP-based approach [61]. After sampling TDOA observations from all microphone pairs, we then approximate the joint pdf of the TDOA using an empirical distribution.

We will proceed by briefly reviewing the MH-GMF [105] in Section 7.1.2.1. Then, we show how this filter can be applied to solve the speaker tracking problem in Section 7.1.2.2, followed by a description and analysis of our experimental setup and results in Section 7.1.2.5.

7.1.2.1 MH-GMF Formulation

KF was originally designed to receive a single observation y_t at time t. In many applied tracking scenarios, however, there are several (K) potential observation candidates $y_t = \left\{y_t^1, \dots, y_t^K\right\}$ available, some of which may be due to the object of interest, whereas others may be due to clutter such as noise, reverberation, etc. This problem is typically treated by taking the single most likely observation or by combining multiple observations in a weighted sum, as it is done in PDAF [91, 106]. As an alternative to considering a single observation, one can use a MH-GMF [105]. This filter treats the multiple observation problem by (see illustration in Figure 7.1):

1. Splitting each KF at time t into K filters. K is the number of observations at time t.
2. Assigning each of the resulting filters to one of the observations.
3. Updating these filters independently as it is done in the classical single observation case. This step leads to a posterior distribution $p(x_t|y_{1:t-1}, y_t^k)$ for each filter k, where y_t^k is the observation at time t, which was assigned to the k^{th} filter.

In doing so, MH-GMF essentially explores multiple, concurrent hypotheses in parallel. Thus, it is expected to increase the tracking robustness to noisy estimates.

In order to integrate the K resulting conditional distributions $p(x_t|y_{1:t-1}, y_t^k)$ in one posterior, $p(x_t|y_{1:t})$ can be written as a marginal distribution of $p(x_t, k|y_{1:t})$, which can be further expanded, under use of $p(x_t, k|y_{1:t}) = p(x_t|k, y_{1:t})p(k|y_{1:t})$, to:

$$p(x_t|y_{1:t}) = \sum_{k=1}^{K} \underbrace{p(x_t|y_t^k, y_{1:t-1})p(k|y_{1:t})}_{=p(x_t,k|y_{1:t})}. \tag{7.3}$$

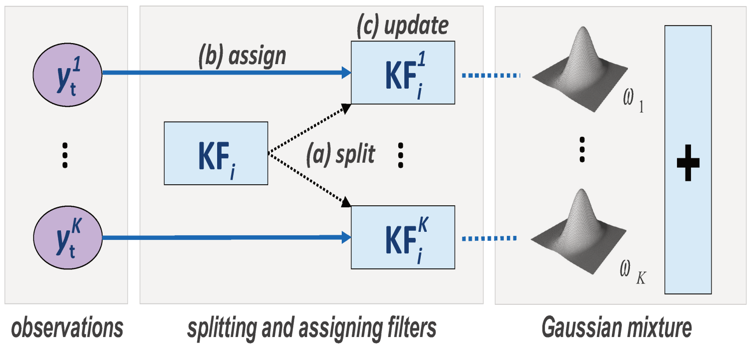


Figure 7.1: *Handling multiple observations with a Kalman filter (KF$_i$).*

This is a Gaussian mixture distribution in which the individual posteriors $p(x_t|y_t^k, y_{1:t-1}) = p(x_t|k, y_{1:t})$ are the Gaussian distributions, whereas $p(k|y_{1:t})$ constitute the corresponding mixture weights. The latter can be obtained with Bayes rule:

$$p(k|y_{1:t}) = \frac{p(y_t|k, y_{1:t-1})p(k|t)}{\sum_{j=1}^{K} p(y_t|j, y_{1:t-1})p(j|t)}, \tag{7.4}$$

$p(k|t)$ is the prior observation probability, which accounts for the confidence or certainty that we put into the kth observation (similarly to [62]). Moreover, $p(y_t|k, y_{1:t-1}) = p(y_t^k|y_{1:t-1})$ are observation likelihoods, which can be evaluated by marginalizing the joint predictive distribution $p(x_t, y_t|y_{1:t-1})$ from step two of the KF with respect to x_t.

As each of the filters in the MH-GMF is split into K filters at each time t, the number of Gaussians in general grows exponentially in time. Hence, we reduce the number of components in the mixture after each iteration by merging Gaussians successively in pairs [105].

Contrary to other tracking filters, MH-GMF treats each observation independently and assigns to it a weight which reflects its *importance* in the updated Gaussian mixture. In doing so, this filter allows us to propagate the observations uncertainty to the tracking stage, as well as incorporate the individual information introduced by each observation. In the following, we propose to apply this filter to the single speaker tracking problem. To do so, we propose a sampling scheme, which captures the uncertainty of the TDOA estimates and propagates it to the tracking stage.

7.1.2.2 MH-GMF-based Single Speaker Tracking

In the Kalman-based solutions proposed in [63, 64], the most likely TDOA is determined individually for each microphone pair. These individual TDOA estimates are subsequently combined to form a joint measurement $y_t = [\hat{\tau}^1, \ldots, \hat{\tau}^Q]$, where the error is assumed to follow a Gaussian distribution. This assumption may be true under ideal conditions. In practice, however, the errors in the GCC functions, i.e., TDOA measurement errors, are expected to follow a multi-modal distribution due to reverberation and background noise [62]. Hence, we propose here to:

1. Consider a larger number of TDOA observation candidates $\{y_t^k\}_{k=1}^K$, that we will refer to as *hypotheses*, with associated confidence weights $\{\gamma_t^k\}_{k=1}^K$.
2. Process these weighted observations using the MH-GMF, described in Section 7.1.2.1, where KF is replaced by UKF due to the non-linearity of the measurement model.

The aim of this procedure is to propagate the uncertainty from the TDOA estimation to the tracking stage, by choosing the weighted observation candidates in such a fashion that they capture the observation uncertainty in the GCC functions. To achieve this, let us first consider the time-independent observation space $\mathcal{Y}$, which can be approximated by the Cartesian product of all possible TDOAs from Q different microphone pairs:

$$\mathcal{Y} = \{y^1, \ldots, y^K\} \triangleq \bigtimes_{q=1}^{Q} \{-\tau_{\max}^q, \ldots, \tau_{\max}^q\}, \tag{7.5}$$

with $y^k = \left[\tau_k^1, \ldots, \tau_k^Q\right]$. $\tau_{\max}^q$ denotes the maximum TDOA of microphone pair q. Note that the $\{y^k\}_k$ here do not have a time index t as they are theoretical combinations, which are independent of time. K is the cardinality of $\mathcal{Y}$. Then, interpreting the GCC as a likelihood function, as was done in [61] for the SRP, and further assuming that the errors in the GCCs are statistically independent [62], the confidence or prior observation likelihood of a particular combination y^k can be calculated as the product of the individual GCC values $R^q(\tau_k^q)$ according to:

$$\gamma_t^k = \prod_{q=1}^{Q} \hat{R}^q(\tau_k^q) \quad \text{with} \quad \hat{R}^q(\tau) \triangleq \frac{R^q(\tau)}{\sum_{\tau'} R^q(\tau')}, \tag{7.6}$$

where the division by $\sum_{\tau'} R^q(\tau')$ normalizes the total probability to 1. This gives us the following observation distribution:

$$p_{\text{measured}}(y_t) = \sum_{k=1}^{K} \gamma_t^k \delta\left(y_t - y^k\right), \tag{7.7}$$

where y^k and γ_t^k are given by (7.5) and (7.6), respectively. As a next step, we could now pass this density to the multiple hypothesis filter from Section 7.1.2.1. But, considering the fact that the Cartesian product results in $K = \prod_{q=1}^{Q} \left(2\tau_{\max}^q + 1\right)$ different combinations, this approach has to be dismissed as intractable. Hence, we reduce the number of observations by approximating (7.7) through a sampling scheme, which samples observations from the high likelihood regions of the observation space in the next section.

7.1.2.3 Estimation of Multiple Observations

In order to obtain a set $\{y_t^1,\dots,y_t^{K'}\}$ of $K' \ll K$ observations from (7.7), we first draw K' TDOA samples from each normalized GCC $\hat{R}_q$ using sampling techniques, e.g., multinomial or importance sampling, and then combine the resulting τ_k^q, from different microphone pairs, to form K' observations $y_t^k = \left[\tau_1^q,\dots,\tau_k^Q\right]$. As a result of sampling, the weights γ_t^k are set to $1/K'$.

Sampling techniques ensure that we draw more TDOAs from regions of high likelihood, i.e., GCC peaks, and less TDOAs from regions of low likelihood, i.e., GCC valleys. So, we statistically focus on combinations y_t^k where the observation probability is high. In fact, this property of sampling is a key factor behind the success of MC methods in general.

The main drawback of this reduction technique, however, consists in sampling the observations τ_k^q from the GCCs of different microphone pairs independently. Therefore, combining these samples randomly may lead to inconsistent observations. By that we mean TDOA combinations $\left[\tau_k^1,\dots,\tau_k^Q\right]$ which do not correspond to a real possible location. In order to alleviate this problem, the filter's predicted observation likelihood $p(y_t^k|y_{1:t})$ is used as an approximate measure of consistency. This motivates the idea of combining the independently drawn τ_k^q in such a fashion that the total observation likelihood is maximized. This is done by:

1. Selecting from each sampled set $\{\tau_1^q,\dots,\tau_{K'}^q\}$, corresponding to microphone pairs q, the TDOA sample $\tau_{k_q}^q$ with the highest projected observation likelihood $p(\tau_{k_q}^q|y_{1:t-1})$.
2. Combining these samples to form a new observation $y_t^k = [\tau_{k_1}^q,\dots,\tau_{k_Q}^q]$.
3. Removing $\tau_{k_q}^q, q = 1,\dots,Q$ from the respective sample sets.

This procedure is repeated until all samples are combined.

7.1.2.4 Voice Activity Detection and Gating

Tracking of acoustic sources can be performed only on speech frames. Therefore, the voice activity detector from [99] is used to discard silence frames, which can cause the filter to diverge. As a further precaution against outliers, the above sampling scheme is restricted to some regions of interest through the integration of gating [106]. This is achieved by:

1. Merging all the predicted observation densities of the proposed MH-GMF into a single Gaussian $p(y_t|y_{1:t-1}) = \mathcal{N}(y_t,\mu,\sigma^2)$.
2. Defining a gating area $\mathcal{G}^q \triangleq \left\{\tau^q \mid \frac{(\tau^q-\mu^q)^2}{(\sigma^q)^2} \leq \psi\right\}$ for each microphone pair q.
3. Sampling the TDOAs τ_k^q from the *gated* GCC-based pdf:

$$\bar{R}^q(\tau^q) = \frac{R^q(\tau^q)\cdot I_{\mathcal{G}^q}(\tau^q)}{\sum\limits_{\tau'=-\tau_{\max}}^{\tau_{\max}} R^q(\tau')\cdot I_{\mathcal{G}^q}(\tau^q)}. \tag{7.8}$$

In these equations, ψ denotes the gating threshold and $I_{\mathcal{G}^q}(\tau^q)$ denotes the indicator function, which is equal to 1 if $\tau^q \in \mathcal{G}^q$ and 0 otherwise.

7.1.2.5 MH-GMF Evaluation: Experimental Setup and Results

In order to evaluate the performance of the proposed algorithm, we performed a set of tracking experiments on the AV16.3 corpus. Section 2.3.1 presents a detailed description of this corpus and the different recordings it offers. In the evaluation below, we report single speaker tracking performance of two types of scenarios, namely, the highly non-stationary scenario, in which a single speaker is quickly moving in the room. This scenario is represented by *seq11-1p-0100*. The goal of this evaluation is to assess the ability of tracking filters to follow the rapid changes in the speaker location, especially that the adopted DSSM is a random walk model, which favors stationary or slowly moving speakers. The second evaluation scenario is the relatively stationary case, in which a speaker is moving through 16 predefined locations while uttering one sentence at each of the positions. This scenario is represented by sequence *seq02-1p-0000*. The goal of this evaluation is to test the capability of the filters to recover and resume the tracking of a speaker who changes his/her position while being silent. These two sequences are 32 and 185 seconds in length, with an average speaker-array distance of 1.18m and 1.53m, respectively, where the minimum distance is 0.57m and the maximum distance is 2.40m. Note that the tracking performance on the remaining single speaker recordings confirm the conclusions we report below. Therefore, we only report results of one sequence for each scenario.

The signal processing is the same as the one we presented and used in all experiments conducted in Chapter 5, with the main difference of using a frame size of 1024 samples (64ms) instead of 512. This change is due to the drastic degradation of GCC-based TDOA estimation when using very short windows. Furthermore, GCC interpolation did not improve the results.

Regarding the tracking setup, MH-GMF uses $K' = 20$ observations, whereas PFs use 100 particles. A larger number of particles did not improve the results. As baseline models, we compare the proposed MH-GMF-based single speaker tracking to:

- The single observation Unscented Kalman Filter (UKF) approach proposed in [63]: This algorithm proposes to use an UKF to counteract the non-linearity of the measurement model. Note that the proposed MH-GMF will be reduced to this approach in the case where $K' = 1$. Thus, MH-GMF can be seen as a generalization of this approach.

- The single observation UKF with gating [64, 106]: This comparison will help us investigate the impact of gating mechanism on the robustness of TDOA measurement to noise.

- The single observation Sequential Importance Resampling Particle Filter (SIR-PF): Combined with SRP [46], this method was proposed as solution to the acoustic source tracking problem [61]. We use this method here as a TDOA-based approach.

- The Multiple Hypothesis Auxiliary Particle Filter (MH-PF) proposed in [62]: This approach proposes to counteract the multi-modality of the GCC function using a multiple hypothesis scheme. This is achieved by considering multiple peaks, and then pass these observations to a multiple hypothesis filter. Thus, MH-PF can be seen as an alternative to MH-GMF when considering multiple hypotheses.

The results are reported in terms of the average Root Mean Square Error (RMSE) and real-time factor, i.e., the processing time divided by the duration of the recording, including silence.

Tracking Approaches	Root Mean Square Error			Real-time Factor
	Azimuth	Elevation	DOA	
UKF	5.56°	15.98°	16.92°	0.336
SIR-PF	4.80°	10.33°	11.40°	0.374
UKF + Gating	4.17°	7.12°	8.24°	**0.329**
MH-PF	3.72°	5.94°	7.00°	0.582
MH-GMF	**2.85**°	**4.25**°	**5.11**°	0.664

(a) Sequence *seq11-1p-0100*: quickly moving speaker

Tracking Approaches	Root Mean Square Error			Real-time Factor
	Azimuth	Elevation	DOA	
UKF	8.66°	19.28°	21.14°	0.410
SIR-PF	7.54°	19.57°	20.98°	0.432
UKF + Gating	2.71°	8.14°	8.58°	**0.329**
MH-PF	3.99°	6.44°	7.58°	0.680
MH-GMF	**2.71**°	**4.07**°	**4.89**°	0.793

(b) Sequence *seq02-1p-0000*: stationary speaker at 16 different locations

Table 7.1: Average RMSE for azimuth, elevation and DOA with respect to the center of the array. The last column shows the real-time factor, i.e., the processing time divided by the duration of the audio recording.

Table 7.1 reports the performance of different tracking filters. As a first conclusion, we can clearly see that increasing the robustness of the filters to noisy TDOA estimates is crucial for improving the tracking performance. More precisely, we can see that the usage of gating drastically improves the performance of the UKF. Replacing this mechanism, which uses a hard approach to reduce the TDOA space, by a soft approach, which uses weighted multiple hypotheses instead leads to a better improvement of the tracking performance.

In a second conclusion, we can say that the proposed MH-GMF performs significantly better than all other methods. That is, its angular error (DOA) is 69% and 79% lower than that of the UKF [63]; 38% and 43% lower than that of the UKF with Gating [106]; and still 27% and 35% lower than that of the MH-PF from [62]. Regarding these results, it should be noted that the main problem of the small aperture microphone arrays, with a planar geometry, consists in obtaining good estimates of the elevation. But, having a closer look at Table 7.1, we can see that it is exactly where our method shows its true strength. If we consider the *Anomaly Rate* (AR); which represents the percentage of estimates with an error $\geq 5°$. We have obtained an AR of 4.79% and 8.34%, for the sequences *seq11-1p-0100* and *seq02-1p-0000*, respectively, and an AR of 8.50% for the sequence *seq01-1p-0000* from the second scenario, this sequence has been reported to have an AR of 29.10% using a different localization (without tracking) algorithm [45]. This result shows that using a tracking approach in combination with a TDOA-based localization technique improves significantly the RMSE, as it smooths out the erroneous estimates.

In terms of real-time implementation, the time factors in Table 7.1, show that all methods run faster than real-time on a standard Intel i7-2600K CPU clocked at 3.4GHz. The plain UKF is roughly 2 times faster than the proposed MH-GMF; and that although the latter runs more than 20 UKFs in parallel. This indicates that most of the computation time is spent in computing the GCC functions and the TDOA estimation.

7.1.3 Improving TDOA Estimation using Tracking Information

As we have discussed earlier in this chapter, increasing the robustness of the Bayesian filters to noisy TDOA estimates is the bottleneck to improving the performance of TDOA-based tracking approaches. While the MH-GMF solution proposes to use multiple observations with confidence scores to tackle this problem, we propose in this section a more general, filter-independent alternative to achieve this purpose.

Previous works, though not directly related to the acoustic source tracking problem, have shown that the use of prior information about the measurements can efficiently improve the observation detection [106], e.g., the *gating* mechanism used in the MH-GMF experiments. Along this line, we present in this section an approach to update the GMM, approximating the GCC function, through use of information that has been obtained in the tracking stage. That is, we propose an approach that enhances the TDOA estimation stage using tracking information, available at each time frame, following these steps:

1. Accounting for the multi-modal aspect of the GCC function resulting from noise and reverberation. To achieve that, we use the GMM approximation of the GCC introduced in Section 4.2.

2. Integrating the speaker knowledge which has been obtained in the tracking stage. In particular, we estimate the pdf of the TDOA, which is expected by the tracking filter at the current time instant, in a first stage, and then use it to update the mixture weights of the above GCC-based GMM. This is done by measuring the *similarity* between each component in the mixture and the predicted pdf of the TDOA. This similarity score reflects the probability that the component generates the true TDOA observation.

3. Extracting the TDOA observations using the estimators described in Section 4.2.4.

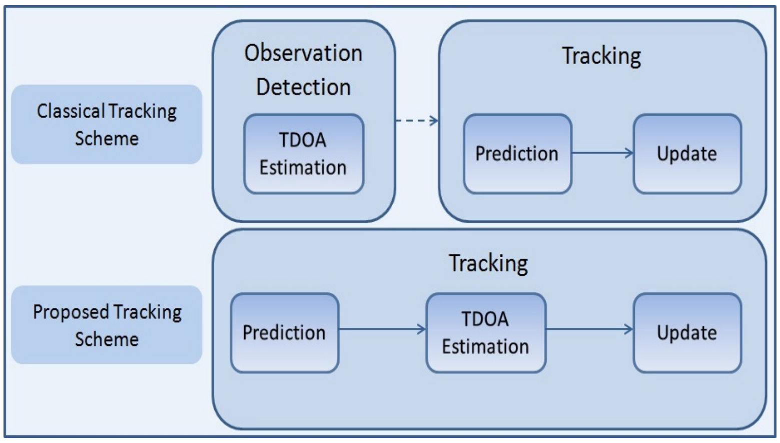


Figure 7.2: *Block diagram comparing the classical tracking scheme to the proposed one. The latter uses the predictive distribution of the tracking filter to improve the TDOA estimation stage.*

In doing so, this approach tightly couples the TDOA estimation stage and the tracking stage. Figure 7.2 shows a schematic view of the different interactions in the proposed tracking approach. The remaining part of of this section is organized as follows, Section 7.1.3.1 shows how tracking information can be integrated into the TDOA estimation stage, whereas Section 7.1.3.2 presents the evaluation of this approach.

7.1.3.1 Update of the TDOA-GMM using Tracking Information

We introduce in this section the mathematical formulation of the proposed tracking-based TDOA estimation. To do so, let $\mathcal{M}^q = \{\mathcal{N}_k^q\}_{k=1}^{K_q} = \{p_k^q, \mu_k^q, \sigma_k^q\}_{k=1}^{K_q}$ be the GM approximating the GCC function of microphone pair $q \in \mathcal{Q}$, obtained using the approach proposed in Section 4.2, and let $\mathcal{N}_p^q$ be the TDOA predicted pdf, which is expected by the tracking filter. This is obtained by marginalizing the joint predicted distribution, obtained in Step 2 of the Bayesian estimation approach described in 3.4, on the state space. For ease of notation, the microphone pair index q is dropped in the rest of this section.

The first step is to calculate the *similarity* between each component in the GM distribution and the predicted pdf. For this purpose, we use two different Similarity Measures (SMs):

$$SM_{\mathrm{KLD}}(\mathcal{N}_p, \mathcal{N}_k) = \frac{1}{1 + \mathrm{KLD}\,(\mathcal{N}_p \,||\, \mathcal{N}_k)}, \tag{7.9}$$

$$SM_{\mathrm{BC}}(\mathcal{N}_p, \mathcal{N}_k) = \int \sqrt{\mathcal{N}_p(x)\,\mathcal{N}_k(x)}\,\mathrm{d}x, \tag{7.10}$$

KLD $(\mathcal{N}_p \,||\, \mathcal{N}_k)$ is the *Kullback-Leibler Divergence* (KLD) between the two Gaussians, whereas the second SM is the *Bhattacharyya Coefficient* (BC) [107]. These two SMs have closed form solutions for Gaussian distributions.

After calculating the similarity scores for each component in the TDOA-GMM, we then update the mixture weights of the GM before estimating the TDOA. The new weight $\bar{p}_k$ of the k^{th} component is given by:

$$\bar{p}_k = \frac{p_k \cdot SM(\mathcal{N}_p, \mathcal{N}_k)}{\sum_{i=1}^{K} p_i \cdot SM(\mathcal{N}_p, \mathcal{N}_i)}. \tag{7.11}$$

The update step smooths out the unlikely mixture components and enhances the ones which are close from the predicted TDOA. Unlike gating [106], this method assigns a weight to each component instead of reducing the measurement space to some regions of interest. This step can be seen as a *correction* of the GMM. Furthermore, this approach is independent of the used tracking filter, which makes it a more attractive choice compared to other solutions to the noisy TDOA measurements problem. Figure 7.3 illustrates the efficiency of the proposed method. The maximal GCC peak corresponds to a TDOA of -9 samples. The use of SM, however, alleviates the estimation error and recovers the true TDOA, which is 8.2 samples.

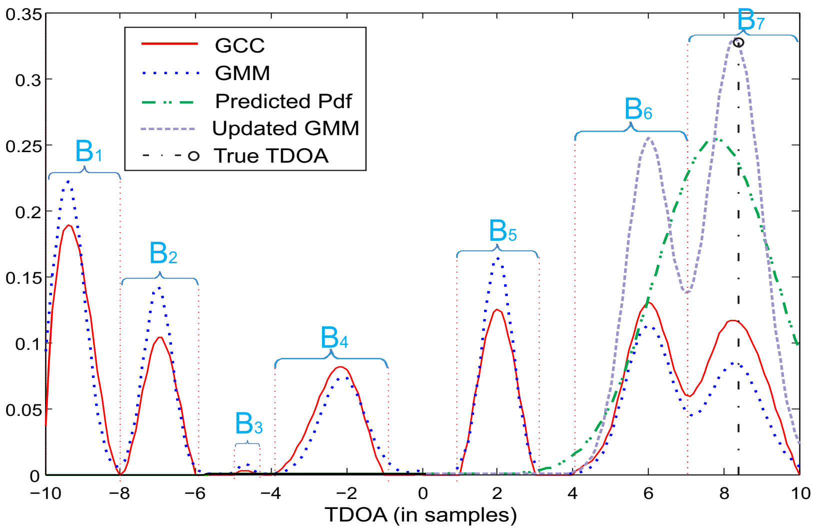


Figure 7.3: *Illustration of the TDOA-GMM and the Gaussian similarity measure (K=7).*

After the GMM update, the TDOA estimation can be then performed using either the *maximum estimate* given by (4.24) or the *mean estimate* introduced in (4.25). The observation vector is constructed by estimating the TDOA $\bar{\tau}_t^q$, for each microphone pair q, and then combine these individual estimates to form a joint measurement $y_t = [\bar{\tau}_t^1, ..., \bar{\tau}_t^Q]$ for each frame t.

7.1.3.2 Experimental Setup and Results

In order to evaluate this approach, we extended the experiments conducted during the evaluation of the MH-GMF to include the proposed similarity measure, which is expected to improve the TDOA estimation. That is, we report the performance of each of the previous tracking filters from Table 7.1 with and without similarity measure. In order to apply this similarity scheme to the multiple hypothesis filters, we replace the multiple GCC peaks considered in the MH-PF by the means of the GM, whereas the mixture weights are used as hypotheses likelihood. Regarding the MH-GMF, the observation detection scheme is replaced by a random selection approach, where the mixture weights are used to randomly draw a component from the GM of each microphone pair and then combine the means to form a joint observation. This process is repeated until we reach the desired number of observations K'.

The evaluation setup is the same as the one described in Section 7.1.2.5. When using the similarity measure, the latter is replaced by gating [106] in the first frames and is used only after a duration T. This precaution is mainly added to ensure that the filter is locked on the true source before using the tracking information. The baseline performance for these experiments is given by Table 7.1. In addition to the tracking performance, we also report the average RMSE of the TDOA measurements over all microphone pairs, this measure will help us evaluate the direct impact of the similarity scheme on the measurement estimation stage. Furthermore, the results reported below were obtained using the second SM, i.e., BC. The use of KLD leads to a similar performance.

Table 7.2 clearly shows that the integration of the prior information about the measurements into the detection stage, through gating or through the proposed approach denoted as“SM”, improves the TDOA estimation and thereby the tracking performance. The results also show that the proposed approach improves the performance of almost all tracking filters, except the MH-GMF when it is applied to sequence *seq11-1p-0100.* This exception is due to the measurement model (7.2), which assumes that the source is stationary, whereas the speaker in this sequence is quickly moving. We can also conclude that the use of this approach is more relevant with single observation tracking algorithms, where the DOA error is 66% and 73% lower for the UKF and 46% and 70% lower for the SIR-PF. This compares to 17% and 16% improvement for the MH-PF and only 4% for the MH-GMF when it is applied to sequence *seq02-1p-0000.* This difference in the improvement was expected given that that the multiple hypothesis filters propose to overcome the multi-modality problem by considering multiple peaks with equal weights, whereas the SM assigns a likelihood weight to each Gaussian before estimating the observations, and thereby, improves the TDOA estimates. We can also notice that, with SM, the performance of the single observation filters, which are computationally more efficient, is close to the performance of the multiple observations filters. This makes the former more attractive in practice.

Table 7.2 also shows that the reason behind this improvement is the reduction of the TDOA root mean square error, which is ≈ 0.63. This value is compared to the inherent 0.5 samples precision error due to the GCC method. Although this could be slightly improved through GCC interpolation, the improvement that we obtained from this additional step is negligible.

Tracking Approaches	Root Mean Square Error			
	Azimuth	Elevation	DOA	TDOA
UKF	5.56°	15.98°	16.92°	2.01
UKF + Gating	4.17°	7.12°	8.24°	1.05
UKF+SM	2.97°	4.92°	5.74°	**0.64**
SIR-PF	4.80°	10.33°	11.40°	2.01
SIR-PF+SM	3.29°	5.12°	6.09°	**0.64**
MH-PF	3.72°	5.94°	7.00°	—
MH-PF+SM	3.25°	4.81°	5.80°	—
MH-GMF	**2.85°**	**4.25°**	**5.11°**	—
MH-GMF+SM	3.21°	5.07°	5.99°	—

(a) Sequence *seq11-1p-0100*: quickly moving speaker

Tracking Approaches	Root Mean Square Error			
	Azimuth	Elevation	DOA	TDOA
UKF	8.66°	19.28°	21.14°	2.33
UKF + Gating	2.71°	8.14°	8.58°	0.99
UKF+SM	2.83°	5.11°	5.84°	0.64
SIR-PF	7.54°	19.57°	20.98°	2.33
SIR-PF+SM	2.97°	5.46°	6.20°	**0.62**
MH-PF	3.99°	6.44°	7.58°	—
MH-PF+SM	3.32°	5.42°	6.36°	—
MH-GMF	2.71°	4.07°	4.89°	—
MH-GMF+SM	**2.60°**	**3.86°**	**4.65°**	—

(b) Sequence *seq02-1p-0000*: stationary speaker at 16 different locations

Table 7.2: Average root mean square error (RMSE), with and without Similarity Measure (SM), for azimuth, elevation and direction of arrival (DOA). The last column shows the average RMSE of the TDOA of 18 microphone pairs, which is calculated only for the single observation filters.

7.2 SRP-based Multiple Speaker Tracking

Multiple speaker tracking approaches are generally designed to overcome few classical problems of multiple object tracking, such as the non-linearity of the state space model dynamics [60, 63, 66], the robustness to noise [17, 68] and the correct estimation of the number of speakers [69]. These approaches, however, do not address two main problems related to the speech nature, namely:

1. The high discontinuity of spontaneous speech, where an active speaker becomes frequently inactive for a short time (100 to 300 ms). See illustration in Figure 7.4.

2. The suppression problem, where the dominant source masks the remaining speakers.

These two problems reduce the speaker detection rate, and thereby makes the tracking of acoustic sources possible *only in short-term*, i.e., while a speaker is talking without being suppressed. Figure 7.4 shows an example where a multiple speaker detector fails in producing location measurements for a short period of time due to the discontinuity of spontaneous speech, which can cause a standard multiple speaker tracking filter to lose track of the speaker.

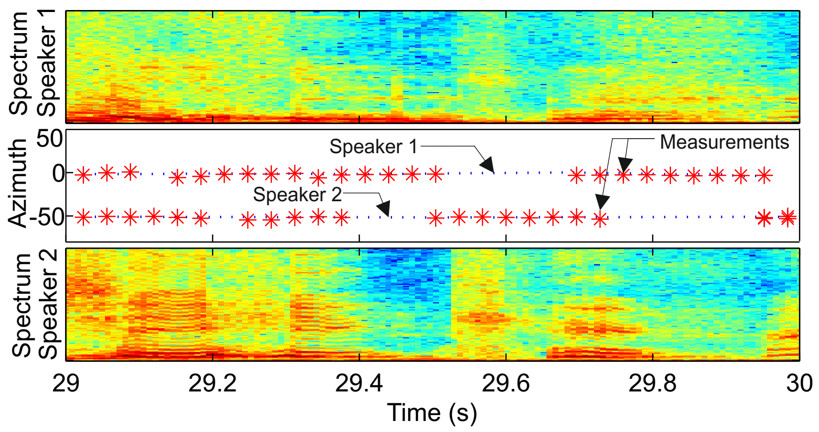


Figure 7.4: *An example of a spontaneous speech segment, where the P-SRP-based instantaneous location detector fails in producing location measurements (stars) during short silence/low energy frames.*

To overcome this problem, the authors of [70] proposed a Short-Term Clustering (STC) approach, which extracts the speaker trajectories as short-term location clusters. Motivated by this work, this section introduces a new multiple speaker tracking framework.

The remaining part of this chapter introduces a novel multiple speaker short-term tracking framework. We can summarize the contributions proposed in this framework as follows:

- The introduction of a more realistic multiple speaker tracking solution, which takes into account the inherent problems related to the speech nature discussed above. This is done using a dynamic bank of parallel KFs to track multiple concurrent speakers, where each speaker follows a temporal HMM. The latter is designed to account for the speech nature by allowing speakers to become inactive for a short period of time [24].

- A new approach that increases the detection rate of suppressed speakers in overlapping speech frames. This is done by extending the single speaker-based similarity measure approach, presented in Section 7.1.3, to the multiple speaker case [25].

7.2.1 SRP-based DSSM

To solve the single speaker tracking problem, presented in Section 7.1, we have considered TDOA-based tracking solutions due to their attractive low computation cost. Extending these approaches, however, to the multiple speaker case is not straightforward and highly depends on the TDOA estimator. Therefore, we propose to use in this case an SRP-like observation detector. That is, we use one of the multiple speaker localization approaches introduced in Chapter 5, combined with the speaker detection approach presented in Chapter 6, to extract multiple and noisy speaker locations, which will be subsequently used as observations in the tracking framework. Thus, the TDOA-based DSSM, introduced in Section 7.1.1, should be modified to account for this change. In particular, the measurement process, which establishes the physical connection between the observation (TDOA) and the state (location) can no longer be used in this case. Therefore, we propose to track the speaker location x_t using the following DSSM:

$$\textit{Process model:} \quad x_t = f(x_{t-1}, v_t) = x_{t-1} + v_t, \tag{7.12}$$

$$\textit{Measurement model}: \; y_t = h(x_t, w_t) = x_t + w_t. \tag{7.13}$$

x_t is the process state at time t, which can be either the DOA given by the azimuth and elevation or the azimuth alone. The latter is considered when the tracking of the elevation becomes unreliable due to the small, planar geometry of the microphone array. The proposed DSSM uses a random walk model for both, the process and measurement models. This assumption is reasonable given the short time frame that is considered in this work, i.e., 32ms. Section 7.1.1 provides a detailed discussion about the random walk model choice.

7.2.2 Multiple Speaker Tracking: Kalman Filter Bank

Multi-party spontaneous speech utterances can be looked at as a sequence of *sporadic* and *concurrent* events [70, 73]. More precisely, speech utterances are generally short and interspersed with many short silences, which results in a sequence of short and isolated segments of speech [70]. Furthermore, the sporadic nature of spontaneous speech increases in the multiple concurrent speaker scenario, where the dominant speaker suppresses the remaining speakers. This property automatically decreases the performance of classical tracking approaches. More precisely, these approaches often require that the object of interest is continuously observable over, relatively, a long period of time. This assumption is violated in the spontaneous speech case, where the instantaneous location estimates, obtained using P-SRP for instance, are often unavailable during silence frames and during speech segments with low energy (see example in Figure 7.4). Moreover, the fast-changing speaker turns and the varying number of active speakers encountered in multi-party speech require very complex models, which allow fast and concurrent transitions in the speaker turns, including the ability to account for overlapping speech when it occurs.

We present in this section a novel short-term filter, which incorporates these two characteristics. This is done using a Kalman Filter Bank (KFB) that 1) models the multiple concurrent speaker scenario, and 2) allows speakers to change their state (speaking, silent, etc.) according to an HMM that models:

1. The frequent and short transitions in a speaker state, e.g., silent, speaking, etc.

2. The time-varying number of speakers, by allowing new speakers to appear (birth state) and existing speakers to disappear (final state). In doing so, the proposed approach presents a more realistic and flexible model to the multiple speaker tracking problem.

This approach takes into account speech nature using short-term processing, similarly to [70], but proposes a more realistic model through use of a KFB, where each filter evolves according to an HMM that models all possible states of a speaker.

In the remaining part of this section, we will introduce the mathematical formulation of the proposed filter in Section 7.2.2.1. Then, Section 7.2.2.2 will show how it can be applied to the multiple speaker case, whereas Section 7.2.2.4 will demonstrate the effectiveness of the proposed multiple speaker tracking approach, in comparison to the STC model [70], on the AV16.3 corpus.

7.2.2.1 Short-Term Tracking (STT) Filter

The STT filter proposes to track multiple speakers using a dynamic bank of KFs running independently and in parallel. Each filter in this bank estimates a single speaker short-term trajectory using the SRP-based DSSM, described in Section 7.2.1, and the recursive Bayesian estimation framework introduced in Section 3.4. Furthermore, the state of each filter is updated according to a temporal HMM (Figure 7.5 is an illustration of the modeled state transitions). More precisely, a filter can be:

1. In the hidden *Birth* state (B): In this state, the filter is initialized to track potential emerging targets.

2. *Active* (A): This hidden state corresponds to filters that are tracking the current active targets in the scene. These include 1) speakers from the previous frame that remained active, 2) speakers that went inactive for a short period of time (100 to 300 ms) and became active again and 3) the new targets that just appeared in the scene.

3. *Inactive* (I): This hidden state models the short silence/break time frames as well as frames with low speech energy (see example in Fig. 7.4). This phenomenon causes a lack of measurements. Therefore, the filter becomes inactive.

4. *Dead* (D): This *final* state models filters that went inactive for a long period of time. This mainly occurs when speakers change turns or when a speaker stops talking. Filters that reach this state are automatically **removed** from the filter bank.

7.2.2.2 Multiple Speaker Tracking Framework

This section introduces the mathematical formulation of the multiple speaker short-term tracking framework. Let $\mathcal{B}_t = \{\mathcal{F}_{t,k}\}_{k=1}^{N_t}$ be a bank of N_t KFs running in parallel at time t. $\mathcal{B}_t$

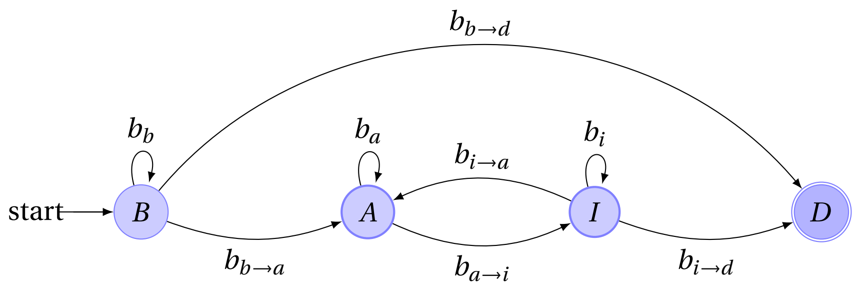


Figure 7.5: *Illustration of the filter state update at time t, given the observed filter activity $T^t_{a,k}$ and the possible HMM state transitions.*

can be divided into three disjoint banks according to each filter state:

$$\mathcal{B}_t = \{\mathcal{F}^a_{t,k}\}_{k=1}^{N_t^a} \bigcup \{\mathcal{F}^i_{t,k}\}_{k=1}^{N_t^i} \bigcup \{\mathcal{F}^b_{t,k}\}_{k=1}^{N_t^b}, \tag{7.14}$$

where $\mathcal{B}^a_t = \{\mathcal{F}^a_{t,k}\}_{k=1}^{N_t^a}$, $\mathcal{B}^i_t = \{\mathcal{F}^i_{t,k}\}_{k=1}^{N_t^i}$ and $\mathcal{B}^b_t = \{\mathcal{F}^b_{t,k}\}_{k=1}^{N_t^b}$ are the active, inactive and potential (new speakers) filter banks, respectively. N^a_t, N^i_t and N^b_t are their respective cardinalities. Let $\mathcal{B}_{t-1}$ be the filter bank at time $t-1$ and let x_t and y_t be the (location) state and observation random variables at time t, respectively. The goal here is to estimate the updated posterior distribution $p^k(x_t|y_{1:t})$ of each filter $\mathcal{F}_{t,k}, k = 1,\ldots,N_t$ in the filter bank $\mathcal{B}_t$ at time t. This time propagation of the posterior distribution is done in four steps:

- **Step 1**. State prediction step: This step uses the process model given by (7.12) to calculate the prior distribution $p^k(x_t|y_{1:t-1}), k = 1,\ldots,N_t$ of each filter $\mathcal{F}_{t,k} \in \mathcal{B}_t$.

- **Step 2**. Joint predictive distribution: In this step, we propagate the predicted prior distribution, calculated in the previous step, from the state space to the augmented joint state-observation space according to the measurement model given by (7.13). We obtain then N_t joint predictive distributions $p^k(x_t, y_t|y_{1:t-1}), k = 1,\ldots,N_t$. In fact, these two steps run the classical Bayesian tracking steps 1 and 2 from Section 3.4 on N_t parallel Kalman filters.

- **Step 3**. Confidence region estimation: For each filter $\mathcal{F}_{t,k}, k = 1,\ldots,N_t$, the joint predictive distribution $p^k(x_t, y_t|y_{1:t-1})$ is marginalized on the state space to obtain the predicted observation distribution $p^k(y_t|y_{1:t-1})$, which characterizes the most likely region to contain the next measurement. This distribution is then used to define the measurement confidence region $\mathcal{C}^k_t$ of the filter $\mathcal{F}_{t,k}$ according to:

$$\mathcal{C}^k_t = \text{Gate} = \left\{ Y_t \in \text{location space} \mid p^k(Y_t|y_{1:t-1}) \geq p_{confid} \right\}, \tag{7.15}$$

  p_{confid} is the confidence threshold (a probability).

- **Step 4**. Target-measurement association and filter bank update: Let $\mathcal{Y}_t = \{Y_t^1, \ldots, Y_t^{M_t}\}$ be the M_t measurements received at time t, and let $\mathcal{A}_{t,k}$ be the target-measurement binary random variable associated to $\mathcal{F}_{t,k}$. The measurement Y_t^m is associated to the target $\mathcal{F}_{t,k_m}$ ($\mathcal{A}_{t,k_m} = 1$) if and only if $Y_t^m \in \mathcal{C}_t^{k_m}$. Then, the corresponding posterior distribution $p^{k_m}(x_t|y_{1:t})$ is updated according to step 3 of the single object Bayesian tracking framework, which is presented in details in Section 3.4.

After the target-measurement association step, the observations, if there are any, $\bar{Y}_t^l, l = 1\ldots,\bar{N}_t$ that were not associated to any target are used to initialize potential new speakers. More precisely, $\bar{N}_t$ Gaussian distributions $\mathcal{N}(x_t, Y_t, \Sigma_{init})$, where the means are given by the observations $Y_t = \bar{Y}_t^l$, are added to the KFB $\mathcal{B}_t^b$. These filters are considered to be at the **birth** state of the HMM (see illustration in Figure 7.5).

7.2.2.3 Update of the Filters State

Once we propagate the posterior distribution of all filters in $\mathcal{B}_t$, we proceed to the update of each filter state according to the proposed HMM (see illustration in Figure 7.5). The new state of each filter is estimated based on its observed *activity* $t_{a,k}$, which is calculated over a history window of duration T_c.

Let L_f be the frame length in seconds. In order to define the activity of a filter $\mathcal{F}_{t,k}$ at time t, we first calculate its **active duration** $\Delta t_{a,k}$ and **inactive duration** $\Delta t_{i,k}$ according to:

$$\Delta t_{a,k} = L_f \cdot \left(\sum_{j=t-T_c}^{t} \mathcal{A}_{j,k} \right), \tag{7.16}$$

$$\Delta t_{i,k} = T_c - \Delta t_{a,k}. \tag{7.17}$$

Essentially, these two measures represent the total duration of frames on which the filter received an observation, or not, over the last T_c frames. In this case, the *filter activity* is defined and calculated according to:

$$t_{a,k} = \max\left(\Delta t_{a,k} - \Delta t_{i,k}, 0\right). \tag{7.18}$$

Now, let $T_{a,k}^t$ be the *observed activity* of the filter $\mathcal{F}_{t,k}$ at time t. Given the current state of the filter and following the proposed HMM, the new state of $\mathcal{F}_{t,k}$ is given by the HMM state which maximizes the transition probabilities:

$$\begin{aligned}
b_{b\to a} &= \begin{cases} 1, & \text{if } \int_0^{T_{a,k}^t} f_b(\theta_b, \mathbf{x})\, \mathrm{d}\mathbf{x} \geq p_{birth}, \\ 0, & \text{otherwise.} \end{cases} && (7.19)\\
b_a = b_{i\to a} &= \mathcal{A}_{t,k}, && (7.20)\\
b_{a\to i} &= 1 - \mathcal{A}_{t,k}, && (7.21)\\
b_b = b_i &= p_{survival} = \int_0^{T_{a,k}^t} f_s(\theta_s, \mathbf{x}) \cdot \mathrm{d}\mathbf{x}, && (7.22)\\
b_{i\to d} = b_{b\to d} &= p_{death} = 1 - p_{survival}, && (7.23)
\end{aligned}$$

$f_x(\theta_x, .)$ $(x \in \{b, s\})$ are two pdfs (with parameters θ_x) modeling the birth and survival processes, respectively. Following the classical use of the exponential pdf as distribution modeling the life duration of objects, these two pdfs are modeled by two exponential distributions with respective means μ_b and μ_s.

The update of the filters state according to the proposed HMM leads to a new bank of **active** filters $\mathcal{B}_t^a = \{\mathcal{F}_{t,k}^a\}_{k=1}^{N_t^a}$. Although $\mathcal{B}_t^a$ can be considered to be the final set of active speakers, the independent update of the filters leads to a high perturbation in the number of active filters over time, which is often undesirable. Therefore, we use the estimated number of **active** filters $\mathcal{B}_t^a$ as a measurement in a second KF to smooth the number of active speakers over time.

7.2.2.4 Evaluation of KFB-based STT

We evaluate the proposed approach using the AV16.3 corpus [71]. The signal processing setup is the same as the one used throughout this thesis, a detailed description of this setup was introduced in the evaluation section of the P-SRP approach. In the experiments reported below, the P-SRP approach, described in Section 5.4, was used as an instantaneous location estimator [18], which extracts a fixed number of potential locations $N_{max} = 6$, for each time frame. These estimates are subsequently classified into speaker/noise classes using the multiple speaker detection approach described in Chapter 6. The location estimates which are classified as speakers are then used as observations in the proposed STT approach to track multiple concurrent speakers. Moreover, the experiments reported below follow the evaluation framework described in Section 2.3.2. The latter derives different statistics from the Gaussian component representing the speaker estimates. More precisely, the results are reported in terms of:

- The precision rate p_s.
- The tracking rate t_r, this is calculated as the correct tracking duration w.r.t. the duration of frames with a (at least one) ground truth location.
- The individual speaker detection rate d_r.
- The average Root-Mean-Square Error (RMSE).

- The real-time factor t of the complete framework on a standard Pentium(R) Quad-Core i5-3550 CPU clocked at 3.30GHz.

Similarly to the work proposed in [70, 73], the tracking is limited to the azimuth angle. This is due to the far-field assumption as well as to the small size of the microphone array. The proposed approach, however, is general and can be applied to 3-D tracking problems with other types of microphone arrays, such as distributed arrays. The tracking parameters setting is as follows, the birth mean is set to $\mu_b = 0.3$s whereas $\mu_s = 0.1$s. The latter aims at excluding filters with a decreasing activity near to 0. The birth probability $p_{birth} = 0.8$, the confidence probability is $p_{confid} = 10^{-2}$, whereas the duration of the context/history window is $T_c = 1$s.

	seq11-1p		seq18-2p		seq24-2p		seq40-3p		seq37-3p	
	STC	STT	STC	STT	STC	STT	STC	STT	STC	STT
Precision rate p_s	87.9	**92.2**	85.0	**99.0**	**81.6**	81.1	94.1	**94.3**	90.6	**94.3**
Tracking rate t_r	69.8	**78.4**	81.5	**90.4**	63.7	**66.8**	75.7	**86.6**	82.2	**84.2**
Time factor t	33.4	**4.8**	42.0	**4.7**	32.0	**4.7**	37.8	**4.8**	36.4	**4.7**

(a) Tracking performance in terms of the precision rate p_s, trajectory estimation rate t_r and real-time factor t.

	seq11-1p		seq18-2p		seq24-2p		seq40-3p		seq37-3p	
	STC	STT	STC	STT	STC	STT	STC	STT	STC	STT
d_r of Speaker 1	69.8	**79.8**	53.1	**61.1**	54.9	**59.0**	39.2	**49.7**	28.8	**29.9**
d_r of Speaker 2	—	—	51.6	**54.8**	34.3	**37.9**	38.4	**40.0**	66.2	**71.0**
d_r of Speaker 3	—	—	—	—	—	—	56.8	**62.6**	**46.7**	40.2
Avg. d_r	69.8	**79.8**	52.3	**58.0**	44.6	**48.4**	44.8	**50.8**	**47.9**	47.0
Avg. RMSE (°)	2.90	**2.81**	**1.96**	2.34	3.07	**3.04**	6.56	**4.70**	2.47	**2.30**

(b) Tracking performance in terms of speaker detection rate d_r and the average root-mean-square error (degree).

Table 7.3: Speaker tracking performance on different recordings from the AV16.3 corpus, namely, recordings with one speaker: *seq11-1p-0100*, with two speakers: *seq18-2p-0101* and *seq24-2p-0111* and with three simultaneous speakers: *seq40-3p-0111* and *seq37-3p-0001*.

Table 7.3 presents the performance of the proposed STT approach on different sequences from the AV16.3 corpus, and compares it to the complete STC framework proposed in [70, 73]. This framework consists of 1) an instantaneous detection-localization approach, followed by 2) an automatic threshold that controls the false alarm rate. The obtained estimates are then 3) clustered into speech utterances using a short-term clustering approach. Finally, 4) a speech/non-speech classification is performed to discard estimates from non-speech frames (more details about this framework can be found in [73]). The STC results were generated using the publicly available STC toolkit [73], using the same parameter setting introduced above.

Table 7.3a shows a clear improvement of STT over the STC approach. More precisely, STT achieves longer correct tracking trajectories, i.e., the increased correct tracking duration rate t_r, while achieving a comparable precision rate p_s. Moreover, the time-factor t shows that STT is 7 to 8 times faster than STC. We can also conclude from this table that the proposed approach achieves a very satisfying tracking rate, i.e., average $t_r \approx 85\%$, and that it mostly tracks the correct acoustic sources shown by the average $p_s \approx 88\%$.

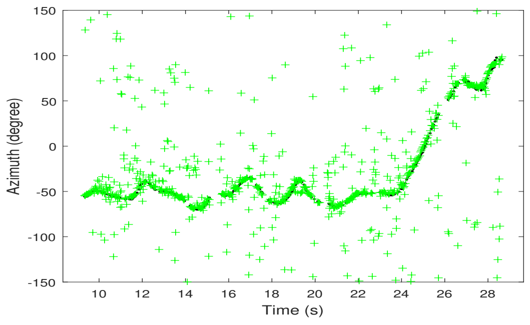
Table 7.3b analyzes the distribution of the precision p_s and the tracking rate t_r performance from Table 7.3a on the individual instantaneous speakers. We can see clearly that STT significantly increases the speaker detection rate d_r without compromising the RMSE, which is comparable for both approaches. We can also see that for sequence *seq24-2p-0111*, which contains very long and frequent intentional segments of silence, the performance of STT decreases and becomes comparable to the performance of STC. This is due to the absence of a speech/non-speech feature that uses speech cues to reject noise estimates during long silence/noise frames. As a result, STT tracks noise sources during these long segments of silence/noise. STC, however, integrates such cues through the usage of MFCCs as classification features. Table 7.3b also shows that the detection rates d_r of the multiple speaker sequences are low compared to the corresponding tracking rate t_r. This is mainly due to the degradation of the simultaneous speaker detection caused by the speaker suppression problem, as well as the high active/inactive transition rate. The next section will present a new solution to overcome this problem.

Figure 7.6 shows the corresponding STT-based azimuth tracking. We can, once again, conclude from this figure that the STT filter mostly tracks the correct speakers and that it properly processes the noisy estimates and clutter. We can also see that the filter is able to track multiple speakers on overlapping speech segments. It is also clear that some of the *noise* trajectories are, in fact, actual speaker locations but the corresponding audio segments are missing the ground truth location. Thus, the filter could not associate them to any speaker during evaluation. This missing information is due to the fact that the ground truth was estimated using visual tracking, which could not be performed on video segments with occlusions or when the speakers leave the predefined *speakers area.* More details about the ground truth evaluation approach can be found in [53].

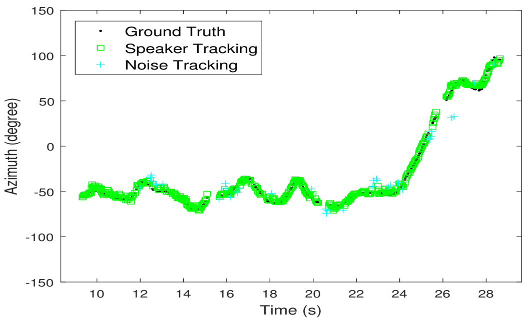
7.2.3 Improving Overlapping Speaker Detection using Tracking Information

The evaluation of the proposed STT approach has shown that we can achieve a good tracking performance with a satisfying precision. This approach, however, similarly to all multiple object tracking solutions, cannot recover the suppressed speakers, which are generally masked by another dominant source. As a result, these approaches lead to a low overlap detection rate. The latter can be very crucial to solve many problems, such as speech separation/enhancement, multi-party speech recognition and speaker diarization [108, 109].

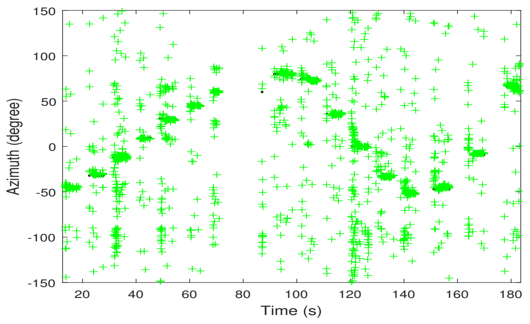
The low overlap detection rate is mainly due to the fact that the multiple object tracking performance is highly dependent on the measurement detection rate, which drastically decreases in multi-party conversational/spontaneous speech. More precisely, in overlapping speech segments, the dominant speaker tends to mask, and therefore suppress, the secondary speakers causing the measurement detector to fail in detecting multiple instantaneous locations. This problem becomes more complex in noisy and/or highly reverberant environments, where the



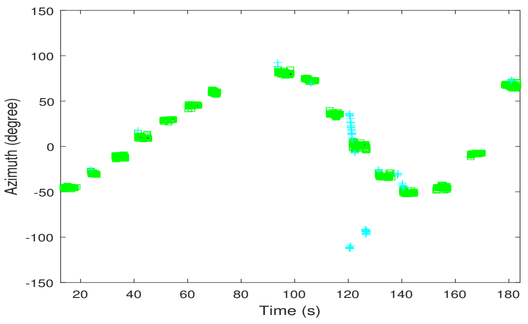
(a) *seq11-1p-0100*: Location measurements



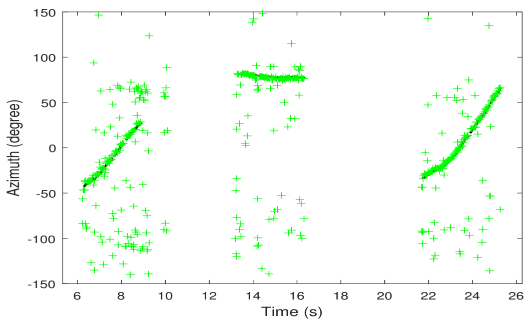
(b) *seq11-1p-0100*: Single speaker tracking



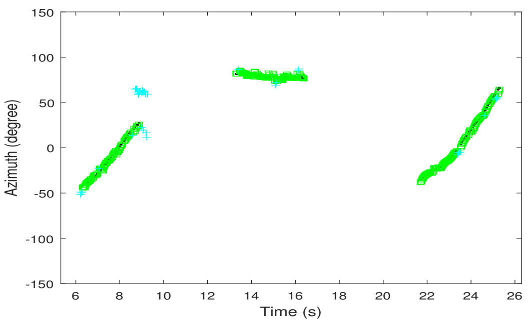
(c) *seq02-1p-0000*: Location measurements



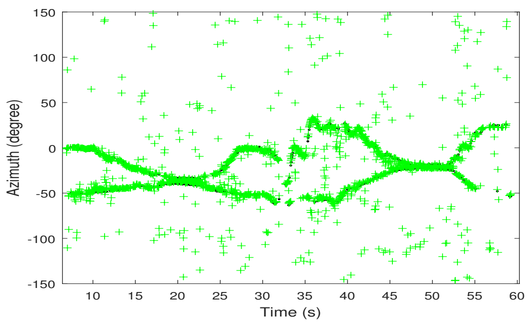
(d) *seq02-1p-0000*: Single speaker tracking



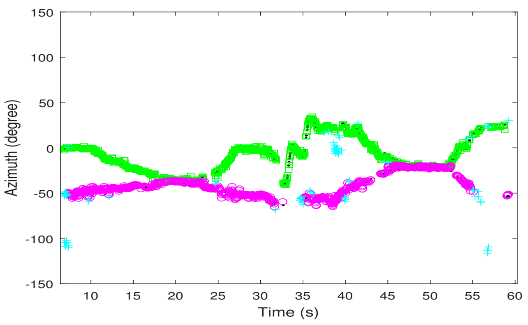
(e) *seq15-1p-0100*: Location measurements



(f) *seq15-1p-0100*: Single speaker tracking

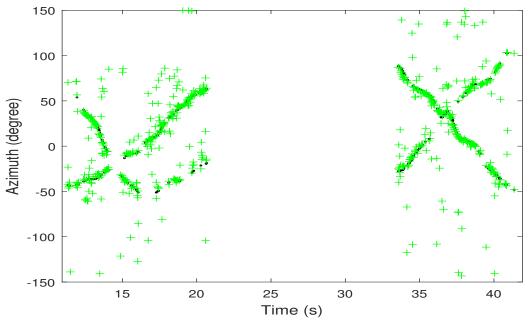


(g) *seq18-2p-0101*: Location measurements

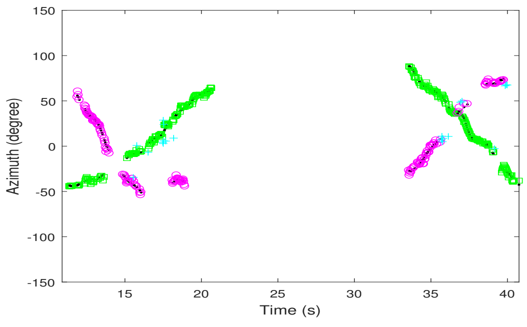


(h) *seq18-2p-0101*: Multiple speaker tracking

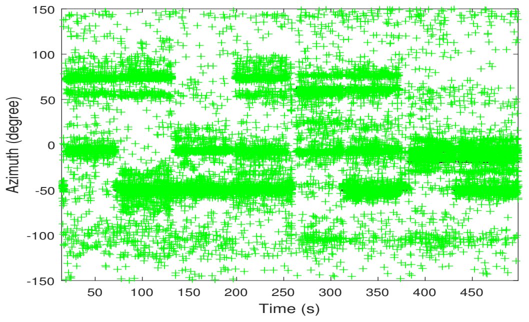
Figure 7.6: *Illustration of the proposed STT tracking on sequences from the AV16.3 corpus. The left column shows the location measurements, which were extracted using P-SRP and then classified as speaker estimates using the online NBC-based speaker detection. The right column shows the final multiple speaker tracking trajectories for the azimuth (in degree).*



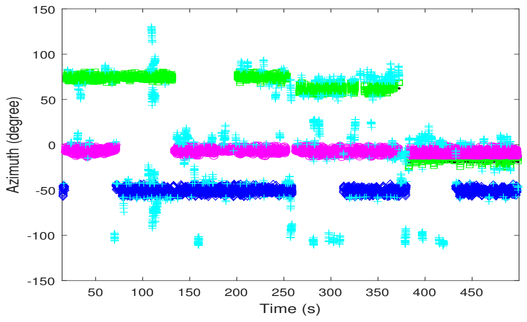
(i) *seq24-2p-0111*: Location measurements



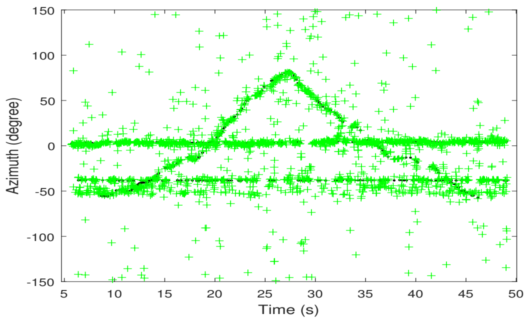
(j) *seq24-2p-0111*: Multiple speaker tracking



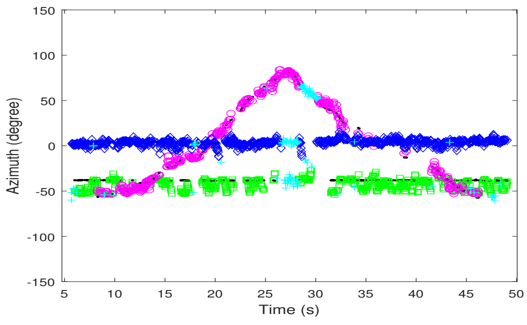
(k) *seq37-3p-0001*: Location measurements



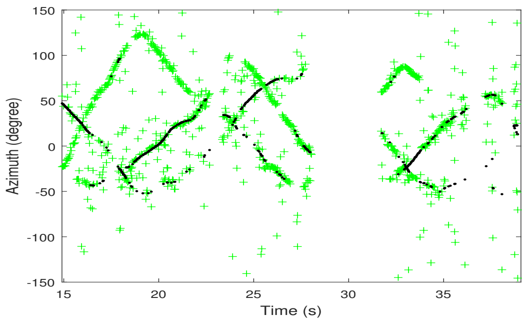
(l) *seq37-3p-0001*: Multiple speaker tracking



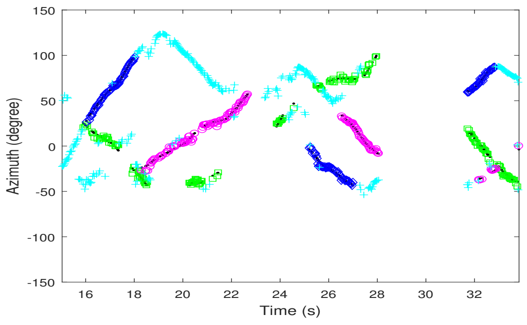
(m) *seq40-3p-0111*: Location measurements



(n) *seq40-3p-0111*: Multiple speaker tracking



(o) *seq45-3p-1111*: Location measurements



(p) *seq45-3p-1111*: Multiple speaker tracking

Figure 7.6: *Illustration of the proposed STT tracking on sequences from the AV16.3 corpus. The left column shows the location measurements, which were extracted using P-SRP and then classified as speaker estimates using the online NBC-based speaker detection. The right column shows the final multiple speaker tracking trajectories for the azimuth (in degree).*

ambient noise sources become competitive to the desired source(s), leading to an increase in the clutter detection rate.

We presented in Section 7.1.3 a new approach to improve the TDOA estimation stage. In particular, this approach counteracts the noise/reverberation problem by enhancing the TDOA detection using tracking information, which leads to more accurate TDOA observations, and thereby to a more robust TDOA-based tracking performance. This solution deals, however, only with the *single* speaker problem, and was designed only for TDOA-based tracking.

We propose in this section an extension of this approach to improve the detection of multiple overlapping speakers using tracking information. More precisely, at each time frame, the proposed approach 1) estimates the P-SRP, presented in Section 5.4, which is used as a measurement detector. This is followed by 2) the estimation of the predicted tracking distributions of all *confirmed* speakers. These pdfs characterize the most likely regions to contain the next measurements. 3) The resulting Gaussians are then used to update the mixture weights of the P-SRP by measuring the *similarity*, as it was done in Section 7.1.3, between each Gaussian component in the P-SRP and the predicted pdfs. Finally, 4) the enhanced P-SRP is used to estimate the location measurements, which are then processed by the multiple speaker STT framework as described in Section 7.2.2.

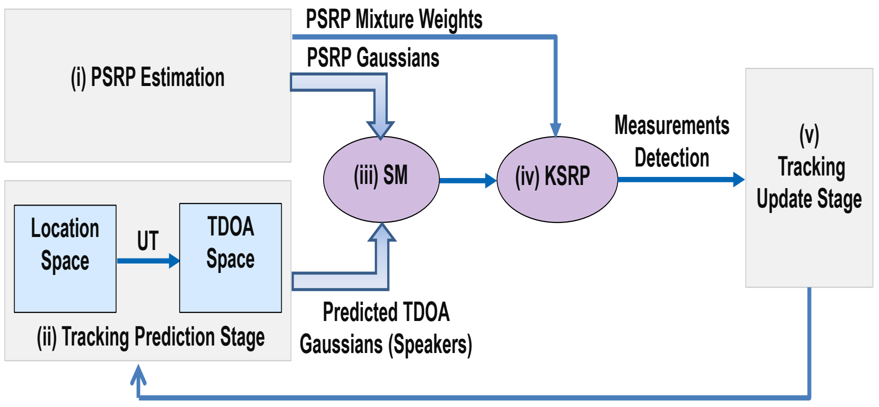


Figure 7.7: *Block diagram of the proposed approach to enhance the P-SRP-based multiple speaker detector using tracking information*

To introduce the new approach, let $SRP_{prob}(\mathbf{l}) \propto \sum_{q\in\mathcal{Q}} \sum_{k=1}^{K^q} p_k^q \cdot \mathcal{N}_k^q\big(\tau^q(\mathbf{l}), \mu_k^q, (\sigma_k^q)^2\big)$ be the P-SRP distribution at time t, which is obtained after approximating each GCC function, of each microphone pair q, by a GM distribution $\mathcal{GM}^q = \{\mathcal{N}_k^q\}_{k=1}^{K_q} = \{p_k^q, \mu_k^q, \sigma_k^q\}_{k=1}^{K_q}$. A detailed description of these approaches was introduced in Chapter 4 and Section 5.4, respectively. For ease of notation, the time index was dropped from these expressions. The next sections introduce the mathematical formulation of the proposed approach.

7.2.3.1 UT for Location-TDOA Mapping

The enhancement of the measurement detection using tracking information is based on reweighting the mixture weights of each of these GMs, which collectively define the P-SRP distribution. This is done by evaluating *how close*, and thereby how relevant, each component in the GMs from the predicted pdfs. Note that this is the same idea that was used in Section 7.1.3 to improve the TDOA estimation using tracking information in the single speaker case. The main issue here, however, is that the predicted pdfs are estimated in the location space, whereas the GCC-based GM components are in the TDOA space. To circumvent this problem, we propose to transform the predicted pdfs using the location-TDOA function given by (4.2). Unfortunately, this function is non-linear, therefore, we propose to use the UT to propagate the mean and covariance of each predicted distribution through the non-linear transformation (4.2) to the TDOA space (step (ii) in Figure 7.7). For ease of notation, the speaker index n is dropped in the rest of this section.

Let d be the location space dimension. In the UT approach, each speaker predicted (location) distribution $\mathcal{G}_S(\mathbf{l}) = \mathcal{N}_S(\mathbf{l}, \mu_S, \Sigma_S)$ is represented as a weighted empirical distribution of $2d+1$ weighted, sigma points $\{\mathcal{L}_i, \mathcal{W}_i\}_{i=0}^{2d}$, calculated according to:

$$\begin{array}{rl|rl} \mathcal{L}_0 = & \mu_\mathbf{l}, & \mathcal{W}_0 = & \kappa/\lambda, \\ \mathcal{L}_{2i+1} = & \mu_\mathbf{l} + \sqrt{\lambda} R_i, & \mathcal{W}_{2i+1} = & 1/(2\lambda), \\ \mathcal{L}_{2i+2} = & \mu_\mathbf{l} - \sqrt{\lambda} R_i, & \mathcal{W}_{2i+2} = & 1/(2\lambda). \end{array} \tag{7.24}$$

$i = 1, \ldots, (d-1)$, where $\lambda = d + \kappa$ for an arbitrary $\kappa \in \mathbb{R}$. In fact, κ specifies how much weight is placed on the mean μ_S, and is set to 1/2, which leads to a weight of $1/d$ for all sigma points. Furthermore, R_i are the rows of the matrix R, result of the Cholesky decomposition $R^T R$ of the covariance Σ_S. The resulting sigma points are then mapped to the TDOA space according to the location-TDOA function (4.2):

$$\mathcal{T}_i^q = \tau^q(\mathcal{L}_i), \quad i = 0, \ldots, 2d, \;\; q \in \mathcal{Q}. \tag{7.25}$$

The mean and covariance of the Gaussian distribution $\mathcal{G}_{T^q}(\tau) = \mathcal{N}(\tau, \mu_{T^q}, \Sigma_{T^q})$, approximating the transformed (predicted) TDOA pdf at the qth microphone pair, are calculated according to:

$$\mu_{T^q} = \sum_{i=0}^{2d} \mathcal{W}_i \cdot \mathcal{T}_i^q, \quad \Sigma_{T^q} = \sum_{i=0}^{2d} \mathcal{W}_i \cdot (\mathcal{T}_i^q - \mu_{T^q})^2. \tag{7.26}$$

7.2.3.2 Similarity-based P-SRP Update

Assuming that there are N_t active speakers at time t, the UT step above leads to N_t predicted TDOA distributions, and that is for each microphone pair $q \in \mathcal{Q}$. Each of these pdfs characterizes the most likely TDOA estimate to be generated by the speaker at the microphone pair q in

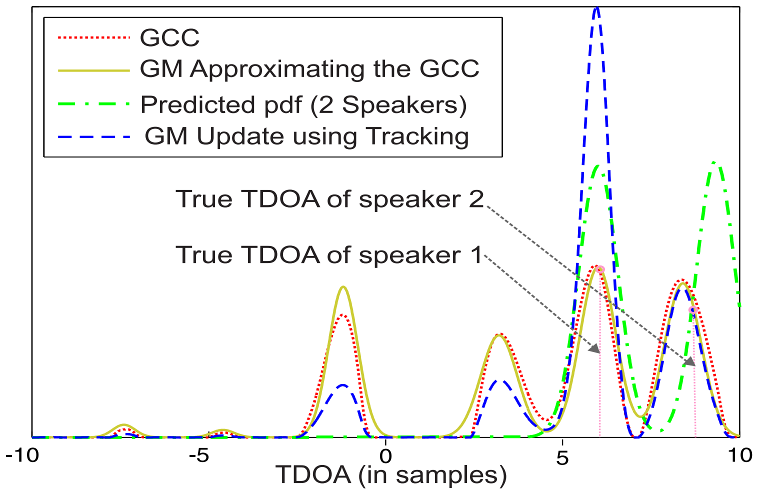


Figure 7.8: *Example of a GCC with a dominant and a masked speaker. We can see that using the tracking information to update the GCC-based GM enhances the low-energy speaker, this corresponds to steps (ii) and (iii) in Figure 7.7.*

the next time frame. Therefore, these distributions can be used as a prior to enhance the most likely Gaussian components in the GM, approximating the GCC function in the P-SRP. This is done through the calculation of similarity scores using the KLD or BC similarity measures given by (7.9) and (7.10), respectively. This part corresponds to step (iii) in Figure 7.7.

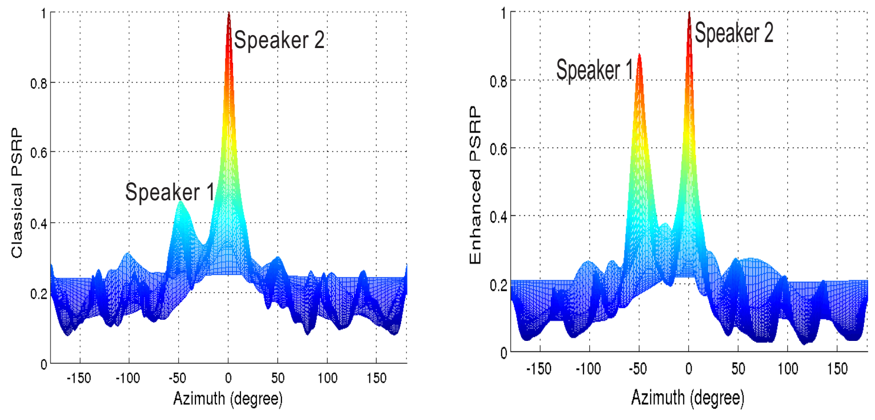
The resulting similarity scores are then used to enhance the mixture weights of all Gaussians in the P-SRP distribution before proceeding to the measurement detection step. More precisely, the new mixture weight $\bar{p}_k^q$ of the kth Gaussian component in the GM approximating the qth GCC function is calculated according to:

$$\bar{p}_k^q = \frac{p_k^q}{Z} \cdot \sum_{n=1}^{N_t} SM(\mathcal{G}_{T^q}^n, \mathcal{N}_k^q). \tag{7.27}$$

Z is the normalization term. Figure 7.8 shows an example of enhancing a TDOA-GM using the tracking information of two active speakers. This illustration shows clearly that the GM components corresponding to the two speakers are either preserved or enhanced, whereas the noise and reverberation components are smoothed out.

The new mixture weights incorporate the tracking prior of *all confirmed* speakers at time t into the detection step. In fact, the update step smooths out the unlikely components in the GM and enhances the ones which are close to the predicted TDOA distributions. The enhanced SRP_{prob}^{track} is given then by:

$$SRP_{prob}^{track}(\mathbf{l}) \propto \sum_{q \in Q} \sum_{k=1}^{K^q} \bar{p}_k^q \cdot \mathcal{N}_k^q(\tau^q(\mathbf{l}), \mu_k^q, (\sigma_k^q)^2). \tag{7.28}$$



(a) Standard P-SRP (b) Tracking-based P-SRP

Figure 7.9: *Figure (a) shows a classical P-SRP with two speakers at azimuth -50° and 0°, respectively (step (i) in Figure 7.7). Figure (b) shows the enhanced P-SRP after combining the updated GM of all microphone pairs. The low-energy speaker at azimuth -50° is clearly enhanced (step (iv) in Figure 7.7).*

7.2.3.3 Accounting for New Emerging Speakers

The updated SRP_{prob}^{track} integrates only the tracking information of the *confirmed* speakers, whereas it smooths out the space regions which do not contain an active target. These regions, however, may contain new emerging speakers (at the birth state of the tracking framework), which will be suppressed in the update step. To counteract this problem, we propose to preserve the new potential targets information provided by the P-SRP according to:

$$SRP_{kalman} = \alpha \cdot SRP_{prob}^{track} + (1-\alpha) \cdot SRP_{prob}. \tag{7.29}$$

α is a confidence factor characterizing how much trust is placed on the tracking information, and how unlikely it is that new speakers appear in the scene, α can also be learned as a time-dependent factor. SRP_{kalman} is our final location measurement detector, which will be used instead of SRP_{prob} to extract potential speaker locations. Once this is done, tracking multiple speakers follows the same procedure described in Section 7.2.2. We will refer to the SRP_{kalman}-based speaker tracking as Kalman Steered Response Power (K-SRP).

7.2.3.4 Evaluation of K-SRP-based Overlap Detection

We evaluate the proposed SRP_{kalman}-based multiple speaker tracking on the AV16.3 corpus using the same experimental setup described in 7.2.2.4. In addition to the evaluation metrics reported in 7.2.2.4, we also report here the overlap detection rate of N simultaneous speakers. The latter is defined as the ratio of the number of detected frames with N correct simultaneous speaker estimates to the total number of frames.

In the experiments reported below, the multiple speaker STT approach, described in Section 7.2.2 is used to estimate the tracking information, which is necessary to enhance the P-SRP, as described in Section 7.1.3. The proposed approach, however, can be integrated into any multiple speaker tracking framework. The tracking confidence factor α is set to 0.9, and KLD is used to calculate the similarity scores. The STT parameters setting is the same as the one used in Section 7.2.2.4. The results are reported without voice activity detection to show the effect of long non-speech/noise segments on the proposed approach.

	seq18-2p-0101			seq24-2p-0111		
	STC	STT	K-SRP	STC	STT	K-SRP
p_s	85.0	**99.0**	97.2	**81.6**	81.1	80.9
t_r	81.5	90.4	**92.9**	63.7	66.8	**72.4**
d_r of Speaker 1	53.1	61.1	**67.5**	54.9	59.0	**62.7**
d_r of Speaker 2	51.6	54.8	**62.5**	34.3	37.9	**46.5**
Avg. d_r	52.3	58.0	**65.0**	44.6	48.4	**54.6**
Avg. RMSE (°)	**1.96**	2.34	2.27	3.07	**3.04**	3.11

(a) Tracking performance on recordings with two simultaneous speakers.

	seq40-3p-0111			seq37-3p-0001		
	STC	STT	K-SRP	STC	STT	K-SRP
p_s	94.1	**94.3**	92.9	90.6	**94.3**	92.9
t_r	75.7	86.6	**88.9**	82.2	84.2	**87.8**
d_r of Speaker 1	39.2	49.7	**53.3**	28.8	29.9	**32.2**
d_r of Speaker 2	38.4	40.0	**45.0**	66.2	71.0	**75.2**
d_r of Speaker 3	56.8	62.6	**66.3**	**46.7**	40.2	44.5
Avg. d_r	44.8	50.8	**54.9**	47.9	47.0	**50.6**
Avg. RMSE (°)	6.56	4.70	**4.59**	2.47	2.30	**2.25**

(b) Tracking performance on recordings with three simultaneous speakers.

Table 7.4: Tracking performance in terms of the precision rate p_s, trajectory estimation rate t_r, speaker detection rate, average RMSE (degree) and real-time factor t.

Table 7.4 present the performance of the original STT approach, which uses the P-SRP approach as a measurement detector to track multiple overlapping speakers, and compares it to the proposed approach (noted as K-SRP), which uses the tracking information provided by the STT method to enhance the P-SRP as described above. Moreover, the results are compared to the complete multiple speaker STC framework proposed in [70, 73]. The details of this framework were introduced in Section 7.2.2.4. More details about this approach can be found in [73].

Table 7.4 shows a clear improvement of K-SRP over the STT and STC approaches. We can see that K-SRP achieves longer correct tracking trajectories, i.e., the increased correct tracking duration rate t_r, as well as higher individual and average speaker detection rates, and that is for most multiple speaker sequences from the AV16.3 corpus. More precisely, the K-SRP approach achieves an average detection rate improvement of 18.7% and 10.5% compared to STC and STT approaches, respectively, whereas the average trajectory estimation rate improvement is 13.0% and 4.5%. These results show that the main improvement of the K-SRP approach is due to 1) the increased speaker detection in frames with low-energy, which is reflected by the improved trajectory rate, and to 2) the increased detection of (overlapping) speakers in low-energy frames or frames where the dominant speaker masks the secondary speakers. This is reflected in Table 7.4 by the improved detection rates.

In order to further analyze and confirm these results, we present in Table 7.5 the distribution of the trajectory rates on frames with N correct estimates. That is, we calculate the percentage of frames with N correct estimates, where $N = 0, \dots, N_{max}$. N_{max} is the maximum number of overlapping speakers for each recording.

N (number of overlapping speakers)	seq18-2p-0101			seq24-2p-0111		
	STC	STT	K-SRP	STC	STT	K-SRP
0 (no detection)	**30.5**	23.4	21.1	**68.3**	66.7	64.0
1 (no overlap)	50.2	**55.6**	48.0	**25.6**	25.3	**25.6**
2 Speakers	19.3	21.0	**30.9**	6.12	8.05	**10.4**

(a) Overlap detection performance on recordings with two simultaneous speakers.

N (number of overlapping speakers)	seq40-3p-0111			seq37-3p-0001		
	STC	STT	K-SRP	STC	STT	K-SRP
0 (No detection)	**36.1**	26.7	24.8	**25.2**	23.4	20.2
1 (No overlap)	28.5	**33.6**	30.0	50.2	**52.0**	51.1
2 Speakers	30.6	37.6	**40.7**	23.6	23.7	**27.3**
3 Speakers	**4.83**	2.18	4.55	0.99	0.92	**1.40**

(b) Overlap detection performance on recordings with three simultaneous speakers.

Table 7.5: Tracking performance in terms of overlap detection rate (%) of N simultaneous speakers. $N = 1$ is the percentage of frames with a single speaker estimate (no overlap).

Table 7.5 generally confirms the conclusion we drew from Table 7.4. In particular, we can see that K-SRP significantly reduces the fraction of frames with no detection, which are typically low-energy/silence frames. Furthermore, K-SRP achieves a higher percentage of frames with (two or three) overlapping speakers, whereas the percentage of frames with a single speaker, i.e., no overlap detection, is decreased. The low overlap detection of three simultaneous speakers shows that speaker overlap mostly occurs between two speakers in spontaneous speech.

Regarding the precision of the different methods, we can confirm that the three approaches lead to a comparable RMSE. The precision rate, however, shows a negligible degradation for K-SRP. This minor degradation introduced by the K-SRP is mainly due to the absence of a speech/non-speech classifier, which uses speech cues to reject noise estimates during long silence/noise frames. As a result, the K-SRP also enhances the noise trajectories during these frames leading to this slight degradation.

Part II

Sequential Neural Estimation: Language Modeling

8 Background: Language Modeling

8.1 Statistical Language Modeling

Traditionally, the goal of a language model is to identify possible sequences of predefined linguistic units, which are typically words. This process is generally guided by the semantic and syntactic properties encoded by the Language Model (LM). A Statistical Language Model (SLM) solves this problem by learning the joint probability distribution over sequences of words, which are typically sentences, from training data. In doing so, SLM provides a principled and unified approach to model and solve a wide range of speech and language technology problems. Some of the early applications that paved the way to the popularity and common usage of SLMs include Automatic Speech Recognition (ASR) [110], Statistical Machine Translation (SMT) [111] and Information Retrieval (IR) [112], to name a few.

Formally, let Y be an observed noisy signal, generated by a hidden sequence of words S, and let us assume that our goal is to maximize the conditional probability $p_{S|Y}$. That is, we would like to extract the sequence of words $\hat{s}$, which best explains the received signal. Using Bayes rule, we can formulate this problem as:

$$\hat{s} = \underset{s}{\operatorname{argmax}}\ p_{S|Y}(s|y) = \underset{s}{\operatorname{argmax}}\ \frac{p_{Y|S}(y|s) \cdot p_S(s)}{p_Y(y)} = \underset{s}{\operatorname{argmax}}\ p_{Y|S}(y|s) \cdot p_S(s). \tag{8.1}$$

We can see that the solution to this problem does not depend on the data prior distribution $p_Y(y)$. Therefore, it is excluded from the right-hand side of this equation. Moreover, $p_{Y|S}(y|s)$

is the likelihood distribution which models the evidence of the data given a hidden sequence of words, whereas $p_S(s)$ models the sequence of words prior captured by the statistical language model. This formulation is also referred to as the *noisy channel model* in many NLP problems. In particular, considering Y to be an acoustic signal leads to the speech recognition solution, whereas considering Y to be an observed sentence in a second language leads to the SMT formulation. This approach can also be applied to information retrieval by simply considering S to be a document. In this case, $p_S(s)$ is the document prior whereas $p_{Y|S}(y|s)$ is a document-based language model. That is, we estimate a prior and a language model for each document, and then extract the one which maximizes the probability of the query given these two quantities. These are few examples that highlight the importance of a high quality SLM for different NLP problems. While these applications may use different corpora to train the SLM of interest, the training approaches used to achieve this goal are, however, very similar. The rest of this section presents a short review and a discussion of statistical language modeling approaches with a particular focus on recent trends and solutions, whereas the next chapter introduces the mathematical formulation of this problem and its different approximations.

8.1.1 N-gram Language Models

Intrinsically, the performance of an SLM can be evaluated based upon its ability to predict the next word given its context. This observation is a direct result of applying the chain rule to express a joint distribution in terms of conditional probabilities. The most common approach to build such models is the word count-based model, which leads to what is commonly known as N-gram LMs [113, 114]. More precisely, N-grams enumerate all possible word combinations over a short span and assign probabilities to them according to the maximum likelihood estimator. Using this simple idea, these models were difficult to outperform for a very long time until the introduction of Neural Network (NN)-based LMs.

Extensive research and a long list of applications using N-gram LMs showed that they generally suffer from different weaknesses that can be summarized as follows:

1. **Data sparsity**: an optimal count-based LM needs to estimate accurate statistics for all possible word combinations. Achieving this goal requires huge amounts of training data, in particular, if large vocabularies are at play. In practice, however, this is hardly feasible, especially for higher order N-grams, due to the limited amount of data available. That is, the trained model will not be able to (or rarely) see many word combinations that are linguistically possible, leading to a poor generalization to unseen data. To illustrate this issue, let us consider a small vocabulary of 10K words and a trigram LM. In this case, we have $M = C(10K, 3) \approx 166B$ possible 3-word combinations. Although many of these trigrams are simple linguistic noise, we can easily see that there is still a large number of valid combination to consider, which highlights the need of large amounts of training data. As a remedy to this problem, different smoothing techniques were introduced [113, 115, 116]. The idea here is to adjust the ML estimator

by either redistributing some probability mass from the estimated N-gram counts on the unseen ones, which is commonly known as *discounting*, or by considering lower order N-grams through *back-off* models or *linear interpolation*. The general difference among smoothing approaches is the considered discounting/back-off function.

2. **Short and fixed context**: the Markov assumption used to approximate the probability of a sequence of words in N-grams is very simplistic. More precisely, language can be generally described as a highly dynamic and complex process that exhibits short and long range patterns. Thus, it cannot be optimally modeled using a fixed and short context window. While different approaches have been proposed to alleviate this problem, such as topic models [117], latent semantic analysis-based models [118] and trigger models [119], dynamic, i.e., cache-based, models [120] are considered to be the widely used solution. The motivation behind cache models is based on the observation that, given a slow change of topic information, the words recently seen are more likely to re-occur. Formally, this is done by learning an additional, small and dynamic N-gram model from the recent history, which is subsequently interpolated with the original static model. Multiple authors, e.g., [121], have reported that while cache models can lead to a significant reduction of the perplexity, they generally perform poorly on speech recognition tasks. This is mainly due to the propagation of recognition errors into the decoding process through the additional cache model, which is learned in this case from the ASR output.

3. **Absence of linguistic features**: N-gram models are based only on the notion of co-occurrence, which is not particular to language. That is, the same principle can also be applied to estimate the maximum likelihood solution for a wide range of discrete sequence prediction problems, regardless of the nature of the data that they consider. From this perspective, N-gram LMs do not learn any inherent linguistic relationship between words that did not appear in the same context, i.e., given the same history. This is in fact one of the most limiting factors of the generalization power of N-gram models to unseen data. One alternative solution to overcome this problem consists of using continuous word representations, which became popular with the introduction of NN language models [122].

8.1.2 Neural Network Language Models (NNLMs)

Similarly to many other applications, such as speech recognition and machine translation, the introduction of neural networks to language modeling led to a significant improvement over standard approaches, in particular, N-gram LMs. One of the most important factors behind their success on this task is the continuous word representation they provide, commonly known as *word embedding*, which overcomes the exponential growth of parameters that N-gram models require to enumerate all word combinations. One of the early NN-based models was introduced by Bengio et al. [122], who proposed to use a Feed-forward Neural Network (FNN), as an alternative to N-grams, to estimate the probability of a given word sequence while considering a fixed context size, i.e., word history. While FNN-LMs were very successful and have been shown to outperform a mixture of different other models [121], and to significantly improve speech recognition performance [123], their fixed context size constraint was a limiting factor for their performance. In order to overcome this constraint and to capture long range dependencies known to be present in language, Mikolov et al. [124, 125] proposed to use a Recurrent Neural Network (RNN), which allows context information to cycle in the network. Although this recurrent architecture, theoretically, allows the context to indefinitely cycle in the network, the authors of [126] have shown that, in practice, the captured information changes quickly in the classical RNN structure [124], and that it is experimentally equivalent to an 8-gram FNN model. As an alternative to model linguistic dependencies, Sundermeyer et *al.* [127] proposed to use a Long-Short Term Memory (LSTM) network to capture long range dependencies and overcome some learning issues that occur in the original RNN. This model has the ability to explicitly control the longevity of context information in the network, and suffers less from the vanishing gradient problem. This chain of novel NN-based LMs continued with more complex and advanced models such as the character-based Convolutional Neural Network (CNN) LM [128] and the autoencoder-based LM [129], to name a few.

In addition to designing new neural architectures to improve language modeling, investigating different strategies to 1) train better character and word embeddings [130], 2) build deep architectures [131], 3) use other modalities [132] and 4) improve the learning capabilities of standard Neural Network Language Models (NNLM)s have been proven to be very effective in improving state-of-the-art LMs. In particular, the introduction of the *dropout* regularization technique [133], which improves the generalization power of neural architectures and helps avoiding over-fitting, was very crucial to improve the performance of recurrent LMs [134]. As an alternative to dropout, the authors of [135] proposed to generalize the learning capabilities of the models by manipulating the training data itself through different noising mechanisms, which were shown to be equivalent to different smoothing techniques in standard N-gram LMs.

Chapter 10 proposes new hybrid NNLMs that combine the learning capabilities of separate standard NN models. In order to introduce these new architectures, Chapter 9 will briefly review the mathematical formulation of language modeling task, and then introduce its approximation according to different standard NNLM architectures.

8.2 Training of Large Vocabulary NNLMs

Training of NNLMs becomes significantly slow and challenging when considering large vocabulary LMs [136]. This is mainly due to the explicit normalization of the output layer, which typically requires the evaluation of the *softmax* function over the complete vocabulary.

In order to overcome this problem, Schwenk et al. [123] proposed to use a short list of frequent words in combination with N-gram models. The performance of this approach, however, significantly depends on the short list size. In a different attempt, Morin et al. [137] proposed to factorize the output probabilities using a binary tree, which results in an exponential speed-up of the training and evaluation, whereas Mikolov et al. [125] proposed to use an additional class layer as an alternative to perform the factorization. The performance of these two approaches significantly depends on the design of the binary tree and the class layer size, respectively. As an alternative to modifying the architecture design, the authors of [138] used importance sampling to approximate the gradient of the objective function. Unfortunately, this approach requires a control of the samples variance, which can lead otherwise to unstable training [139]. In a similar work, Mnih et al. [140] proposed to use Noise Contrastive Estimation (NCE) [141] to speed-up the training. NCE treats the learning as a binary classification problem between a target word and noise samples, which are drawn from a noise distribution. Moreover, NCE considers the normalization term as an additional parameter that can be learned during training; or fixed beforehand. In this case, the network learns to *self-normalize*. This property makes NCE more attractive compared to other sampling methods, such as importance sampling [138], which would still require the use of the softmax function during evaluation. In batch mode training, however, the implementation of NCE cannot be directly formulated using dense matrix operations, which compromises their speed-up gains. One simple solution for NCE approach is to share noise samples across the targets in the batch [142]. Along this line, chapter 11 proposes a new solution to train large vocabulary LMs using NCE in batch mode.

8.3 Evaluation Framework of Language Modeling

In order to evaluate the LM approaches proposed in this thesis, we use three different corpora, namely, we use the Penn Treebank (PTB) corpus and the Large Text Compression Benchmark (LTCB) [143] to test the new hybrid NNLMs that we will introduce in Chapter 10, whereas the One Billion Word Benchmark (OBWB) and LTCB are used to evaluate the performance and speed-up gains of the B-NCE approach, described in Chapter 11. This section will briefly review these corpora and the evaluation metrics that we use to assess the performance of the proposed NNLMs in comparison to the baseline models.

8.3.1 Evaluation Corpora

8.3.1.1 Penn Treebank Corpus (PTB)

All proposed hybrid NNLMs are evaluated, in a first set of experiments, on the Penn Treebank section of the Wall Street Journal corpus, which we will briefly refer to as Penn Treebank. Although this benchmark is relatively small given the current available corpora and available resources, PTB remains one of the most widely used and studied corpora in the context of language modeling. Moreover, most evaluations reported on this corpus use the same data processing and model setup, which makes analysis and comparison to other models reported in the literature straightforward. The experiments conducted in this thesis will follow the standard setup and data split, e.g., [125, 144]. That is, sections 0-20 are used for training while sections 21-22 and 23-24 are used for validation and testing, respectively. The vocabulary was limited to the most frequent 10K words while the remaining words were all mapped to the token <unk>. Similarly to the RNNLM toolkit[1] [125], we have used a single end of sentence tag </s> between each two consecutive sentences, whereas the begin of sentence tag <s> was not included[2].

8.3.1.2 Large Text Compression Benchmark (LTCB)

In order to evaluate how the proposed models scale to large corpora, we run a second set of experiments on the Large Text Compression Benchmark (LTCB) [143]. This corpus was primarily designed to conduct research on text compression tasks under the rationale that solving such problem requires efficient language modeling combined with powerful coding. Moreover, the authors of this corpus argue that language modeling is easy for humans but hard for computers since it requires vast knowledge, whereas coding is easy for computers but hard for humans because it requires deterministic computation. From this perspective, developing efficient language models for computers is at the heart of solving the compression task.

LTCB is an extract of the enwik9 dataset, which contains the first 10^9 bytes of the XML text dump of the English version of Wikipedia. The training-validation-test split and data processing used in our experiments follow the recipe proposed in [144]. That is, all but the most frequent 80K words were replaced by <unk>, whereas the data split uses the toolkit provided by [144]. Similarly to the RNNLM toolkit[1] and the PTB experiments, we have used a single end of sentence tag between each two consecutive sentences, whereas the begin of sentence tag was not included[2]. Table 8.1 presents more details about the size and the OOV rate of words that were mapped to <unk> for PTB and LTCB.

	Train		Dev		Test	
Corpus	#W	<unk>	#W	<unk>	#W	<unk>
PTB	930K	6.52%	82K	6.47%	74K	7.45%
LTCB	133M	1.43%	7.8M	2.15%	7.9M	2.30%

Table 8.1: Corpus size in number of words and <unk> rate.

8.3.1.3 One Billion Word Benchmark (OBWB)

Assessing the relevance of LMs should take into account different aspects. That is, in addition to the model performance which is our driving factor, some of the other influential criteria to consider are: 1) the amount of data used to train and evaluate the model, 2) the time needed during training, 3) the retrieval speed, i.e., time needed to retrieve the probability of a word sequence (e.g., a query) and 4) the size of the model, which governs the memory needed to

[1]The RNN LM toolkit is available at `http://www.rnnlm.org/`

[2]This explains the difference in the corpus size compared to the numbers reported in [144].

store it and can impact the previous factor. While the third and forth aspects, which are related to model deployment in real world applications, are becoming less of an issue due to the increasing computation and storage resources available on servers, the first and second problems are still crucial for language modeling. In particular, there is a popular belief that using large amounts of data is the bottleneck to achieving a satisfying performance, which would in return require a longer training time, especially if large vocabularies are at play. The One Billion Word Benchmark (OBWB) [136] was collected in order to facilitate an empirical investigation of this belief, in addition to bridging the gap between models trained in controlled environments on small corpora and real world applications, which would generally require larger models. To do so, OBWB offers a corpus with ≈ 0.8B tokens with a vocabulary size of ≈ 0.8M words.

The data processing we adopt in this thesis follows the description provided in [136], which leads to an <unk> rate (i.e., OOV rate) of $\approx 0.3\%$. Similarly to LTCB, a single </s> tag is added at the end of each sentence. In the experiments described below, we also use the same data split proposed in [136]. Furthermore, the first 5 held-out sets are used for validation whereas the remaining 45 sets are used for testing. The results that we report in the rest of this thesis showed, however, that the models perplexity on these two sets is comparable with an average difference of less than 0.5 point in perplexity.

The OBWB is used in this thesis for two purposes: 1) to evaluate the B-NCE approach which would highlight its potential when large corpora/vocabularies are considered. This will be done after a first set of experiments on LTCB, and 2) to provide a baseline study of standard NNLMs on this large corpus, which is needed to draw strong conclusions about NNLMs. In particular, we will train, evaluate and analyze the performance of different FNN, RNN and LSTM models with comparable number of parameters on this corpus. In fact, the baseline we report in this thesis is complimentary to the one conducted in [136].

8.3.2 Evaluation Metrics

In order to conduct a balanced evaluation of the LMs that we present in this thesis, we report their performance according to different metrics that reflect the LM relevance factors that we discussed in Section 8.3.1.3. In particular, the results are reported in terms of:

1. **The perplexity (PPL)**: Formally, PPL is calculated according to:

$$\mathrm{PPL} = \left(\prod_{t=1}^{T} \frac{1}{p(w_t|c_t)} \right)^{\frac{1}{T}} = 2^{\left(-\frac{1}{T}\sum_{t=1}^{T} \log_2 \, p(w_t|c_t)\right)} \tag{8.2}$$

where T is the number of tokens in the test set and c_t is the context considered during the prediction of the word w_t, e.g., c_t of an FNN model is given by the last $N-1$ words whereas c_t of an RNN model is the recurrent context h_t.

Perplexity is an indirect way of evaluating the estimated cross-entropy according to a Monte Carlo approximation. Given that the cross-entropy of a given model is lower-bounded by the entropy of the actual, unknown distribution of the data, we can conclude that a good LM is the one which achieves the lowest cross-entropy possible and thereby the lowest perplexity. The reason to use perplexity instead of a direct usage of entropy is merely due to the more convenient scale of values it offers and its simplicity to report, remember and compare for different models. That is, perplexity is generally reported as whole numbers (or sometimes as decimal numbers with a rounding of 0.5), whereas entropy is evaluated in bits, where the decimal part is significant during evaluation even for small differences.

Another common metric to assess LMs is the word error rate, which is generally used to evaluate the performance of an ASR system. The latter combines the language model with an acoustic model to perform this task. While this metric is very relevant for many speech applications, the models developed in this thesis are mainly used in the context of "Information Density and Linguistic Encoding" (IDeaL) [3], which is a collaborative research center that investigates the hypothesis that language use may be driven by the optimal use of the communication channel. Thus, we can hypothesize that language complexity may be appropriately indexed by Shannon's notion of information, which is also referred to as surprisal. From this perspective, perplexity seems to be a more adequate measure to evaluate our LMs. That has been said, we agree that a study of the application of these models to speech recognition (and machine translation) should also be conducted in the future.

2. **The Number of Parameters (NoP)**: This metric allows us to conduct a more informative comparison of the models. More precisely, it is well-known today that increasing the number of parameters in a neural network combined with a good choice of a regularization technique, to avoid over-fitting, can significantly improve the performance of the models. Thus, it is very important to consider models with a comparable number of parameters during comparison and analysis. Another important aspect would be to investigate the slope of the model performance with an increasing number of parameters.

3. **The Training Speed (TS):** This is defined as the number of words processed per second (w/s) during training on a GTX TITAN X GPU. While training new deeper and more complex models can generally improve the performance, different architectures and models need different training time due to the different types and number of connections they incorporate. For instance, a convolutional layer is typically slower compared to a fully connected layer. Hence, it is also important to consider the extra-time needed to train a model knowing the improvement it can achieve as an additional aspect during model comparison.

[3]Information Density and Linguistic Encoding: http://www.sfb1102.uni-saarland.de

9 Basics: Neural Network Language Models

9.1 Language Modeling: Formulation

The goal of a language model is to estimate the probability distribution $p(w_1^T)$ of word sequences $w_1^T = w_1, \cdots, w_T$. Using the chain rule, this distribution can be expressed as:

$$p(w_1^T) = \prod_{t=1}^{T} p(w_t | w_1^{t-1}). \tag{9.1}$$

This probability is generally approximated under different simplifying assumptions, which are typically derived based on different linguistic observations. All these assumptions, however, aim at 1) modeling the context information, syntactic and/or semantic, needed to perform the word prediction, and 2) to simplify the estimation process. The resulting models can be broadly divided into two categories: long and short range context models. The rest of this chapter presents a brief overview of these two categories with a particular focus on NNLMs.

9.1.1 Short vs Long Range Context

Short range context LMs approximate (9.1) based on the Markov dependence assumption of order $N-1$. That is, the prediction of the current word depends only on the last $N-1$ words in the history. In this case, the distribution in (9.1) becomes

$$p(w_1^T) \approx \prod_{t=1}^{T} p(w_t | w_{t-N+1}^{t-1}). \tag{9.2}$$

The most popular methods that subscribe in this category are N-grams [113, 114] and FNN models [122]. N-gram models calculate each term in this product using the maximum likelihood estimate given by:

$$p(w|h) = \frac{\text{count}\,(h,w)}{\text{count}\,(h)}, \tag{9.3}$$

where h represents the context, given by the last N-1 words, and count (h, w) denotes the number of times that the word w appeared after the context h, which is then divided by the count of h. We can see here that these probabilities are independent of the time stamp t. Thus, these statistics are generally calculated and stored offline. FNN language models, on the other hand, estimate each of the terms involved in the product above, i.e, $p(w_t|w_{t-N+1}^{t-1})$, in a single bottom-up evaluation of the network. Section 9.2.1 will introduce the mathematical formulation used to perform this estimation.

Although short context LMs perform well and are easy to learn, natural languages that they try to encode, however, cannot not be modeled as a Markov process due to their dynamic nature and the long range dependencies they manifest. Alleviating this assumption led to an extensive research to develop more suitable modeling techniques.

In order to illustrate the dynamic and long range dependencies present in language, let us consider the word triggering problem. That is, we would like to investigate how does the occurrence of a word impacts the probability of occurrence of other words over a large context. To do so, we use the triggering correlation defined over a distance d according to:

$$c_d(w_1, w_2) = \frac{p(w_1, w_2)}{p(w_1)\, p(w_2)}. \tag{9.4}$$

A value greater than 1 shows that it is more likely that the word w_1 follows w_2 at a distance d than expected without the occurrence of w_2. Figure 9.1 shows the variation of this correlation for pronouns with the distance d. It can be observed that seeing another "he" about twenty words after having seen a first "he" is much more likely. A similar observation can be made for the word "she". It is, however, surprising that seeing "he" after "he" is three times more likely than seeing "she" after "she", so "he" is much more predictive. In the cases of cross-word triggering of "he" → "she" and "she" → "he", we find that the correlation is suppressed in comparison to the same word triggering for distances larger than three. In summary, Figure 9.1 demonstrates that word triggering information exists at large distances, even up to one thousand words. These conclusions were confirmed by similar correlation experiments that we conducted for different types of words and triggering relations.

In order to model this long-term correlation and overcome the restrictive Markov assump-

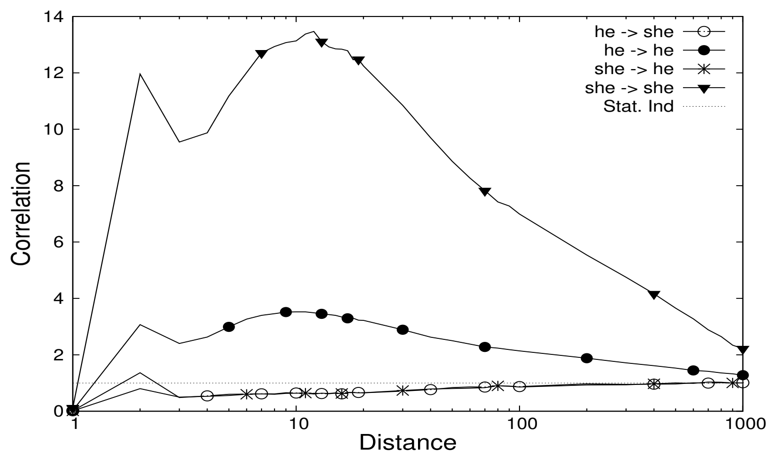


Figure 9.1: *Variation of word triggering correlation for pronouns over large distances.*

tion, recurrent LMs have been proposed to approximate (9.1) according to:

$$p(w_1^T) \approx \prod_{t=1}^{T} p(w_t | w_{t-1}, h_{t-1}) = \prod_{t=1}^{T} p(w_t | h_t). \tag{9.5}$$

In NN-based recurrent models, h_t is a context vector which represents the complete history, and modeled as a hidden state that evolves within the network. The main difference between recurrent language models is the function used to calculate h_t based on h_{t-1}.

9.2 Baseline Neural Network Language Models

NNLMs use different architectures to approximate the probability of a word sequence $p(w_1^T)$, given by (9.1). This is done by approximating and then separately estimating each of the terms involved in this product, i.e, $p(w_t|w_1^{t-1}) \approx p(w_t|h_t)$, as we discussed earlier.

Formally, let U be a word embedding matrix and let W be the hidden-to-output weight matrix. NNLMs that consider word embeddings as input, approximate each of these terms in a bottom-up evaluation of the network according to:

$$H_t = \mathcal{M}(\mathcal{P}, \mathcal{R}_{t-1}, U), \tag{9.6}$$

$$O_t = g(H_t \cdot W), \tag{9.7}$$

where $\mathcal{M}$ represents a particular NN-based model, which can be a deep architecture, H_t and O_t are the last hidden and output layers of the network at time t, respectively, $\mathcal{P}$ denotes the parameters characterizing this particular model and $\mathcal{R}_{t-1}$ denotes its recurrent information at time $t-1$. $g(\cdot)$ is the softmax function.

The rest of this section briefly introduces $\mathcal{M}$, $\mathcal{P}$ and $\mathcal{R}_{t-1}$ for the basic NNLMs architectures. That is, we review FNN, RNN and LSTM-based language models.

9.2.1 FNN-based Language Modeling

Similarly to N-gram models, FNN uses the Markov assumption of order N-1 to approximate (9.1), which leads to (9.2). Subsequently, each of the terms involved in the probability product, i.e, $p(w_t|w_{t-N+1}^{t-1})$, is estimated separately in a single bottom-up evaluation of the network. In this case, the model $\mathcal{M}$ of a one-layer FNN is given by:

$$E_{t-i} = X_{t-i} \cdot U \ , \qquad i = N-1, \ldots, 1 \tag{9.8}$$

$$H_t = f\left(\sum_{i=1}^{N-1} E_{t-i} \cdot V_i\right). \tag{9.9}$$

X_{t-i} is a one-hot encoding of the word w_{t-i}. Thus, E_{t-i} is the continuous representation of the word w_{t-i}. $f(\cdot)$ is an activation function and $V = [V_1, \cdots, V_{N-1}]$ are the connection weights of the hidden layer. Figure 9.2 shows an example of an FNN with a fixed context size $N-1=3$ with a single hidden layer. To summarize, for an FNN model $\mathcal{M}$, $\mathcal{P} = \{V_i\}_{i=1}^{N-1}$ and $\mathcal{R}_{t-1} = \emptyset$.

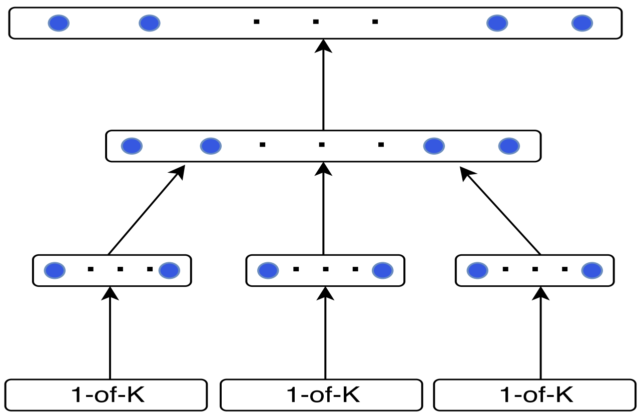


Figure 9.2: *FNN architecture of a model with one hidden layer and a context of three words.*

9.2.2 Elman RNN-based Language Modeling

RNN proposes to capture the complete history in a context vector h_t, which represents the state of the network and evolves in time. Therefore, RNN approximates each term in (9.1), i.e., $p(w_t|w_1^{t-1})$, according to:

$$p(w_t|w_1^{t-1}) \approx p(w_t|w_{t-1}, h_{t-1}) = p(w_t|h_t). \tag{9.10}$$

This approximation shows that the context at time t is continuously updated based on the context at time $t-1$ and the last seen word w_{t-1}. RNN models this transformation as:

$$H_t = f(X_{t-1} \cdot U + H_{t-1} \cdot V). \tag{9.11}$$

Thus, for an RNN model $\mathcal{M}$, $\mathcal{P} = V$ and $\mathcal{R}_{t-1} = H_{t-1}$. Figure 9.3 shows an example of the standard RNN architecture, also known as *vanilla* RNN.

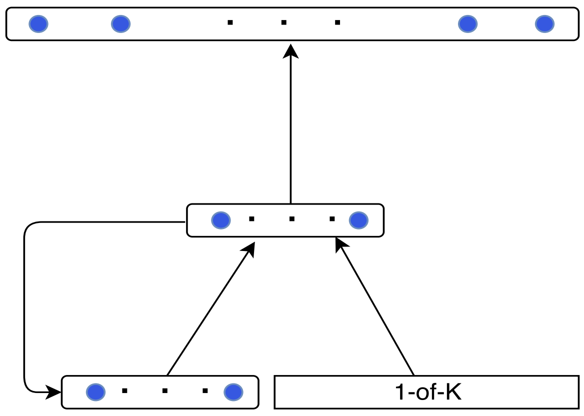


Figure 9.3: *Elman RNN architecture.*

Theoretically, the recurrent connections of an RNN allow the context to indefinitely cycle in the network and thus, modeling long context. In practice, however, the authors of [126] have shown that this information changes quickly over time, and that it is experimentally equivalent to an 8-gram FNN. This is mainly due to the static nature of the connection weights that are applied to the context at each time step, which turns them into some forgetting/smoothing factors. This observation was also confirmed by the experiments conducted in this thesis. The next section introduces a different recurrent model that can improve long range modeling.

9.2.3 Long-Short Term Memory Network

In order to alleviate the rapidly changing context in standard RNNs and control the longevity of the dependencies modeling in the network, LSTM network [127] introduces an internal memory state C_t, which explicitly controls the amount of information to forget or to add to the network before estimating the current hidden state. Formally, an LSTM model $\mathcal{M}$ is given by:

$$E_{t-1} = X_{t-1} \cdot U, \tag{9.12}$$

$$\{i, f, o\}_t = \sigma\left(V_w^{i,f,o} \cdot E_{t-1} + V_h^{i,f,o} \cdot H_{t-1}\right), \tag{9.13}$$

$$\tilde{C}_t = f\left(V_w^c \cdot E_{t-1} + V_h^c \cdot H_{t-1}\right), \tag{9.14}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \tag{9.15}$$

$$H_t = o_t \odot f(C_t), \tag{9.16}$$

where $\odot$ is the element-wise multiplication operator and $\tilde{C}_t$ is the memory candidate, whereas i_t, f_t and o_t are the input, forget and output gates of the network, respectively. Figure 9.4 illustrates the recurrent module of an LSTM network. Hence, for an LSTM model $\mathcal{M}$, $\mathcal{R}_{t-1} = \{H_{t-1}, C_{t-1}\}$ and $\mathcal{P} = \{V_w^{i,f,o,c}, V_h^{i,f,o,c}\}$.

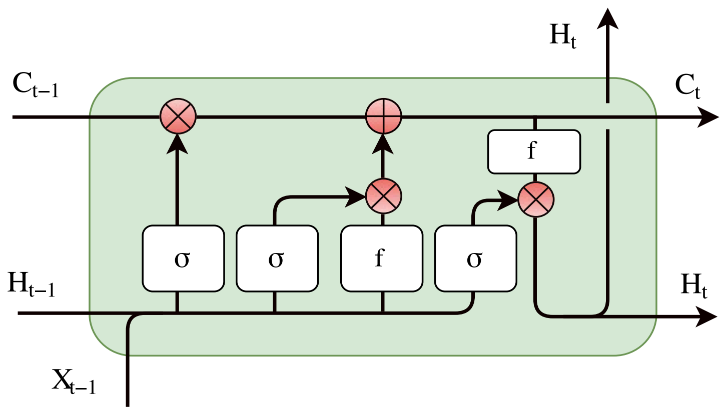


Figure 9.4: *Block diagram of the recurrent module of an LSTM network.*

Although LSTM models have been shown to outperform classical RNN in modeling long range dependencies, they do not explicitly model long/short context but rather use a single state to encode the global linguistic context.

10
Hybrid Neural Network Language Models

While dynamic context models, such as LSTM, Gated Recurrent Unit (GRU) and highway networks, propose to intrinsically model the linguistic dependencies with no explicit control over its range [145], there is another category of approaches that follow a *model-then-merge* approach to overcome the short context constraint and capture long range dependencies. That is, these approaches propose to use separate models to capture different linguistic properties (or range of dependencies) and then merge them. One of the early models that subscribe in this category was proposed by Bellegarda [118], who used Latent Semantic Analysis (LSA) to capture the global context, and then combined it with standard N-gram models, which capture the local context. In a similar but more recent approach, the authors of [146] have shown that RNN-LM performance can be significantly improved using an additional global topic information obtained using Latent Dirichlet Allocation (LDA). In fact, combining different models was proven to be very effective in improving language models and became the key to success for many new approaches. As a summary, recent work regarding model combination can be divided into two broad categories:

1. Model stacking: This category is inspired by computer vision neural models, which generally use different architectures to capture different features at the different layers of the network. For instance, the model proposed in [128] combines a CNN, an LSTM and a highway network to solve an LM task.

2. Output-level combination: These approaches propose to train separate models but combine them at the word prediction level. For instance, the maximum entropy RNN model,

introduced in [147], uses direct N-gram connections to the output layer, whereas standard linear interpolation uses pre-defined factors to perform the combination [148].

Following the model combination principle, the work presented in this thesis proposes new hybrid NNLMs that are developed as mixtures of classical architectures. In doing so, these models combine the learning capabilities of different NNLMs. The work presented here introduces three new models that can be summarized as follows:

1. SRNN-LM: This new model proposes to unfold the RNN model during word prediction (evaluation) as it is done during training. That is, the input word embeddings of an FNN model are updated based on the recurrent information, available at each word position, before propagating this information to the next layers. In fact, this model is inspired by residual networks, e.g.. [149], but it is applied to a recurrent model (Section 10.1).

2. LSRC-LM: This network proposes to separately and explicitly model short and long range dependencies using two separate RNN and LSTM models, respectively. The word prediction is performed after an update of the local context in a first step, which is then used to update the global context in a second step (Section 10.2).

3. Neural Mixture Model (NMM): While the two previous networks propose explicit combination of particular models, this new architecture is proposed as a generalization that does not impose any constraints on the number or type of networks to combine. More precisely, this model proposes to use a dynamic number of heterogeneous neural architectures as kernels to detect different features, which are then combined using an additional mixture layer (Section 10.3).

The remaining part of this chapter presents the mathematical formulation of each of these models in addition to their evaluation on different corpora, with a comparison to multiple state-of-the art models.

10.1 Sequential Recurrent Neural Network (SRNN) LMs

We have seen in Chapter 8 that recurrent models predict the next word based only on the last seen word w_{t-1}, and the recurrent context information from previous time step h_{t-1}. Therefore, they are expected to lose information about word position rather quickly, and cannot explicitly model short range dependencies/patterns as it is done in FNN and traditional N-grams. For example, English has position-dependent patterns such as "he $*$ he" (e.g., "he said he", "he mentioned he", ...). The position of "he" is essential for making the right prediction in this case, and the recurrent models are not designed to explicitly encode that. Rather, they are better for smooth incremental updates and hence for longer range dependencies.

We introduce in this section a novel approach that models short range dependencies like FNN and long range dependencies like RNN. In particular, the hidden layers combine explicit encoding of the local context with a recurrent architecture, which allows the context information to sequentially evolve in the network at the projection layer. In the first step, the word representations are enhanced using the context information. This step maps the word representations from a universal embedding space into a context-based space. Then, the model performs the next word prediction as it is typically done in FNN. The learning of the network parameters uses the Back-Propagation Through Time (BPTT) algorithm similarly to RNN. The main difference here is the additional network error resulting from the additional sequential connections. In addition to the new architecture, we also show that the learning of word-dependent sequential connections can substantially improve the performance of the proposed network.

10.1.1 SRNN as a Combination of FNN and RNN

The main difference between an RNN and an FNN model is the context representation. More precisely, The context layer H_t of an FNN architecture is estimated based on a fixed context size, i.e., the last $N-1$ words, whereas in an RNN model, H_t is constantly updated (at each time iteration) using only the last word and the recurrent context at time $t-1$. We propose here an architecture that captures short range dependencies over the last $N-1$ word positions as it is done in FNN, and the long range context through recurrence, similarly to RNN. This structure is motivated by the fact that although the last word in the history is crucial to the next word prediction for an RNN, the farther words still carry an additional information which improves the prediction. This is also the reason behind the slight improvement of FNN performance when increasing the context window.

The design of SRNN structure is motivated by the inefficiency of RNN to model position dependent patterns, which are particularly frequent in conversational speech. RNN loses information about word position quickly and therefore cannot efficiently model short range dependencies. FNN and N-gram models, however, are designed as position-dependent models, which deal only with short-term context.

Extending RNN structure to explicitly represent the short term history as it is done in FNN will 1) help improve the modeling of short range context, as it will 2) allow the network to capture any residual/additional context information that may be present in the past $i = t-N+1, \cdots, t-2$ time steps, but which may have been lost during the last context update that is based only on the last word at $t-1$ (See illustration in Figure 10.1). In the worst case scenario, the context information will be simply redundant and is expected to not harm the performance. In fact, this new architecture is inspired by residual networks [149], which achieve state-of-the-art performance on computer vision tasks. Thus, this model can be seen as an attempt to extend a recurrent network using residual/sequential connections.

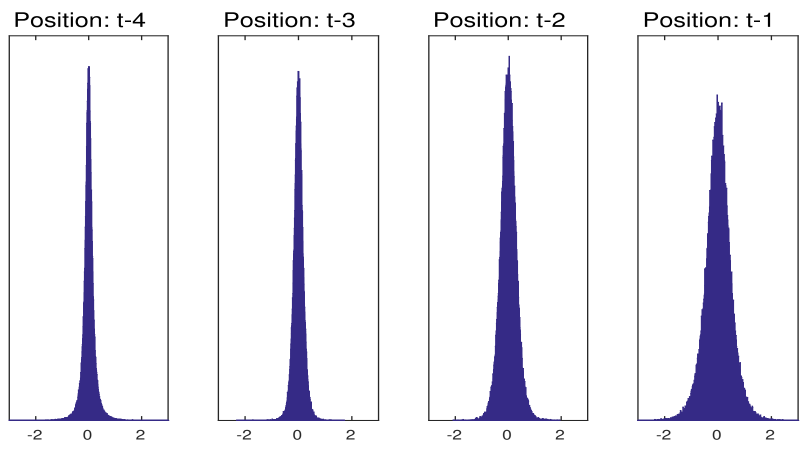


Figure 10.1: *Histograms of the projection-to-hidden weights V_1, V_2, V_3 and V_4 (see Figure 10.2) for each of the 4 word positions of an SRNN (N=5) trained on LTCB. These histograms show that the magnitude of the weights decays with the word position (from $t-1$ to $t-4$) but does not nullify. Thus, the farther word positions still capture some residual/additional context.*

10.1.2 SRNN-based Language Modeling

An SRNN of order $N-1$ approximates (9.1) according to:

$$p(w_1^T) \approx \prod_{t=1}^{T} p(w_t | w_{t-N+1}^{t-1}, h_{t-N+1}) = \prod_{t=1}^{T} p(w_t | h_{t-N+2}^{t}). \tag{10.1}$$

The proposed SRNN model approximates the probability of a word sequence, given by (9.1), according to (10.1). We can see that this approximation explicitly represents the history over the last $N-1$ word positions, as it is done in the approximation of FNN given by (9.2), while it enhances the actual word representations using the recurrent context information, which propagates sequentially within the network (see illustration in Figure 10.2). Furthermore, restricting the context to a one-word history window (N=2) in (10.1) leads to the RNN approximation given by (9.5). Therefore, the proposed approach can be seen as an extension of the standard RNN to explicitly model and capture short range context.

The additional sequential connections allow the context information to propagate from the past to the future within the network. These connections can be defined as a Word-Independent (WI) recurrence vector, which fixes the amount of context information allowed to propagate in the network, as they can be designed as Word-Dependent (WD) vectors. In this case, each word will have its own context weight vector, which will learn to scale the context "neurons" according to their relevance for that particular word.

The network evaluation is performed similarly to FNN, the main difference occurs in (9.8), which becomes in the case of the word-independent model:

$$P_{t-i} = f_s(X_{t-i} \cdot U + C \odot P_{t-i-1}), \quad i = N-1, \cdots, 1, \tag{10.2}$$

whereas it becomes in the case of the word-dependent model:

$$P_{t-i} = f_s(X_{t-i} \cdot U + C_{w_{t-i}} \odot P_{t-i-1}),\ i = N-1, \cdots, 1. \tag{10.3}$$

$f_s(\cdot)$ is a sequential activation function and $\odot$ is the element-wise product operator. C is the word-independent recurrence weight vector, whereas $C_{w_{t-i}}$ is the word-dependent context weights corresponding to the word w_{t-i}. Note that C can also be defined as a full matrix, instead of a vector, as it is done in the standard RNN. The work conducted in this thesis, however, showed that using a simple vector is sufficient to improve the performance. Figure 10.2 shows an example of an SRNN with three additional sequential connections ($N = 4$) and a single hidden layer.

We can see that the proposed SRNN model is a general architecture that includes different networks. In particular, setting the context weights to $C = [0, \ldots, 0]$ and using an identity activation function, i.e., $f_s(x) = x$, leads to the standard FNN architecture, whereas using a one-word context, $N = 2$, results in the standard RNN architecture with a diagonal recurrence matrix and an additional non-recurrent layer. Moreover, setting C to a fixed value in $[0, 1]$ and $f_s(x) = x$ leads to the Fixed-size Ordinally-Forgetting Encoding (FOFE) [144] architecture, which was proposed to uniquely encode word sequences.

Regarding the word representations, SRNN replaces the universal word embeddings at the projection layer of an FNN, i.e., $\{E_{t-i}\}_{i=1}^{N-1}$, by context-dependent word embeddings $\{P_{t-i}\}_{i=1}^{N-1}$. More particularly, both equations given by 10.2 and 10.3 show that each word representation is enhanced using the context information before proceeding to the next word prediction. Therefore, we can interpret this particular step as a transformation from the universal embedding space into the context-dependent space for a better discrimination of word representations.

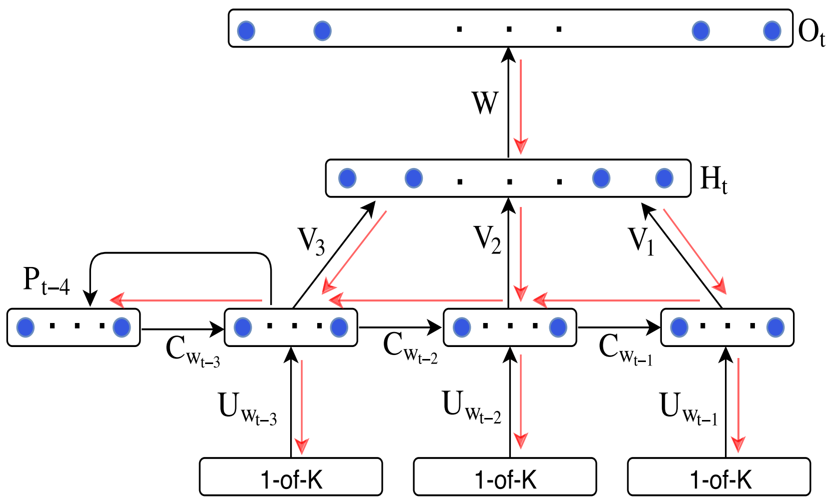


Figure 10.2: *SRNN architecture. The backward path (red arrows) shows the error propagation during training (unfolding of the network during BPTT is not shown here).*

10.1.3 Training of SRNN-LMs

The parameters to train for an SRNN architecture are the word embeddings U, the project-to-hidden weights $V = [V_1, \cdots, V_{N-1}]$, the hidden-to-output weights W and the context weight vector C for the WI model, or $C = [C_1^\intercal, \cdots, C_K^\intercal]^\intercal$ (K is the vocabulary size) for the WD model. In this case, each word w in the vocabulary will be characterized by two learnable embeddings, namely, the continuous representation U_w, which is also learned by other NN models, in addition to the context weight C_w, which is newly introduced by SRNN. During training, we also add the constraint that each $C_w(i) \in [0,1]$, which reflects their role as scaling factors.

Similarly to RNN, the parameter learning of an SRNN architecture follows the standard BPTT algorithm. The main difference occurs at the projection layer, where the additional error vectors, resulting from the sequential connections, should be taken into account (See example of error propagation in Figure 10.2) before unfolding the network in time.

10.1.4 SRNN Evaluation

10.1.4.1 Experimental Setup

To evaluate the proposed SRNN approach, we conducted a set of LM experiments on the PTB and LTCB corpora, which are introduced in Section 8.3.1.1 and Section 8.3.1.2, respectively.

The proposed SRNN approach is compared to different models, including the Kneser-Ney (KN) N-gram model and different feed-forward and recurrent neural architectures. For feed-forward networks, the baseline systems include FNN-based LM [122] as well as the 2) FOFE approach, which was proposed as a sentence-based, feed-forward model [144]. The FOFE results were obtained using the FOFE toolkit [1] [144]. The results are reported for different N-gram sizes (N=2,3 and 5) and different numbers of hidden layers (1 or 2). Regarding recurrent models, we compare SRNN to 3) the original RNN without classes [125], 4) to RNN with maximum entropy (RNNME) [147], 5) to a Deep RNN (D-RNN) [2] [150], which investigates different ways of adding hidden layers to RNN, and finally 5) to the LSTM architecture [127], which explicitly regulates the amount of information that propagates in the network. For a complete analysis, we will also report the performance of other models on the PTB during our discussion below.

10.1.4.2 SRNN Evaluation on PTB

For the PTB experiments, the FNN, FOFE and SRNN architectures have similar configurations. That is, the hidden layer(s) size is 400 with all hidden units using the Rectified Linear Unit

[1]All the data processing steps described in this thesis for PTB and LTCB were performed using the FOFE toolkit in [144], which is available at `https://wiki.eecs.yorku.ca/lab/MLL/_media/projects:fofe:fofe-code.zip`.

[2]The deep RNN results were obtained using L_p and maxout units, dropout regularization and gradient control techniques, which are known to significantly improve the performance. None of these techniques, however, were used in our experiments.

(ReLu), i.e., $f(x) = \max(0, x)$, as activation function, whereas the word representation (embedding) size was set to 200 for FNN, FOFE and LSTM and 100 for SRNN. Moreover, all recurrent models use $f = \tanh(\cdot)$ as activation function for all recurrent layers, including the sequential activation function of SRNN, whereas $f = \text{sigmoid}(\cdot)$ is used for the input, forget and output gates of LSTM. The hidden layer size of RNN and LSTM were set to 400 and follow the original configuration proposed in [125] and [127], respectively. For LSTM, we also report the performance after tuning the model, i.e., after optimizing the learning parameters. We also use the same learning setup adopted in [144], namely, we use the stochastic gradient descent algorithm with a mini-batch size of 200, the learning rate is initialized to 0.4, the momentum is set to 0.9, the weight decay is fixed at 4.10^{-5} and the training is done in epochs. The weights initialization follows the normalized initialization proposed in [151]. Similarly to [124], the learning rate is halved when no significant improvement in the log-likelihood of the validation data is observed. Then, we continue with seven more epochs while halving the learning rate after each epoch. The BPTT was set to 5 time steps. In the tables below, WI-SRNN refers to the word-independent SRNN model proposed in (10.2), whereas WD-SRNN refers to the word-dependent model introduced in (10.3). For both models, the context connection weights C were randomly initialized in $[0,1]$. In order to compare to the FOFE approach, we also report results when C is reduced to a scalar forgetting factor that is fixed at 0.7. This is denoted as WI-SRNN* in the tables below.

	Model			Model+KN5			NoP	TS (w/s)
N-1=	1	2	4	1	2	4	4	4
	1 Hidden Layer							
FNN	176	131	119	132	116	107	6.32M	24.3K
FOFE	123	111	112	108	100	101	6.32M	17.2K
WI-SRNN*	117	110	109	105	100	99	5.16M	12.9K
WI-SRNN	112	107	107	102	98	97	5.16M	11.2K
WD-SRNN	109	106	106	99	96	95	6.16M	10.4K
	2 Hidden Layers							
FNN	176	129	114	132	114	102	6.48M	21.8K
FOFE	116	108	109	104	98	97	6.48M	16.6K
WI-SRNN*	114	108	107	102	98	96	5.32M	10.8K
WI-SRNN	109	105	104	99	96	94	5.32M	9.6K
WD-SRNN	108	103	104	97	94	94	6.32M	9.2K
	Recurrent Models							
RNN	117			104			8.16M	20.6K
Deep RNN	107.5			—			6.96M	—
LSTM	113			99			6.96M	7.6K
Tuned LSTM	105			98			6.96M	7.6K

Table 10.1: SRNN performance on the PTB test set.

Table 10.1 shows the LMs evaluation on the PTB test set. We can clearly see that the proposed approach outperforms all other models (on par with the tuned LSTM) using the lowest number of model parameters among all configurations. This also includes other models that were reported in the literature, such as RNN with maximum entropy [147], random forest LM [152], structured LM [153] and syntactic neural network LM [154]. More particularly, SRNN with two hidden layers achieves a comparable performance to a mixture of RNNs [148]. We can also conclude that the explicit modeling of short range dependencies through sequential connections improves the performance. More precisely, the results show that increasing the history window (1, 2 and 4) improves the performance of all SRNN models. Table 10.1 also shows that using a fixed forgetting factor scalar (WI-SRNN*) leads to a slight improvement over FOFE approach, which is mainly due to the additional non-linear activation function f_s. Furthermore, the word-dependent (WD-SRNN) model slightly outperforms the word-independent model (WI-SRNN) but with a significant increase in the number of parameters. Regarding the training speed, we can conclude that training an SRNN model requires approximately twice the time needed for FNN and RNN, whereas it needs less time compared to LSTM.

10.1.4.3 SRNN Evaluation on LTCB

The LTCB experiments use the same PTB experimental setup reported above with few minor changes, which are mainly due to the significant training time required for this corpus. The results shown in Table 10.2 follow the same experimental setup used in [144]. More precisely, these results were obtained without usage of momentum or weight decay, whereas the mini-batch size was set to 400. FNN and FOFE architectures contain 2 hidden layers of size 600 (or 400), whereas RNN and SRNN have a single hidden layer of size 600. In order to compare SRNN to FOFE [144], the forgetting factor C of WI-SRNN* was fixed at 0.6.

	Model			NoP
Context Size M=N-1	1	2	4	4
KN	239	156	132	—
FNN [M*200]-600-600-80k	235	150	114	64.84M
FOFE [M*200]-400-400-80k	120	115	108	48.48M
FOFE [M*200]-600-600-80k	112	107	100	64.84M
WI-SRNN* [M*200]-600-80k	110	102	94	64.48M
WI-SRNN [M*200]-600-80k	85	80	77	64.48M
WD-SRNN [M*200]-600-80k	77	74	72	80.48M
RNN [600]-600-80k	85			96.36M

Table 10.2: SRNN performance on the LTCB test set.

The LTCB results shown in Table 10.2 generally confirm the PTB conclusions. In particular, we can see that SRNN outperforms all other models while requiring comparable or fewer model parameters. Moreover, the WI-SRNN* model with a single hidden layer slightly outperforms FOFE (2 hidden layers), whereas WI-SRNN achieves a comparable performance to RNN (for

N=2). These results, however, show a more significant improvement for the WD-SRNN model and for the increased window size (from 1 to 4) compared to the improvement obtained on the PTB corpus. This is mainly due to the larger amount of training data available in the LTCB corpus, which allows us to train richer WD context vectors.

in		strictly		germany	
U_w	C_w	U_w	C_w	U_w	C_w
into	at	solely	purely	italy	japan
throughout	on	rigidly	totally	france	russia
through	for	broadly	physically	britain	italy
during	their	purely	solely	switzerland	france
within	to	ostensibly	technically	england	spain
towards	an	generally	fairly	spain	britain
across	by	essentially	equally	hungary	finland
on	from	largely	freely	poland	canada
toward	a	rigorously	basically	belgium	turkey
around	his	expressly	obviously	austria	switzerland

Table 10.3: Examples of top 10 similar words.

Table 10.3 shows some word examples with their top 10 cosine similarities for word embeddings U_w and Euclidean distance for context weights C_w. These examples show a general trend; not valid for every example, that the embeddings capture semantic (conceptual) similarities and the context weights model syntactic (functional) similarities. Confirming this hypothesis, however, was not conducted in this thesis.

10.2 Long-Short Range Context (LSRC) LMs

We introduced in Chapter 8 two standard recurrent models to capture long range dependencies, namely, RNN and LSTM, as we have seen that LSTM is a better choice due to its capability to dynamically adapt the context modeling through the different gates it uses. RNN, however, uses static connection weights, which cannot capture the dynamic aspect of the context information. In fact, the authors of [126] have shown that, in practice, the captured information changes quickly in the classical RNN structure [124], and that it is experimentally equivalent to an 8-gram FNN model. Therefore, we can say that RNN is rather a local context model than a long term one. Furthermore, while LSTM can adapt its learning to capture dynamic context, it does not differentiate between long and short range dependencies as it is typically done is multi-span models. This is mainly due to its usage of a single hidden state to model the optimal information with no direct control over its range. The work presented in this section proposes an extension of LSTM to achieve this goal.

10.2.1 Multi-Span Language Models

We have seen in the introduction of Chapter 10 that many language modeling techniques propose to explicitly learn short range dependencies, which generally model the syntactic properties of a language and/or long range dependencies, which are semantic in nature. These attempts to learn and combine these two types of dependencies led to what is known as multi-span LMs [118]. The goal of these models is to learn the various constraints, both local and global, that are present in a language. This is typically done using two different models, which separately learn the local and global context, and then combine their resulting linguistic information to perform the word prediction. For instance, the authors of [155] proposed to use LSA to capture the global context, and then combined it with the standard N-gram models, which capture the local context, whereas [146] proposed to model the global topic information using LDA, which is then combined with an RNN-based LM. This idea is not particular to language modeling but has also been used in other natural language processing tasks, e.g., the authors of [156] used a local/global model to perform a spoken language understanding task.

We will continue along this line by proposing a new multi-span architecture, which separately models the short and long context information while it dynamically merges them to perform the language modeling task. This is done through a novel recurrent Long-Short Range Context (LSRC) network, which explicitly models the local (short) and global (long) context using two separate hidden states that evolve in time. This new architecture is an adaptation of the LSTM cell to take into account the linguistic properties.

10.2.2 LSRC-based Language Modeling

Following the line of thoughts in [146, 155], we introduce a new multi-span model, which takes advantage of the LSTM ability to model long range context while, simultaneously, learning and integrating the short context through an additional recurrent, local state. In doing so, the resulting LSRC network is able to separately model short and long context while it dynamically combines them to predict the next word. Formally, the LSRC model is defined as:

$$E_{t-1} = X_{t-1} \cdot U \tag{10.4}$$

$$H_t^l = f\left(E_{t-1} + U_l^c \cdot H_{t-1}^l\right), \tag{10.5}$$

$$\{i, f, o\}_t = \sigma\left(V_l^{i,f,o} \cdot H_t^l + V_g^{i,f,o} \cdot H_{t-1}^g\right), \tag{10.6}$$

$$\tilde{C}_t = f\left(V_l^c \cdot H_t^l + V_g^c \cdot H_{t-1}^g\right), \tag{10.7}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \tag{10.8}$$

$$H_t^g = o_t \odot f(C_t), \tag{10.9}$$

$$O_t = g\left(W \cdot H_t^g\right). \tag{10.10}$$

The learning of an LSRC model requires the training of the local parameters $V_l^{i,f,o,c}$ and U_l^c, the global parameters $V_g^{i,f,o,c}$, the word embeddings U and the hidden-to-output connection weights W. Similarly to SRNN, this is achieved using the BPTT algorithm, which is typically used to train recurrent networks.

LSRC uses two hidden states, namely, H_t^l and H_t^g to model short and long range context, respectively. More particularly, the local state H_t^l evolves according to (10.5), which is nothing but a simple recurrent model that evolves according to (9.10). In doing so, H_t^l is expected to have a similar behavior to RNN, which has been shown to capture local/short context (up to 10 words), whereas the global state H_t^g follows the LSTM model, which is known to capture longer dependencies (see example in Figure 10.4). The main difference here, however, is the dependence of the network modules (gates and memory candidate) on the previous local state H_t^l instead of the last seen word E_{t-1}. This model is based on the assumption that the local context carries more linguistic information and is, therefore, more suitable to combine with the global context and to update LSTM compared to the last seen word. Figure 10.3 illustrates the recurrent module of an LSRC network. It is worth mentioning that this model was not particularly developed to separately learn syntactic and semantic information. This may come, however, as a result of the inherent local and global nature of these two types of linguistic properties.

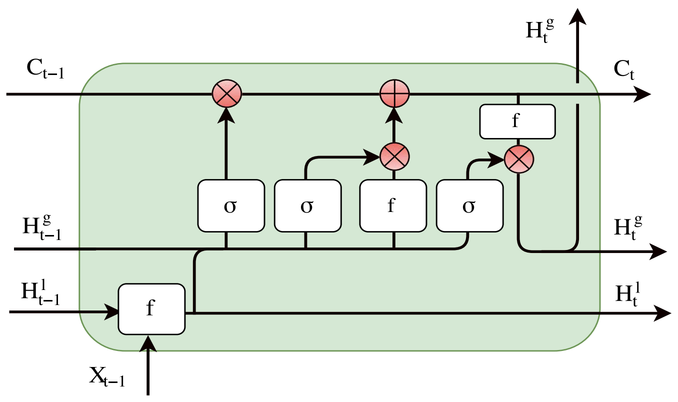


Figure 10.3: *Block diagram of the recurrent module of an LSRC network.*

10.2.3 Context Range Estimation of LSRC

For many NLP applications, capturing the global context information can be a crucial component to develop successful systems. This is mainly due to the inherent nature of language, where a single topic can span over few sentences, paragraphs or a complete document. LSA-like approaches take advantage of this property and aim at extracting some hidden "concepts" that best explain the data in a low-dimension "semantic space". To some extent, the hidden layer of LSRC/LSTM can be seen as a vector in a similar space. The information stored in this vector, however, changes continuously based on the processed words. Moreover, interpreting its content is generally difficult. As an alternative, measuring the temporal correlation of this hidden vector can be used as an indicator of the ability of the network to model short and long context dependencies. Formally, the temporal correlation of a hidden state H over a distance d can be calculated according to:

$$c_d = \frac{1}{T}\sum_{t=1}^{t=T} \mathrm{SM}\,(H_t, H_{t+d}), \tag{10.11}$$

where T is the test data size in words and “SM” is a similarity measure such as the *cosine similarity*. This measure allows us to evaluate how fast does the information stored in the hidden state change over time.

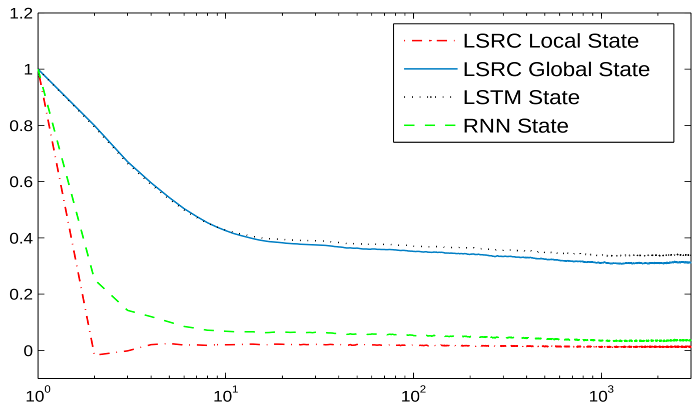


Figure 10.4: *Temporal correlation of the LSRC model in comparison to LSTM and RNN.*

Figure 10.4 shows the variation of this temporal correlation for the local and global states of the proposed LSRC network, in comparison to RNN and LSTM, for various values of the distance d (up to 3000 words). This figure was obtained on the test set of the PTB corpus, described in Section 10.2.4. The main conclusion we can draw from this figure is the ability of the LSRC local and global states (trained jointly) to behave in a similar fashion compared to RNN and LSTM states (trained separately), respectively. We can also conclude that the LSRC global state and LSTM are able to capture long range correlations, whereas the context changes rapidly over time in RNN and the LSRC local state.

10.2.4 LSRC Evaluation

The evaluation of the proposed LSRC model follows a similar experimental setup to that described in Section 10.1.4. That is, we ran a set of experiments on the PTB corpus and LTCB and then compared it to the same set of baseline models, which include 1) the commonly used Kneser-Ney (KN) N-gram model [113] with and without cache [157], 2) the FNN-based LM [122] and 3) the FOFE approach [144]. For these short size context models, we report the results of different context sizes (1, 2 and 4). Regarding the recurrent models, we compared LSRC to 4) the original RNN [125], 5) to a Deep RNN (D-RNN)[2] [150] and finally 6) to the LSTM model [127].

The performance of the recurrent models is reported for different numbers of hidden layers (1 or 2). In order to investigate the impact of deep models on the LSRC architecture, we added a single hidden, non-recurrent layer (of size 400 for PTB and 600 for the LTCB experiments) to the LSRC model (D-LSRC). This was sufficient to improve the performance with a negligible increase in the number of parameters required by the model.

10.2.4.1 LSRC Evaluation on PTB

For the PTB experiments, FNN, FOFE, RNN and LSTM models use the same experimental setup used in SRNN evaluation. The LSRC configuration was setup such that it follows the experimental setup used by these models. That is, the word embedding size was set to 200 and 100, i.e., LSRC(100), respectively, this choice was made to illustrate the effectiveness of the LSRC model when using a small embedding size, the training uses the BPTT algorithm for 5 time steps, the mini-batch was set to 200 and the additional non-recurrent layer in D-LSRC uses the ReLu activation function. The learning rate, however, was set to 1.0 and the weight decay was fixed at 5×10^{-5}, whereas the use of the momentum did not lead to any additional improvement. Moreover, the data is processed sequentially without any sentence independence assumption. Thus, the recurrent models will be able to capture long range dependencies that exist beyond the sentence boundary.

	Model			Model+KN5			NoP
N-1=	1	2	4	1	2	4	4
KN	186	148	141	—	—	—	—
KN+cache	168	134	129	—	—	—	—
	1 Hidden Layer						
FNN	176	131	119	132	116	107	6.32M
FOFE	123	111	112	108	100	101	6.32M
	Recurrent Models (1 Layer)						
RNN	117			104			8.16M
LSTM (1L)	113			99			6.96M
Tuned LSTM (1L)	105			98			6.96M
LSRC(100)	109			96			5.81M
LSRC(200)	104			94			7.00M
	2 Hidden Layers						
FNN	176	129	114	132	114	102	6.96M
FOFE	116	108	109	104	98	97	6.96M
	Deep Recurrent Models						
D-LSTM (2L)	103			97			8.42M
D-RNN^2 (3L)	107.5			NR			6.16M
D-LSRC(100)	103			93			5.97M
D-LSRC(200)	102			92			7.16M

Table 10.4: LSRC performance on the PTB test set.

Table 10.4 shows the performance of different LMs on the PTB test set. As a first observation, we can clearly see that the LSRC approach outperforms all other models for all configurations, in particular, RNN and LSTM. More particularly, we can conclude that LSRC, with an embedding size of 100, achieves a better performance than all other models, except the tuned LSTM, while reducing the number of parameters by $\approx 29\%$ and $\approx 17\%$ compared to RNN and LSTM, respectively. Increasing the embedding size to 200, which is used by the other models, improves its performance with a resulting NoP comparable to LSTM. The significance of the improvement obtained here over LSTM was confirmed through a statistical significance t-test, which led to p-values $\leq 10^{-10}$ for a significance level of 5% and 0.01%, respectively.

The results of the deep models in Table 10.4 also show that adding a single non-recurrent hidden layer to LSRC can significantly improve the performance. In fact, the additional layer bridges the gap between the LSRC models with an embedding size of 100 and 200, respectively. The resulting architectures outperform the other deep recurrent models with a significant reduction of the number of parameters (for the embedding size of 100), and without usage of dropout regularization, L_p and maxout units or gradient control techniques compared to the deep RNN^2(D-RNN) model.

We can conclude from these experiments that the explicit modeling of short and long range dependencies using two separate hidden states improves the performance while significantly reducing the number of parameters.

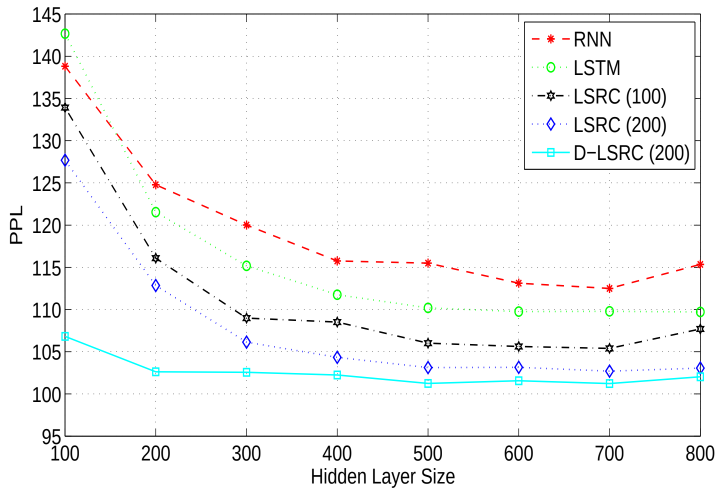


Figure 10.5: *Perplexity of different NNLMs with different hidden layer sizes on the PTB test set.*

In order to show the consistency of the LSRC improvement over the other recurrent models, we report the variation of the models performance with respect to the hidden layer size, without tuning, in Figure 10.5. This figure shows that increasing the LSTM or RNN hidden layer size could not achieve a similar performance compared to the one obtained using LSRC with a small layer size (e.g., 300). It is also worth mentioning that this observation holds when comparing an LSTM with two recurrent layers to an LSRC with an additional non-recurrent layer.

10.2.4.2 LSRC Evaluation on LTCB

The LTCB experiments follow the same experimental setup reported in Section 10.1.4. In addition to the results reported for SRNN, we also report here the performance of LSTM with 1 and 2 recurrent layers, respectively, whereas we add a non-recurrent layer to LSRC. The recurrent layers are marked with an "R" in Table 10.5.

	Model			NoP
Context Size M=N-1	1	2	4	4
KN	239	156	132	—
KN+cache	188	127	109	—
FNN [M*200]-600-600-80K	235	150	114	64.84M
FOFE [M*200]-400-400-80K	120	115	108	48.48M
FOFE [M*200]-600-600-80K	112	107	100	64.84M
RNN [600]-R600-80K	85			96.36M
LSTM [200]-R600-80K	66			65.92M
LSTM [200]-R600-R600-80K	61			68.80M
LSRC [200]-R600-80K	63			65.96M
LSRC [200]-R600-600-80K	59			66.32M

Table 10.5: LSRC performance on the LTCB test set.

The results shown in Table 10.5 generally confirm the conclusions we drew from the PTB experiments above. That is, we can see that the proposed LSRC model largely outperforms all other models. In particular, LSRC clearly outperforms LSTM with a negligible increase in the number of parameters (resulting from the additional $200 \times 200 = 0.04M$ local connection weights U_l^c) for the single layer results. We can also see that this improvement is maintained for deep models (2 hidden layers), where the LSRC model achieves a slightly better performance while reducing the number of parameters by $\approx 2.5M$ and speeding-up the training time by $\approx 20\%$ compared to the two layers LSTM model.

The PTB and LTCB results clearly highlight the importance of recurrent models to capture long range dependencies for LM tasks. The training of these models, however, requires large amounts of data to significantly outperform short context models. This can be seen in the performance of RNN and LSTM on the PTB and LTCB corpora, reported in the tables above. We can also conclude from these results that the explicit modeling of long and short context in a multi-span model can lead to an additional improvement over state-of-the-art models.

10.3 Neural Mixture Models (NMMs)

Motivated by the model combination principle (see discussion in the beginning of this chapter) that is commonly used to improve the modeling capabilities of standard LMs, we have introduced in Section 10.1 a new neural LM, which tightly combines FNN and RNN architectures. In doing so, the new model is able to enhance the learning capabilities of a recurrent architecture through explicit modeling of word position information. Following the same line of thoughts, we proposed in Section 10.2 a different architecture that combines LSTM and RNN to explicitly and separately model short and long range dependencies. Experiments conducted on the PTB corpus and LTCB have shown that these hybrid models lead to significant improvements when compared to the performance of the separate models.

We propose in this section a generalized framework to combine different heterogeneous NN-based architectures using a single mixture model. More particularly, the proposed architecture uses 1) a hidden feature layer to separately learn each of the models to be combined, and 2) a hidden mixture layer, which combines the resulting model features. Moreover, this architecture uses a single word embedding matrix, which is learned from all models, and a single output layer. This framework is, in principle, able to combine different NN-based LMs (e.g., FNN, RNN, LSTM, etc.) with no direct constraints on the number of models to combine or their configuration.

10.3.1 Model Combination for Language Modeling

On the contrary to a large number of research directions on improving or designing particular neural architectures for language modeling, the work presented in this section is an attempt to design a general architecture, which is able to combine different types of existing heterogeneous models rather than investigating new ones.

The work presented here is motivated by recent research showing that model combination can lead to a significant improvement in LM performance [148]. This is typically done either 1) by designing deep networks with different architectures at the different layers, as it was done in [128], this category of model combination, however, requires a careful selection of the architectures to combine for a well-suited feature design, as it can be difficult/slow to train, or 2) by combining different models at the output layer, as it is done in the maximum entropy RNN model [147] or using the classical linear interpolation [148]. This category typically leads to a significant increase in the number of parameters when combining multiple models.

Our previous two attempts to circumvent these problems, namely SRNN and LSRC, were solely designed to combine particular models, therefore, they are not well-suited to generalize to other architectures. This section continues along this line of work by proposing a general architecture to combine different heterogeneous neural models with no direct constraints on their number or type.

10.3.2 NMM-based Language Modeling

This section introduces the mathematical formulation of the proposed mixture model, which uses a variable number of basic architectures as kernels to combine. The latter share the output layer and the word embeddings, which are trained jointly while the parameters of each model in the mixture are trained separately. The work conducted in thesis mainly focuses on the basic architectures as possible components in the mixture.

To introduce the mathematical formulation of NMM, let $\{\mathcal{M}_m\}_{m=1}^M$ be a set of M models to combine, and let $\{\mathcal{P}_m, \mathcal{R}_m^t\}_{m=1}^M$ be their corresponding model parameters and recurrent information at time t, respectively. For the basic NNLMs, namely FNN, RNN and LSTM, $\mathcal{M}_m$, $\mathcal{P}_m$ and $\mathcal{R}_m^t$ were introduced in Section 9.2. Moreover, let U be the shared word embedding matrix, which is learned during training from all models in the mixture. The neural mixture model is given by the following steps (see illustration in Figure 10.6):

1. *Feature layer: update each model and calculate its features (i.e., its last hidden layer):*

$$H_m^t = \mathcal{M}_m(\mathcal{P}_m, \mathcal{R}_m^{t-1}, U), \quad m = 1, \cdots, M. \tag{10.12}$$

2. *Mixture layer: combine the different features:*

$$H_{\text{mixture}}^t = f_{\text{mixture}}\left(\sum_{m=1}^{M} H_m^t \cdot S_m\right). \tag{10.13}$$

3. *Output layer: calculate the output using a softmax function:*

$$O^t = g\left(H_{\text{mixture}}^t \cdot W\right). \tag{10.14}$$

f_{mixture} is a non-linear mixing function, whereas S_m, $m = 1, \cdots, M$ are the mixture weights (matrices). Although the experiments conducted in this work mainly focus on FNN, RNN and LSTM, the set of possible model selection for $\mathcal{M}_m$ is not restricted to these but includes all NN-based models that take word embeddings as input.

The proposed mixture model uses a single word embedding matrix and a single output layer with predefined and fixed sizes. The latter are independent of the sizes of the mixture models. In doing so, this model does not suffer from the significant parameter growth when increasing the number of models in the mixture. We can also see that this architecture does not impose any direct constraints on the number of models to combine, their size or their type. Hence, we can combine for instance models of the same type but with different sizes/configurations, as we can combine heterogeneous models, such as recurrent and non-recurrent models, in a single mixture. Moreover, the mixture models can also be deep architectures with multiple hidden layers.

10.3.3 Training of NMMs and Model Dropout

NMM training follows the standard back-propagation algorithm used to train neural architectures. More particularly, the error at the output layer is propagated to all models in the mixture. At this stage, each model receives a network error, updates its parameters and propagates its error to the shared word embedding (input) layer. We should also mention here that recur-

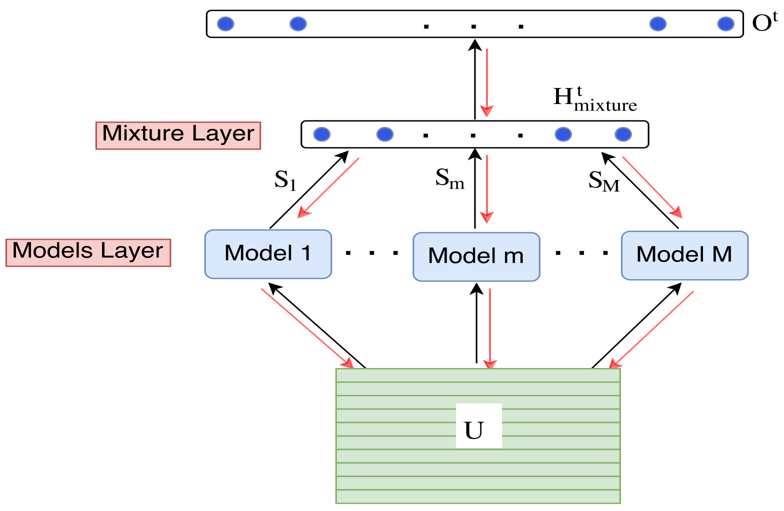


Figure 10.6: *NMM architecture. Red (back) arrows show the error propagation during training.*

rent models can be "unfolded" in time, independently of the other models in the mixture, as it is done for standard networks. Once each model is updated, the continuous word representations are then updated as well while taking into account the individual network errors emerging from the different models in the mixture (see illustration in Figure 10.6).

The joint training of the models in the mixture is expected to lead to a *complementarity* effect. We mean by complementarity here that the mixture models perform poorly when evaluated separately but lead to a much better performance when tested jointly. This is expected to be a result of the models learning and modeling, eventually, different features. Moreover, the joint learning is also expected to lead to a richer and more expressive word embeddings.

In order to 1) enforce models co-training and 2) avoid network over-fitting when the number of models in the mixture is large. We use a model dropout technique, which is inspired by the standard dropout regularization [133] that is widely used to train neural networks. The idea here is to have "models" replace "neurons" in the standard dropout. Therefore, for each training example, a model is to be dropped with a probability p_d. Then, only models that are selected contribute to the mixture and have their parameters and mixing weights S_m updated. Similarly to standard dropout, model dropout is applied only to the non-recurrent mixture models.

10.3.4 NMM Evaluation

To evaluate NMM, we follow the same experimental setup we used to evaluate SRNN and LSRC. Section 10.1.4 gives a detailed description of this setup including the baseline LMs. The comparison and analysis also includes our previous models, namely, SRNN and LSRC.

Although the proposed approach was not designed for a particular mixture of models, we only report results for different combinations of FNN, RNN and LSTM, which are considered to be the baseline NNLMs. For clarity, an NMM result is presented as $F^{N_1,\cdots,N_f}_{S_1,\cdots,S_f} + R_{S_1,\cdots,S_r} + L_{S_1,\cdots,S_l}$, where f is the number of FNN models in the mixture, $S_m, m = 1, \cdots, f$ are their corresponding hidden layer sizes (that are fed to the mixture) and $N_m, m = 1, \cdots, f$ are their fixed history sizes. The same notation holds for RNN and LSTM, where r and l represent the number of RNN and LSTM models in the mixture, respectively, with $N_r, N_l = 1$. The number of models in the mixture is given by $f + r + l$. Moreover, the notation $F_{S_f}^{N_b - N_e}$ means that this model combines $N_e - N_b + 1$ consecutive FNN models with respective history sizes $N_b, N_b + 1, \cdots, N_e$, with all models having the same hidden layer size S_f.

10.3.4.1 NMM Evaluation on PTB

For the PTB experiments, FNN, RNN and LSTM models use the same experimental setup used in SRNN and LSRC evaluation, with some minor differences. That is, all models have a hidden layer size of 400, with FNN and SRNN using the ReLu as activation function and having 2 hidden layers. ReLu is also used as activation function for the mixture layer in the NMM models, which use a single hidden layer. The embedding size is 100 for SRNN and NMM, whereas it is set to 400 for RNN and 200 for FNN and LSTM. The training is performed using the stochastic gradient descent algorithm with a mini-batch size of 200. As a reminder of the training configuration, the learning rate is initialized to 0.4, the momentum is set to 0.9, the weight decay is fixed at 4×10^{-5}, the model dropout is set to 0.4 and the training is done in epochs. The weights initialization follows the normalized initialization proposed in [151]. Similarly to [124], the learning rate is halved when no significant improvement in the log-likelihood of the validation data is observed. Then, we continue with seven more epochs while halving the learning rate after each epoch. The BPTT was set to 5 time steps for all recurrent models.

In addition to the perplexity (PPL), and Number of model Parameters (NoP) metrics, which were used to evaluate the previous models, we also report here the performance in terms of Parameter Growth (PG) for NMM, which is defined as the relative increase in the number of parameters of NMM w.r.t. the baseline model in the table. In order to demonstrate the power of the joint training, we also report the perplexity PPL and NoP of the Linearly Interpolated (LI) models in the mixture after training them separately. In this case, each model learns its own word embedding and output layer.

The PTB results reported in Table 10.6 show clearly that combining different small-size models with a reduced word embedding size results in a better perplexity performance compared to the baseline models, with a significant decrease in the NoP required by the mixture. More particularly, we can see that adding a single FNN model to a small size LSTM or RNN is sufficient to outperform the baseline models while reducing the number of parameters by 24% and 36%, respectively. The same conclusion can be drawn when combining an RNN with an LSTM. We can also see that adding more FNN models to each of these mixtures leads to additional improvements while keeping the number of parameters significantly small. Table 10.6 also shows that training the small size models (in the mixture) separately, and then linearly interpolating them, results in a slightly worse performance compared to the mixture model with

	PPL	NoP	PG	PPL(LI)	NoP(LI)
FNN (N=5)	114	6.49M	—	—	—
$F_{200}^{2,3}$	117	5.27M	-18.8%	120.0	6.10M
F_{200}^{2-5}	110	5.61M	-13.6%	112.0	12.28M
LSTM	105	6.97M	—	—	—
LSRC	104	7.0M	—	—	—
$L_{100}+F_{200}^{2}$	102	5.25M	-24.7%	114	5.12M
$L_{100}+R_{100}$	102	5.18M	-25.7%	118	4.09M
RNN	117	8.17M	—	—	—
RNNME	117	10G	—	—	—
WD-SRNN	104	6.33M	—	—	—
WI-SRNN	104	5.33M	—	—	—
$R_{100}+F_{200}^{2}$	109	5.18M	-36.6%	119	5.05M
$R_{100}+F_{200}^{2-6}$	105	5.86M	-28.3%	108	17.41M

Table 10.6: NMM performance on the PTB test set.

a noticeable increase in the NoP. This conclusion emphasizes the importance of the joint training. Moreover, we can also see that mixing RNN with FNN models leads to a comparable performance to SRNN, which was particularly designed to enhance RNN with FNN information. The proposed approach, however, does not particularity encode the individual characteristics of the models in the mixture, which reflects its ability to include different types of NNLMs. We can also conclude that combining FNN with recurrent models leads to a more significant improvement when compared to mixtures of FNNs. This conclusion shows, similarly to other work, e.g., [147], that recurrent models can be further improved using N-gram/feed-forward information given that they model different linguistic features.

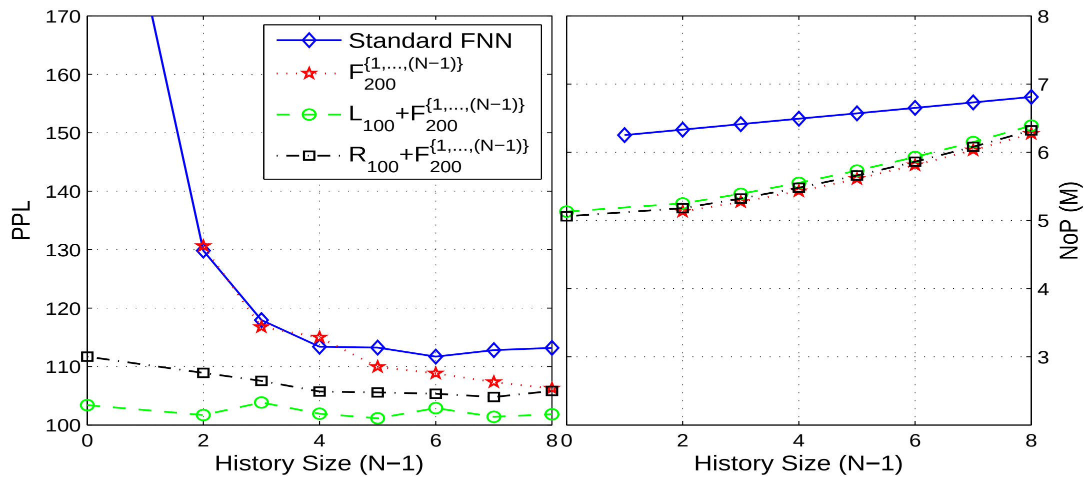


Figure 10.7: *Perplexity vs parameter growth of different mixture models while iteratively increasing the number of FNN models in the mixture.*

Figure 10.7 is an extension of Table 10.6, which shows the change in perplexity and NoP of different NMM models while iteratively increasing the number of FNNs in the mixture. This figure confirms that combining heterogeneous models (combining LSTM or RNN with FNN models) achieves a better performance compared to combining only FNN models. We can also conclude from this figure that the improvement becomes very slow after adding 4 FNN models to each mixture.

10.3.4.2 NMM Evaluation on LTCB

The results shown in Table 10.7 follow the same experimental setup used during SRNN and LSRC experiments. More precisely, these results were obtained without usage of momentum, model dropout or weight decay whereas the mini-batch size was set to 400. The FNN architecture contains 2 hidden layers whereas RNN, LSTM, SRNN and NMM have a single hidden layer. All hidden layers have a size of 600.

	PPL	NoP	PG
FNN[4*200]-600-600-80K	110	64.92M	—
F_{600}^{2-4}	102	66.24M	+2.03%
F_{100}^{2-7}	92	64.98M	+0.09%
RNN[600]-600-80K	85	96.44M	—
WI-SRNN[4*200]-600-80K	77	64.56M	—
WD-SRNN[4*200]-600-80K	72	80.56M	—
$R_{200}+F_{400}^{4}$	84	64.80M	-32.8%
$R_{200}+F_{600}^{2-4}$	77	66.40M	-31.2%
LSTM[600]-600-80K	66	65.92M	—
LSRC[200]-600-80K	63	65.96M	—
$L_{400}+R_{200}$	64	65.44M	-1.51%
$L_{300}+F_{300}^{4}$	64	65.28M	-1.75%
$L_{600}+F_{600}^{4}$	58	67.21M	+1.16%

Table 10.7: NMM performance on the LTCB test set.

The LTCB results shown in Table 10.7 generally confirm the PTB conclusions. In particular, we can see that combining recurrent models with half (third for RNN) their original size and a single FNN model leads to a performance comparable to that of the baseline models. Moreover, increasing the size of the mixture models (for LSTM) or increasing the number of FNN models in the mixture (for RNN) improves the performance further with no noticeable increase in the NoP. Similarly to the PTB experiments, we can also see that the NMM model achieves a similar performance to that of the WI-SRNN architecture with a reduction of the number of parameters by $\approx$ 31% compared to the original RNN model. The FNN mixture results show a more significant improvement when combining multiple small-size (100) models compared to mixing few large models (600). This conclusion shows that the strength of the NMM architecture lies in its ability to combine the learning capabilities of different models, even with small sizes.

11 Batch Noise Contrastive Estimation

Training large vocabulary NNLMs is a difficult task due to the explicit requirement of the output layer normalization, which typically involves the evaluation of the full softmax function over the complete vocabulary. One possible solution to overcome this problem is Noise Contrastive Estimation (NCE) approach [141], which solves this problem by casting the learning of the output layer as a binary classification task between a target word and noise samples, drawn from a given noise distribution. This approach, however, is not well-suited for batch training. More particularly, in standard NCE, each target word in the batch uses a different set of noise samples, which makes it difficult to formulate the learning using dense matrix operations.

This chapter introduces an extension of this approach to batch training. The main idea here is to restrict the vocabulary, at each iteration, to the words in the batch and replace the standard *softmax* function by NCE. In particular, the target words (to predict) in the batch play the role of targets and noise samples at the same time. In doing so, the proposed approach does not require any sampling and can be fully formulated using dense matrix operations, leading to significant speed-up gains with no noticeable degradation of the performance. Moreover, we can also show that this approach optimally approximates the unigram noise distribution, which is widely used in NCE-based language modeling. For clarity, the terms batch and mini-batch will be used interchangeably as synonymous in the rest of this chapter.

11.1 Noise Contrastive Estimation

Probabilistic NNLMs generally require the normalization of the output layer to produce meaningful probability distributions. Using the *softmax* function, the probability of the next word w, given the context c and the model parameters θ, is calculated according to:

$$p_\theta^c(w) = p(w|c,\theta) = \frac{\exp(\mathcal{M}_\theta(w,c))}{\sum\limits_{v\in V}\exp(\mathcal{M}_\theta(v,c))}. \tag{11.1}$$

V is the vocabulary and $\mathcal{M}_\theta$ is the neural scoring function. Learning the parameters θ generally requires the evaluation of the normalization term for each context c. This evaluation involves the complete vocabulary and, therefore, becomes very challenging and resource demanding for large vocabulary corpora.

11.1.1 NCE as a Binary Classifier

The authors of [141] proposed to use noise contrastive estimation to train unnormalized probabilistic models, where the normalization term can be treated as an additional parameter that can be learned during training as well. This idea can be directly applied as an alternative to the full softmax. In this case, $p_\theta^c(w)$ is approximated according to:

$$p_\theta^c(w) \approx \frac{\exp(\mathcal{M}_\theta(w,c))}{Z_c}. \tag{11.2}$$

Z_c is a context-dependent normalization term that can be learned during training. In practice, however, Z_c is fixed to a constant value Z. For instance, the authors of [142] used $Z = 1$, whereas it was set to $Z = \exp(9)$ in [158]. The latter approximates the average normalization term when using the full softmax, which helps speed-up and stabilize the training. This value will be used in the experiments conducted in Section 11.3.

The idea behind NCE is to turn the density estimation problem into a binary classification task, which learns to discriminate between real samples drawn from the data distribution p_D^c and noise samples drawn from a given noise distribution p_n^c. Although p_n^c is context-dependent, it has been empirically shown (e.g., [140]) that context-independent noise distributions, such as unigram, are sufficient to achieve a good performance. Thus, we will use $p_n^c = p_n$ in the rest of this chapter.

Formally, if K denotes the number of noise samples, the probability that the word w was generated from the data distribution ($L = 1$); or from the noise distribution ($L = 0$), is given by:

$$p_c^w(L=1) \triangleq p(L=1 \mid w,c) = \frac{p_\theta^c(w)}{p_\theta^c(w) + K\cdot p_n(w)}, \tag{11.3}$$

$$p_c^w(L=0) \triangleq p(L=0 \mid w,c) = 1 - p(L=1 \mid w,c). \tag{11.4}$$

According to NCE, the model distribution p^c_θ is expected to converge towards the data distribution p^c_D after minimizing the following objective function on the data D:

$$\mathcal{J}(\theta) = -\sum_{w_i}^{D} \left(\log(p^{w_i}_{c_i}(1)) + \sum_k \log\left(p^{w_i^k}_{c_i}(0) \right) \right), \tag{11.5}$$

where $\{w_i^k\}_{k=1}^K$ denote the K noise samples, which are drawn from the noise distribution p_n to train the model on the target word w_i. Moreover, the gradient of $\mathcal{J}(\theta)$ is given by:

$$\frac{\partial \mathcal{J}(\theta)}{\partial \theta} = -\sum_{w_i}^{D} \left(p^{w_i}_{c_i}(0) \frac{\partial \log(p^{c_i}_\theta(w_i))}{\partial \theta} - \sum_k p^{w_i^k}_{c_i}(1) \frac{\partial \log(p^{c_i}_\theta(w_i^k))}{\partial \theta} \right). \tag{11.6}$$

NCE-based training of a neural network follows the standard back-propagation algorithm applied to the objective function given by (11.5) and its gradient in (11.6). More details about NCE and its gradient derivation can be found in [141].

11.1.2 NCE vs Importance Sampling

The authors of [159] have shown that NCE and Importance Sampling (IS) are closely related. More precisely, the main difference between these two approaches is that NCE is defined as a binary classifier between samples drawn from the data and the noise distributions using a logistic loss, whereas IS is a multi-class classifier, which uses *softmax* and a cross-entropy loss. Hence, the authors concluded that IS is theoretically a better choice compared to NCE. The results reported, however, showed a minor difference in performance (a difference of 2.4 in perplexity). Moreover, training using IS can be very difficult and requires a careful control of the samples variance, which can lead otherwise to unstable learning as it was reported in [139]. Hence, an adaptive IS may use a large number of samples to solve this problem, whereas NCE is more stable and requires a fixed and small number of noise samples (e.g., 100) to achieve a good performance [140, 142]. Furthermore, the network learns to *self-normalize* during training using NCE. As a results, and on the contrary to IS, the *softmax* can be avoided during evaluation, which makes NCE an attractive choice to train large vocabulary NNLM. The next section will show how NCE can be efficiently implemented in batch mode training.

11.2 Batch Noise Contrastive Estimation

Although NCE is a good alternative to train large vocabulary LMs, it is not well-suited for batch training. More particularly, each target word in the batch uses a different set of noise samples, which makes it difficult to formulate the learning using dense matrix operations. As a result, the training speed significantly decreases. To alleviate this problem, noise samples can be shared across the batch as was demonstrated in [142].

We propose here an extension of NCE to batch training (B-NCE). This approach does not require any sampling as it can be formulated using dense matrix operations. Furthermore, we can show that this solution optimally approximates sampling from a unigram distribution, which has been shown to be sufficient to achieve a good performance [140, 142].

The main idea here is to restrict the vocabulary, at each forward-backward pass, to the target words in the batch (i.e., words to predict) and then replace the *softmax* function by NCE. In particular, these words play alternatively the role of targets and noise samples. That is, for a target word w_i, at mini-batch index i, the rest of the target mini-batch (i.e., the remaining target words at the other indices j, $j \neq i$) are considered to be the noise samples. The rest of this section introduces the mathematical formulation of B-NCE, which efficiently calculates the error with respect to the output layer weights and biases, as well as the error at the previous layer during mini-batch training, using the objective function (11.5) and its gradient (11.6).

11.2.1 LM Training using B-NCE

Let B, H and V be the sizes of the mini-batch, the last hidden layer and the vocabulary, respectively. The matrix L^t (size $= B \times H$) will denote the evaluation of the last hidden layer at time t on the current batch. Let $V^t = \{w_b^t\}_{b=1}^{B}$ be the target words in the batch at time t, and let W (size $= H \times V$) and C (size $= 1 \times V$) denote the hidden-to-output weight matrix and bias vector, respectively. Our goal here is to calculate the error (i.e., delta) of the output weights W and biases C, as well as the error at the previous layer L^t.

The output layer evaluation in a feed-forward pass of B-NCE, at time t, is calculated by restricting the output layer to V^t. That is, we use the sub-matrix weights W^t (size $= H \times B$) and sub-vector bias C^t (size $= 1 \times B$), which are obtained by restricting W and C to V^t. Thus, the output O^t of the B-NCE-based network, at time t, is given by:

$$O^t = \frac{\exp(L^t \cdot W^t \oplus C^t)}{Z} \quad (\text{size} = B \times B). \tag{11.7}$$

$\oplus$ adds the vector C^t to each row of the left-hand matrix. In fact, the expression given by (11.7) is the B-NCE matrix formulation of (11.2) in batch training.

Now, let $N^t = p_n(\{w_b^t\})$ (size $= 1 \times B$) be the probability of the words in the batch according to the noise distribution p_n. In order to evaluate the gradient of the objective function w.r.t. the output weights and biases, we first define the normalization matrix Y^t according to:

$$Y^t = O^t \oplus (B-1) \cdot N^t \quad (\text{size} = B \times B). \tag{11.8}$$

Y^t is simply the normalization term in equations (11.3) and (11.4) with $K = B-1$. This is a direct result of using the rest of the words in the batch as NCE noise samples for each target word w_b^t, $b = 1, \ldots, B$. In doing so, we eliminate the sampling step.

In order to calculate (11.6) w.r.t. the output weights and biases, we first introduce the auxiliary B-NCE gradient matrix $\mathcal{G}^t$ (size = $B \times B$), at time t, according to:

$$\mathcal{G}^t(i,j) = \begin{cases} \dfrac{O_B^t(i,j)}{Y^t(i,j)}, & \text{if } i \neq j \\ \dfrac{-(B-1)\cdot N^t(i)}{Y^t(i,j)}, & \text{otherwise .} \end{cases} \tag{11.9}$$

$\mathcal{G}^t$ is nothing but the element-wise division of O^t by Y^t after replacing the diagonal of the former by $-(B-1)\cdot N^t$. Then, by applying the NCE gradient derivation given by (11.6), B-NCE calculates the output weight error ΔW^t, the output bias error ΔC^t as well as the error $E(L^t)$ at the previous layer according to:

$$\Delta W^t = {L^t}^\top \cdot \mathcal{G}^t, \tag{11.10}$$

$$\Delta C^t = \sum_{row} \mathcal{G}^t, \tag{11.11}$$

$$E(L^t) = \mathcal{G}^t \cdot {W^t}^\top. \tag{11.12}$$

Note that these equations are similar to the ones obtained when using the *softmax* function combined with the cross-entropy based loss function. The main difference here consists of replacing the full gradient matrix in the *softmax*-based derivation by its B-NCE counterpart, given by $\mathcal{G}^t$. Once the error $E(L^t)$ is propagated to the last hidden layer L^t using (11.12), training the rest of the network follows the standard back-propagation algorithm.

After processing the complete training data, each word w in the vocabulary will be used exactly $(B-1)\times \text{count}(w)$ times as a noise sample. This is strictly equivalent to sampling from a unigram noise distribution, which shows that B-NCE is an optimal implementation of NCE using unigram as noise distribution. We should also mention here that a batch may contain more than occurrence of a given word. In this case, this problem can be solved either by 1) reducing the batch to one occurrence of each word and ignoring the duplicates, or 2) by accumulating the weight deltas for each word as it is typically done for the word embeddings. The B-NCE experiments conducted in Section 11.3 follow the second strategy.

11.2.2 Adaptive B-NCE

The proposed B-NCE approach, as defined above, uses a fixed number of noise samples (i.e., $B-1$), which is dependent on the batch size. In the case where the latter is small (e.g., $B < 100$), B-NCE can be extended to use an additional K noise samples. This can be done by simply drawing an additional K samples form the noise distribution p_n, and share them across the batch as it was done in [142]. The adaptive B-NCE follows the exact same steps described above using the extended output weight sub-matrix W_{B+K}^t (size = $H \times (B+K)$), and the extended sub-vector bias C_{B+K}^t (size = $1 \times (B+K)$) to evaluate (11.7), whereas (11.8) becomes

$$Y^t = O^t \oplus (B+K-1)\cdot N_{B+K}^t \quad (\text{size } = B \times (B+K)), \tag{11.13}$$

where $N^t_{B+K} = \left[N^t, N^t_K\right]$ with N^t_K denoting the probabilities of the additional K noise samples according to the noise distribution p_n.

11.3 B-NCE Evaluation

To evaluate the proposed B-NCE approach, we conducted a set of LM experiments on two different corpora, namely, LTCB and OBWB, which were introduced in Section 8.3.1.2 and Section 8.3.1.3, respectively.

The primary motive of using LTCB, with its medium vocabulary size (i.e., 80K), is to be able to compare the performance of LMs trained using NCE to their counterparts that are trained using the full *softmax*. When using NCE to train the models, the evaluation is either performed using the NCE constant Z for normalization (PPL^n), in this case the target word probabilities are given by (11.2); or using the *softmax* function (PPL^f), which calculates these probabilities using (11.1). The difference in performance between these metrics will evaluate the ability of the models to learn to self-normalize after training. For a comprehensive comparison of the different models, we also report the Number of Parameters (NoP) required by each model as well as its Training Speed (TS), which is calculated as the number of words processed per second (w/s) during training. All experiments were conducted on a **single Titan-X GPU**.

11.3.1 Baseline Models

In order to assess the gap among established NNLMs, this paper also presents a comparative study of different standard architectures with comparable NoPs. That is, we report results for the standard FNN model [122], Elman RNN-based LM [125] as well as the LSTM model [127]. The RNN model we consider here uses a projection weight matrix to decouple the word embedding and the hidden layer sizes. In doing so, we can evaluate different NNLMs with comparable hidden layer sizes and NoPs. We also report results after adding a fully-connected, bottleneck ReLu layer right before the output layer in the recurrent models. These architectures are marked with the prefix *ReLu* in the tables below. Each model is trained using the proposed B-NCE approach and the shared noise NCE (S-NCE) [142]. For the LTCB corpus, we also report results of the models trained with the full *softmax* function. This is the primary motive for using this corpus. We would like also to highlight that the goal of the work presented in this chapter is not about improving LM performance, but rather showing how a significant training speed-up can be achieved, without compromising the models performance, when considering large vocabulary LMs. Hence, we solely focus our experiments on NCE as a major approach to achieve this goal [140, 142, 158] in comparison to the reference *softmax* function. Comparison to other training approaches such as importance sampling is out of the scope of this thesis.

11.3.2 LTCB Experiments

For the LTCB experiments, the embedding size is fixed at 200, the 5-gram FNN model has two hidden layers, whereas the RNN and LSTM networks use a single recurrent layer. All non-recurrent layers use ReLu as activation function. More details about the models architectures are shown in Table 11.1, where "(R)" stands for recurrent and "(B)" for bottleneck. The batch size is fixed at 400 and the initial learning rate is set to 0.4. The latter is halved when no improvement on the validation data is observed for an additional 7 epochs. We also use a norm-based gradient clipping with a threshold of 5 but we do not use dropout. On the contrary to the SRNN and LSRC experiments, the recurrent models are unfolded for 20 time steps during training, which was used by other researchers when considering the one billion word corpus [142, 159]. Regarding the NCE configuration, B-NCE and S-NCE use unigram as noise distribution p_n. Furthermore, following the setup proposed in [140, 142], S-NCE uses $K = 100$ noise samples, whereas B-NCE uses only the target words in the batch (K=0). Note that S-NCE will process and update $B + K$ words at its output layer during each forward-backward pass, whereas B-NCE updates only B words. Similarly to [158], the NCE normalization constant is set to $Z = \exp(9)$, which approximates the average normalization term when using the *softmax*.

Model	Architecture
5-gram FNN	4×200–600–400(B)–V
RNN	200–600(R)–V
ReLu-RNN	200–600(R)–400(B)–V
LSTM	200–600(R)–V
ReLu-LSTM	200–600(R)–400(B)–V

Table 11.1: Model architecture for LTCB experiments.

The LTCB results reported in Table 11.2 clearly show that B-NCE reduces the training time by a factor of 4 to 8 with a slight degradation in the models performance compared to *softmax*. Moreover, we can also see that B-NCE slightly outperforms S-NCE while being faster and simpler to implement. In particular, B-NCE does not require the sampling step since it uses the rest of the output words in the batch itself as noise samples to train the model on each target word. This can be efficiently implemented using dense matrix operations (see Section 11.2). Table 11.2 also shows that PPL^n is close from PPL^f, which typically reflects that the models trained using NCE are able to self-normalize, where the normalization term using softmax is, in average, very close from the NCE constant Z. We have also observed in our experiments that the models degradation and the gap between PPL^f and PPL^n strongly depend on the amount of training data, the vocabulary size as well as the size of the last hidden layer. More particularly, increasing the training data leads to a more stable learning and therefore to a smaller gap between these two metrics and a much lower degradation of the models performance (see OBWB experiments below).

	PPL^n	PPL^f	TS (w/s)	NoP
5-gram FNN (*softmax*)	—	110.2	8.4K	48.8M
5-gram FNN (S-NCE)	129.8	125.4	29.1K	48.8M
5-gram FNN (B-NCE)	119.4	113.7	35.1K	48.8M
RNN(*softmax*)	—	79.7	5.9K	64.6M
RNN (S-NCE)	88.7	84.2	37.8K	64.6M
RNN (B-NCE)	87.6	82.5	43.7K	64.6M
ReLu-RNN (*softmax*)	—	69.5	8.6K	48.8M
ReLu-RNN (S-NCE)	80.2	77.3	30.9K	48.8M
ReLu-RNN (B-NCE)	79.4	76.0	36.7K	48.8M
LSTM (*softmax*)	—	62.5	8.9K	66.0M
LSTM (S-NCE)	77.4	73.1	27.2K	66.0M
LSTM (B-NCE)	70.9	68.3	37.1K	66.0M
ReLu-LSTM (*softmax*)	—	59.2	8.2K	50.3M
ReLu-LSTM (S-NCE)	68.4	67.1	26.9K	50.3M
ReLu-LSTM (B-NCE)	64.9	62.4	32.0K	50.3M

Table 11.2: Performance of B-NCE-based LMs on LTCB.

We can also conclude from Table 11.2 that the additional ReLu layer improves the performance while significantly decreasing the number of parameters (NoP). This conclusion is valid for both, RNN and LSTM. These results confirm that adding a fully-connected bottleneck layer can significantly boost the performance of the recurrent models. This idea has been previously used in computer vision tasks in combination with Convolutional Neural Networks (CNN) [160] as well as in speech recognition [161], where the fully-connected layer is used as pat of the LSTM recurrent module.

11.3.3 One Billion Word Benchmark Experiments

The OBWB experiments follow a similar setup to the one used for LTCB with minor differences. That is, the embedding size is set to 500 for all models, the batch size is fixed at 500, S-NCE uses $K = 200$ noise samples and the initial learning rate is set to 1.0. Given that the vocabulary size is $\approx 0.8M$, it was not possible to train the language models using the full *softmax* function. Therefore, we only report results for models trained using B-NCE and S-NCE. During evaluation, however, we also report performance using the full softmax. More details about the models configuration are shown in Table 11.3.

The OBWB results in Table 11.4 generally confirm the LTCB conclusions. That is, B-NCE slightly outperforms S-NCE while being faster and simpler to train (training speed in 3^{rd} column). Moreover, these results also show a much smaller difference between PPL^f and PPL^n compared to LTCB, which suggests that the models learned to better self-normalize due to the larger amount of training data. Similarly to LTCB, we can see that the additional ReLu helps reducing the NoPs while improving or maintaining the models performance for RNN and LSTM.

Model	Architecture
5-gram FNN	4×500—1500—600(B)—V
RNN	500—1500(R)—V
ReLu-RNN	500—1500(R)—600(B)—V
LSTM	500—1500(R)—V
ReLu-LSTM	500—1500(R)—600(B)—V

Table 11.3: Model architecture for OBWB experiments.

	PPL^n	PPL^f	TS (w/s)	NoP
5-gram FNN (S-NCE)	86.3	84.4	12.3K	0.88B
5-gram FNN (B-NCE)	81.9	80.6	13.4K	0.88B
RNN (S-NCE)	70.8	67.6	24.9K	1.59B
RNN (B-NCE)	63.4	61.4	27.8K	1.59B
ReLu-RNN (S-NCE)	59.6	59.1	20.1K	0.88B
ReLu-RNN (B-NCE)	56.9	56.6	23.1K	0.88B
LSTM (S-NCE)	51.3	50.3	12.0K	1.60B
LSTM (B-NCE)	48.6	48.1	13.1K	1.60B
ReLu-LSTM (S-NCE)	51.0	50.9	13.0K	0.89B
ReLu-LSTM (B-NCE)	49.2	48.8	14.7K	0.89B

Table 11.4: Performance of B-NCE-based LMs on OBWB.

In comparison to other reported results on the OBWB. We can see that the small ReLu-RNN achieves a close performance from the very large RNN model (PPL = 51.3 and NoP = 20B) proposed in [136]. Moreover, the performance of the small ReLu-LSTM is comparable to the LSTM models proposed in [142] and [159], which use large hidden layers. In particular, the first paper trains a large 4-layers LSTM model using S-NCE on 4 GPUs (PPL = 43.2 and NoP = 3.4B), whereas the second one uses a recurrent bottleneck layer [161] and a total of $K = 8192$ noise samples with importance sampling on 32 Tesla K40 GPUs.

Table 11.4 can be considered as a baseline performance of standard NNLMs on OBWB. To the best of our knowledge, this is the first experimental study conducted on the OBWB that evaluates and compares standard NNLMs with similar configurations, and more importantly, with a comparable number of parameters. While we can clearly see that recurrent models significantly outperform the feed-forward models, the gap between RNN and LSTM is, however, small. This is mainly due to the random shuffling of the sentences in the one billion word corpus, which limits the ability of LSTM models to learn context dependencies beyond the sentence boundaries. Thus, an evaluation of the gap between RNN and LSTM still needs to be conducted on a large corpus.

Part III

Conclusions and Future Work

12 Conclusions and Future Work

We have presented in this thesis different solutions to solve two sequential data problems, namely, 1) multiple overlapping speaker localization, detection and tracking using the sequential multi-channel audio signal received by a small aperture microphone array, and 2) the language modeling task, which aims at predicting the next word given the sequence of previous words (i.e., its history). The proposed solutions take into account the fundamental differences regarding the problem formulation of these two sequential problems. Thus, we introduced a new sequential Bayesian framework to solve the first problem while we proposed a sequential neural estimation approach to tackle the second task as summarized below.

12.1 Multiple Speaker Localization, Detection and Tracking

12.1.1 Summary and Achieved Objectives

We introduced in this thesis a novel sequential Bayesian estimation framework to solve the multiple speaker localization, detection and tracking problem. This framework consists actually of a complete pipeline with multiple components that include 1) new observation detectors, which extract a fixed number of potential location estimates per time frame, 2) an unsupervised Bayesian detector, which classifies these estimates into noise/speaker classes and 3) new Bayesian filters, which use the estimates from the speaker class to track multiple overlapping speakers. In doing so, this new framework is able to answer three important questions regarding the received signals:

1. The localization step answers the question: “Where does the signal originate from?”.
2. The second step in the pipeline, i.e., acoustic source detection, answers the question: “How many real simultaneous (overlapping) speakers are in each time-frame?”.
3. The last short-term tracking step returns local identifiers that answer the question: “Who are (locally) the speakers?”.

The extensive experimental studies that we conducted on the AV16.3 corpus showed that the proposed framework answers these questions while achieving the following objectives:

- No significant predefined assumptions or constraints regarding the problem formulation and the proposed solutions. In particular, regarding the location of the speakers, their number and their movement.
- The framework achieves a satisfying performance in a standard meeting room environment.
- We can also conclude that the approach is quite robust to the typical in-door noise and reverberation that are present in the AV16.3 corpus.
- The proposed solutions are able to run online with a near real-time performance, although they were not particularly optimized for this goal. We believe that significant speed-up can be further obtained by optimizing the implementation.
- This framework was particularly successful in increasing the overlap detection rate of overlapping speakers, which is still an open issue for many speech processing applications, such as speech separation, speech recognition and speaker diarization.

12.1.2 Conclusions and Insights

The work conducted in this thesis shows that solving the problem of noisy estimates (caused by noise sources and reverberation), which highly decreases multiple speaker localization and tracking performance, can be dealt with in two different ways, namely, either 1) by increasing the robustness of tracking filters to clutter and noise, which can be achieved using multiple hypothesis filters, or 2) by using the tracking information itself, at each time step, to improve the location measurement stage, leading to more accurate estimates. This is in contrast to time-averaging and clustering approaches, which generally combine multiple successive time frames to extract location estimates and, thereby, increase the robustness to noise. Moreover, these two proposed principles to deal with noise actually bridge the gap between TDOA-based and SRP-based localization approaches for the single speaker case.

We can also conclude that using a Bayesian approach to solve all stages of this problem, i.e., from the TDOA estimation to the multiple speaker tracking stage, provides a convenient framework that deals with noise and multi-modality in a probabilistic manner while removing the extensive search step required by standard approaches. Another important conclusion we can draw from this work is that speech features are important but not necessary to solve the noise/speaker detection problem, where we have shown that the nature of the acoustic source ("point" vs "distributed") can actually be used as an alternative criterion to separate noise estimates from real speakers. Obviously, such approach can benefit from adding speech features but the achieved performance clearly shows that this new criterion is very effective.

From a practical point-of-view, we can also conclude that the Bayesian approach is a better and more realistic choice for real-world applications since it allows an online and near real-time performance while using a short-time frame, i.e., 32ms, without imposing any direct constraints or assumptions on the speakers.

12.1.3 Future Work

While the proposed framework was successful on the AV16.3, there are additional research directions that could be investigated to achieve further gains. In particular:

- Additional speech cues should be investigated in order to improve the detection and tracking approaches. More precisely, speech feature extractors, such as MFCC, could be added to augment the feature vectors used by the naive Bayesian classifier in order to improve the noise/speaker classification.
- Currently, the proposed approach can only track a speaker while being active, i.e., speaking, but cannot map separate short-term trajectories to the same speaker after a short-/long silence. Following the previous proposition, speech cues can also be used here to connect the short-term trajectories and thereby achieve a long term-tracking of each speaker. Potential work in this direction should investigate and merge techniques from speaker diarization and speaker clustering.
- An extensive evaluation of this approach when it is used as a pre-processing step in other tasks, such as speech separation and multi-party distant speech recognition, should be investigated.

12.2 Hybrid Neural Network Language Models

12.2.1 Summary and Achieved Objectives

In the second part of this thesis, we introduced new hybrid neural network models to solve the language modeling task. These new architectures were developed, following the model combination principle, as different mixtures of classical architectures. In doing so, these new models are able to combine the learning capabilities of different NNLMs. In particular, the work presented in this part introduced an:

- SRNN architecture which unfolds the RNN model during word prediction (evaluation) as it is done during training. That is, we used an explicit N-gram history, as it is done in FNN model, while enhancing the word embeddings using the recurrent information, available at each word position, through additional sequential connections. In doing so, we merged the benefits of FNN and RNN in a single model.
- LSRC model which separately and explicitly models short and long range dependencies using two separate local and global states, respectively. The word prediction is performed after an update of the local context in a first step, which is then used to update the global context. In doing so, we separately capture short and long range dependencies. This model can be seen as an attempt to merge RNN and LSTM into a single architecture.
- Neural mixture model which is an attempt to generalize the previous models through an architecture that does not impose any constraints on the number or type of networks to combine. More precisely, this model proposes to use a dynamic number of heterogeneous neural architectures as kernels to detect different linguistic features, which are then combined using an additional mixture layer.

The evaluation of these models on the PTB corpus and LTCB showed a significant reduction of the perplexity compared to different baseline models. This improvement is due to many advantages that are either inherited from the generalization power of the NN solutions; or from the new architectures that were designed to model some linguistic aspects. In particular, these models are able to fully or partially solve the following problems:

- Data sparsity: This is solved through the continuous word representations that are learned by the NNLMs. We should highlight here that word representations obtained by different architectures can lead to a significant difference when applied to different linguistic tasks. In particular, we expect the SRNN context embeddings to capture a different type of information compared to standard word2vec embeddings[162]. This hypothesis should be investigated in the future work.
- Dynamic context modeling: All proposed architectures try to explicitly model and combine short and long range context information. This is done by combining heterogeneous models that can capture a dynamic range of context due to their recurrent architectures.
- Absence of linguistic features: This problem is inherently solved in the SRNN and LSRC models while it is explicitly dealt with in the NMM architecture. The latter uses different heterogeneous NNLMs as kernels that can capture different linguistic features, which are subsequently combined in a mixture layer.

In addition to the new hybrid neural language models, we also proposed an adaptation of the noise contrastive estimation approach to simplify and speed-up the training of NNLMs in batch mode. This approach removes the sampling step needed in the standard model, optimally approximates the unigram noise distribution and can be fully formulated using dense matrix operation. Experiments conducted on LTCB and OBWB showed a significant speed-up compared to the standard softmax approach with a negligible decrease in the performance.

12.2.2 Conclusions and Insights

The work conducted in the second part of this thesis shows that designing new neural architectures for language modeling should explicitly model short and long range dependencies, with a preference to have separate models or components to learn each of them. This choice is motivated by the fact that these two types of dependencies are often of different natures. More particularly, long range dependencies generally carry semantic information, whereas short context mostly encodes syntactic properties and short linguistic patterns. This conclusion is confirmed by the models proposed in this thesis, in addition to the general improvement we observe when we combine heterogeneous models that are reported in the literature.

We can also conclude from this work that FNN, RNN and LSTM model and capture different type of linguistic information and context range. In particular, the three models developed in this work confirmed that the FNN model has the unique advantage of explicitly modeling word-position information and short patterns, which improves the performance when it is combined with the RNN model. The latter learns a smoothed representation of the context and therefore loses the word-position information. Thus, we can see that these two models are actually complementary to each other. Furthermore, the experiments conducted in this thesis have shown that RNN is in fact a short to medium range context model. That is, RNN is experimentally equivalent to an 8-gram FNN model. This conclusion is also confirmed by the temporal correlation of the hidden state of the RNN model, which shows that the encoded information completely changes after processing, approximately, ten words. The temporal correlation of LSTM, on the other hand, showed that it is able to learn a dynamic range which tends to be much longer compared to RNN, i.e., over hundreds of words. In many applications, however, the processed data is of a short nature, e.g., online queries, dialogue systems, question/answering, etc., which makes this type of models sub-optimal. In fact, this work led to the conclusion that solving the language modeling task, depending on the target application, should take into account different important factors, such as 1) the explicit modeling of short and long range context (as mentioned above), 2) modeling the word-position information and 3) an explicit integration of the role (semantic, syntactic, etc.) of each word in the history, during each prediction step. While we have presented solutions to the first two aspects, the third one should be dealt with using mechanisms such as attention or residual connections (similarly to SRNN). This will be investigated in the future work.

12.2.3 Future Work

While the evaluation of the proposed hybrid NNLMs on PTB and LTCB showed a good improvement of the performance regarding the perplexity and training speed, there are additional research directions that could be investigated to achieve further gains. In particular:

- Investigation of attention mechanism to accurately extract the target word, which can be present in the last few sentences. Note that this research direction is required for tasks that do not necessarily target perplexity but rather accuracy, such as sentence completion task and speech recognition, although these two metrics generally correlate.

- Investigation of the additional context embeddings that are learned by the SRNN model. In particular, a study to understand the linguistic nature of the information that they capture would be of great interest.

- Extension and application of the neural mixture model to other tasks. In particular, this architecture can be investigated for LM adaptation by retraining few kernels on the adaptation data, as it can be applied to solve multi-task problems.

- While the work presented in this thesis was mainly conducted to solve information density and linguistic encoding problems, an evaluation of the proposed approaches for other tasks such speech recognition and machine translation should be also conducted in future work.

List of Abbreviations

AC	Agglomerative Clustering
AED	Adaptive Eigenvalue Decomposition
AEJD	Acoustic Event Joint Distribution
AI	Artificial Intelligence
ASR	Automatic Speech Recognition
B-NCE	Batch Noise Contrastive Estimation
BF	Beamforming
BIC	Bayesian Information Criterion
BPTT	Back-Propagation Through Time
BSS	Blind Source Separation
C-SRP	Cumulative Steered Response Power
cdf	cumulative density function
CNN	Convolutional Neural Network
CRFs	Conditional Random Fields
CV	Computer Vision
DCP	Dominance-based Cumulative Power
DL	Deep Learning
DOA	Direction of Arrival
DSB	Delay-and-Sum Beamformer
DSSM	Dynamic State Space Model
EKF	Extended Kalman Filter
EM	Expectation-Maximization
FAR	False Alarm Rate
FNN	Feed-forward Neural Network
GCC	Generalized Cross-Correlation
GM	Gaussian Mixture
GMM	Guassian Mixture Model
HMM	Hidden Markov Model
IR	Information Retrieval
K-SRP	Kalman Steered Response Power

KF	Kalman Filter
KFB	Kalman Filter Bank
LM	Language Model
LS	Least Squares
LSRC	Long-Short Range Context
LSTM	Long-Short Term Memory
LTCB	Large Text Compression Benchmark
MC	Monte Carlo
MCCC	Multi-Channel Cross-Correlation
MFCC	Mel Frequency Cepstral Coefficents
MH-GMF	Multiple Hypothesis Gaussian Mixture Filter
ML	Maximum Likelihood
MLE	Maximum Likelihood Error
NBC	Naive Bayesian Classifier
NCE	Noise Contrastive Estimation
NLP	Natural Language Processing
NMM	Neural Mixture Model
NN	Neural Network
NNLM	Neural Network Language Models
OBWB	One Billion Word Benchmark
P-SRP	Probabilistic Steered Response Power
PDAF	Probabilistic Data Association Filter
pdf	probability density function
PF	Particle Filter
PTB	Penn Treebank
ReLu	Rectified Linear Unit
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
SCM	Spatial Correlation Matrix
SLM	Statistical Language Model
SMT	Statistical Machine Translation
SNR	Signal-to-Noise Ratio
SRNN	Sequential Recurrent Neural Network
SRP	Steered Response Power
SRR	Signal-to-Reverberation Ratio
SS	Speech Synthesis
STC	Short-Term Clustering
STT	Short-Term Tracking
SVM	Support Vector Machine

TDOA	Time Difference Of Arrival
TTS	Text-To-Speech
UCA	Uniform Circular Arrays
UKF	Unscented Kalman Filter
UT	Unscented Transform
WD	Word-Dependent
WEM	Weighted Expectation-Maximization
WI	Word-Independent

List of Figures

List of Tables

Bibliography

[1] D. Forsyth and J. Ponce, *Computer vision: a modern approach.* Upper Saddle River, NJ; London: Prentice Hall, 2011.

[2] G. Bradski and A. Kaehler, *Learning OpenCV: computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.

[3] L. R. Rabiner and B.-H. Juang, "Fundamentals of speech recognition," 1993.

[4] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[5] T. Dutoit, *An introduction to text-to-speech synthesis.* Springer Science & Business Media, 1997, vol. 3.

[6] D. Zelterman, "Bayesian artificial intelligence," 2005.

[7] C. M. Bishop, *Pattern recognition and machine learning (information science and statistics)*, 1st ed. Springer, Oct. 2007.

[8] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction.* MIT press Cambridge, 1998, vol. 1.

[9] L. Rabiner and B. Juang, "An introduction to hidden markov models," *IEEE assp magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[10] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[11] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

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

[13] T. Dietterich, “Machine learning for sequential data: A review,” *Structural, syntactic, and statistical pattern recognition*, pp. 227–246, 2002.

[14] A. Doucet, S. Godsill, and C. Andrieu, “On sequential monte carlo sampling methods for bayesian filtering,” *Statistics and computing*, vol. 10, no. 3, pp. 197–208, 2000.

[15] M. S. Arulampalam, S. Maskell, and N. Gordon, “A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking,” *IEEE Transactions on Signal Processing*, vol. 50, pp. 174–188, 2002.

[16] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[17] Y. Oualil, F. Faubel, M. Magimai.-Doss, and D. Klakow, “A TDOA Gaussian mixture model for improving acoustic source tracking,” in *20th European Signal Processing Conference (EUSIPCO)*, Aug. 2012, pp. 1339–1343.

[18] Y. Oualil, M. Magimai.-Doss, F. Faubel, and D. Klakow, “Joint detection and localization of multiple speakers using a probabilistic interpretation of the steered response power,” in *Statistical and Perceptual Audition Workshop*, Sep. 2012.

[19] ——, “A probabilistic framework for multiple speaker localization,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2013.

[20] Y. Oualil, F. Faubel, and D. Klakow, “A fast cumulative steered response power for multiple speaker detection and localization,” in *21st European Signal Processing Conference (EUSIPCO)*. IEEE, 2013, pp. 1–5.

[21] ——, “An unsupervised Bayesian classifier for multiple speaker detection and localization,” in *Proc. INTERSPEECH*, Aug. 2013.

[22] Y. Oualil, R. M. Toroghi, and D. Klakow, “Online unsupervised overlapping speaker detection using enhanced classification history-based features,” in *14th International Workshop on Acoustic Signal Enhancement (IWAENC)*. IEEE, 2014, pp. 228–232.

[23] Y. Oualil, F. Faubel, and D. Klakow, “A multiple hypothesis Gaussian mixture filter for acoustic source localization and tracking,” in *International Workshop on Acoustic Signal Enhancement (IWAENC)*, Sep. 2012, pp. 233–236.

[24] Y. Oualil and D. Klakow, “Multiple concurrent speaker short-term tracking using a kalman filter bank,” in *IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP)*, Florence, Italy, May 4-9 2014, pp. 1444–1448.

[25] Y. Oualil, R. M. Toroghi, and D. Klakow, “Improving overlapping speaker detection using multiple speaker tracking information,” in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Atlanta, GA, USA, Dec. 3-5 2014, pp. 552–556.

[26] Y. Oualil, C. Greenberg, M. Singh, and D. Klakow, “Sequential recurrent neural networks for language modeling,” in *17th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, San Francisco, CA, USA, Sep. 8-12 2016, pp. 3509–3513.

[27] Y. Oualil, M. Singh, C. Greenberg, and D. Klakow, “Long-short range context neural networks for language modeling,” in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Austin, Texas, USA, Nov. 2016, pp. 1473–1481.

[28] Y. Oualil and D. Klakow, “A neural network approach for mixing language models,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2017, pp. 5710–5714.

[29] ——, “A batch noise contrastive estimation approach for training large vocabulary language models,” in *18th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, Stockholm, Sweden, Aug. 20-24 2017.

[30] M. Brandstein and D. Ward, *Microphone arrays: signal processing techniques and applications.* Springer Science & Business Media, 2013.

[31] S. Makino, T.-W. Lee, and H. Sawada, *Blind speech separation.* Springer, 2007, vol. 615.

[32] R. Zelinski, “A microphone array with adaptive post-filtering for noise reduction in reverberant rooms,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1988, pp. 2578–2581.

[33] X. Sheng and Y.-H. Hu, “Maximum likelihood multiple-source localization using acoustic energy measurements with wireless sensor networks,” *IEEE Transactions on Signal Processing*, vol. 53, no. 1, pp. 44–53, 2005.

[34] D. N. Zotkin, R. Duraiswami, and L. S. Davis, “Joint audio-visual tracking using particle filters,” *EURASIP Journal on Applied Signal Processing*, vol. 2002, no. 1, pp. 1154–1164, 2002.

[35] R. O. Schmidt, “A new approach to geometry of range difference location,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-8, no. 6, pp. 821–835, Nov. 1972.

[36] H. C. Schau and A. Z. Robinson, “Passive source localization employing intersecting spherical surfaces from time-of-arrival differences,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 35, no. 8, pp. 1223 – 1225, Aug. 1987.

[37] J. O. Smith and J. S. Abel, "Closed-form least-squares source location estimation from range-difference measurements," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 35, no. 12, pp. 1661 – 1669, Dec. 1987.

[38] M. Brandstein, J. Adcock, and H. Silverman, "A closed-form method for finding source locations from microphone-array time-decay estimates," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 5, pp. 3019–3022, 1995.

[39] M. S. Brandstein, J. E. Adcock, and H. F. Silverman, "A closed-form location estimator for use with room environment microphone arrays," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 7, no. 1, pp. 45–50, Jan. 1997.

[40] Y. Chan and K. Ho, "A simple and efficient estimator for hyperbolic location," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 42, no. 8, pp. 1905 –1915, Aug. 1994.

[41] J. C. Chen, K. Yao, and R. E. Hudson, "Source localization and beamforming," *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 30–39, 2002.

[42] J. Chen, J. Benesty, and Y. Huang, "Robust time delay estimation exploiting redundancy among multiple microphones," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 11, no. 6, pp. 549–557, 2003.

[43] J. Benesty, "Adaptive eigenvalue decomposition algorithm for passive acoustic source localization," *Journal of the Acoustical Society of America*, vol. 107, no. 1, pp. 384–391, 2000.

[44] J. Dmochowski, J. Benesty, and S. Affes, "Direction of arrival estimation using the parameterized spatial correlation matrix," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1327 –1339, May 2007.

[45] ——, "The generalization of narrowband localization methods to broadband environments via parametrization of the spatial correlation matrix," in *15th European Signal Processing Conference (EUSIPCO)*, 2007, pp. 763–767.

[46] J. H. DiBiase, "A high-accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays," Ph.D. dissertation, Brown University, 2000.

[47] J. P. Dmochowski, J. Benesty, and S. Affes, "Fast steered response power source localization using inverse mapping of relative delays," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 289–292.

[48] H. Do, H. F. Silverman, and Y. Yu, "A real-time SRP-PHAT source location implementation using stochastic region contraction (SRC) on a large-aperture microphone array," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2007, pp. 121–124.

[49] H. Do and H. F. Silverman, "A fast microphone array SRP-PHAT source location implementation using coarse-to-fine region contraction(CFRC)," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Oct. 2007, pp. 295 –298.

[50] G. Lathoud and I. A. McCowan, "A sector-based approach for localization of multiple speakers with microphone arrays," in *Proc. SAPA Workshop*, Oct. 2004.

[51] G. Lathoud and M. Magimai.-Doss, "A sector-based, frequency-domain approach to detection and localization of multiple speakers," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2005.

[52] M. Cobos, A. Marti, and 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, Jan. 2011.

[53] G. Lathoud, M. Magimai.-Doss, , and B. Hervé, "Threshold selection for unsupervised detection, with an application to microphone arrays," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toulouse, France, May 2006.

[54] M. Nilesh and M. Rainer, "A scalable framework for multiple speaker localization and tracking," in *International Workshop on Acoustic Signal Enhancement (IWAENC)*, 2008.

[55] O. Yilmaz and S. Rickard, "Blind separation of speech mixtures via time-frequency masking," *IEEE Transactions on signal processing*, vol. 52, no. 7, pp. 1830–1847, 2004.

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

[57] H. Do and H. Silverman, "A method for locating multiple sources from a frame of a large-aperture microphone array data without tracking," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2008, pp. 301 –304.

[58] C. Weiping, Z. Xiaoyan, and W. Zhenyang, "Localization of multiple speech sources based on sub-band steered response power," in *International Conference on Electrical and Control Engineering (ICECE)*, Jun. 2010, pp. 1246 –1249.

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

[60] A. Levy, S. Gannot, and A. P. Habets, "Multiple-hypothesis extended particle filter for acoustic source localization in reverberant environments," *IEEE Transactions on Acoustics, Speech and Signal Processing*, 2010.

[61] D. B. Ward and R. C. Williamson, “Particle filter beamforming for acoustic source localization in a reverberant environment,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, May 2002, pp. 1777–1780.

[62] J. Vermaak and A. Blake, “Nonlinear filtering for speaker tracking in noisy and reverberant environments,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 5, May 2001, pp. 3021–3024.

[63] S. Gannot and T. G. Dvorkind, “Microphone array speaker localizers using spatial-temporal inforamtion,” *EURASIP Journal on Applied Signal Processing*, pp. 174–174, 2006.

[64] U. Klee, T. Gehrig, and J. McDonough, “Kalman filters for time delay of arrival-based source localization,” *EURASIP Journal on Applied Signal Processing*, pp. 167–167, 2006.

[65] A. Levy, S. Gannot, and E. A. P. Habets, “Multiple-hypothesis extended particle filter for acoustic source localization in reverberant environments,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, no. 6, pp. 1540–1555, 2011.

[66] T. Gehrig and J. McDonough, “Tracking multiple speakers with probabilistic data association filters,” in *Proc. CLEAR*, 2007, pp. 137–150.

[67] A. Masnadi-Shirazi and B. Rao, “Separation and tracking of multiple speakers in a reverberant environment using a multiple model particle filter glimpsing method,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 2516–2519.

[68] X. Zhong and J. Hopgood, “Nonconcurrent multiple speakers tracking based on extended kalman particle filter,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 293–296.

[69] A. Quintan and F. Asano, “Tracking a varying number of speakers using particle filtering,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 297–300.

[70] G. Lathoud and J. M. Odobez, “Short-term spatio-temporal clustering applied to multiple moving speakers,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 5, p. 15, July 2007.

[71] G. Lathoud, J.-M. Odobez, and D. Gatica-Perez, “AV16.3: An audio-visual corpus for speaker localization and tracking,” in *Proc. MLMI 04 Workshop*, May 2006, pp. 182–195.

[72] D. Moore, “The idiap smart meeting room,” IDIAP, Tech. Rep. Idiap-Com-07-2002.

[73] G. Lathoud, “Spatio-temporal analysis of spontaneous speech with microphone arrays,” Ph.D. dissertation, École Polytechnique Fédérale de Lausanne, Switzerland, Dec. 2006.

[74] R. E. Walpole, R. H. Myers, S. L. Myers, and K. Ye, *Probability and statistics for engineers and scientists.* Macmillan New York, 1993, vol. 5.

[75] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal processing magazine*, vol. 13, no. 6, pp. 47–60, 1996.

[76] G. McLachlan and T. Krishnan, *The EM algorithm and extensions.* John Wiley & Sons, 2007, vol. 382.

[77] J. Ghosh and R. Ramamoorthi, *Bayesian nonparametrics*, ser. Springer Series in Statistics. Springer, 2003.

[78] A. Norets and J. Pelenis, "Bayesian modeling of joint and conditional distributions," *Journal of Econometrics*, vol. 168, no. 2, pp. 332 – 346, 2012.

[79] S. M. Kay, *Fundamentals of statistical signal processing.* Prentice Hall PTR, 1993.

[80] F. Faubel and D. Klakow, "A transformation-based derivation of the Kalman filter and an extensive unscented transform," in *Proc. SSP*, Sep. 2009, pp. 161–164.

[81] R. E. Kalman *et al.*, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[82] A. C. Harvey, *Forecasting, structural time series models and the Kalman filter.* Cambridge university press, 1990.

[83] S. J. Julier and J. K. Uhlmann, "A new extension of the kalman filter to nonlinear systems," in *Int. symp. aerospace/defense sensing, simul. and controls*, vol. 3. Orlando, FL, 1997, pp. 182–193.

[84] S. J. Julier, "The scaled unscented transformation," in *American Control Conference*, vol. 6. IEEE, 2002, pp. 4555–4559.

[85] S. S. Haykin *et al.*, *Kalman filtering and neural networks.* Wiley Online Library, 2001.

[86] D. Alspach and H. Sorenson, "Nonlinear bayesian estimation using guassian sum approximations," *IEEE Transactions on Automatic Control*, vol. 17, no. 4, pp. 439–448, 1972.

[87] E. Brookner, *Tracking and Kalman filtering made easy*, ser. A Wiley-Interscience publication. Wiley, 1998.

[88] F. Faubel, "Statistical signal processing techniques for robust speech recognition," Ph.D. dissertation, Spoken Language Systems Group, Saarbrücken, Germany, 2013.

[89] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," in *Proceedings of the IEEE*, vol. 92, no. 3, Mar. 2004, pp. 401–422.

[90] F. Faubel and D. Klakow, "A transformation-based derivation of the kalman filter and an extensive unscented transform," in *IEEE Workshop on Statistical Signal Processing*, Sep. 2009.

[91] T. Gehrig, U. Klee, J. McDonough, S. Ikbal, M. Wölfel, and C. Fügen, "Tracking and beamforming for multiple simultaneous speakers with probabilistic data association filters," in *Proc. Interspeech*, Sep. 2006.

[92] D. B. Ward, E. A. Lehmann, and R. C. Williamson, "Particle filtering algorithms for tracking an acoustic source in a reverberant environment," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 11, pp. 826–836, 2003.

[93] A. Badali, J. M. Valin, F. Michaud, and P. Aarabi, "Evaluating real-time audio localization algorithms for artificial audition in robotics," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2009, pp. 2033–2038.

[94] Y. Cho, D. Yook, S. Chang, and H. Kim, "Sound source localization for robot auditory systems," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 3, pp. 1663–1668, Aug. 2009.

[95] J. Valin, F. Michaud, B. Hadjou, and J. Rouat, "Localization of simultaneous moving sound sources for mobile robot using a frequency-domain steered beamformer approach," *EEE International Conference on Robotics and Automation (ICRA)*, pp. 1033–1038, 2004.

[96] J. Borwein and A. Lewis, *Convex analysis and nonlinear optimization : theory and examples*, ser. CMS Books in Mathematics. Springer, 2005.

[97] J. Nocedal and S. J. Wright, *Numerical optimization*, 2nd ed. Springer, 2006.

[98] A. Ruszczynski, *Nonlinear optimization*, ser. Nonlinear Optimization. Princeton University Press, 2011.

[99] J. Sohn, N. S. Kim, and W. Sung, "A statistical model-based voice activity detection," *IEEE Signal Process. Lett.*, vol. 6, no. 1, pp. 1–3, 1999.

[100] G. J. McLachlan and T. Krishnan, *The EM algorithm and extensions (Wiley Series in Probability and Statistics)*, 2nd ed. Wiley-Interscience, Mar. 2008.

[101] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification (2nd Edition)*, 2nd ed. Wiley-Interscience, Nov. 2000.

[102] H. L. Van Trees, *Optimum array processing (detection, estimation, and modulation theory, part IV)*, 1st ed. Wiley-Interscience, Mar. 2002.

[103] K. Kumatani, J. McDonough, B. Rauch, D. Klakow, P. Garner, and W. Li, “Beamforming with a maximum negentropy criterion,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 5, pp. 994–1008, July 2009.

[104] J. LeBlanc and P. D. Leon, “Speech separation by kurtosis maximization,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, May 1998, pp. 1029–1032 vol.2.

[105] F. Faubel, M. Georges, B. Fu, and D. Klakow, “Robust Guassian mixture filter based mouth tracking in a real environment,” in *Proc. Visual Computing Research Conference (IVCI, Saarbrucken)*, Dec. 2009.

[106] Y. Bar-Shalom and T. E. Fortmann, *Tracking and data association.* Academic Press, 1988.

[107] T. Kailath, “The divergence and Bhattacharyya distance measures in signal selection,” *IEEE Transactions on Communication Theory*, vol. 15, pp. 52–60, 1967.

[108] X. A. Miro, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, “Speaker diarization: A review of recent research,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 356–370, Feb. 2012.

[109] J. Ajmera and C. Wooters, “A robust speaker clustering algorithm,” in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Nov. 2003, pp. 411–416.

[110] S. Katz, “Estimation of probabilities from sparse data for the language model component of a speech recognizer,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 3, pp. 400–401, Mar. 1987.

[111] P. F. Brown, J. Cocke, S. A. D. Pietra, V. J. D. Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin, “A statistical approach to machine translation,” *Computational Linguistics*, vol. 16, no. 2, pp. 79–85, Jun. 1990.

[112] J. Ponte and W. Croft, “A language modeling approach to information retrieval,” pp. 275–281, 1998.

[113] R. Kneser and H. Ney, “Improved backing-off for m-gram language modeling,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Detroit, Michigan, USA, May 1995, pp. 181–184.

[114] R. Rosenfeld, “Two decades of statistical language modeling: Where do we go from here?” in *Proceedings of the IEEE*, vol. 88, 2000, pp. 1270–1278.

[115] S. F. Chen and J. Goodman, “An empirical study of smoothing techniques for language modeling,” *Computer Speech & Language*, vol. 13, no. 4, pp. 359 – 394, 1999.

[116] S. F. Chen and R. Rosenfeld, "A survey of smoothing techniques for ME models," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 1, pp. 37–50, 2000.

[117] D. Gildea and T. Hofmann, "Topic-based language models using EM," in *Sixth European Conference on Speech Communication and Technology*, 1999.

[118] J. R. Bellegarda, "A multi-span language modeling framework for large vocabulary speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 6, no. 5, pp. 456–467, Sep. 1998.

[119] R. Lau, R. Rosenfeld, and S. Roukos, "Trigger-based language models: A maximum entropy approach," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, 1993, pp. 45–48.

[120] F. Jelinek, B. Merialdo, S. Roukos, and M. Strauss, "A dynamic language model for speech recognition," in *Workshop on Speech and Natural Language*, ser. HLT '91, Stroudsburg, PA, USA, 1991, pp. 293–295.

[121] J. Goodman, "A bit of progress in language modeling, extended version," Microsoft Research, Tech. Rep. MSR-TR-2001-72, 2001.

[122] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research (JMLR)*, vol. 3, pp. 1137–1155, Mar. 2003.

[123] H. Schwenk and J. Gauvain, "Training neural network language models on very large corpora," in *Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Oct. 2005, pp. 201–208.

[124] T. Mikolov, M. Karafiát, L. Burget, J. C̃ernocký, and S. Khudanpur, "Recurrent neural network based language model," in *11th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, Makuhari, Chiba, Japan, Sep. 2010, pp. 1045–1048.

[125] T. Mikolov, S. Kombrink, L. Burget, J. C̃ernocký, and S. Khudanpur, "Extensions of recurrent neural network language model," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2011, pp. 5528–5531.

[126] L. Hai Son, A. Allauzen, and F. Yvon, "Measuring the influence of long range dependencies with neural network language models," in *NAACL-HLT Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT.* Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 1–10.

[127] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *13th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, Portland, OR, USA, Sep. 2012, pp. 194–197.

[128] K. Yoon, J. Yacine, S. David, and R. A. M., "Character-aware neural language models," in *30th AAAI Conference on Artificial Intelligence*, 2016.

[129] S. Chandar A P, S. Lauly, H. Larochelle, M. Khapra, B. Ravindran, V. C. Raykar, and A. Saha, "An autoencoder approach to learning bilingual word representations," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 1853–1861.

[130] L. Verwimp, J. Pelemans, H. V. hamme, and P. Wambacq, "Character-word lstm language models," in *EACL*, 2017.

[131] E. Arisoy, T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Deep neural network language models," in *NAACL-HLT Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, ser. WLM '12, Stroudsburg, PA, USA, 2012, pp. 20–28.

[132] R. Kiros, R. Salakhutdinov, and R. Zemel, "Multimodal neural language models," in *31st International Conference on Machine Learning (ICML)*, 2014, pp. 595–603.

[133] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research (JMLR)*, vol. 15, no. 1, pp. 1929–1958, 2014.

[134] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[135] Z. Xie, S. I. Wang, J. Li, D. Lévy, A. Nie, D. Jurafsky, and A. Y. Ng, "Data noising as smoothing in neural network language models," *CoRR*, vol. abs/1703.02573, 2017.

[136] C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson, "One billion word benchmark for measuring progress in statistical language modeling," *CoRR*, vol. abs/1312.3005, 2013.

[137] F. Morin and Y. Bengio, "Hierarchical probabilistic neural network language model," in *10th International Workshop on Artificial Intelligence and Statistics*, 2005, pp. 246–252.

[138] Y. Bengio and J.-S. Sénécal, "Quick training of probabilistic neural nets by importance sampling," in *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2003.

[139] ——, "Adaptive importance sampling to accelerate training of a neural probabilistic language model," *IEEE Transactions on Neural Networks*, vol. 19, no. 4, pp. 713–722, 2008.

[140] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," in *29th International Conference on Machine Learning (ICML)*, 2012, pp. 1751–1758.

[141] M. U. Gutmann and A. Hyvärinen, "Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics," *Journal of Machine Learning Research (JMLR)*, vol. 13, pp. 307–361, Feb. 2012.

[142] B. Zoph, A. Vaswani, J. May, and K. Knight, "Simple, fast noise-contrastive estimation for large RNN vocabularies," in *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, California, USA, Jun. 12-17 2016, pp. 1217–1222.

[143] M. Mahoney, "Large text compression benchmark," 2011. [Online]. Available: http://mattmahoney.net/dc/textdata.html

[144] S. Zhang, H. Jiang, M. Xu, J. Hou, and L. Dai, "The fixed-size ordinally-forgetting encoding method for neural network language models," in *53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing ACL*, vol. 2, July 2015, pp. 495–500.

[145] K. Irie, Z. Tuske, T. Alkhouli, R. Schluter, and H. Ney, "LSTM, GRU, Highway and a bit of Attention: an empirical overview for language modeling in speech recognition," Sep. 2016.

[146] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *2012 IEEE Spoken Language Technology Workshop (SLT)*, Miami, FL, USA, Dec. 2-5 2012, pp. 234–239.

[147] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Cernocký, "Strategies for training large scale neural network language models," in *IEEE Workshop on Automatic Speech Recognition & Understanding (ASRU)*, Waikoloa, HI, USA, Dec. 11-15, 2011, pp. 196–201.

[148] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Cernocký, "Empirical evaluation and combination of advanced language modeling techniques," in *12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, Florence, Italy, Aug. 27-31, 2011, pp. 605–608.

[149] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[150] R. Pascanu, Ç. Gülçehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," *CoRR*, vol. abs/1312.6026, 2013.

[151] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Chia Laguna Resort, Sardinia, Italy, May 2010, pp. 249–256.

[152] P. Xu and F. Jelinek, "Random forests and the data sparseness problem in language modeling," *Computer Speech & Language*, vol. 21, no. 1, pp. 105–152, 2007.

[153] D. Filimonov and M. P. Harper, "A joint language model with fine-grain syntactic tags," in *Conference on Empirical Methods in Natural Language Processing (EMNLP), A meeting of SIGDAT, a Special Interest Group of the ACL*, Singapore, Aug. 2009, pp. 1114–1123.

[154] A. Emami and F. Jelinek, "Exact training of a neural syntactic language model," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Montreal, Quebec, Canada, May 2004, pp. 245–248.

[155] J. R. Bellegarda, "Exploiting both local and global constraints for multi-span statistical language modeling," in *IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP)*, Seattle, Washington, USA, May 12-15 1998, pp. 677–680.

[156] T. Anastasakos, Y. Kim, and A. Deoras, "Task specific continuous word representations for mono and multi-lingual spoken language understanding," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 4-9 2014, pp. 3246–3250.

[157] R. Kuhn and R. De Mori, "A cache-based natural language model for speech recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 6, pp. 570–583, 1990.

[158] X. Chen, X. Liu, M. J. F. Gales, and P. C. Woodland, "Recurrent neural network language model training with noise contrastive estimation for speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5411–5415.

[159] R. Józefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu, "Exploring the limits of language modeling," *CoRR*, vol. abs/1602.02410, 2016.

[160] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[161] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *15th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, Sep. 14-18 2014, pp. 338–342.

[162] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., 2013, pp. 3111–3119.