DIRECTION OF ARRIVAL ESTIMATION AND LOCALIZATION OF MULTIPLE SPEECH SOURCES IN ENCLOSED ENVIRONMENTS

Mikael Swartling

Blekinge Institute of Technology
Doctoral Dissertation Series No. 2012:03

School of Engineering

Direction of Arrival Estimation and Localization of Multiple Speech Sources in Enclosed Environments

Mikael Swartling

Blekinge Institute of Technology doctoral dissertation series
No 2012:03

Direction of Arrival Estimation and Localization of Multiple Speech Sources in Enclosed Environments

Mikael Swartling

Doctoral Dissertation in
Applied Signal Processing

School of Engineering
Blekinge Institute of Technology
SWEDEN

2012 Mikael Swartling
School of Engineering
Publisher: Blekinge Institute of Technology,
SE-371 79 Karlskrona, Sweden
Printed by Printfabriken, Karlskrona, Sweden 2012
ISBN: 978-91-7295-226-3
ISSN 1653-2090
urn:nbn:se:bth-00520

Abstract

Speech communication is gaining in popularity in many different contexts as technology evolves. With the introduction of mobile electronic devices such as cell phones and laptops, and fixed electronic devices such as video and teleconferencing systems, more people are communicating which leads to an increasing demand for new services and better speech quality. Methods to enhance speech recorded by microphones often operate blindly without prior knowledge of the signals. With the addition of multiple microphones to allow for spatial filtering, many blind speech enhancement methods have to operate blindly also in the spatial domain. When attempting to improve the quality of spoken communication it is often necessary to be able to reliably determine the location of the speakers. A dedicated source localization method on top of the speech enhancement methods can assist the speech enhancement method by providing the spatial information about the sources.

This thesis addresses the problem of speech-source localization, with a focus on the problem of localization in the presence of multiple concurrent speech sources. The primary work consists of methods to estimate the direction of arrival of multiple concurrent speech sources from an array of sensors and a method to correct the ambiguities when estimating the spatial locations of multiple speech sources from multiple arrays of sensors. The thesis also improves the well-known SRP-based methods with higher-order statistics, and presents an analysis of how the SRP-PHAT performs when the sensor array geometry is not fully calibrated. The thesis is concluded by two envelope-domain-based methods for tonal pattern detection and tonal disturbance detection and cancelation which can be useful to further increase the usability of the proposed localization methods.

The main contribution of the thesis is a complete methodology to spatially locate multiple speech sources in enclosed environments. New methods and improvements to the combined solution are presented for the direction-of-arrival estimation, the location estimation and the location ambiguity correction, as well as a sensor array calibration sensitivity analysis.

Preface

This doctoral thesis summarizes my work within the field of signal processing. The research work has been carried out at the Department of Electrical Engineering at Blekinge Institute of Technology, Sweden.

Acknowledgements

To begin with, I would like to express my sincere gratitude to the two most important persons during my long and very interesting journey as a Ph.D. candidate: Professor Ingvar Claesson for giving me the opportunity to become a Ph.D. candidate and Doc. Nedelko Grbić for being my supervisor. Your professionalism and enthusiastic, encouraging and inspiring support and guidance have been tremendous assets during my journey. I also wish to thank Dr. Benny Sällberg and Dr. Mikael Nilsson for the all the assistance and day-to-day discussions about even the smallest details and ideas. It has been an honor to join and work in such a solid, open and interesting research group.

I would also like to thank former and present colleagues and my good friends for all the time and activities outside the workplace. A special thank you goes to my dear friends Josef for convincing me to study the field of signal processing in the first place and getting me on the track that later led to this journey, Thomas for the long walks in both good and bad weather and for all the discussions that have come up, and to Anders, Magnus and Anton for the board game evenings with all the strategy analysis, arguments and game mechanics abuse.

A special thank you also goes to my family for all the support I have received throughout the years.

Mikael Swartling

Karlskrona, December 2011

Publications

The thesis is based on the following publications:

[1] Mikael Swartling, Nedelko Grbić, and Ingvar Claesson. Direction of Arrival Estimation for Multiple Speakers using Time-Frequency Orthogonal Signal Separation. In *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing*, volume 4, pages 833–836, May 2006.

[2] Benny Sällberg, Mikael Swartling, Nedelko Grbić, and Ingvar Claesson. Real Time Implementation of a Blind Beamformer for Subband Speech Enhancement using Kurtosis Maximization. In *Proceedings of International Workshop on Acoustic Echo and Noise Control*, September 2006.

[3] Mikael Swartling, Mikael Nilsson, and Nedelko Grbić. Detection of Vehicle Mounted Auditory Reverse Alarm using Hidden Markov Model. In *Proceedings of ELMAR-2007*, pages 163–166, September 2007.

[4] Mikael Swartling, Benny Sällberg, and Nedelko Grbić. Direction of Arrival Estimation for Speech Sources using Fourth Order Cross Cumulants. In *Proceedings of IEEE International Symposium on Circuits and Systems*, pages 1696–1699, May 2008.

[5] Mikael Swartling, Mikael Nilsson, and Nedelko Grbić. Distinguishing True and False Source Locations when Locating Multiple Concurrent Speech Sources. In *Proceedings of IEEE Sensor Array and Multichannel Signal Processing Workshop*, pages 361–364, July 2008.

[6] Mikael Swartling and Nedelko Grbić. Calibration Errors of Uniform Linear Sensor Arrays for DOA Estimation: an Analysis with SRP-PHAT. *Signal Processing*, 91(4):1071–1075, April 2011.

[7] Mikael Swartling, Benny Sällberg, and Nedelko Grbić. Source Localization for Multiple Speech Sources Using Low Complexity Non-Parametric Source Separation and Clustering. *Signal Processing*, 91(8):1781–1788, August 2011.

[8] Mikael Swartling, Benny Sällberg, and Nedelko Grbić. Tone Detection and Cancelation in the Modulated Envelope Domain. *IEEE Signal Processing Letters*, December 2011. Submitted.

Contents

Chapter 1

Introduction

1.1 Motivation and Thesis Topic

Speech communication is gaining in popularity in many different contexts as technology evolves. With the introduction of mobile electronic devices such as cell phones and laptops, and fixed electronic devices such as video and teleconferencing systems, more people are communicating which leads to an increasing demand for new services and better quality. Methods to enhance speech recorded by microphones often operate blindly without prior knowledge of the signals. With the addition of multiple microphones to allow for spatial filtering, many blind speech enhancement methods have to operate blindly also in the spatial domain. When attempting to improve the quality of spoken communication it is often necessary to be able to reliably determine the location of the speakers. A dedicated source localization method on top of the speech enhancement methods can assist the speech enhancement method by providing the spatial information about the sources. With fewer blind domains, potentially more specialized and efficient methods for speech enhancement can be employed.

The demand for automation is also increasing in areas such as audio-based camera steering for applications such as video conferencing and security and surveillance systems. It is essential in such applications to reliably determine the location of a speaker, or any acoustic event in general. This information can then be used to steer the camera, either for the purpose of transmitting the recorded video for distribution in a conference system, or to focus the surveillance on areas where activity has been detected via auditory cues, as opposed to visual cues such as motion sensing. Auditory source localization is also of interest outside speech-related applications, such as in noise field measurements when localizing noise sources in and around a car environment, or auditory alarms within an industrial environment.

This thesis addresses the problem of speech-source localization, with a focus on the problem of localization in the presence of multiple concurrent speech sources. The thesis presents a series of assumptions and theories about the signal and sensor models, and the source localization methods and signal structures that must be considered in relation to signal separation. Methods that address the problems of locating multiple concurrent sources are then proposed, both for direction estimation and for spatial location estimation. Constraints throughout the thesis are the focus on real-time implementations and the resource efficiency necessary if the methods are to be possible to implement on embedded platforms.

The primary work presented in this thesis consists of methods to estimate the direction of arrival of multiple concurrent speech sources from an array of sensors and a method to correct the ambiguities when estimating the spatial locations of multiple speech sources from multiple arrays of sensors. The thesis also improves the well-known SRP-based methods with higher-order statistics, and presents an analysis of how the SRP-PHAT performs when the sensor array geometry is not fully calibrated. The thesis is concluded by two envelope-domain-based methods for tonal pattern detection and tonal disturbance detection and cancelation which can be useful to further increase the usability of the proposed localization methods. In addition to the research contributions, a real-time audio processing framework for Matlab is also introduced. This framework has been a tremendously useful tool for rapid development, implementation, evaluation and demonstration of real-time signal processing methods.

1.2 Thesis Overview

The thesis is divided into three logical parts that describe the background and the introduction to the general topic of source localization, the main contributions to the research field, and supplementary work.

1.2.1 Part One

The first part consists of the introductory chapters. These chapters introduce the signal processing framework and the background leading up to the research contributions in Part Two.

- Chapter 1 presents the background and motivation for the work that is presented, and introduces the notational conventions used throughout the thesis.

- Chapter 2 presents the basic models, tools and conventions used throughout the thesis. This includes a description of a model of how signals are assumed to propagate and interact with an array of sensors and how the sparse and disjoint structure of independent speech sources can be exploited for signal separation. This chapter also outlines essential tools such as the filterbank for time-frequency-domain decomposition of a time-domain signal.

- Chapter 3 introduces the concept of source localization, describes different meanings of this concept and presents various approaches to perform source localization.

1.2.2 Part Two

The second part consists of the main topic of the thesis and the majority of the work.

- Chapters 4 and 5 present two methods to estimate the direction of arrival for multiple concurrent speech sources. The major contributions in these chapters are two methods to perform direction-of-arrival estimation of multiple concurrent speech sources that, contrary to many other similar methods, algorithmically allow the number of sources they can simultaneously locate to be greater than the number of sensors.

- Following a successful direction-of arrival estimation of multiple concurrent sources, chapter 6 presents a method to estimate the physical location of sources. The major contribution in this chapter is a method to handle the problem of having multiple combinations of intersection points that occurs when using multiple independent sensor arrays.

- Chapter 7 presents an analysis of how the popular steered response power with phase transform method behaves when the sensors within the sensor array are not sufficiently position-calibrated. The major contribution in this chapter is the analysis that shows that the steered response power with phase transform method is (or can be made) robust and tolerant to moderately uncertain sensor positions.

- Chapter 8 presents a method that extends the steered response power method for direction-of-arrival estimation with the help of higher-order statistics. The major contribution in this chapter is the extension from second-order statistics to fourth-order statistics which, in general, provides a way to distinguish speech sources from noise.

An important element in these contributions has been to ensure that the methods allow for an efficient real-time implementation. The efficiency requirement refers to both computational efficiency and memory efficiency. In other words, the complexities must be low enough to allow implementation on small embedded platforms with limited resources. The real-time requirement refers to that the method must be able to execute in a continuously executing environment with only an acceptable delay. Thus, it must be causal and not rely on future events, nor shall block-processing be performed in unnecessary large blocks so that the delay between action and response becomes unusable in a practical application.

Both the efficiency and the real-time requirements are very loosely defined and there are often no distinct boundaries for what is considered to be efficient, or when a method cannot operate in real-time. The boundaries can be both subjective, as in the case of a perceptual delay between action and response, or well defined physical limitations such as the actual processing power and memory available on a target platform.

1.2.3 Part Three

The third part consists of complements to the main topic of the thesis.

- Chapter 9 presents a method for the detection of a vehicle-mounted audible reverse alarm. The major contribution in this chapter is a framework to detect an alarm signal that consists primarily of amplitude modulated tonal components.

- Chapter 10 presents a method for tone detection and cancelation. The major contribution in this chapter is an envelope-domain-based method for the detection and cancelation of multiple concurrent tonal disturbances.

- Chapter 11 presents the development of a real-time audio processing framework for Matlab. The framework is a very helpful tool for rapid development, evaluation and demonstration of real-time audio signal processing within Matlab.

1.3 Notations

The following notations define the base notations and conventions used throughout the thesis. Exceptions exist, but are then noted as such when used.

1.3.1 Typographical Notations

The following typographical notations are used to denote value type, order and dimensionality:

x	Scalar value (lower case, italics).
x	Universal constant values (lower case, upright).
$\mathbf{x}$	Column vector (lower case, bold).
$\mathbf{X}$	Matrix and Set (upper case, bold).

The following typographical notations and parenthesis or brackets are used to denote signal domain and continuous or discrete time or time-frequency points:

x	Time domain signal (lower case).
X	Frequency or time-frequency domain signal (upper case).
$x(t)$	Signal value at time t.
$x[n]$	Signal value at sample index n.
$X(\omega)$	Signal value at real angular frequency ω.
$X[k]$	Signal value at subband index k.
$X(\tau,\omega)$	Signal value at time-frequency point (τ,ω).
$X[n,k]$	Signal value at time-frequency index $[n,k]$.
ω_k	Normalized angular frequency for the center of the subband k; $\omega_k = 2\pi k/K$.

1.3.2 Constants

The following universal constants are used:

c	Propagation speed of sound; $c \approx 344\,\mathrm{m\,s^{-1}}$.
e	Natural logarithm base.
j	Imaginary unit.

1.3.3 Operators

The following operators are used:

$E[\cdot]$	Expectation operator.
$\hat{E}[\cdot]$	Sample based estimate of the expectation operator.
$V[\cdot]$	Variance operator.
$\Im[\cdot]$	Imaginary part of complex value.
$\mathrm{int}[\cdot]$	Rounding to nearest integer towards zero.
$\lvert\cdot\rvert$	Absolute value.
$\lVert\cdot\rVert$	Euclidean vector norm or matrix norm.
$\angle$	Angle of complex value.
$\star$	Convolution.
$\circ$	Vector or matrix element-wise multiplication.
$\square^{\star}$	Complex conjugate.
$\square^{T}$	Vector or matrix transpose.
$\square^{H}$	Vector or matrix Hermitian transpose.
$\hat{\square}$	Estimated value.
$\tilde{\square}$	Time-reversed sequence.

1.3.4 Functions and Transforms

The following functions are used:

$\delta(\cdot)$	The Dirac delta function.
$\mathcal{F}\{\cdot\}$	The forward Fourier transform; discrete or continuous.
$\mathcal{F}^{-1}\{\cdot\}$	The inverse Fourier transform; discrete or continuous.
$\mathrm{sinc}(\cdot)$	The Sinc function; $\mathrm{sinc}(x) \equiv \frac{\sin(\pi x)}{\pi x}$ and $\mathrm{sinc}(0) \equiv 0$.

1.3.5 Names for Common Quantities

The following names are given to common quantities:

I, i	Number of sources and source index.
J, j	Number of sensors and sensor index.
K, k	Number of subbands and subband index.
$\mathbf{P}, \{p, q\}$	Set of sensor index pairs and sensor index pair.
s, S	Original source signal.
x, X	Received sensor signal.
v, N	Noise signal.
h, g	Signal propagation path filter from source to sensor; with ($h(t)$ or $h[n]$) and without ($g(t)$ or $h[n]$) direct path propagation components.
G	Cross-power spectrum.
$\mathbb{M}$	Time-frequency mask.
τ	Propagation delay or time-difference of arrival.
a	Propagation attenuation.
α	Angle of arrival.
d	Sensor spacing.

$\mathbf{v}$	Direction-of-arrival vector; unit norm.
$\mathbf{s}$	Source location.
$\mathbf{p}$	Sensor location.

1.3.6 Subscripts and Decorations

Many of the quantities are associated with a subscript notation to indicate, most commonly, an association with a source and/or a sensor. Reoccurring subscript constellations and their typical uses include:

- Subscript i denotes a source i:

$s_i(t)$	Source signal from the source i.
τ_i	Propagation delay for the source i to some reference point.
$\mathbf{v}_i$	Direction-of-arrival vector towards the source i.

- Subscript j denotes a sensor j:

$x_j(t)$	Sensor signal recorded at the sensor j.
$\mathbf{p}_j$	Location of the sensor j.

- Subscript i, j denotes a source i and a sensor j:

$\mathbf{h}_{i,j}$	Signal propagation path filter from the source i to the sensor j.
$\mathbb{M}_{i,j}$	Time-frequency mask to mask the source i from the sensor signal j.

- Subscript p, q denotes the pair of the sensors p and q:

$\tau_{p,q}$	Propagation delay difference from a source between the sensor p and the sensor q.
$G_{p,q}$	Cross-power spectrum between the signals from the sensor p and the sensor q.

- Additional subscripts, as well as other decorations, may be present and follow the same convention:

$\hat{G}_{i,p,q}$	Estimate of the cross-power spectrum for the source signal i between the sensor p and the sensor q.

1.3.7 Figures

The following symbols are used in the figures:

★	Source.
●	Sensor or sensor array.

Chapter 2

Source and Signal Models

Much of the work done when new signal processing methods are researched includes theoretical and mathematical work, yet the theory and mathematics have to be connected to the real world and thus to the physical domain. The connections between theory and the physical domain are described by models and assumptions that define a framework in which signal processing can be performed.

This chapter introduces the most important models and assumptions of this thesis and provides the foundation on which all proposed methods are based. The specific models concern mainly how signals are assumed to propagate in space and time, and how they interact with a spatially spread sensor array. A model of the temporal structure of speech signals, which are of particular interest in this thesis, is also presented. Finally, some additional tools are presented that by themselves are not actual models or assumptions but which are nonetheless important when carrying out the specified set of tasks. These tools comprise a fast and efficient filterbank structure and a room impulse response model for the approximation of reverberation effects in an enclosed environment.

2.1 The Signal Propagation Models

A signal propagation model describes how a signal is assumed to propagate from a transmitting source to a receiving sensor. Two signal propagation models are commonly used: the convolutive signal propagation model and the anechoic signal propagation model.

2.1.1 The Convolutive Signal Propagation Model

The convolutive signal propagation model describes the signal propagation path from the transmitting source to the receiving sensor as a filtering operation:

$$x(t) = s(t) \star h(t) + \nu(t). \tag{2.1}$$

The corresponding frequency-domain model then becomes

$$X(\omega) = S(\omega)\, H(\omega) + N(\omega). \tag{2.2}$$

The filter $h(t)$ models the linear physical properties of the propagation path, such as propagation delay and attenuation, and multi-path propagation or reverberation. The noise signal $\nu(t)$ is added to the received sensor signal to model the environment and the sensor noise. The noise is typically considered to be both temporally and spatially white Gaussian noise.

The convolutive signal propagation path is typically very complex to model. In this thesis, however, the specific details of the filter are of lesser importance, but the fact that the signal propagation path can be modeled by a filter is assumed. A method to estimate the impulse response for a rectangular room is presented in section 2.6 for the purpose of generating evaluation signals.

2.1.2 The Anechoic Signal Propagation Model

The anechoic signal propagation model is a simplification of the convolutive signal propagation model in that the signal is assumed to propagate in an anechoic environment. The signal is here assumed to only have a direct propagation path between the source and the receiver, and no indirect paths due to reflections from the environment. The filter $h(t)$ from the convolutive signal model is then reduced to model only the propagation delay and the attenuation:

$$x(t) = a\, s(t-\tau) + \nu(t). \tag{2.3}$$

The corresponding frequency-domain model then becomes

$$X(\omega) = a\, S(\omega)\, \mathrm{e}^{-\mathrm{j}\omega\tau} + N(\omega). \tag{2.4}$$

Due to the propagation distance between the transmitting source and the receiving sensor, denoted the *mixing parameters*, the source signal $s(t)$ is subject only to an attenuation factor a and a time delay τ.

2.1.3 The Multi-Source and the Multi-Sensor Model

The work presented in this thesis is focused on arrays consisting of multiple sensors. The fundamental signal propagation models themselves do not change to account for multiple sensors, except in the naming of the parameters of the signal propagation models. The individual sensors within the sensor array can be modeled independently, with an additional subscript to denote a sensor index.

Multiple sources, on the other hand, interact additively with each other due to the linearity of the propagation models, so the multi-source signal propagation models have to be extended to account for multiple sources. The additive interaction is simply the sum of single-source signal propagation models, Thus, the anechoic multi-source and multi-sensor signal propagation model is defined as

$$x_j(t) = \sum_{i=0}^{I-1} a_{i,j}\, s_i(t - \tau_{i,j}) + \nu(t) \tag{2.5}$$

and the corresponding frequency-domain model as

$$X_j(\omega) = \sum_{i=0}^{I-1} a_{i,j}\, S_i(\omega)\, \mathrm{e}^{-\mathrm{j}\omega\tau_{i,j}} + N(\omega). \tag{2.6}$$

Since each source location has its own unique combination of mixing parameters not only from their location in physical space, but also to where the receiving sensors are located, the mixing parameters $a_{i,j}$ and $\tau_{i,j}$ from the source i to the sensor j are defined for each source and sensor combination.

The convolutive multi-source and multi-sensor signal propagation model is extended accordingly, but with the filter $h_{i,j}(t)$ describing the propagation path from the source i to the sensor j instead.

2.2 Mixing Parameter Normalization

When working with a small sensor array, some observations are made that lead to a simplified signal propagation model. The first observation is that, in a passive system where only the received sensor signals are observed, the source signal itself cannot be observed. Therefore, it is impossible to estimate the global mixing parameters $a_{i,j}$ and $\tau_{i,j}$ used in the signal model presented earlier. Instead of having global mixing parameters, the mixing parameters are normalized with respect to an arbitrary reference point within the sensor array.

If $a_{i,\mathrm{r}}$ and $\tau_{j,\mathrm{r}}$ correspond to the mixing parameters from the source i to an arbitrary reference point r within the sensor array, these parameters represent the global propagation attenuation and delay of the source signals. The local mixing parameters can be normalized with respect to the reference point:

$$a'_{i,j} = \frac{a_{i,j}}{a_{i,\mathrm{r}}} \tag{2.7}$$

$$\tau'_{i,j} = \tau_{i,j} - \tau_{i,\mathrm{r}}. \tag{2.8}$$

To normalize the mixing parameters, a common propagation attenuation factor is factored out, and a common propagation delay is subtracted from the global mixing parameters with respect to a local reference point. The local mixing parameters describe the additional propagation attenuation and delay with respect to the reference point (or, depending on the relative location of the reference point, a gain or a forward time "delay"). Note that the reference point need not be one of the sensors, but can be an arbitrary point close to (within or outside) the sensor array.

The first simplification, or rather a change of notation, that is made to the signal propagation model concerns the notation and the meaning of the mixing parameters. Since the source signals cannot be observed, it is reasonable for the signal model to exclude the full path from the source and instead focus on the local mixing parameters. The change of notation is simply that $a_{i,j}$ and $\tau_{i,j}$ now denote the *local* mixing parameters instead of the global mixing parameters.

The second simplification that is made to the signal model concerns the propagation attenuation. Since the sensor array is presumed to be small, it is assumed that $a_{i,\mathrm{r}} \approx a_{i,j}$. Therefore, the normalized local mixing parameters have a near-unit local propagation attenuation, meaning that $a'_{i,j} \approx 1$. The anechoic multi-source signal model is simplified to

$$x_j(t) = \sum_{i=0}^{I-1} s_m(t - \tau_{i,j}) + \nu_j(t) \tag{2.9}$$

and the corresponding frequency-domain model to

$$X_j(\omega) = \sum_{i=0}^{I-1} S_i(\omega)\mathrm{e}^{-\mathrm{j}\omega\tau_{i,j}} + N_j(\omega) \tag{2.10}$$

where $\tau_{i,j}$ now denotes the local propagation delay for the source i between the sensor array reference point and the sensor j. The attenuation mixing parameter has been excluded from the signal propagation model.

2.3 Far-Field Source Location

The removal of the propagation attenuation mixing parameter was based on the assumption that the sensor array is small and that the propagation attenuation difference between the reference point and any sensor within the sensor array is negligible. This is equivalent to assuming that the source is located far away from the sensor array and, furthermore, that the propagating wave, once it reaches the sensor array, is a planar wave local to the sensor array.

Consider two sensors p and q located at the spatial positions $\mathbf{p}_p$ and $\mathbf{p}_q$, respectively, and a source located at $\mathbf{s}$. The time-difference of arrival between the two sensors is then

$$T(\mathbf{p}_p, \mathbf{p}_q, \mathbf{s}) = \frac{\left\|\mathbf{p}_q - \mathbf{s}\right\| - \left\|\mathbf{p}_p - \mathbf{s}\right\|}{\mathrm{c}}. \tag{2.11}$$

In a source localization framework where the goal is to determine $\mathbf{s}$, the delay $T(\cdot)$ is observed from the received sensor signals while $\mathbf{p}_p$ and $\mathbf{p}_q$ are known from the sensor array geometry. Given the observed time delay $\tau_{p,q}$ between the two received sensors signals, the source is located at $\hat{\mathbf{s}}$ such that

$$\tau_{p,q} = T(\mathbf{p}_p, \mathbf{p}_q, \hat{\mathbf{s}}). \tag{2.12}$$

The solution to $\hat{\mathbf{s}}$ is located on one sheet of a hyperboloid of two sheets which axis of symmetry is the vector between the two sensors. This means that there is no single point that uniquely satisfies the equation. Instead, the set of solutions is all points that are located on the surface of the hyperboloid. The location of the source cannot, therefore, be determined by $\hat{\tau}_{p,q}$ alone. Figure 2.1(a) shows the relationship between the source location, the sensor locations, and the time-difference of arrival. In the figure, the thin dashed line shows the hyperboloid set of solutions to (2.11): the source $\mathbf{s}$ can be located anywhere along this line and satisfy the equation in (2.11). In three dimensions, this line revolves around the sensor array base line and becomes a surface.

The hyperboloid model of the set of possible source locations is known as the near-field model. By studying the behavior of the hyperboloid as a set of possible solutions under the assumption that the source is located sufficiently far away from the sensor array, the hyperboloid approaches, and can thus be approximated by, a cone [1]. The cone model is also known as the far-field model.

As a result of the far-field model, the propagating wave fronts are no longer assumed to be propagating spherically in all directions from a point in space, but instead travel as parallel wave fronts propagating in a single direction. By replacing the source position **s** by a unit vector **v** pointing from the sensor array towards the source, or equivalently describing the inverse direction of propagation of a wavefront, the time delay between two sensors becomes

$$T(\mathbf{p}_p, \mathbf{p}_q, \mathbf{v}) = \frac{\mathbf{p}_q^{\mathrm{T}}\mathbf{v} - \mathbf{p}_p^{\mathrm{T}}\mathbf{v}}{\mathrm{c}}. \tag{2.13}$$

It is assumed that the sensor array baseline is also the reference for the coordinate system in which the sensor locations and the direction-of-arrival vector or the angle of arrival is defined. The baseline along which the sensor are located defines the first principal axis. The direction-of-arrival vector is then related to the angle of arrival as

$$\mathbf{v} = \begin{bmatrix} \sin(\alpha) & \cos(\alpha) \end{bmatrix}^{\mathrm{T}} \tag{2.14}$$

and the time-difference of arrival can be expressed as

$$\begin{aligned} T(\mathbf{p}_p, \mathbf{p}_q, \mathbf{v}) &= \frac{\left\|\mathbf{p}_q - \mathbf{p}_p\right\|}{\mathrm{c}} \cdot \sin(\alpha) & (2.15) \\ &= \frac{d}{\mathrm{c}} \cdot \sin(\alpha) & (2.16) \end{aligned}$$

where d denotes distance between the sensors, and the product $d \sin(\alpha)$ is the *effective sensor spacing*. The effective sensor spacing is the sensor spacing as seen from the direction of the source. Figures 2.1(b) shows the relationship between the source location, the sensor locations, and the time-difference of arrival. Figure 2.1(c) shows the relationship between the direction-of-arrival vector and the corresponding angle of arrival. In figure 2.1(b), the thin dashed line again shows the hyperboloid set of solutions to (2.11) and how it is approximated by a straight line in the far field.

An important conceptual point to note here is that although the direction-of-arrival vector is presented as a two-dimensional vector, implying that the sensor location vectors are also two-dimensional, this notation is not restricted to two dimensions. In three dimensions, there is a rotational symmetry of the cone (and the hyperboloid in the near-field model), and the second dimension fully represents the remaining spatial dimensions due to the rotational symmetry around the baseline. Thus, the first principal axis defines the sensor array baseline and the second dimension represents the remaining spatial dimensions.

2.4 The Multi-Sensor Array

A generalization of the two-sensor array is to use an array with an arbitrary number of sensors. A convention has to be set that defines how directions and time delays are defined in the presence of multiple sensors. For the two-sensor array, the time delay is simply defined as the time delay between the only two sensors. The angle α typically has two conventions: either it is the angle from the baseline, or, as defined herein, it is the angle perpendicular to the baseline. The difference is simply a change of sines and cosines in the subsequent models.

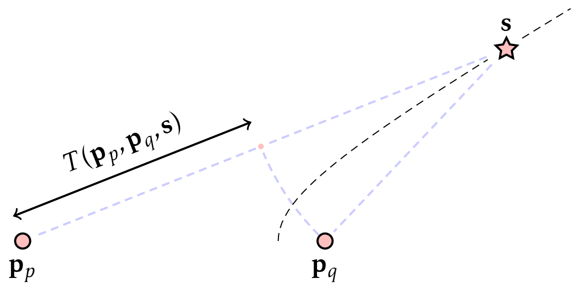
2.4.1 Arbitrary Sensor Array

An arbitrary sensor geometry, for example the circular sensor array shown in figure 2.2(a), does not necessarily have a natural reference point from where a single direction of arrival or a time delay describing the direction of arrival of a source can be defined. Here, the vector $\mathbf{v}$ describes the direction relative to a fixed coordinate system.

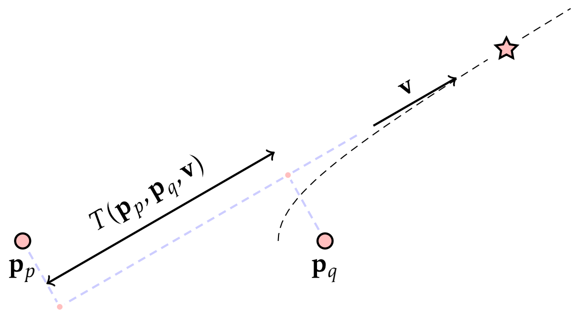
2.4.2 Uniform Linear Sensor Array

A common type of sensor array is the uniform linear sensor array, where the sensors are uniformly distributed along a common array baseline, as shown in figure 2.2(b). As in the case of the two-sensor array, the angle of arrival can be defined as the angle relative to the baseline, or perpendicular to the baseline as shown in the figure.

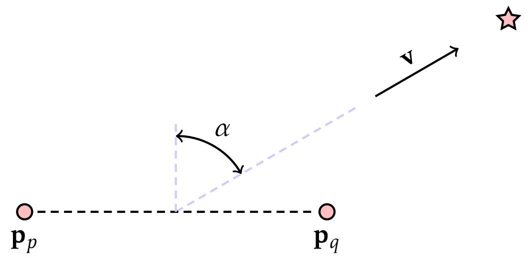
An interesting property of the uniform linear sensor array is that the time-difference of arrival is the same for all pairs of adjacent sensor within the array. Similarly, by extension the time-difference of arrival for pairs of second-order adjacent sensors is twice that of adjacent pairs. The same is also true for kth order adjacent pairs. As a result of this, it is possible to define a single time-difference of arrival for the whole sensor array that corresponds to



(a) The hyperboloid model with time-difference of arrival.

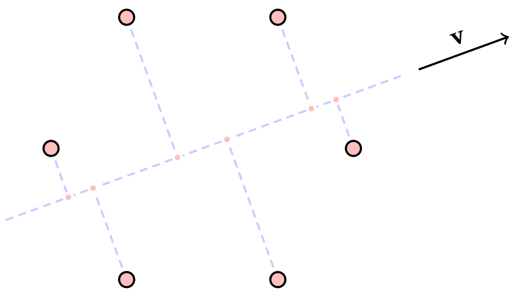


(b) The cone model with time-difference of arrival.

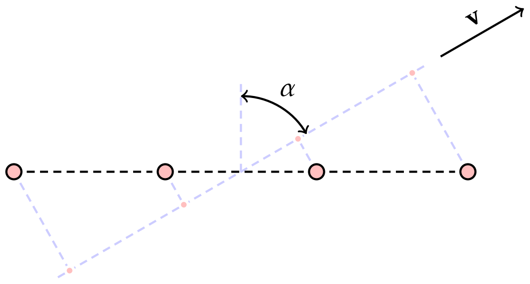


(c) The cone model with the relationship between the direction-of-arrival vector and the angle of arrival.

Figure 2.1: The hyperboloid model and the cone model: (a) the time-difference of arrival for radial wave fronts, (b) the time-difference of arrival for planar wave fronts, and (c) the relationship between the direction-of-arrival vector and the angle of arrival for the cone model.



(a) Circular sensor array geometry.



(b) Uniform linear sensor array.

Figure 2.2: A six-sensor circular array representing an arbitrary sensor array geometry, and a four-sensor uniform linear sensor array.

the time delay for pairs of adjacent sensors. This common time-difference of arrival can be scaled accordingly for higher-order pairs.

It is, of course, possible to define angles of arrival for an arbitrary sensor array as well, since it is possible to uniquely map the angles and the vectors. There must, after all, be a coordinate system defined in which sensor positions and direction vectors are defined. The difference between the arbitrary and specialized array geometries is that the specialized geometries, such as the uniform linear sensor array, may have a reasonable reference based on the sensor array geometry alone. In the case of an arbitrary sensor array geometry, the reference points may have to be based on the coordinate system instead.

2.5 Estimation Error and Variance

Equation (2.16) describes how the time-difference of arrival relates to an angle of arrival, or equivalently, to a direction-of-arrival vector. In practical localization applications, it is often desired to determine the direction of arrival given a known time-difference of arrival instead. This is easily accomplished by solving for α:

$$\alpha = \arcsin\left(\frac{\mathrm{c}}{d} \cdot \tau\right). \tag{2.17}$$

The equation is a non-linear mapping from the time-difference of arrival τ to the corresponding angle of arrival α.

To see how the estimation errors of the time-difference of arrival τ propagate through the non-linear transformation to the angle of arrival α, it is first necessary to distinguish the true values from the estimated values. The true time-difference of arrival and angle of arrival are denoted τ and α. The estimated time-difference of arrival and angle of arrival, which are corrupted by measurement noise, are denoted $\hat{\tau}$ and $\hat{\alpha}$.

Assuming that the probability density function of the observations is concentrated near its mean value, the moments of the random variable $\hat{\alpha} = f(\hat{\tau})$ can be approximated from the moments of the random variable $\hat{\tau}$ and the Taylor series expansion of the transforming function around the mean value of the random variable. A first-order Taylor series expansion of the transforming function yields

$$\mathrm{E}[\hat{\alpha}] \approx f(\tau) \tag{2.18}$$

and

$$\mathrm{V}[\hat{\alpha}] \approx \left[\frac{\mathrm{d}}{\mathrm{d}\tau} f(\tau)\right]^2 \cdot \mathrm{V}[\tau]. \tag{2.19}$$

The non-linear transforming function is

$$\alpha = f(\tau) = \arcsin\left(\frac{\mathrm{c}}{d} \cdot \tau\right) \tag{2.20}$$

and its first derivative is

$$\frac{\mathrm{d}}{\mathrm{d}\tau} f(\tau) = \frac{\mathrm{c}}{d} \cdot \frac{1}{\sqrt{1 - \left(\frac{\mathrm{c}}{d} \cdot \tau\right)^2}}. \tag{2.21}$$

Furthermore, observe that

$$\arcsin\left(\frac{c}{d}\cdot\tau\right)=\alpha \leftrightarrow \frac{c}{d}\cdot\tau=\sin(\alpha). \tag{2.22}$$

The variance of the estimated angle of arrival becomes

$$\begin{aligned}
V[\hat{\alpha}] &\approx \frac{c^2}{d^2}\cdot\frac{1}{1-\left(\frac{c}{d}\cdot\tau\right)^2}\cdot V[\tau] && (2.23)\\
&= \frac{c^2}{d^2}\cdot\frac{1}{1-\sin^2(\alpha)}\cdot V[\tau] && (2.24)\\
&= \frac{c^2}{d^2}\cdot\frac{1}{\cos^2(\alpha)}\cdot V[\tau]. && (2.25)
\end{aligned}$$

An important observation is that the variance of the estimated angle of arrival increases as the true angle of arrival approaches ±90°. This means that, assuming that the estimate of the time-difference of arrival has a constant variance for all angles of arrival, both the expectation error and the variance of the estimated angle of arrival are lowest where α is 0°, and increase as α approaches ±90°. These points are denoted the *broadside* and the *endfires* of a uniform linear sensor array, respectively.

2.6 Room Impulse Response Modeling

A substitute to real recordings when evaluating an algorithm is to synthesize the evaluation signals. The signals recorded by a sensor array can be modeled by a simple ideal direct path propagation model, which provides a clean and undistorted theoretical representation of the received sensor signals. In a real recording environment there are many destructive elements that, sometimes severely, degrade and distort the received sensor signals. These destructive elements can also be considered in the model so as to construct a more realistic representation of an actual recording environment.

Typical destructive elements are acoustic feedback, noise and reverberation. Of these, reverberation is often considered the most difficult element to handle [2]. Reverberation is a process where the propagating wave bounces off of the walls and other surfaces of an enclosed environment to create a multitude of differently delayed and attenuated echoes. It is, therefore, necessary to simulate the reverberation of an enclosed space as a significant and realistic destructive element to the synthesized evaluation signals.

There are typically three approaches, each associated with multiple methods, to simulate how the propagating wave interacts with an enclosed space: [3]

- **Wave-based models** attempt to solve the wave equation, analytically or numerically [4]. Analytical solutions exist for trivial room shapes and conditions, but in most cases numerical methods, such as finite element models and boundary element models, are needed. These methods are typically more accurate than the ray-based models discussed below, but they are also numerically expensive and are only practical when dealing with relatively small enclosures.

- **Ray-based models** model the propagating wave field as a set of geometrical rays reflecting off of surfaces, for example by mirroring the source location over the surfaces and representing the reflection with corresponding direct paths from the reflected positions [5].

- **Statistical models** model the statistical behavior rather than the temporal behavior of the enclosed environment. Thus, this approach is suitable for noise modeling but not for impulse response generation.

The properties of reverberating acoustic waves differ greatly for different frequencies and their relation to the dimension of the enclosed environment. No single method can practically handle the relatively wide range of frequencies typically encountered in acoustic signal processing. The wave-based models are accurate, but only practical for small enclosures, or equivalently, for long wavelengths. As the wavelength decreases, the ray-based models become increasingly more accurate.

2.6.1 The Image Method

Image methods approximate the room impulse response by transforming multipath reflections from a single source into direct paths from multiple virtual sources. The virtual sources are determined by mirroring the original source over the surfaces of the room, and the multi path impulse response is modeled as the sum of the ideal direct paths for a set of virtual sources [6, 7]. Figure 2.3 shows some virtual sources obtained from mirroring the original source in a 5×5 grid. Although mirroring is typically performed in all three spatial dimensions, only two-dimensional reflections are shown for the purpose of illustration. The original source is within the original room at the reflection index $\{0,0\}$, and all virtual sources within the grid are shown along with their direct paths. The virtual source at the reflection index $\{2,2\}$ has been emphasized, and also shows the corresponding reflection points in the virtual mirrored space (the points along the virtual direct path line), as well as the actual reflection points and trace in the original room.

The discrete-time room impulse response is modeled as

$$h[n] = \sum_{i,j,k} \alpha_{i,j,k} \cdot \beta_{i,j,k} \cdot \operatorname{sinc}(n - \tau_{i,j,k}) \tag{2.26}$$

or in the frequency domain as

$$H[k] = \sum_{i,j,k} \alpha_{i,j,k} \cdot \beta_{i,j,k} \cdot \mathrm{e}^{-\mathrm{j}\omega_k \tau_{i,j,k}} \tag{2.27}$$

where α is the reflection loss attenuation, β is the propagation attenuation, τ is the propagation delay and $\{i,j,k\}$ is the reflection index. This model is the same as the anechoic signal model presented earlier, the only difference being that there are two attenuation factors instead of one. A description of each of these parameters follows.

Virtual Source Location

The location of a virtual source with the reflection index $\{i,j,k\}$ can be expressed as

$$\mathbf{s}_{i,j,k} = \operatorname{diag}\begin{bmatrix} -1^i & -1^j & -1^k \end{bmatrix} \mathbf{s} + \operatorname{diag}\begin{bmatrix} i & j & k \end{bmatrix} \mathbf{r} \tag{2.28}$$

where $\mathbf{s}$ is the location of the original source, $\mathbf{r}$ is a vector containing the dimension of the room, and where the origin of the coordinate system is located at the center of the room. The virtual source indices i, j and k are the indices along the three spatial axes of the room. As a consequence, indices can be both positive and negative, and the reflection index $\{i,j,k\} = \{0,0,0\}$ is the real source location.

Propagation Delay and Attenuation

Given a sensor position $\mathbf{p}$, the virtual direct path propagation vector is

$$\mathbf{d}_{i,j,k} = \mathbf{p} - \mathbf{s}_{i,j,k}. \tag{2.29}$$

The propagation delay is simply

$$\tau_{i,j,k} = \left\|\mathbf{d}_{i,j,k}\right\| \cdot \frac{f_\mathrm{s}}{\mathrm{c}} \tag{2.30}$$

where f_s is the sampling frequency. This is the time taken, in samples, to propagate the length of the virtual direct path propagation vector. The propagation attenuation is modeled by

$$\beta_{i,j,k} = \frac{1}{\left\|\mathbf{d}_{i,j,k}\right\|}. \tag{2.31}$$

Reflection Loss

An interesting property of the reflections of the rectangular room and the indexing scheme used is that the values $|i|$, $|j|$ and $|k|$ correspond to the number of reflections along the corresponding axes. For the purpose of modeling the reflection loss, this trivializes the calculation of the attenuation factor as a part of the direct path attenuation for each virtual source:

$$\alpha_{i,j,k} = \alpha^{|i|+|j|+|k|} \tag{2.32}$$

where $|\alpha| < 1$ is the reflection loss coefficient for a single reflection, and $\alpha_{i,j,k}$ is the total reflection loss from the virtual source index $\{i, j, k\}$.

Reflections in a rectangular environment can only occur on the axes of alternate facing surfaces. A single surface can never result in two or more consecutive reflections along the corresponding axis. Referring to figure 2.3, an even number of reflections in, say, index i corresponds to $i/2$ reflections in each of the left and right surfaces. An odd number of reflections means an additional reflection in either the left or the right surface. If $i < 0$, the additional reflection is in the left surface, and if $i > 0$, the additional reflection is in the right surface.

The reflection loss can be generalized with individual reflection loss coefficients defined for each surface based on the reflection index alone:

$$\alpha_i = \alpha_{\mathrm{i}+}^{\mathrm{int}[|i|/2]} \cdot \alpha_{\mathrm{i}-}^{\mathrm{int}[|i|/2]} \cdot \begin{cases} \alpha_{\mathrm{i}-} & i < 0 \\ 1 & i = 0 \\ \alpha_{\mathrm{i}+} & i > 0 \end{cases} \tag{2.33}$$

where $\alpha_{\mathrm{i}+}$ and $\alpha_{\mathrm{i}-}$ are the reflection loss coefficients for the surface bordering on the positive and the negative sides of the axis, for the i reflection index.

The reflection loss coefficients α_j and α_k are calculated accordingly with the corresponding j and k reflection indices, so that the total reflection loss coefficient is

$$\alpha_{i,j,k} = \alpha_i \cdot \alpha_j \cdot \alpha_k. \tag{2.34}$$

The points of reflection for a single virtual direct path component can be determined by tracing the straight line between the virtual source and the receiving sensor, finding all intersection points between the path and the boundaries of the reflected rooms. The intersection points can then be reflected back to the index $\{i, j, k\} = \{0, 0, 0\}$ to obtain the corresponding sequence of reflection points between the original source and the receiving sensor for any virtual source point.

Ideally, the summations in (2.26) and (2.27) shall be over all i, j and k, but this is not realistic in practice. Since both the propagation attenuation and the reflection loss attenuation increases as the reflection order increases, summation can easily be limited with this indexing scheme to reflections below the order N_{ref} by ensuring that

$$|i| + |j| + |k| < N_{\text{ref}}. \tag{2.35}$$

Reflections of a specific order is obtained at equality. Limiting the total number of reflections creates a diamond-shaped set of virtual source locations. Reflections can also, as in figure 2.3, be limited along the reflection indices individually, creating a rectangular-shaped set of virtual source locations.

Figure 2.4 shows the discrete-time impulse response generated by the example in figure 2.3. The room is modeled to be $4\,\text{m} \times 4\,\text{m}$, the sample rate is $8\,\text{kHz}$, the reflection attenuation coefficient is $\alpha = 0.5$, and only reflections in two dimensions are considered to match the figure. The whole impulse response corresponds to $h[n]$, which can be decomposed into the dominant direct path $\text{sinc}(n - \tau)$, where τ is the direct path propagation delay, and $g[n]$ is the reverberation impulse response. The direct path is dominant to all the reverberation paths in the environment, which fade rapidly as the reflection order and the propagation distance increase.

2.7 Time-Frequency Decomposition and Filterbanks

The relation between the time domain and the frequency domain is often expressed with the well-known Fourier transform. A signal is transformed from

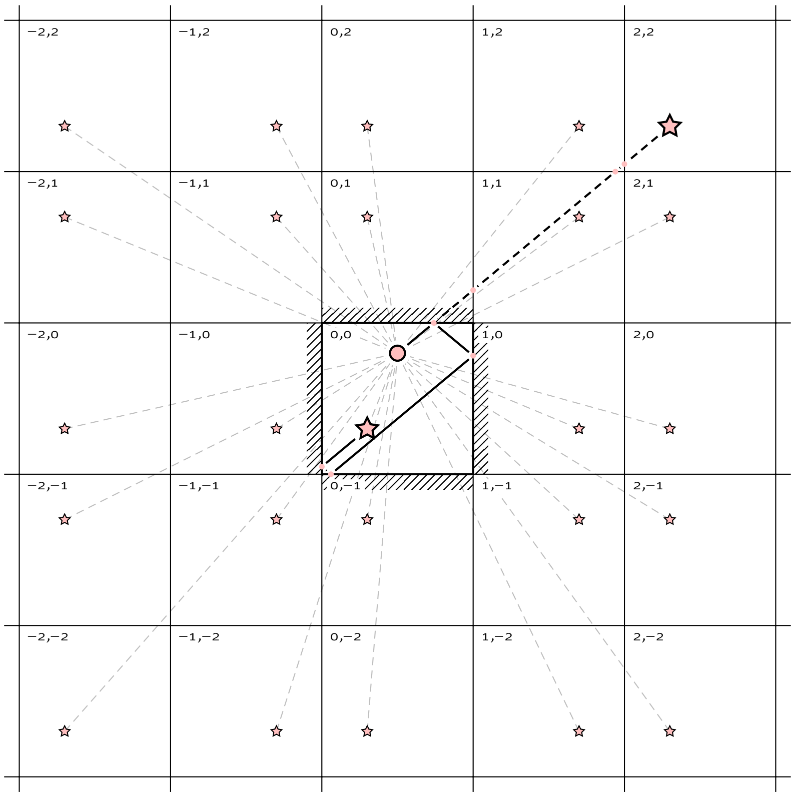


Figure 2.3: The image method simulating reverberation by using the direct paths of a 5×5 grid of reflected virtual sources.

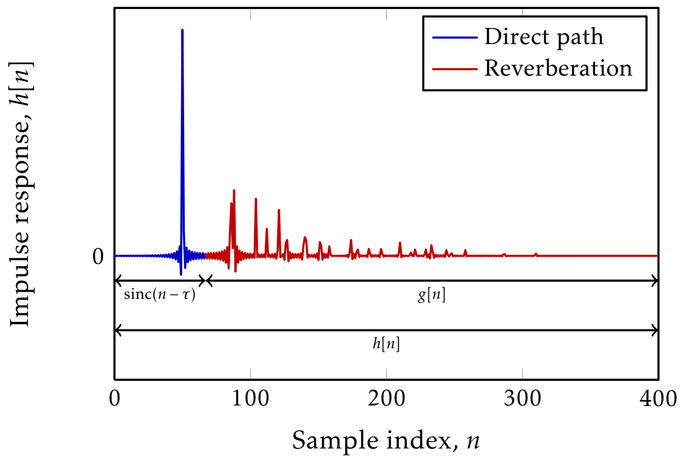


Figure 2.4: An impulse response generated by the image method simulating reverberation by using the direct paths of a 5×5 grid of reflected virtual sources.

the time domain into the frequency domain by the forward Fourier transform, and from the frequency domain into the time domain with the inverse Fourier transform. One issue with the forward and inverse Fourier transform is that the signal is transformed between the two domains exclusively. A method is often desired to transform the signal into a hybrid domain expressing both time *and* frequency, allowing for both a frequency-domain formulation of the problem at hand, and a time-domain implementation of the solution.

2.7.1 The Short-Time Fourier Transform

A hybrid domain is the time-frequency domain which allows frequency domain signal processing to be performed, but retains a temporal dimension to allow continuous processing over time. This can be defined through the short-time Fourier transform:

$$X(\tau,\omega) = \int_{-\infty}^{\infty} x(t)\, w(t-\tau)\, \mathrm{e}^{-\mathrm{j}\omega t}\, \mathrm{d}t \tag{2.36}$$

where $w(t)$ is a window with finite support. This effectively transforms a time-limited and windowed frame of the signal, where τ defines the location of the frame in time, while the support of the window is the duration of the frame in time. The time-frequency domain signal $X(\tau, \omega)$ is generally complex valued and represents the amplitude and the phase at the time τ and at the frequency ω.

One way to look at the short-time Fourier transform is in terms of filtering and modulation. The short-time Fourier transform can be rewritten as

$$X(\tau, \omega) = \mathrm{e}^{-\mathrm{j}\omega\tau} \int_{-\infty}^{\infty} x(t)\, w(t-\tau)\, \mathrm{e}^{-\mathrm{j}\omega(t-\tau)}\, \mathrm{d}t. \tag{2.37}$$

Here, the output signal $X(\tau, \omega)$ can be seen as the result of filtering the signal $x(t)$ with the filter $w(-t)\mathrm{e}^{\mathrm{j}\omega t}$. This filter can be further decomposed into a prototype filter $w(-t)$ and a modulation $\mathrm{e}^{\mathrm{j}\omega t}$. The frequency response of the window typically has a low-pass characteristic with a very low cut-off frequency. Treating the window as a low-pass filter, the modulation then modulates the low-pass filter into a band-pass filter which pass-band is centered at ω. Following the band-pass filtering is a demodulation of the filtered signal to center it at the zero-frequency. The signal $X(\tau, \omega)$ can, therefore, be seen as a time-domain signal over τ that has been filtered by a narrow-band band-pass filter which center frequency is ω and which has subsequently been demodulated to the zero-frequency.

2.7.2 The Filterbank

The general structure of the analysis part of a discrete-time multi-rate filterbank structure shown in figure 2.5 [8] employs a set of narrow-band analysis filters $h_k[n]$ to limit the bandwidth of the input signal to a narrow frequency band, followed by factor D decimators to reduce the sample rate of the bandwidth-limited subband signals. The synthesis part, structured accordingly, employs a set of factor D interpolators to increase the sample rate, followed by the synthesis filters $g_k[n]$ to create a full-rate signal from the low-rate subband signals.

The decimated subband signals can be expressed in the z-domain as

$$X_k[z] = \frac{1}{D} \sum_{d=0}^{D-1} H_k[z^{1/D}\, \mathrm{e}^{-\mathrm{j}2\pi d/D}] X[z^{1/D}\, \mathrm{e}^{-\mathrm{j}2\pi d/D}] \tag{2.38}$$

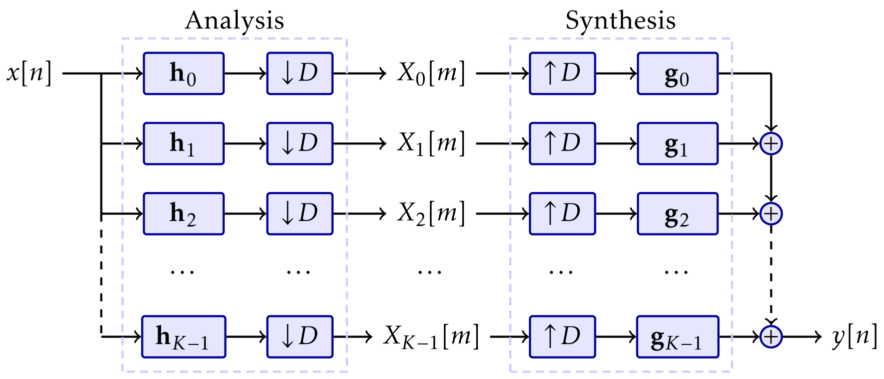


Figure 2.5: A multi-rate filterbank with K subbands which sample rate is decimated D times.

where the D terms in the summation represent the D aliasing terms arising in the expansion of the spectrum due to the decimation. The term at $d = 0$ is the desired term, and the terms at $d \in \{1, \ldots, D-1\}$ are the undesired aliasing terms. The full-band output signal is then, after processing the subband signals, reconstructed as

$$Y[z] = \sum_{k=0}^{K-1} X_k[z^D] G_k[z]. \tag{2.39}$$

2.7.3 The Modulated Uniform DFT Filterbank

A direct implementation of the filterbank as presented in figure 2.5 is computationally demanding as the K input signals and expanded subband signals are all filtered by the analysis and the synthesis filters at full sample rate. A direct implementation is also computationally wasteful as all but every Dth sample is effectively discarded in the decimation process. The filterbank can be reformulated, with some additional constraints on the design of the analysis and synthesis filters, the number of subbands and the decimation factor, to achieve a computationally efficient fast-Fourier transform-based polyphase implementation [8, 9, 10, 11].

Polyphase Decomposition of the Analysis Filter

A filterbank is said to be a DFT filterbank if the kth subband analysis filters $h_k[n]$ are related to the prototype filter $h_0[n]$ as

$$h_k[n] = h_0[n]\,\mathrm{e}^{-\mathrm{j}2\pi nk/K} \tag{2.40}$$

$$= h_0[n]\,\mathrm{e}^{-\mathrm{j}\omega_k n} \tag{2.41}$$

where $\omega_k = 2\pi k/K$ is the normalized angular frequency of the center of the kth subband. In the z-domain, the prototype filter becomes

$$H_k[z] = H_0[z\,\mathrm{e}^{\mathrm{j}2\pi k/K}]. \tag{2.42}$$

Assuming that the prototype filter is of the length $L = P \cdot D$, it can be written as a sum of its polyphase components in the following way:

$$H_0[z] = \sum_{d=0}^{D-1} z^{-d}\, E_d[z^D] \tag{2.43}$$

where

$$E_d[z] = \sum_{p=0}^{P-1} z^{-p}\, h_0[pD + d] \tag{2.44}$$

is the polyphase component d of the prototype filter. The modulated subband filters can, consequently, also be written as a sum of the polyphase components of the prototype filter:

$$H_k[z] = H_0[z\,\mathrm{e}^{\mathrm{j}2\pi k/K}] \tag{2.45}$$

$$= \sum_{d=0}^{D-1} z^{-d}\,\mathrm{e}^{-\mathrm{j}2\pi dk/K} E_d[z^D\,\mathrm{e}^{\mathrm{j}2\pi kD/K}] \tag{2.46}$$

$$= \sum_{d=0}^{D-1} z^{-d}\,\mathrm{e}^{-\mathrm{j}2\pi dk/K} \sum_{p=0}^{P-1} z^{-pD}\,\mathrm{e}^{-\mathrm{j}2\pi pkD/K}\, h_0[pD + d]. \tag{2.47}$$

The modulated subband filters are divided into $O = K/D$ subband groups, where each subband group consists of D subbands. The modulated subband filter for the subband $k = iO + o$, where $o \in \{0,\dots,O-1\}$ is the subband group and $i \in \{0,\dots,D-1\}$ is the subband within the subband group, then becomes

$$H_{iO+o}[z] = \sum_{d=0}^{D-1} z^{-d}\,\mathrm{e}^{-\mathrm{j}2\pi d(iO+o)/K} \sum_{p=0}^{P-1} z^{-pD}\,\mathrm{e}^{-\mathrm{j}2\pi p(iO+o)D/K}\, h_0[pD+d] \tag{2.48}$$

$$= \sum_{d=0}^{D-1} z^{-d}\,\mathrm{e}^{-\mathrm{j}2\pi di/D} \sum_{p=0}^{P-1} z^{-pD}\,\mathrm{e}^{-\mathrm{j}2\pi o(pD+d)/K}\, h_0[pD+d] \tag{2.49}$$

$$= \sum_{d=0}^{D-1} z^{-d}\,\mathrm{e}^{-\mathrm{j}2\pi di/D} E_{d,o}[z^D] \tag{2.50}$$

where

$$E_{d,o}[z] = \sum_{p=0}^{P-1} z^{-p}\,\mathrm{e}^{-\mathrm{j}2\pi o(pD+d)/K}\, h_0[pD+d]. \tag{2.51}$$

is the polyphase component d for the subband group o of the low-pass prototype filter. An implementation of the analysis filterbank structure in (2.50) is shown in figure 2.6.

To see the implementation in figure 2.6, equation (2.50) can be expanded into its three factors: the delay line z^{-d}, the modulation $\mathrm{e}^{-\mathrm{j}2\pi di/D}$, and the filter $E_{d,o}[z^D]$. The modulation term is one row of a $D \times D$ DFT matrix, and the sum of all D rows corresponds to the complete DFT transformation. The sum of the differently modulated terms thus corresponds to the forward DFT operation, which can be implemented with an efficient forward FFT algorithm. Each subband group o replicates this implementation with the same delay line and modulation, but with different subband filters. Finally, according to the Noble identities [8], the decimation can be performed before the filtering due to the polyphase implementation of the filters.

Polyphase Decomposition of the Synthesis Filter

The polyphase components of the synthesis filters are the time-reverse complex conjugate of the corresponding polyphase components of the analysis filters:

$$e'_{d,o}[n] = e^{\star}_{d,o}[P-1-n] \tag{2.52}$$

or in the z-domain:

$$E'_{d,o}[z] = z^{P-1}\, E^{\star}_{d,o}[z^{-1}]. \tag{2.53}$$

The synthesis filters can be reformulated similar to the analysis filters by dividing the filters into $O = K/D$ subband groups and D subbands per group so that

$$H_{iO+o}[z] = \sum_{d=0}^{D-1} z^{-D+1+d}\, \mathrm{e}^{\mathrm{j}2\pi di/D}\, E'_{d,o}[z^{D}]. \tag{2.54}$$

An implementation of the synthesis filterbank structure in (2.54) is shown in figure 2.7.

2.8 W-Disjoint Orthogonality

A common assumption among many time-frequency masking-based signal separation methods, such as the methods presented in this thesis, is that the time-frequency representations of the source signals to be separated are sparse and that their supports are disjoint in the time-frequency domain. Denoted W-disjoint orthogonality, the time-frequency masking-based signal separation methods exploit the sparse and disjoint structure of speech signals in the time-frequency domain for blind signal separation [12]. The W-disjoint orthogonal property of two signals $s_p(t)$ and $s_q(t)$, or the corresponding time-frequency signals $S_p(\tau,\omega)$ and $S_q(\tau,\omega)$, can be formulated as

$$S_p(\tau,\omega)\, S_q(\tau,\omega) = 0. \tag{2.55}$$

The assumption is that there is no energy overlap in the time-frequency domain between the two signals.

From the W-disjoint orthogonality it follows that, in the discrete time-frequency domain, there exists a set of time-frequency masks that can separate an additive mixture of orthogonal signals into its original components. Under the assumption that the sources are indeed W-disjoint orthogonal, no more than one source contributes with energy at a specific time-frequency point $[n,k]$, and a binary mask can therefore extract the time-frequency point from a specific source. Independent speech sources have been shown to be nearly W-disjoint orthogonal, and are therefore suitable for the time-frequency masking methods used to separate multiple speech sources [12].

A method to quantify the W-disjoint orthogonality is the preserved-signal ratio (PSR) and the signal-to-interference ratio (SIR) [12]. The preserved-signal ratio represents how well the signal S_p is restored from the additive mixture $S_p + S_q$ by time-frequency masking, and the signal-to-interference ratio represents how well the signal S_q is suppressed from the same mixture. The W-orthogonality is the combined preservation and suppression ratios such that orthogonal signals can be well preserved and characterized by little separation cross-talk.

The preserved-signal ratio is calculated as

$$\mathrm{PSR}(\theta) = \frac{\left\|\mathbb{M}_{p,q,\theta} \circ S_p\right\|^2}{\left\|S_p\right\|^2} \tag{2.56}$$

and the signal-to-interference ratio as

$$\mathrm{SIR}(\theta) = \frac{\left\|\mathbb{M}_{p,q,\theta} \circ S_p\right\|^2}{\left\|\mathbb{M}_{p,q,\theta} \circ S_q\right\|^2}. \tag{2.57}$$

The binary time-frequency mask is

$$\mathbb{M}_{p,q,\theta}[n,k] = \begin{cases} 1 & 20 \log_{10} \frac{|S_p[n,k]|}{|S_q[n,k]|} > \theta \\ 0 & \text{otherwise} \end{cases} \tag{2.58}$$

and is designed such that the time-frequency points in signal S_p, which dominates S_q by at least θ dB in magnitude, are passed through. The mask threshold θ determines by how much the source p must dominate the source q to mask it.

The term "dominate" may be a bit misleading in this context, because it is not strictly necessary for the mask threshold θ to be greater than 0 dB, which would be the natural requirement for one source to dominate another. A threshold less than 0 dB will mask a greater part of the desired signal and increase the preserved-signal ratio, but at the same time the lowered threshold will also decrease the signal-to-interference ratio as a greater part of the interference signal is masked. There are three cases for the threshold:

$\theta = 0\,\mathrm{dB}$ The masking is defined by which signal's amplitude is the largest in any time-frequency point. Ignoring the special case

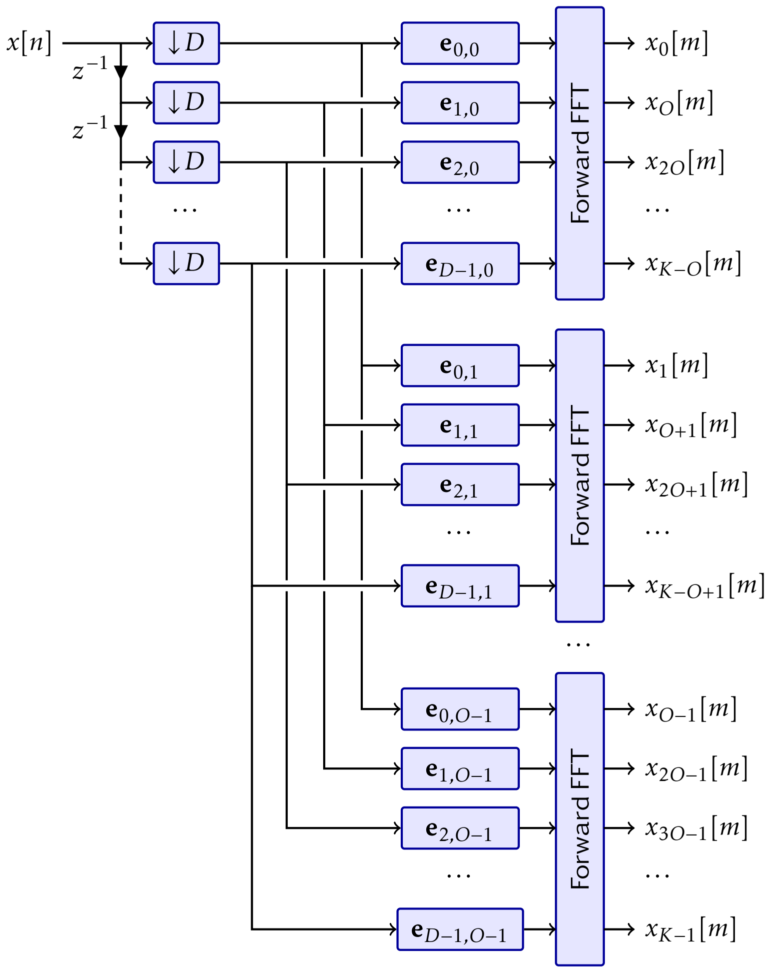


Figure 2.6: Analysis filterbank.

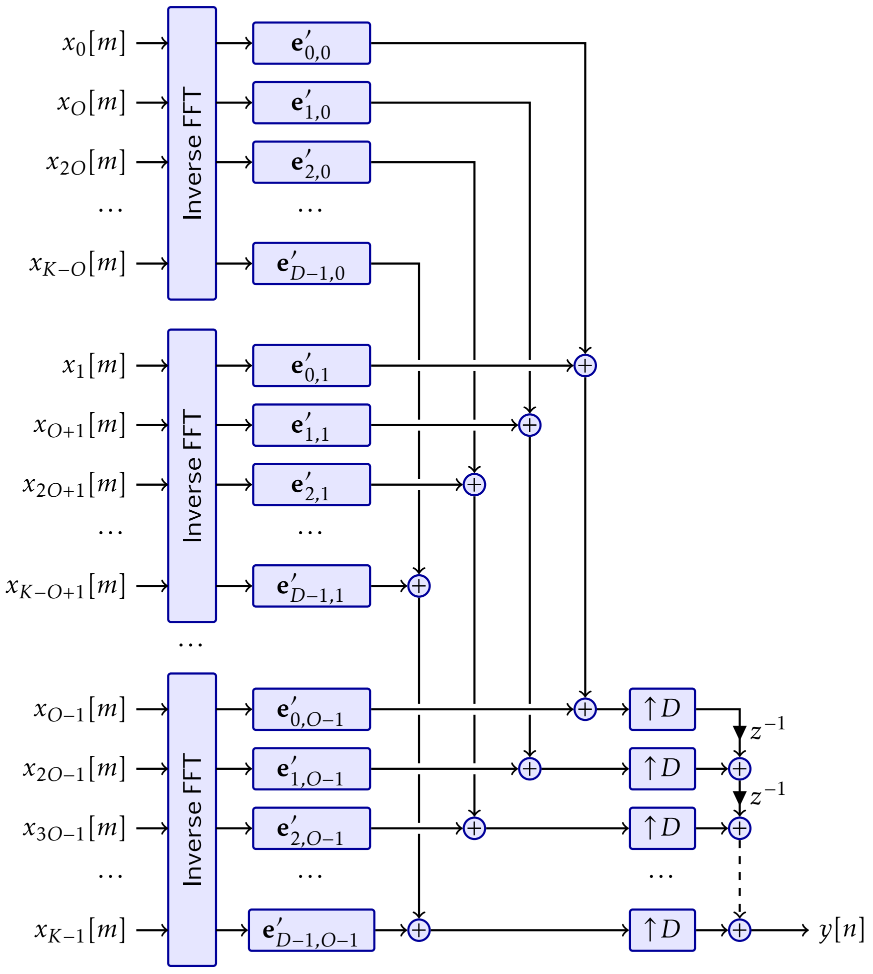


Figure 2.7: Synthesis filterbank.

where energies are exactly identical in some time-frequency points, the two masks $\mathbb{M}_{p,q,0}$ and $\mathbb{M}_{q,p,0}$ are each others complement.

$\theta > 0\,\text{dB}$ This is a stronger definition of "domination" than in the case of $\theta = 0\,\text{dB}$. The energy of one source must not only be greater than the other, but must be greater by a factor of θ dB. There are time-frequency points where neither $\mathbb{M}_{p,q,\theta}$ nor $\mathbb{M}_{q,p,\theta}$ mask their source, and the set of such points can be considered where neither source is dominant or, equivalently, have similar energy.

$\theta < 0\,\text{dB}$ This is a weaker definition of "domination" than in the case of $\theta = 0\,\text{dB}$. There are time-frequency points where both $\mathbb{M}_{p,q,\theta}$ and $\mathbb{M}_{q,p,\theta}$ mask their source, and the set of such points can be considered where neither source is dominant or, equivalently, have similar energy.

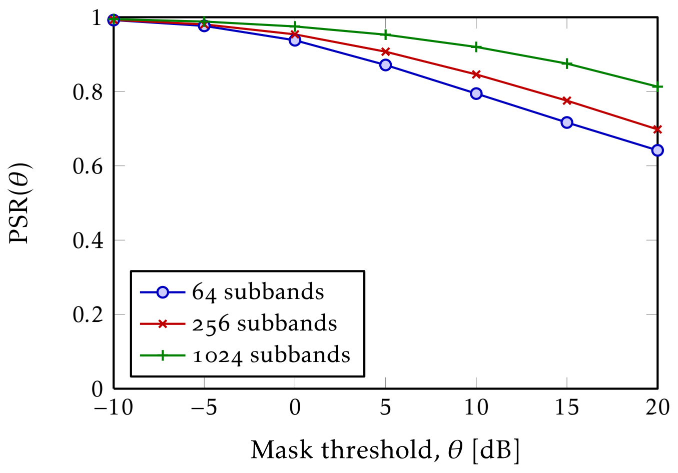
The difference between the stronger and weaker definition of "domination" is whether the set of time-frequency points with equivalent energy masks its source or not. If the source is not masked, as in the stronger definition, the preserved-signal ratio decreases while the signal-to-interference ratio increases. On the other hand, if the source is masked, as in the weaker definition, the preserved-signal ratio increases while the signal-to-interference ratio decreases.

Figure 2.8 shows the average PSR and SIR for speech signal pairs from the TIMIT database [13]. Time-frequency transformation is performed using a uniform DFT filterbank, with 64, 256 and 1024 subbands. At a mask threshold of 0 dB, the PSR suggests that around 95 % of a signal can be restored from a mixture of two speech sources using a binary time-frequency mask. Even when the threshold is raised to 20 dB, as much as 60 % to 80 % of the energy can still be restored from the additive mixture since the high-energy formants of the speech dominate the time-frequency points. Because the important formants of different speech signals rarely overlap, the formants of different sources can be masked and separated.

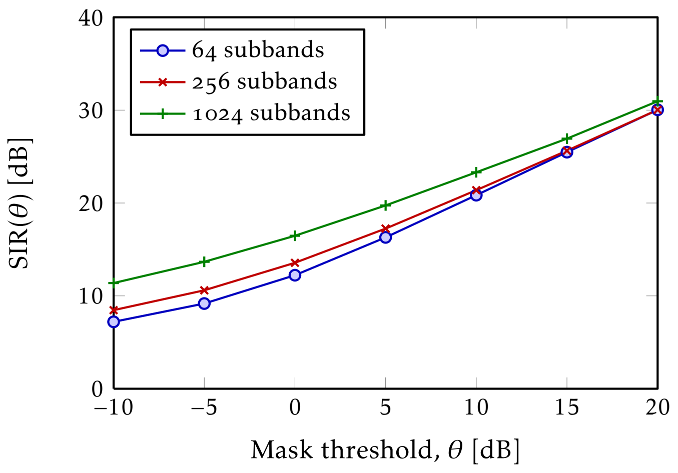
Figure 2.9 shows the average PSR and SIR, evaluated at a threshold of 0 dB, and where the interfering signal is the sum of multiple independent speech signals. Although modeled as a single interfering speech source for the PSR and SIR calculations, the interfering source is

$$s_q(t) = \sum_i s_{q_i}(t) \tag{2.59}$$

where $s_{q_i}(t)$ is the ith independent interfering speech source. The figure shows a decreasing PSR and SIR as the number of interfering sources increases. Similar to figure 2.8, the PSR and SIR increase with the number of subbands of the filterbank.

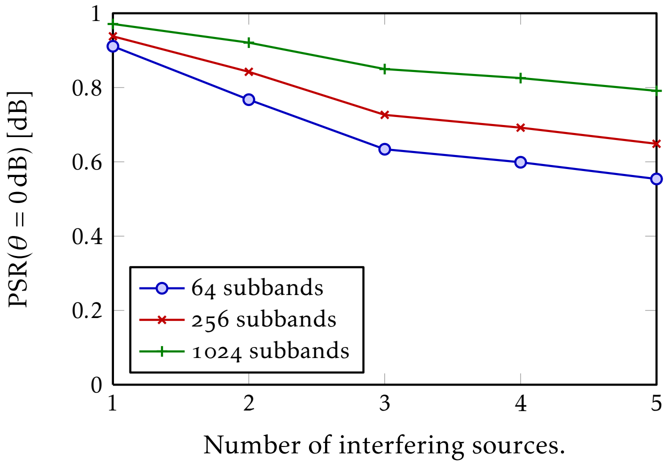


(a) PSR

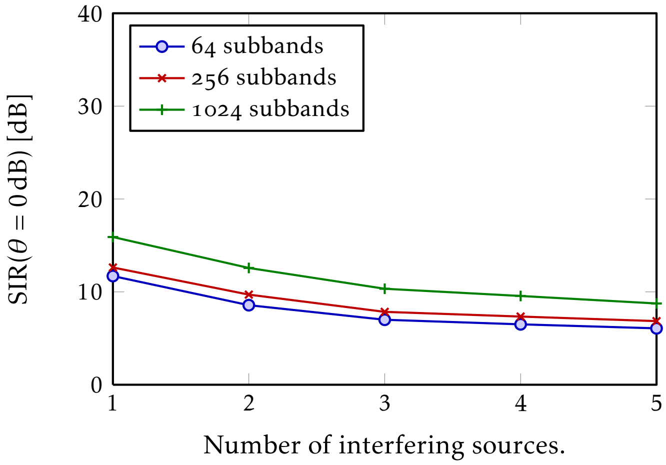


(b) SIR

Figure 2.8: Orthogonality for varying a mask threshold: (a) average PSR (preserved-signal ratio), and (b) SIR (signal-to-interference ratio) of all the speech signal pairs used during evaluation.



(a) PSR



(b) SIR

Figure 2.9: Orthogonality for interfering sources: (a) average PSR (preserved-signal ratio), and (b) SIR (signal-to-interference ratio) of speech signals used during evaluation.

Chapter 3

Source Localization

The term *source localization* is ambiguous in some contexts. It refers to the localization of an acoustic source in a general sense, but it may not be entirely clear in what space the source is to be found in each case. This thesis covers two different applications of source localization. The first application is to find the location of a source from a sensor array's point of view. In a far-field model, where only directions can be estimated, source localization is about finding the direction towards the source. This is equivalent to estimating the parameters $\mathbf{v}$, τ or α from the far-field sensor array model in chapter 2.

The second application of source localization is to find the spatial location of a source in a physical environment. This is equivalent to estimating $\mathbf{s}$ from the near-field sensor array model in chapter 2. The mapping from the source location $\mathbf{s}$ to the corresponding direction-of-arrival vector $\mathbf{v}$ is unique in that a source position corresponds to a single direction-of-arrival parameter. The reverse is not true though, which was shown with the hyperboloid model where a time-difference-of-arrival parameter corresponded to all source locations on the hyperboloid surface. This means that other steps have to be taken to disambiguate the locations from the hyperboloid or the cone model.

The two applications will herein be denoted as *direction-of-arrival estimation* and *source location estimation*, respectively, while *source localization* is the general concept of locating a source in some context.

3.1 Source Localization Methods

Source localization methods are commonly divided into three different approaches: high-resolution spectral-based estimators, time-difference of arrival-based estimators and beamformer-based estimators [14, 15]. The concepts will be covered briefly and the beamformer-based estimators will be discussed in more detail later as they constitute an important base for this thesis.

3.1.1 High-Resolution Spatial Spectral-Based Estimators

The high-resolution spatial spectral estimation-based methods can roughly be described as methods based on the spatial correlation matrix of the received sensor signals. These methods for direction-of-arrival estimation are closely related to temporal spectrum estimation but employed in the spatial domain instead of in the temporal domain, which is why they are referred to as spatial spectral estimators. An overview of spatial spectral-based estimators can be found in [16].

An example of a high-resolution spatial spectral estimator for source localization is the MUSIC estimator [17, 18, 19]. The MUSIC estimator is based on decomposing the spatial correlation matrix into the eigenvectors that span the space of a set of narrow band signals and the remaining noise space. In its original form, it was a method for far-field direction-of-arrival estimation using a uniform linear sensor array, but it has subsequently been extended for near-field arbitrary sensor array source location estimation [20]. Even though it focuses on narrow-band sources, a wide-band source like speech can be decomposed into narrow-band signals by a time-frequency decomposition, such as the subband signals from a filterbank.

One limitation with MUSIC is that the number of sources has to be less than the number of sensors. Using a principle similar to the one presented in chapters 4 and 5, where a specific source signal structure is assumed to relax the restriction on the number of sources, a MUSIC-based multi-source direction-of-arrival estimation method has been proposed in [21].

Although spatial spectral-based estimators can be used for speech source localization, they are typically sensitive to sensor placement errors and coherent signals from reverberation paths [15, 22]. For these reasons, the spatial spectral-based estimators have not been researched any further within the scope of this thesis.

3.1.2 Time-Difference of Arrival-Based Estimators

The time-difference of arrival-based estimators are methods utilizing the estimated time-difference between various sensor pairs. The actual way in which the time differences are estimated is a transparent process and will be covered in the following sections. The time-difference of arrival-based location estimator presented here is an example of how such an estimator can work.

The signal model in (2.11) presented the time delay between the two sensors p and q:

$$T(\mathbf{p}_p, \mathbf{p}_q, \mathbf{s}) = \frac{\left\|\mathbf{p}_q - \mathbf{s}\right\| - \left\|\mathbf{p}_p - \mathbf{s}\right\|}{c}. \tag{3.1}$$

As has been shown, a single sensor pair cannot be used for location estimation. However, by employing several sensor pairs, and the corresponding time delays, the location can be estimated. Since each sensor pair constrains the set of possible source locations in space, multiple sensor pairs add additional constraints to further reduce the set of possible locations. A simple optimization criterion is the least squares time-difference-of-arrival estimator (TDOA-LS) which optimizes the cost function

$$J_{\text{TDOA-LS}}(\mathbf{s}) = \sum_{\{p,q\}} \left|\tau_{p,q} - T(\mathbf{p}_p, \mathbf{p}_q, \mathbf{s})\right|^2 \tag{3.2}$$

and the source location is determined by

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} J_{\text{TDOA-LS}}(\mathbf{s}). \tag{3.3}$$

The least squares cost function optimization searches for a source location $\hat{\mathbf{s}}$ that minimizes the difference between the theoretical time delays for a source location and the measured time delays. Given enough sensor pairs, the source location ambiguity in the hyperboloid model can be reduced to a unique solution.

3.1.3 Beamformer-Based Estimators

A basic type of beamformer is the delay-and-sum beamformer:

$$y(t) = \sum_{j=0}^{J-1} x_j(t + T(\mathbf{p}_j, \mathbf{s})). \tag{3.4}$$

The name is derived from the operations it performs: delay the sensor signals and sum them to produce the beamformer output. The received sensor signals are aligned in time with respect to the propagation delay $T(\mathbf{p}_j, \mathbf{s})$ between the source location $\mathbf{s}$ and the sensor location $\mathbf{p}$, and summed coherently to produce the beamformer output signal.

The steering delays $T(\cdot)$ compensate for the propagation delays from the point $\mathbf{s}$ to the sensor array and the sensor signals are then aligned in time and added constructively during the summation stage. Signals from other locations will not be aligned in time and will be added destructively in the summation stage provided that no spatial aliasing occurs. This effectively attenuates signals coming from outside the focus point of the beamformer.

The power of the output signal of a beamformer that is steered towards a location **s** can be used as a cost function:

$$J_{\mathrm{SRP}}(\mathbf{s}) = \int \left|y(t)\right|^2 \mathrm{d}t. \tag{3.5}$$

The maximum beamformer output power is achieved when the beamformer is steered towards the point where the source is located:

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}} J_{\mathrm{SRP}}(\mathbf{s}). \tag{3.6}$$

At this location, similar to the TDOA-LS method, the beamformer steering vectors match the theoretical steering vectors for the true source location, and the constructive interference in the delay-and-sum beamformer is maximized. This approach is known as the steered response power (SRP) and it is the primary method focused on in this thesis.

3.1.4 Near-Field Location Estimation Variance

The examples given for the time-difference of arrival and beamformer-based estimators estimate the spatial location of the source. An important issue to focus on is the location estimation variance. Both theoretical derivations and practical experiments on the estimation variance have been performed [1, 23]. Looking at the location from a polar point of view, the source location can be described, relative a reference point, in terms of a bearing angle and a radial distance. Similarly, the variance of the location estimate can also be described in terms of variance in the angular and radial directions.

The variance of the bearing angle of the source location has been shown to be proportional to the variance of the time-difference-of-arrival estimates and inverse proportional to the effective sensor spacing. This means that the variance of the bearing angle is effectively independent of the radial distance to the source.

The variance of the radial direction, however, has been shown to be proportional to the variance of the true radial distance to the *fourth* power, besides being proportional to the variance of the bearing angle. The important consequence of this is that the ability to estimate the radial distance deteriorates very rapidly as the true radial distance from the sensor array increases. As the source is located further and further away from the sensor array, only the bearing angle can be reliably estimated.

A conclusion from these observations is that location estimation is only possible in the near field, and not in the far field. The far field is when the source is considered to be located so far away that range estimation is no longer possible in practice. However, there are no definite rules where this limit is. The focus in this thesis is on small sensor arrays, so the far-field model is assumed to hold.

3.2 Time-Difference of Arrival Estimation

One remaining point to describe within the topic of source localization is how to actually estimate the time-difference of arrival between some sensor signals. This section will show the process by which the time delays can be estimated from the sensor signals. A steered beamformer-based estimator will be explained and simplified to the specific case with a uniform linear sensor array. The steered beamformer is based on the correlation-based group of time-difference-of-arrival estimations [24, 25, 26].

3.2.1 Correlation-Based Method

To estimate the time delay between two signals, it is necessary to examine the relative temporal structure of the two signals. One such method is the cross correlation between the two received sensor signals $x_p(t)$ and $x_q(t)$:

$$r_{p,q}(\tau) = \mathrm{E}\left[x_p(t) \cdot x_q(t-\tau)\right]. \tag{3.7}$$

The received sensor signals can be expressed in terms of the source signal instead. By separating the propagation path filters into an ideal direct path delay component τ and an additional reverberation path filter component g, the signal propagation path becomes $h(t) = \delta(t-\tau) + g(t)$, and the cross correlation becomes

$$\begin{aligned} r_{p,q}(\tau) &= \mathrm{E}\left[s(t) \star h_p \cdot s(t-\tau) \star \tilde{h}_q\right] && (3.8) \\ &= \mathrm{E}\left[s(t) \star \delta(t-\tau_p) \cdot s(t-\tau) \star \delta(t+\tau_q-\tau)\right] + g\text{-based terms.} && (3.9) \end{aligned}$$

The g-based terms is the result of the convolution of the source signal with the reverberation filters, excluding the direct path. Since the direct path is assumed to be the dominant term, as demonstrated in figure 2.4, the g-based terms are assumed to be negligible for the purpose of deriving the basis for the correlation-based time-difference-of-arrival estimation methods. Further expanding the cross correlation, it becomes

$$r_{p,q}(\tau) = \mathrm{E}\Big[s(t-\tau_p)\cdot s(t+\tau_q-\tau)\Big] + g\text{-based terms} \tag{3.10}$$

$$= r_{s,s}(\tau-\tau_q+\tau_p) + g\text{-based terms.} \tag{3.11}$$

The source signal is, therefore, subject to an ideal delay to each sensor corresponding to the direct path, and to an additional filtering describing the reverberation paths on top of the ideal direct path.

The cross correlation between the two received sensor signals $x_p(t)$ and $x_q(t)$ is equal to the autocorrelation of the source signal $s(t)$ shifted by the corresponding propagation delay difference between the source and the two sensors. The autocorrelation $r_{s,s}(\tau)$ of the source signal, or of any signal in general, attains its maximum at $\tau = 0$. The maximum of $r_{s,s}(\tau-\tau_q+\tau_p)$, and consequently of $r_{p,q}(\tau)$ if the direct path is dominant to all reverberation paths, is assumed to be located at $\tau = \tau_q - \tau_p$. As a consequence, the time-difference of arrival can be estimated by finding the maximum point of the cross correlation between the two received sensor signals. A correlation-based time-difference-of-arrival estimation method can be described as

$$\hat{\tau}_{p,q} = \arg\max_{\tau} r_{p,q}(\tau) \tag{3.12}$$

where $\hat{\tau}_{p,q}$ is the estimated time-difference of arrival between the sensor signals $x_p(t)$ and $x_q(t)$. A global maximum is assumed for the delay caused by the direct path, but spurious local maxima may also be caused by the reverberation paths.

The time-difference of arrival between the two signals is estimated by finding the time shift at which the cross correlation attains a maximum value. In a discrete-time sampled system, the resolution of the cross correlation, and consequently the resolution of the time-difference-of-arrival estimate, is limited to the time shifts at which the signals are sampled. Unless an interpolation scheme is employed to increase the resolution of the cross correlation, the resolution of the time-difference-of-arrival estimate is effectively limited by the sample rate of the signals.

An approach to handle the resolution issue is to decouple the delay parameter τ from the sensor signals. This can be achieved by frequency-domain transformation of the cross correlation via the Fourier transform:

$$\hat{\tau}_{p,q} = \arg\max_{\tau} \int_{-\infty}^{\infty} G_{p,q}(\omega)\, \mathrm{e}^{-\mathrm{j}\omega\tau}\, \mathrm{d}\omega \tag{3.13}$$

where

$$G_{p,q}(\omega) = \mathrm{E}\left[X_p(\omega)\, X_q^{\star}(\omega)\right] \tag{3.14}$$

is the cross-power spectrum of the sensor signals p and q, and where $X_p(\omega)$ and $X_q(\omega)$ are the frequency-domain representations of the corresponding time-domain signals. By transforming the optimization problem to the frequency-domain, the resolution of the time shift parameter is decoupled from the resolution of the signals. Optimization of the cross correlation can then be performed with a continuous variable that is not constrained in resolution even for discrete-time sampled signals.

3.2.2 The Generalized Cross Correlation

The generalized cross correlation extends the computation of the cross correlation by pre-filtering the received sensor signals with the general filters $g'_{p \text{ or } q}(t)$ [24]. The general cross correlation is instead calculated as

$$r_{p,q}(\tau) = \mathrm{E}\left[x_p(t) \star g'_p \cdot x_q(t-\tau) \star \tilde{g}'_q\right]. \tag{3.15}$$

In the frequency-domain, the generalized cross correlation becomes

$$r_{p,q}(\tau) = \int_{-\infty}^{\infty} \left[G'_p(\omega)\, X_p(\omega)\right] \left[G'^{\star}_q(\omega)\, X_q^{\star}(\omega)\right] \mathrm{e}^{-\mathrm{j}\omega\tau} \mathrm{d}\omega \tag{3.16}$$

$$= \int_{-\infty}^{\infty} G'_p(\omega)\, G'^{\star}_q(\omega)\, X_p(\omega)\, X_q^{\star}(\omega)\, \mathrm{e}^{-\mathrm{j}\omega\tau} \mathrm{d}\omega \tag{3.17}$$

$$= \int_{-\infty}^{\infty} \psi_{p,q}(\omega)\, X_p(\omega)\, X_q^{\star}(\omega)\, \mathrm{e}^{-\mathrm{j}\omega\tau} \mathrm{d}\omega \tag{3.18}$$

where

$$\psi_{p,q}(\omega) = G'_p(\omega)\, G'^{\star}_q(\omega) \tag{3.19}$$

and $G'(\omega)$ are the frequency-domain representation of the general filters. The function $\psi_{p,q}(\omega)$ is known as a general frequency weighting function or a processor.

Looking at the process from the source signal to the received and filtered signal, it can be seen that there are a number of cascaded filters being applied to the source signal before computing the generalized cross correlation: the ideal direct path propagation delay, the additional reverberation paths, and the general filters. Denoting the post-filtered signal as $y(t)$, or as $Y(\omega)$ in the frequency domain, the generalized cross correlation becomes

$$r_{p,q}(\tau) = \int_{-\infty}^{\infty} Y_p(\omega) Y^{\star}_q(\omega)\, \mathrm{e}^{-\mathrm{j}\omega\tau} \mathrm{d}\omega \tag{3.20}$$

where

$$\begin{aligned} Y(\omega) &= G'(\omega)\, X(\omega) && (3.21) \\ &= G'(\omega)\, G(\omega)\, \mathrm{e}^{-\mathrm{j}\omega\tau}\, S(\omega). && (3.22) \end{aligned}$$

Thus, from right to left, the received and filtered sensor signals are expressed as the source signal $S(\omega)$ filtered by the direct propagation path component $\mathrm{e}^{-\mathrm{j}\omega\tau}$, the reverberation component $G(\omega)$, and the general filter $G'(\omega)$. If the general filters are chosen such that

$$G'(\omega) = \frac{1}{G(\omega)} \tag{3.23}$$

then the general filters will cancel the multi-path propagation effects, leaving only the ideal direct-path propagation. However, this requires complete knowledge of the propagation channel, which is typically unrealistic. There are a multitude of processors proposed, from simple unit-weighted cross correlations, to statistically optimal weighing functions, and to ad hoc experimental weighting functions [24].

3.2.3 The Phase Transform

The phase transform processor will the focus on here and it can be defined as

$$\psi_{p,q}(\omega) = \frac{1}{\left|X_p(\omega)X_q^\star(\omega)\right|}. \tag{3.24}$$

This processor was primarily meant as an ad hoc method to whiten the received sensor signals.

At first it may appear that the phase transform processor accentuates the contribution to the cross correlation for frequencies with low energy, and suppresses the contribution for frequencies with high energy. This may appear illogical, and it may seem that the weighting should have been the other way around. However, there are reasonable explanations and motivations for why this works. If the source is assumed to be white, then the recorded sensor signal is actually an estimate of the propagation path:

$$S(\omega) = 1 \rightarrow X(\omega) = G(\omega) \tag{3.25}$$

subject only to a phase shift caused by the ideal direct path. Thus, by choosing the general filters as

$$G'(\omega) = \frac{1}{X(\omega)} \tag{3.26}$$

the multi-path propagation effects can be eliminated. If the sensors are closely located, it can be assumed that

$$G_p(\omega) \approx G_q(\omega) \rightarrow X_p(\omega)X_q^\star(\omega) \approx \left|X_p(\omega)X_q^\star(\omega)\right|. \tag{3.27}$$

The phase transform uses the received sensor signal as an estimate of the propagation path to inverse-filter the received sensor signal.

The phase transform has gained popularity and is recognized as a fairly robust method useful in low-noise but reverberant environments. Even though it has been around for a long time, it was not until recently that it was shown why it actually performs so well [27]: as the variance of the noise $v(t)$ decreases, the phase transform processor approaches the maximum likelihood estimator.

3.3 The Steered Response Power

As demonstrated earlier in this chapter, the class of steered beamformer direction-of-arrival estimators is based on steering a beamformer across a search space to maximize the power of the beamformer output signal. The delay-and-sum beamformer is, as presented earlier, defined as

$$y(t) = \sum_{m=0}^{M-1} x_m(t + T(\mathbf{p}_m, \mathbf{s})) \tag{3.28}$$

where the function

$$T(\mathbf{p}, \mathbf{s}) = \frac{||\mathbf{s} - \mathbf{p}||}{c} \tag{3.29}$$

is the propagation delay between the source location $\mathbf{s}$ and the sensor location $\mathbf{p}$. The delays $T(\cdot)$, referred to as the steering vector or the steering delays, focus the beamformer towards the desired location $\mathbf{s}$.

The frequency-domain representation of the delay-and-sum beamformer is

$$Y(\omega) = \sum_{m=0}^{M-1} X_m(\omega)\, e^{j\omega T(\mathbf{p}_m, \mathbf{s})} \tag{3.30}$$

and the output power is

$$P(\mathbf{s}) = \int_{-\infty}^{\infty} Y(\omega)\, Y^{\star}(\omega)\, d\omega. \tag{3.31}$$

The location of the source is then determined by maximizing the output power of the beamformer:

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}} P(\mathbf{s}). \tag{3.32}$$

3.3.1 The Generalized Steered Beamformer

The class of correlation-based time-difference-of-arrival estimators and the generalization of them were presented in the previous section. The steered beamformer-based estimators can also be generalized in a similar manner, and, in fact, the generalized steered beamformer method can be seen as a further generalization of the generalized cross correlation. This can be seen by expanding the beamformer output power in terms of the received sensor signals and adding the pre-filters from the generalized cross correlation to the received sensor signals:

$$P(\mathbf{s}) = \sum_{q=0}^{M-1} \sum_{p=0}^{M-1} \int_{-\infty}^{\infty} \psi_{p,q}(\omega) X_p(\omega) X_q^{\star}(\omega) \mathrm{e}^{\mathrm{j}\omega\left(T(\mathbf{p}_q,\mathbf{s})-T(\mathbf{p}_p,\mathbf{s})\right)} \mathrm{d}\omega. \tag{3.33}$$

When comparing this with the generalized cross correlation-based method from the previous section, given that $\tau \leftrightarrow T(\mathbf{p}_q,\mathbf{s})-T(\mathbf{p}_p,\mathbf{s})$, it can be seen that the generalized steered beamformer response power is the sum of all pair-wise generalized cross correlations of the received sensor signals. The generalized steered beamformer allows for an arbitrary sensor geometry and an arbitrary search space to locate the source in. Two simplifications will be made based on two major assumptions: the far-field model is assumed over the near-field model, and the sensor array is assumed to be a uniform linear sensor array.

3.3.2 The Far-Field Approximation

As shown earlier, source location estimation is not possible in the far field. Similarly, it is not possible to focus the beamformer on a specific location in the far field. While it is possible to design the steering vectors, the beamformer will not be able to focus with any greater accuracy onto the desired point except in the angular direction. In the radial direction, the accuracy is virtually non-existent, which is why only direction estimation, and not location estimation, is possible in the far field. Therefore, the steering vectors should ideally be defined by $T(\mathbf{p},\mathbf{v})$, which steers the beamformer towards the direction $\mathbf{v}$ instead of a single point $\mathbf{s}$. This, however, poses a problem since determining the propagation delay becomes impossible since the far-field model effectively assumes that the source is located at a point with infinite radius.

Similar to the mixing parameter normalization in chapter 2, the steering vectors are normalized to a reference point. Assuming that the reference point is also chosen to be the origin of the coordinate system, the local steering vectors can be defined as

$$T(\mathbf{p},\mathbf{v}) = \frac{\mathbf{p}^{\mathrm{T}}\mathbf{v}}{\mathrm{c}}. \tag{3.34}$$

A steering vector is simply calculated by projecting the sensor position $\mathbf{p}$ onto the direction-of-arrival vector $\mathbf{v}$.

3.3.3 The Uniform Linear Sensor Array

Looking at the definition of the steered response power of the generalized steered beamformer in (3.33), it can be seen that the steering vectors of the cross correlations depend on the relative propagation delay differences $T(\mathbf{p}_q,\cdot)-T(\mathbf{p}_p,\cdot)$ between the two sensors p and q. Assuming that the sensor array is a uniform linear sensor array, the location of a sensor can be described as

$$\mathbf{p}_n = \mathbf{p}_0 + \mathbf{b}\cdot nd \tag{3.35}$$

where $\mathbf{b}$ is a unit-vector representing the axis of the sensor array baseline, $\mathbf{p}_0$ is the location of a reference sensor, and d is the distance between two adjacent sensors. The steering vectors for the generalized steered response power then become

$$\begin{aligned} T(\mathbf{p}_q,\mathbf{v})-T(\mathbf{p}_p,\mathbf{v}) &= \frac{\mathbf{p}_q^{\mathrm{T}}\mathbf{v}-\mathbf{p}_p^{\mathrm{T}}\mathbf{v}}{\mathrm{c}} && (3.36)\\ &= \frac{(\mathbf{p}_0+\mathbf{b}\cdot qd)^{\mathrm{T}}\mathbf{v}-(\mathbf{p}_0+\mathbf{b}\cdot pd)^{\mathrm{T}}\mathbf{v}}{\mathrm{c}} && (3.37)\\ &= \frac{\mathbf{b}^{\mathrm{T}}\mathbf{v}\cdot qd-\mathbf{b}^{\mathrm{T}}\mathbf{v}\cdot pd}{\mathrm{c}} && (3.38)\\ &= \frac{\mathbf{b}^{\mathrm{T}}\mathbf{v}\cdot d}{\mathrm{c}}\cdot(q-p) && (3.39)\\ &= \frac{d\,\sin(\alpha)}{\mathrm{c}}\cdot(q-p) && (3.40) \end{aligned}$$

where α is the cone angle corresponding to the direction-of-arrival vector $\mathbf{v}$ in the far-field model. An important observation is that the steering vectors for the uniform linear sensor array differ only by a linear scale factor $(q-p)$ of the sensor array-wide constant $d\,\sin(\alpha)/\mathrm{c}$.

3.3.4 Sensor Pair Selection

The strict definition of the steered response power is the sum of the generalized cross correlations for all possible combinations of sensor pairs. For a sensor array with N sensors, the set of pairs $\mathbf{P}$ is

$$\mathbf{P} = \{\{p,q\} : p \in \{0,\ldots,N-1\},\ q \in \{0,\ldots,N-1\}\}. \tag{3.41}$$

For localization and computational efficiency purposes, the set $\mathbf{P}$ has some undesired properties.

- The set contains pairs $\{p,q\}$ such that $p = q$. These pairs effectively attempt to perform the time-difference-of-arrival estimation between a sensor signal and itself. In terms of beamforming and the steered response power, a single sensor has no spatial resolution and those pairs contribute uniformly to the steered response power in any direction.

- The set contains symmetric pairs such that if $\{p,q\}$ is an element of the set, where $p \neq q$, then $\{q,p\}$ is also a pair of the set. These pairs can be used for time-difference-of-arrival estimation, but the symmetry pair does not add any additional information to the location of the maximum steered response power.

- The set contains the pairs $\{p,q\}$ where $p \ll q$ or $p \gg q$. These pairs are, from a numerical point of view, the pairs that provide the best spatial resolution and accuracy of the time-difference-of-arrival estimate. A larger distance between the sensors means that an estimation error in the time-difference-of-arrival estimation translates into a smaller error in the direction-of-arrival estimation. However, from a practical point of view, these pairs are problematic. As the distance increases, the coherence between the sensor signals decreases, introducing large errors in the time-difference-of-arrival estimation, and also lowering the maximum allowed frequency of the signal according to the spatial sampling theorem.

The first two points are mostly issues related to computational efficiency. The original set of pairs contains N^2 pairs. However, if eliminating the self pairs and the symmetric pairs from the set, the set only contains $\binom{N}{2}$ pairs. Such a set can be defined as

$$\mathbf{P} = \{\{p,q\} : p \in \{0,\ldots,N-2\},\ q \in \{p+1,\ldots,N-1\}\}. \tag{3.42}$$

The third point reduces the size of the set of pairs further by eliminating pairs which sensors are separated above some defined limit. By defining the set with only pairs of adjacent sensors, eliminating any higher-order neighbors, the set only contains $N-1$ pairs. Such a set can be defined as

$$\mathbf{P} = \{\{p,q\} : p \in \{0,\ldots,N-2\},\ q = p+1\}. \tag{3.43}$$

Even though only adjacent pairs are used here, sets with neighbors up to a specific order can be made. This particular set used here has the additional advantage that $q - p = 1$ for all elements $\{p, q\}$ of the set, so that the steering vectors in (3.33) as defined by (3.40) are the same for all generalized cross correlations. The computational complexity is thus reduced further by using a single steering vector for all sensor pairs.

The reduced size of the sets means that fewer generalized cross-power spectra are computed for the generalized steered response power. The three common sets presented have a total of N^2, $\binom{N}{2}$ and $N-1$ pairs, respectively. This reduction in computational complexity that can be accomplished by selecting the latter sets becomes significant even for sensor arrays with just a few sensors. As a comparison, for $N = 4$ sensors, the reduction in complexity is from $N^2 = 16$ cross-power spectrum estimations for a full set of pairs, to $\binom{N}{2} = 6$ and $N - 1 = 3$ cross-power spectrum estimations.

3.4 The GCC-PHAT and the SRP-PHAT

Given the models and the assumptions presented so far, the final estimator of the time-difference of arrival between the sensor signals p and q, based on the generalized cross correlation with the phase transform processor (GCC-PHAT), is expressed as

$$\hat{\tau}_{p,q} = \arg\max_{\tau} \int_{-\infty}^{\infty} \psi_{p,q}(\omega)\, X_p(\omega)\, X_q^{\star}(\omega)\, \mathrm{e}^{\mathrm{j}\omega\tau}\, \mathrm{d}\omega \tag{3.44}$$

where

$$\psi_{p,q}(\omega) = \frac{1}{\left|X_p(\omega) X_q^{\star}(\omega)\right|}. \tag{3.45}$$

Likewise, the final estimator, based on the steered response power with the phase transform processor (SRP-PHAT) for a uniform linear sensor array, is expressed as

$$\hat{\tau} = \arg\max_{\tau} \sum_{\{p,q\}} \int_{-\infty}^{\infty} \psi_{p,q}(\omega)\, X_p(\omega)\, X_q^{\star}(\omega)\, \mathrm{e}^{\mathrm{j}\omega\tau(q-p)}\, \mathrm{d}\omega. \tag{3.46}$$

With real frequencies, the unit of the delay is seconds. In the discrete time-frequency domain, the integration over frequencies is replaced with a summation over subbands instead, and the GCC-PHAT and the SRP-PHAT become

$$\hat{\tau}_{p,q} = \arg\max_{\tau} \sum_{k=0}^{K-1} \psi_{p,q}[k]\, X_p[k]\, X_q^{\star}[k]\, \mathrm{e}^{\mathrm{j}\omega_k \tau} \tag{3.47}$$

and

$$\hat{\tau} = \arg\max_{\tau} \sum_{\{p,q\}} \sum_{k=0}^{K-1} \psi_{p,q}[k]\, X_p[k]\, X_q^{\star}[k]\, \mathrm{e}^{\mathrm{j}\omega_k \tau (q-p)}. \tag{3.48}$$

The unit of the delay here is samples.

3.5 Location Estimation

Since location estimation has been shown to be impossible in the far field, and since the performance deteriorates rapidly with distance in the near-field model, other approaches have to be taken to spatially locate a source. One approach is to triangulate the source using multiple sensor arrays and intersect direction-of-arrival vector estimates from different sensor arrays. Using multiple sensor arrays and linear intersection of direction-of-arrival vector pairs does not change the fact that location estimation is only possible on the near-field. Each sensor array is still small and assumes a far-field source, so only the direction of arrival can be estimated at each sensor array. Multiple sensor arrays are then used to create an array of sensor arrays, satisfying the near-field model.

3.5.1 Linear Intersection

Given the reference location $\mathbf{p}$ of a sensor array and the corresponding estimated direction-of-arrival vector $\mathbf{v}$, the points of possible source locations $\mathbf{s}$ are defined by the bearing line

$$\mathbf{s} = t \cdot \mathbf{v} + \mathbf{p} \tag{3.49}$$

where $\mathbf{p}$ is the origin, the unit vector $\mathbf{v}$ is the direction and t is a scalar defining a point along the bearing line. Two spatially separated sensor arrays can be utilized to estimate the location of the source. If the bearing lines from the two sensor arrays are

$$\begin{cases} \mathbf{s}_p = t_p \cdot \mathbf{v}_p + \mathbf{p}_p \\ \mathbf{s}_q = t_q \cdot \mathbf{v}_q + \mathbf{p}_q \end{cases} \tag{3.50}$$

then the source location can be determined by finding a common point along the two bearing lines. If an intersection point is assumed to exist, the problem can be formulated as

$$\mathbf{s}_p = \mathbf{s}_q \tag{3.51}$$

$$t_p \cdot \mathbf{v}_p + \mathbf{p}_p = t_q \cdot \mathbf{v}_q + \mathbf{p}_q \tag{3.52}$$

$$t_p \cdot \mathbf{v}_p - t_q \cdot \mathbf{v}_q = \mathbf{p}_q - \mathbf{p}_p. \tag{3.53}$$

In matrix form, the equation system becomes

$$\mathbf{V}\,\mathbf{t} = \mathbf{p} \tag{3.54}$$

where

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_p & \mathbf{v}_q \end{bmatrix} \tag{3.55}$$

$$\mathbf{t} = \begin{bmatrix} t_p & -t_q \end{bmatrix}^{\mathrm{T}} \tag{3.56}$$

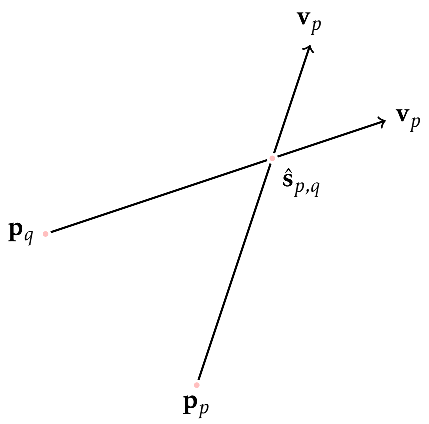
$$\mathbf{p} = \mathbf{p}_q - \mathbf{p}_p. \tag{3.57}$$

Solving for $\mathbf{t}$, the solution can be inserted into the parametric forms of the lines to obtain the intersection point between the two lines:

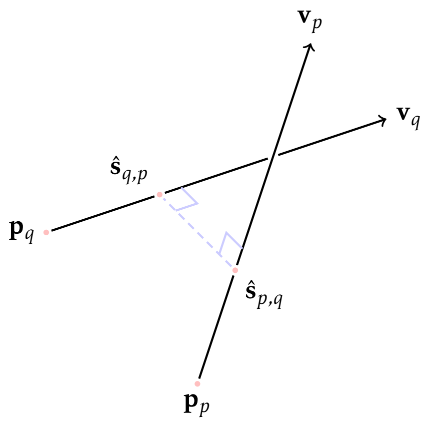
$$\hat{\mathbf{s}}_{p,q} = \begin{cases} t_p \cdot \mathbf{v}_p + \mathbf{p}_p \\ t_q \cdot \mathbf{v}_q + \mathbf{p}_q. \end{cases} \tag{3.58}$$

In two spatial dimensions, and assuming that the matrix $\mathbf{V}$ is full rank, a unique solution to (3.54) exists and corresponds to the intersection point between the two bearing lines. This is illustrated in figure 3.1(a). The final location estimate in this case is any of the two solutions, which are identical.

In three spatial dimensions, however, a single intersection point may not exist even if the matrix $\mathbf{V}$ is full rank as can be seen in figure 3.1(b). In this case, the two solutions in (3.58) provide the two points in figure 3.1(b). These two points are the points on each bearing line closest to the other bearing line. Assigning the point $\hat{\mathbf{s}}_{p,q}$ to be the point on the bearing line p closest to the bearing line q, the two solutions are uniquely identified by $\hat{\mathbf{s}}_{p,q}$ and $\hat{\mathbf{s}}_{q,p}$.



(a) Bearing line intersection with a single intersection point.



(b) Bearing line intersection with no single intersection point.

Figure 3.1: Bearing line intersection with and without a single intersection point.

3.5.2 Ill-Conditioned Intersection Matrix

The solution to (3.54) depends on a matrix inverse. If the matrix is ill-conditioned, the variance of the parameters are amplified when propagating to the solution, and the variance of the location estimate increases. An estimate of the intersection point location error is modeled as exemplified in figure 3.2 where the bearing line p is parallel displaced, and the error is modeled as the corresponding intersection point displacement along the bearing line q. Given a parallel-displacement δ_p of the bearing line p, the displacement of the intersection point along the bearing line q is

$$\epsilon_{p,q} = \frac{\delta_p}{\sin(\alpha_{p,q})} \tag{3.59}$$

where $\alpha_{p,q}$ is the angle of separation between the two bearing lines. The parallel displacement is an approximation near the intersection point of a small angle-of-arrival variation of the bearing line. The resulting intersection point displacement is the location estimation error resulting from the angle-of-arrival variation. Minimum error is achieved at 90° separation between bearing lines.

When the separation angle between the two bearing lines approaches 0° or 180°, the error approaches infinity. This happens when the two bearing lines are parallel, at which point the matrix $\mathbf{V}$ becomes rank deficient so that there is no longer an intersection point between the two bearing lines. The square of the sine of the angle of the separation between the two bearing lines can, therefore, be seen as a measure of the condition of the matrix.

The matrix $\mathbf{V}$ is ill-conditioned if there is a linear dependence between the two basis vectors of the matrix. There are two cases where this happens:

- The source is located in the far-field of the array of sensor arrays, and the bearing lines are parallel.
- The source is located between two sensor arrays, and the bearing lines are parallel, although in opposite directions.

In a practical application, care must be taken when positioning the sensor arrays to ensure that the direction-of-arrival vectors do not become parallel.

3.5.3 Multi-Line Linear Intersection

The approach to intersect the bearing lines works on a sensor array pair basis and has to be extended to handle three or more sensor arrays with corre-

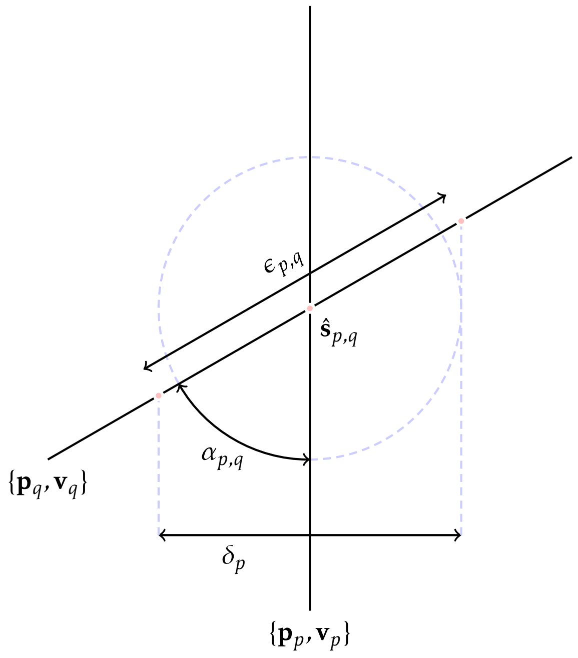

$\epsilon_{p,q}$
$\hat{\mathbf{s}}_{p,q}$
$\alpha_{p,q}$
$\{\mathbf{p}_q, \mathbf{v}_q\}$
δ_p
$\{\mathbf{p}_p, \mathbf{v}_p\}$

Figure 3.2: Approximation of the intersection point error by parallel-displacement of the bearing lines.

sponding bearing lines. Similar to how two non-parallel lines may not have a unique intersection point in three or more dimensions, three or more mutually non-parallel vectors may not intersect at a unique point even in two dimensions.

A closed form solution based on pair-wise linear intersections, denoted the linear intersection estimate, was presented by Brandstein et al. [28, 29]. The linear intersection estimate is a statistically weighted average of the intersection points from all sensor array pairs. It is assumed that the statistical distribution of the direction-of-arrival vector estimates is, fully or partly, known or estimated.

Another weighting approach presented here is based solely on the instantaneous observations themselves. Consider, as shown earlier, that $\hat{\mathbf{s}}_{p,q}$ is the point on the bearing line from the sensor array p that is closest to the bearing line from the sensor array q. In accordance with the model of the intersection point error, the variance of the location estimate increases as the direction-of-arrival vectors become less orthogonal, or, specifically, inverse proportional to the sine of the angle of separation between the two bearing lines. Based on that observation, the intersection point weighting approach consists of weighting the estimate based on the mutual projection of the corresponding direction-of-arrival estimates. Given the intersection points between all bearing line pairs, the final location estimate is

$$\hat{\mathbf{s}} = \frac{\sum_{\{p,q\}} W_{p,q}\, \hat{\mathbf{s}}_{p,q}}{\sum_{\{p,q\}} W_{p,q}} \tag{3.60}$$

where

$$W_{p,q} = 1 - \left|\mathbf{v}_p^{\mathrm{T}} \mathbf{v}_q\right|^2 . \tag{3.61}$$

The weight is, assuming that the direction-of-arrival vectors are unit vectors, the square of the sine of the angle of separation between the two bearing lines. Maximum weight contribution is achieved when the bearing lines are orthogonal, and minimum weight contribution occurs when the bearing lines are parallel.

Another approach to estimate the location of the source from an arbitrary number of sensor arrays and corresponding bearing lines is to minimize a point-to-line distance measure instead of calculating a weighted average of line-line intersection points. The least squares solution to the point closest to the bearing lines defined by $\mathbf{p}$ and $\mathbf{v}$ is

$$\hat{\mathbf{s}} = \left[\sum_n \left(\mathbf{I} - \mathbf{v}_n \mathbf{v}_n^{\mathrm{T}}\right)\right]^{-1} \left[\sum_n \left(\mathbf{I} - \mathbf{v}_n \mathbf{v}_n^{\mathrm{T}}\right) \mathbf{p}_n\right]. \tag{3.62}$$

The derivation for the least squares solution is presented in appendix A.

Chapter 4

Multi-Source DOA Estimation

the DUET-based Approach

This chapter is based on the following article:

Mikael Swartling, Nedelko Grbić, and Ingvar Claesson. Direction of Arrival Estimation for Multiple Speakers using Time-Frequency Orthogonal Signal Separation. In *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing*, volume 4, pages 833–836, May 2006.

This chapter presents and evaluates a method to estimate the directions of arrival of multiple concurrent speech sources. The method consists of two stages: a blind signal separation stage to separate the individual speech sources, and a single-source direction-of-arrival estimation stage to estimate the direction of arrival of each individual source. By preprocessing the received sensor signals using the blind signal separation method, the problem of direction-of-arrival estimation is reduced from finding multiple sources in one set of sensor signals, to finding single delays in multiple sets of sensor signals.

The goal of the blind signal separation stage is to separate a set of unknown signals, or sources, from a set of observed signals. The signals are the output from a sensor array, where the different sensors receive (sometimes marginally) different mixtures of the source signals. The term "blind" in this context means that the unknown source signals are not observed, and that no information is available about the mixing system [30]. To compensate for the lack of information about the unknown sources, their propagation to the sensor array and the mixing system, some assumptions must be made about the sources being separated. Such assumptions include that the sources are statistically independent, or that sources are W-disjoint orthogonal.

A recently developed method for blind signal separation is the Degenerate Unmixing and Estimation Technique, or DUET [31, 12]. The DUET method can separate more sources than there are received sensor signals, an ability referred to as degenerate demixing. Degenerate demixing is a challenging task in that the mixing matrix is not invertible, and traditional methods based on estimating the inverse of the mixing matrix do not work. After separating the original sources from the received mixtures, a steered beamformer-based method is used to estimate the direction of arrival for each source.

4.1 Blind Signal Separation

The signal separation method used in the method proposed in this chapter is based on the DUET method and the real-time DUET method [31, 32]. A modification of the real-time DUET method is introduced and employed to handle sensor arrays with multiple sensors.

4.1.1 Mixing Parameter Estimation

The original DUET method assumes an anechoic signal propagation model. Because it does not neglect the propagation attenuation factor, this method is slightly different from the anechoic signal propagation model presented in chapter 2. The signal propagation model assumed by the DUET method is

$$x_p(t) = \sum_{i=0}^{I-1} s_i(t) \tag{4.1}$$

$$x_q(t) = \sum_{i=0}^{I-1} s_i(t-\tau_i)\, a_i \tag{4.2}$$

where a_i and τ_i denote the propagation attenuation and the propagation delay for the source i at the sensor q normalized to the sensor p. The pair $\{a_i, \tau_i\}$ is referred to as the mixing parameters for the source i. In the frequency-domain, the signal propagation model can be expressed as

$$X_p(\tau,\omega) = \sum_{i=0}^{I-1} S_m(\tau,\omega) \tag{4.3}$$

$$X_q(\tau,\omega) = \sum_{i=0}^{I-1} S_m(\tau,\omega)\, a_i\, \mathrm{e}^{\mathrm{j}\omega\tau_i}. \tag{4.4}$$

The primary assumption is that the sources are W-disjoint orthogonal which implies that no more than one source can be active at any time-frequency point (τ, ω). The signal propagation model can be reformulated as

$$X_p(\tau,\omega) = S_i(\tau,\omega) \tag{4.5}$$

$$X_q(\tau,\omega) = S_i(\tau,\omega)\, a_i\, \mathrm{e}^{\mathrm{j}\omega\tau_i} \tag{4.6}$$

where i indicates the single active source at the corresponding time-frequency point (τ, ω). The mixing parameters can be observed from the received sensor signals X_p and X_q by recognizing that

$$a_i = \frac{\left|X_p(\tau,\omega)\right|}{\left|X_q(\tau,\omega)\right|} \tag{4.7}$$

and

$$\tau_i = -\frac{1}{\omega} \angle\left[X_p(\tau,\omega)\, X_q^{\star}(\tau,\omega)\right] \tag{4.8}$$

for the single active source i at the time-frequency point (τ, ω). The DUET method distinguishes the individual source signals by assuming that the observed mixing parameters form clusters near the true mixing parameters corresponding to the present sources. A clustering approach was taken to assign discrete time-frequency observations to the estimated clusters and, consequently, assign them to one of the sources.

The clustering-based approach taken by the DUET method is not suitable for real-time processing and can only be performed in an off-line environment, or in large batches. An alternative approach to estimating the mixing parameters resulted in the real-time DUET method [32]. Instead of the off-line-based clustering approach, the real-time DUET method defines a maximum likelihood cost function that is minimized online using a gradient-based search method in order to continuously find and track the mixing parameters. The mixing parameters are updated as

$$a_i[n] = a_i[n-1] - \mu_a \frac{\partial}{\partial a_i} J[n] \tag{4.9}$$

and

$$\tau_i[n] = \tau_i[n-1] - \mu_\tau \frac{\partial}{\partial \tau_i} J[n] \tag{4.10}$$

where $J[n]$ is the maximum likelihood cost function [32], and μ is the learning rate.

Some modifications for small arrays are made to the original method in order to improve the performance. Assuming that the relative attenuation mixing parameter is unity, there is no need to track the attenuation mixing parameter. Thus, the expression for the partial derivative for updating the delay mixing parameter is simplified. In this case, the attenuation mixing parameter is ignored, and the delay mixing parameter is updated as in (4.10), where

$$\frac{\partial}{\partial \tau_i} J_{p,q}[n] = \sum_{k=0}^{K-1} \frac{-\omega_k \, \mathrm{e}^{-\lambda \rho_i}}{\sum_{i=0}^{I-1} \mathrm{e}^{-\lambda \rho_i}} \operatorname{Im}\left[X_p[n,k] X_q^{\star}[n,k] \mathrm{e}^{\mathrm{j}\omega_k \tau_i} \right] \tag{4.11}$$

and where

$$\rho_i = \rho(\tau_i, n, k) = \frac{1}{2} \left| X_p[n,k] \mathrm{e}^{-\mathrm{j}\omega_k \tau_i} - X_q[n,k]) \right|^2 . \tag{4.12}$$

4.1.2 Demixing

The original method performs the demixing using binary masks. The mask is defined as

$$\mathbb{M}_m[n,k] = \begin{cases} 1 & \text{if } \rho_i \text{ is minimum for all } i \in \{0, \dots, I-1\} \\ 0 & \text{otherwise} \end{cases} \tag{4.13}$$

and the original source estimates in the time-frequency representation are

$$\hat{S}_i[n,k] = \mathbb{M}_i[n,k] X_{p \text{ or } q}[n,k]. \tag{4.14}$$

Any of the two sensor signals p or q can be used to demix the original sources. Due to symmetries in the mixing parameters when changing the order of the p and q indices that denote the two sensors, and in the close proximity of the sensors so that the signals are assumed to only differ by a relative phase shift, demixing from the different sensor signals also differs only in a relative phase shift. The estimates of the original source signals that are demixed from the different sensor signals are otherwise identical.

4.2 Delay Estimation

The DUET and real-time DUET methods aim to reconstruct the original source signals by transforming the time-frequency representation into time domain signals. In this chapter, the aim is instead to estimate the direction of arrival of the separated source signals. The proposed method makes two modifications to the approach taken by the DUET and real-time DUET methods:

- The first modification builds on the assumption that to reconstruct the source signals, it is only necessary to demix a single sensor signal. The direction-of-arrival estimation methods require two sensor signals to estimate the time-difference of arrival, and, consequently, the direction of arrival. Therefore, all source signals have to be demixed from *all* sensor signals, not just from a single sensor signal.
- The second modification assumes that for increased direction-of-arrival estimation accuracy, it is desired to use a multi-sensor array. This requires a modification to the DUET and the real-time DUET methods to account for an arbitrary set of sensor pairs.

The first point is not strictly a modification to either the DUET or to the real-time DUET method. This method simply requires demixing from all sensor signals instead of a single sensor signal so that

$$\hat{S}_{i,j}[n,k] = \mathbb{M}_i[n,k]\,X_j[n,k] \tag{4.15}$$

not only for all i and any j, but for all i and all j. The second point is handled by a simple extension to the real-time DUET mixing parameter update equation, where the mixing parameter is updated according to the sum of the corresponding gradients for all sensor pairs:

$$\tau_i[n] = \tau_i[n-1] - \mu \sum_{\{p,q\}} \frac{\partial}{\partial \tau_i} J_{p,q}[n]\,. \tag{4.16}$$

Once the source signals have been demixed from the sensor signals, the multi-source direction-of-arrival estimation has been reduced to multiple single-source direction-of-arrival estimations. Cross-power spectra for all sensor pairs $\{p,q\}$ for a source i is estimated by

$$G_{i,p,q}[k] = \hat{\mathrm{E}}\left[\hat{S}_{i,p}[k]\,\hat{S}_{i,q}[k]\right]. \tag{4.17}$$

The time-difference of arrival for the source i is then estimated by the generalized steered response power method using the cross-power spectra $G_{i,p,q}[k]$ for all $\{p,q\}$.

4.3 Evaluation

The proposed method to estimate the direction of arrival for multiple concurrent speech sources is evaluated using real room recordings. The room is a conference room of the size 4 m × 5 m × 2.6 m. Two speech sources are represented by loudspeakers, playing pre-recorded speech consisting of random phrases from the TIMIT database [13], and the loudspeakers are placed at different locations to simulate a varying angular source spreading. A primary source is located at angles of arrival of 0°, 20°, 40° and 60°, and a secondary source is located at an angle of arrival of −30°.

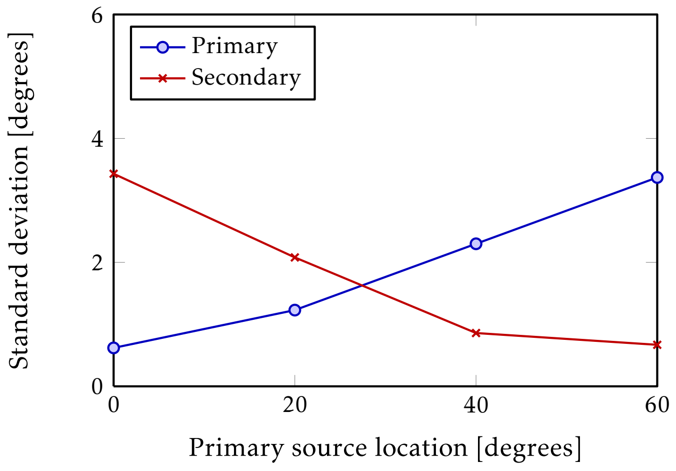
The sensor signals are recorded at a sample rate of 16 kHz, and time-frequency decomposition is performed using a two-times oversampled uniform DFT filterbank with two polyphase subband filter taps. The sensor array consists of four sensors, uniformly and linearly distributed with a sensor spacing of 4 cm. The tests focus on measuring the standard deviation and mean estimation error of the estimated angles when the primary source is positioned at each of its four locations. The standard deviation is the mean deviation from the mean estimated angle, while the mean estimation error is the mean difference between the mean estimated angle and the true angle.

Figures 4.1(a) and 4.2(a) show the standard deviation. Two important properties are shown: when the angle for the primary source approaches the endfire of the array, the standard deviation increases, as suggested by (2.25); when the two sources are spatially close to each other, the standard deviation also increases. It should also be noted that when the sources are spatially well separated, the standard deviation of the secondary source remains constant.

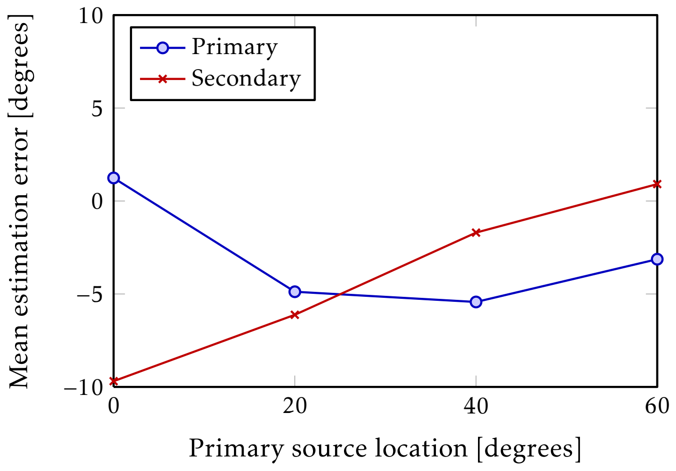
Figures 4.1(b) and 4.2(b) show the mean estimation error. Again, when the two sources are spatially separated, the mean estimation error for the secondary source remains constant. The mean estimation error for the primary source increases as the source approaches the endfire.

4.4 Conclusion

Real room recordings have shown that the proposed method constitutes a robust method for angle-of-arrival estimation for multiple concurrent speech sources. Good results were obtained in environments with moderate reverberation. All steps involved in estimating the angles are suitable for real time applications, which is important as the system is intended to be used in real-time speech localization.

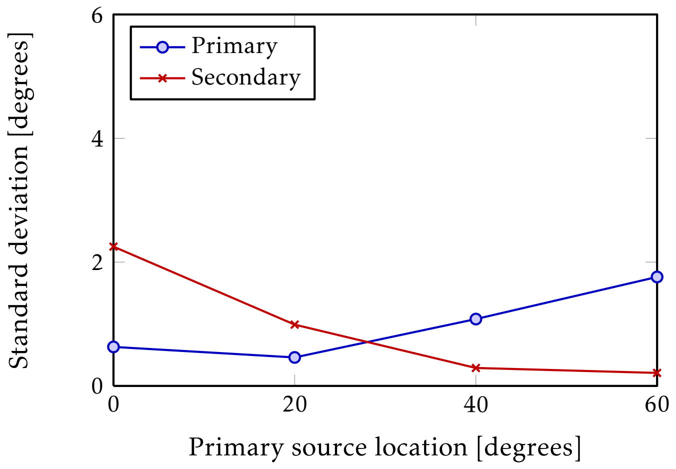


(a)

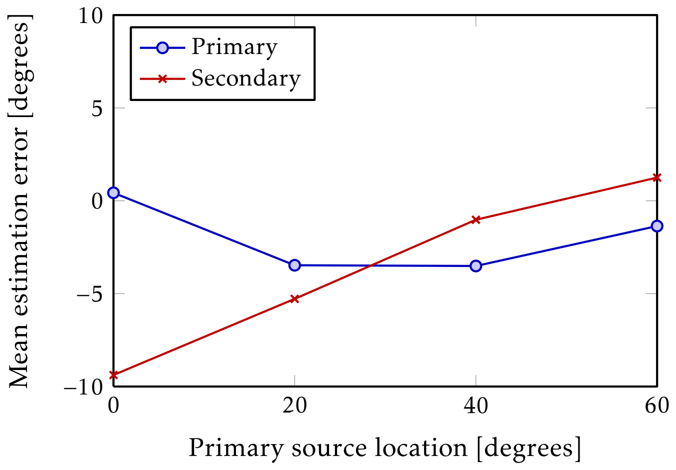


(b)

Figure 4.1: Standard deviation and mean estimation error for the primary and the secondary source, when the primary source is positioned at different angles. The number of DFT filterbank subbands is 256.



(a)



(b)

Figure 4.2: Standard deviation and mean estimation error for the primary and the secondary source, when the primary source is positioned at different angles. The number of DFT filterbank subbands is 1024.

Chapter 5

Multi-Source DOA Estimation

the SRP-based Approach

This chapter is based on the following article:

Mikael Swartling, Benny Sällberg, and Nedelko Grbić. Source Localization for Multiple Speech Sources Using Low Complexity Non-Parametric Source Separation and Clustering. *Signal Processing*, 91(8):1781–1788, August 2011.

Traditional blind signal separation methods have been designed and used for signal separation aiming for the reconstruction or estimation of original source signals, and to maintain the audio quality of the reconstructed source signals. One approach to blind speech signal separation is to exploit the sparse and disjoint structure of the signals in the time-frequency domain. It has been shown that the masking of time-frequency points can separate the additive mixtures of speech signals [12]. Various methods to design the masks have been presented, including methods based on signal model mixing parameters, observation vectors or direction-of-arrival estimates, and independent components [12, 33, 34, 35, 36, 37]. Masking in the time-frequency domain generally introduces music distortion in the reconstructed signals due to time-varying binary masking functions, and there is often a tradeoff between the amount of separation and the level of introduced music distortion. Research aiming to reduce the music distortion through masking while maintaining good signal separation has been conducted in [38, 39, 40].

When estimating the direction of arrival of the separated sources, the signals do not necessarily have to be reconstructed. Any audible music distortion that would have appeared in the reconstructed signals due to masking does not have to be a concern for the direction-of-arrival estimation. The method presented in chapter 4 used a blind signal separation method based on the real-time DUET method [12] with the aim of estimating the direction of arrival of the separated sources instead of reconstructing the source signals. It was possible to tune the parameters of the blind signal separation stage for a more aggressive and faster converging separation, and the direction of arrival of each source was then estimated by using the robust steered response power with the phase transform method. Even though the separated speech sources empirically had a significantly lower perceptual quality, due to aggressive separation when the signals were reconstructed to the time domain, the direction-of-arrival estimation performed well.

Accurate information about the location of sources is beneficial to many applications in signal processing. Blind methods or automatic tracking methods can sometimes take advantage of this additional knowledge to improve their performance. This includes applications such as speech enhancement and separation in hand-held and mobile devices, laptops and conference telephony systems, and also automatic camera tracking in video conference, surveillance and security systems.

This chapter presents a new method for direction-of-arrival estimation of multiple concurrent speech sources. The method presented in chapter 4 performed the blind signal separation and direction-of-arrival estimation in two independent stages: the first stage separated the sources with the help of a conventional signal separation method based on the real-time DUET method, and the second stage performed the direction-of-arrival estimation. In this chapter, the new method combines the two stages into a new simultaneous blind signal separation and direction-of-arrival estimation method. The new method eliminates the real-time DUET method used in the signal separation stage in the previous method, and instead replaces it with feedback from the direction-of-arrival estimation stage. In addition to evaluating the performance with both real recordings and simulations, a real-time prototype of the proposed method has been implemented on a DSP platform to evaluate the computational and the memory complexities in a real application.

5.1 The Multi-Source DOA Estimation Stage

A method based on cluster-based blind signal separation using time-frequency domain masking is proposed. The observed time delay estimates for each time-frequency point are clustered around a set of cluster centroids. After masking the time-frequency points into the separate clusters, the proposed method estimates the direction of arrival of each source and adjusts the cluster centroids using the SRP-PHAT method.

5.1.1 Blind Signal Separation

The anechoic signal propagation model, where the sensor signals are subject to a time-shift and an attenuation due to the propagation distance, is assumed. For a small sensor array, the attenuation is assumed to be equal all over the sensor array, and is modeled by a unit gain. The sensor signal for the sensor j is modeled as

$$x_j(t) = \sum_{i=0}^{I-1} s_i(t - \tau_{i,j}) + \nu_j(t). \tag{5.1}$$

The source signals $s_i(t)$ are subject only to a time-shift $\tau_{i,j}$ from the source i to the sensor j. The noise $\nu_j(t)$ is modeled as zero-mean independent white Gaussian noise. It is assumed that the time-shift can, in the absence of noise, be modeled as a phase-shift in the time-frequency domain [41]:

$$x_p(t) = x_q(t - \tau_{p,q}) \leftrightarrow X_p(\tau, \omega) = X_p(\tau, \omega)\,\mathrm{e}^{-\mathrm{j}\omega\tau_{p,q}}. \tag{5.2}$$

In the discrete time-frequency domain, and with the propagation delay $\tau_{i,j}$ in samples, the signal model is

$$X_j[n,k] = \sum_{i=0}^{I-1} S_i[n,k]\,\mathrm{e}^{-\mathrm{j}\omega_k\tau_{i,j}} + N_j[n,k]. \tag{5.3}$$

Under the assumption that the sources are W-disjoint orthogonal, no more than one source is active at any time-frequency point. The signal model can thus be reduced to

$$X_j[n,k] = S_i[n,k]\,\mathrm{e}^{-\mathrm{j}\omega_k\tau_{i,j}} + N_j[n,k] \tag{5.4}$$

where i is the index of the single active source at the corresponding time-frequency point $[n,k]$. The cross-power spectrum for two sensor signals then becomes

$$G_{p,q}[n,k] = \mathrm{E}\Big[X_p[n,k]\,X_q^{\star}[n,k]\Big] \tag{5.5}$$

$$= |S_i[n,k]|^2\,\mathrm{e}^{-\mathrm{j}\omega_k(\tau_{i,q}-\tau_{i,p})}. \tag{5.6}$$

For each time-frequency point, a time delay $T_{p,q}[n,k]$ between sensors p and q can be derived from the cross-power spectrum as

$$\begin{aligned} T_{p,q}[n,k] &= \tau_{i,q} - \tau_{i,p} && (5.7) \\ &= \frac{1}{\omega_k} \cdot \angle G_{p,q}[n,k]. && (5.8) \end{aligned}$$

For a small uniform linear sensor array with more than two sensors, an average time delay for a time-frequency point can be calculated as the average of the time delays for all sensor pairs:

$$T[n,k] = \frac{1}{|\mathbf{P}|} \sum_{\{p,q\}} \frac{T_{p,q}[n,k]}{q-p}. \tag{5.9}$$

Due to the assumed W-disjoint orthogonality, and the anechoic propagation path, the time delays depend only on what source is active in the given time-frequency point. Time delay estimates for all time-frequency points that are dominated by the same source will have the same value.

The estimated time delay for each time-frequency point is dependent on the phase of the cross-power spectrum. If the sensor array is too large and introduces spatial aliasing, the phase at some points in the cross-power spectrum is ambiguous. It is therefore necessary to ensure that the inter-sensor distance d for any sensor pair $\{p,q\}$ is $d < \mathrm{c}/(2 \cdot \mathrm{f}_{\mathrm{max}})$, where $\mathrm{f}_{\mathrm{max}}$ is the highest signal frequency in hertz, meaning that there is no spatial aliasing. The sensor array used in the evaluation section has an inter-sensor spacing d of 0.04 m, so $\mathrm{f}_{\mathrm{max}}$ must be less than 4287.5 Hz. Thus, sampling at 8 kHz ensures that no spatial aliasing occurs.

A number of cluster centroids C_i, one for each source present in the received sensor signals, is used to track the clusters of time delays for each time-frequency point. From the cluster centroids, a binary time-frequency mask for the source i is calculated as

$$\mathbb{M}_i[n,k] = \begin{cases} 1 & \text{if } T[n,k] \text{ is closest to cluster } C_i \\ 0 & \text{otherwise.} \end{cases} \tag{5.10}$$

The original source i as recorded by the sensor j is demixed in the time-frequency domain as

$$\hat{S}_{i,j}[n,k] = \mathbb{M}_i[n,k] X_j[n,k]. \tag{5.11}$$

For source reconstruction, any one of the received sensor signals can be demixed and the time-frequency representation is transformed back to the time domain. For source localization, which is the focus of this article, it is necessary to demix all sources from all sensor signals.

5.2 Proposed Cluster Centroid Tracking Stage

The proposed method is a novel method for updating cluster centroids in an online centroid tracking environment with the aim to estimate the direction of arrival of the separated sources. The previously presented method in chapter 4 consists of two separate stages for estimating the direction of arrival for multiple speech sources. The first stage was based on the real-time DUET method in which the signal model mixing parameters are tracked online employing a gradient-based search method that makes use of a maximum likelihood cost function [32]. The tracked mixing parameters serve as cluster centroids for the observed mixing parameters, and are used to create the masks to separate the individual sources. The second stage estimates the direction of arrival of the separated signals using the robust SRP-PHAT method. In the new method, the two stages are combined and the direction-of-arrival information from the robust SRP-PHAT is fed back to the cluster centroid tracking stage. The real-time DUET method is thus eliminated from the signal separation stage and replaced with the proposed cluster centroid tracking stage.

The cluster centroids track the time delay signal mixing parameter. The observed feature vectors for clustering are the observed time delays $T[n,k]$ for each time-frequency point $[n,k]$. After separation, the SRP-PHAT method updates the cluster centroids. The estimated time-difference of arrival $\hat{\tau}_i$ for the source i from the SRP-PHAT equals the new cluster centroid, so that the cluster update is $C_i = \hat{\tau}_i$.

Figure 5.1 illustrates the recursive process of signal separation and time-difference-of-arrival estimation. The clustering and masking stages makes up the signal separation stage, and the time-difference-of-arrival estimation stage makes up the cluster update stage.

5.3 Evaluation

The evaluation is performed in both a simulated room environment and using real room recordings. The blind signal separation and the direction-of-arrival estimation stages are evaluated using simulated sensor array data to cover a

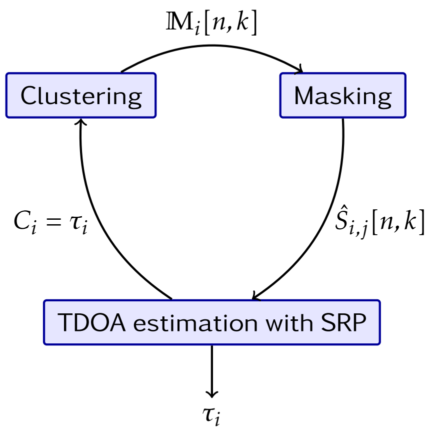


Figure 5.1: The recursive process for signal separation and the time-difference-of-arrival estimation for multiple sources.

range of controlled environments. Real room recordings are used to evaluate the intersection point selection stage to show its practical use. Room impulse responses for simulation are generated by the image method [5], where reflection coefficients for the room boundaries are adjusted for different reverberation times. Sensor array data are finally generated by filtering a source signal by the generated synthetic room impulse responses.

The real sensor data are recorded in a typical small conference room environment measuring 4 m × 4 m × 2.6 m. The same room dimensions and sensor and source placements are modeled to generate the synthetic impulse responses. The two sensor arrays consist of four sensors each, with 4 cm sensor spacing. The sample rate is 8 kHz, and the time-frequency transformation of the sensor signals is performed using oversampled uniform DFT filterbanks [8, 42]. The number of subbands vary, but a two-times oversampling and two-tap polyphase subband filters have been used throughout the evaluations. The speech sequences used both in the simulation and in the real recordings are taken randomly from the TIMIT database.

5.3.1 Multi-Source DOA Estimation

The blind signal separation is evaluated using simulated room impulse responses. Two sources are placed at a distance of 1.5 m from a single sensor array: a primary source is located at angles of arrival of $0°$, $20°$, $40°$ and $60°$, and a secondary source is located at an angle of arrival of $-30°$. The proposed SRP-based method is compared to the previous DUET-based multi-source direction-of-arrival estimation method for a varying number of subbands, reverberation times and primary source locations.

The evaluation results are presented in table 5.1. Estimates of the angle of arrival for each of the two sources are produced at every sample period of the analysis filterbank output. For 64, 256 and 1024 subbands and a two-times oversampling factor, direction-of-arrival estimates are produced every 4 ms, 16 ms and 64 ms, respectively. The average standard deviation and mean estimation errors over the different positional configurations are presented in the table. The mean estimation error is calculated as MEE = $\hat{E}[|\alpha| - |\hat{\alpha}|]$ and represents the mean error in the estimated angle of arrival in comparison to the true angle of arrival. A positive estimation error is an offset towards the endfires. As shown in the table, the proposed method performs with a lower standard deviation and mean estimation error than the previous DUET-based method in all setups.

The results show a general trend towards larger standard deviation and an increased mean estimation error when the number of subbands is reduced. However, the number of subbands also controls the delay of the filterbanks and is a tradeoff between signal separation, direction-of-arrival estimation performance and the delay between the input and output signals. The delay can be reduced by decreasing the polyphase subband filter length at the expense of increased subband aliasing, or by increasing the oversampling at the expense of a higher subband sample rate and higher processing requirements. The filterbank structure can furthermore be restricted by other processing, such as speech enhancement, if the same filterbanks are utilized.

5.4 Implementation Details

5.4.1 Computational Complexity

Common steps in both the presented SRP-based method and the previous DUET-based method is the masking of the sensor signals to separate the original sources, the estimation of cross-power spectrums (CPS) and the DOA

Table 5.1: Evaluation of the proposed SRP-based method in comparison to the previous DUET-based method. The estimated angle of arrival is evaluated for the standard deviation (STD) and the mean estimation error (MEE) in degrees. The results are averaged over five speech recordings and four positional configurations.

(a) Reverberation time RT_{60} is 100 ms.

	SRP		DUET	
Subbands	STD	MEE	STD	MEE
64	2.37°	-4.08°	5.01°	-6.51°
256	1.04°	-1.97°	1.82°	-3.60°
1024	0.65°	-1.36°	0.96°	-2.79°

(b) Reverberation time RT_{60} is 250 ms.

	SRP		DUET	
Subbands	STD	MEE	STD	MEE
64	3.48°	-7.34°	5.05°	-13.30°
256	1.53°	-6.60°	2.67°	-10.93°
1024	0.77°	-6.34°	1.28°	-10.09°

estimation using SRP-PHAT. The difference lies in how the cluster centroids are updated and how the masks are calculated. Table 5.2 lists the approximate number of operations needed to perform the steps that make the difference between the two methods. The listed operations are: complex valued additions and subtractions, multiplications and divisions, exponential and trigonometric functions, and searching for a minimum value among a set of values. The number of operations is listed for each subband and time-step.

Table 5.3 lists the approximate number of operations needed to estimate the cross-power spectrum (CPS) and to evaluate the SRP-PHAT method in (3.46) for a single time delay τ. The cross-power spectrum is estimated using a first-order AR-process, making the computational complexity low and independent of the effective integration time for the estimated value. The number of

Table 5.2: Complexity comparison of the differences between the proposed SRP-based method and the previous DUET-based method. The complexity is an approximate calculation of the number of operations performed for each subband and time instant.

Operation	SRP	DUET
Addition	$\lvert\mathbf{P}\rvert + I$	$\lvert\mathbf{P}\rvert \cdot I \cdot 3 + I$
Multiplication	0	$\lvert\mathbf{P}\rvert \cdot I \cdot 9 + I$
Trigonometry	1	$\lvert\mathbf{P}\rvert \cdot I \cdot 4$
Minimum	1 (out of I)	1 (out of $\lvert\mathbf{P}\rvert \cdot I$)

times needed to evaluate (3.44) depends on the numerical search method employed to find the point of maximum value. The search method used during evaluation was a two-step approach. First, a coarse search across the search space at evenly distributed time delays was performed, followed by a golden section search with parabolic interpolation around the initial highest steered response power to narrow in on the peak [43]. The point of the maximum steered response power was generally found in 18 to 25 evaluations.

The complexity of the proposed SRP-based method in comparison to the previous DUET-based method is significantly reduced. However, the total complexity is dominated by the SRP-PHAT and the search for maximum steered response power. Table 5.4 shows the complexity improvements for a varying number of sensors and sources. The complexity improvements are calculated from the approximated complexities in table 5.2 and 5.3, where I is the number of sources to locate and J is the number of sensors in the array. Sensor pairs are selected as adjacent sensors within the array, so $\lvert\mathbf{P}\rvert = J - 1$, and the SRP-PHAT is assumed to be evaluated 20 times per direction-of-arrival estimate during the optimization process. The total complexity is reduced by 12 % to 13 %. If the search method is improved such that the number of SRP-PHAT evaluations are reduced, the total reduction by using the proposed method is increased. As an example, if the number of SRP-PHAT evaluations is reduced to 10, the improvement in table 5.4 is increased to approximately 25 %.

Table 5.3: Complexity listing of the common steps; the cross-power spectrum (CPS) estimation and the SRP-PHAT evaluation. The complexity is an approximate calculation of the number of operations performed for each subband and time instant.

Operation	CPS estimation	SRP-PHAT
Addition	$\lvert\mathbf{P}\rvert \cdot I \cdot 2$	$\lvert\mathbf{P}\rvert \cdot I + \lvert\mathbf{P}\rvert$
Multiplication	$\lvert\mathbf{P}\rvert \cdot I \cdot 3 + \lvert\mathbf{P}\rvert + J$	$\lvert\mathbf{P}\rvert \cdot I \cdot 2$
Trigonometry	0	$\lvert\mathbf{P}\rvert \cdot I \cdot 2$

Table 5.4: Complexity improvement for the proposed SRP-based method over the previous DUET-based method. The number of operations is calculated from table 5.2 and 5.3 for different numbers of sources I and sensors J.

Configuration	SRP	DUET	Improvement
$I = 2$ and $J = 2$	239	271	12 %
$I = 2$ and $J = 8$	1637	1867	12 %
$I = 8$ and $J = 2$	881	1015	13 %
$I = 8$ and $J = 8$	6059	7003	13 %

5.4.2 Implementation

The proposed method has been designed with focus a on real-time online processing. A prototype implementation has been implemented on a custom floating point DSP platform. The two most prominent factors that could limit an implementation on the platform are the computational complexity and the memory complexity. The computational complexity was evaluated by comparing the frame time and the processing time for each of the stages. The frame time is the sample time of the output of the filterbanks used to transform the sensor signals into the time-frequency domain. The memory complexity was evaluated as the persistent states for each of the stages and the maximum temporary working set.

A critical factor that calls for immediate observation during the implementation is the result of the signal separation stage. A cross-power spectrum is estimated for each source and for each sensor so that the direction of arrival can be estimated for each separated source. An approach to reduce the memory impact of the signal separation stage is to assume that the set of sensor pairs $\{p, q\}$ for the direction-of-arrival estimation consists only of pairs of adjacent sensors. The memory impact can then be significantly reduced by only keeping one cross-power spectrum for each source in memory. The memory impact is also reduced by using an AR-process to integrate the cross-power spectrums. A variable integration time is thus achieved at a low constant memory impact. Another limiting factor is the number of filterbank subbands. Better localization performance can be achieved with a higher number of subbands, but this adds the cost of an increased memory impact. While the number of subbands increases the computational complexity per frame, the frame time is increased accordingly and the relative computational complexity remains constant.

Table 5.5 shows the computational and the memory complexity of the prototype implementation. The computational complexity is listed as the number of cycles, and the memory complexity is the storage requirement for persistent states. For as many as five concurrent sources being separated and located, the total computational complexity is 10.2 % of the available processing cycles, and the memory complexity is 14.1 % of the available memory on the particular DSP platform used for the implementation. The DSP platform is an Analog Devices ADSP-21262 DSP with 2 Mibit of memory and running at 100 MHz. The sampling and filterbank setup is otherwise the same as in the evaluation section: a 8 kHz sampling rate, 256 subbands, two-times oversampling and two-tap polyphase subband filters.

5.5 Conclusion

The chapter presents a new method for simultaneous blind signal separation and direction-of-arrival estimation of multiple speech sources. The improvements over the previously presented DUET-based method is the elimination of any signal or environment-dependent tuning parameters, improved direction-of-arrival estimation performance and a significantly reduced computational complexity. The proposed method performs overall better for a wide range of simulated environments, and real room recordings show the method's practical use.

The most computationally demanding stage is the SRP-PHAT step conducted to search for the peak in the steered response power from a sensor

Table 5.5: Computational and memory complexity for the stages.

Stage	Computations [cycles]	Memory [32 bit words]
Filterbanks	2.2×10^6	2048
Signal separation	0.8×10^6	5632
DOA estimation	7.2×10^6	10
Maximum working set	—	1536
Total	10.2×10^6	9226

array. While most overall improvements can be made by reducing the number of evaluations of the steered response power, the proposed method offers a significant computational reduction over the previous DUET-based method. Over all, the computational complexity is improved by 12 % to 13 %. A prototype implementation on an embedded DSP platform demonstrates that, for as many as five concurrent sources, the computational and memory complexity is well within the bounds of the platform.

Chapter 6

Multi-Source Location Estimation

This chapter is based on the following articles:

Mikael Swartling, Mikael Nilsson, and Nedelko Grbić. Distinguishing True and False Source Locations when Locating Multiple Concurrent Speech Sources. In *Proceedings of IEEE Sensor Array and Multichannel Signal Processing Workshop*, pages 361–364, July 2008.

Mikael Swartling, Benny Sällberg, and Nedelko Grbić. Source Localization for Multiple Speech Sources Using Low Complexity Non-Parametric Source Separation and Clustering. *Signal Processing*, 91(8):1781–1788, August 2011.

When employing multiple sensor arrays for source localization in the far field, intersecting the bearing lines from different sensor arrays yields the physical location of the source [29]. In the presence of multiple sources, there are multiple bearing lines at each sensor array and also multiple combinations of intersection points of the bearing lines, many of which do not correspond to real physical sources. This chapter presents a method to handle the problem of multiple intersection points. The proposed method integrates well with the methods presented in chapter 4 and 5, and is based on correlating parameters from the blind signal separation stage. This allows the localization stage to intersect the correct bearing lines from multiple sensor arrays so that the resulting location estimate corresponds to a true source location. Previously proposed methods to solve the multiple intersection point problem include methods based on the clustering of intersection points and the correlation of signals [44, 45].

Figure 6.1 shows the problem when intersecting bearing lines for the localization of three sources with the help of two sensor arrays. In the figure, there are nine intersection points and six possible ways to combine them to

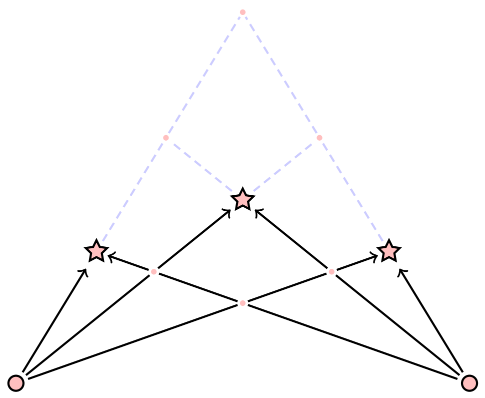


Figure 6.1: The problem when intersecting the bearing lines from three sources at two sensor arrays. Many of the multiple intersection points do not correspond to real physical sources. One of six possible source location configurations is shown in the figure.

estimate the location of the sources. In the general case, for M sources, there are M^2 intersections and $M!$ combinations which might originate from real physical sources. The terms true and false intersection points are used to describe whether an intersection point corresponds to a real source location or not. A true intersection point is the result of pairing bearing lines for the same source from different sensor arrays, while false intersections result from pairing bearing lines from different sources.

6.1 Intersection Point Selection

By separating the sources and estimating the direction of arrival at multiple sensor arrays, the resulting bearing lines can be intersected to determine the locations of the sources. The intersection point selection problem consists of intersecting the correct bearing lines from the sensor arrays when there are multiple sources present.

A pair of time-frequency masks for the same source produced by the blind signal separation stage at different sensor arrays are assumed to be correlated. The proposed methods to match signals and the corresponding bearing lines between arrays are based on correlating the masks and the signals from the blind signal separation stage. Masks and signal estimates that correlate the most are assumed to also belong to the same source, and so the corresponding bearing lines intersect at a real physical source location.

By separating the sources at each sensor array, a set of masks $\mathbb{M}_i^k$ is estimated for each of the sources i at each sensor array k. A straightforward way to select a combination of bearing line pairs from two sensor arrays is to calculate the $I \times I$ matrix

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & \cdots & a_{1,I} \\ \vdots & \ddots & \vdots \\ a_{I,1} & \cdots & a_{I,I} \end{bmatrix} \tag{6.1}$$

where

$$a_{i_1,i_2} = \left\| \mathbb{M}_{i_1}^1 \circ \mathbb{M}_{i_2}^2 \right\|. \tag{6.2}$$

If the matrix element a_{i_1,i_2} has a larger value, this means that there are many common time-frequency points, which implies a larger probability that the corresponding bearing lines intersect at a true source location.

To estimate a set of bearing line pairs that intersect at true source locations, successively select the largest element from the matrix $\mathbf{A}$, while the corresponding row and column should be excluded from the following iterations. The row and column indices correspond to the bearing line index for the corresponding sensor arrays. By excluding the selected row and column indices from the following iterations, it is ensured that each bearing line is used exactly once in the estimation of a source location. This approach assumes that there is only one source for each bearing line. However, this is a valid assumption, since the blind signal separation stage is assumed to separate the sensor signals into the individual source signals.

When correlating the binary masks, the energy contained in the time-frequency points of the separated signals is not weighted into the mask with the result that noise will contribute to the correlation estimates. To weigh the correlation contribution for each time-frequency point, the masks can be weighted by the energy content in the corresponding time-frequency points. The elements of the matrix then become

$$a_{i_1,i_2} = \left\| \hat{S}_{i_1,j}^1 \circ \hat{S}_{i_2,j}^2 \right\|. \tag{6.3}$$

Since the separated signals $\hat{S}_{i,j}$ are already masked during the separation of the sources from the mixtures, the energy-weighted masks are equal to the energy of the separated signals. Similar to the demixing in (5.11), any one of the individual demixed sensor signals j can be used.

Finding many common time-frequency points does not only imply a good bearing line match, but it also suggests that a higher number of common dominant time-frequency points is used for the direction-of-arrival estimation. To exploit this assumption, a scoring system is also proposed. A scoring system that turned out to work well was to directly use the value of a_{i_1,i_2} as the score for the intersection point of the bearing line pair $\{i_1, i_2\}$. An additional observation is the order in which the pairs are selected: the pairs are selected from $\mathbf{A}$ in decreasing order of the corresponding score a_{i_1,i_2}. The number of selected pairs may, therefore, be limited not only by score, but by the number of pairs with the highest score as well.

The difference in the two proposed scoring systems is what mutual information between the signals that is emphasized. The score for correlating the binary masks is a measure of the number of common dominant time-frequency points. However, for time-frequency points where none of the signals contain a significant amount of energy, the mask is noisy, which contributes equally to the final score. When correlating the weighted masks, the score is almost exclusively determined by the correlation of the high energy formants of the speech signals. Even a single erroneously masked formant can then significantly alter the score in favor of false intersections.

6.2 Evaluation

The intersection point selection method is evaluated with simulations and real room recordings. The real recording setup consists of two four-sensor uniform linear sensor arrays and three sources. Uniform DFT filterbanks with 256 subbands are used for time-frequency decomposition.

Figure 6.2 shows a part of a single recording and the corresponding localization results. The figure represents an overhead view of the room where each dot is a location estimate at each time instant. A location estimate for each of the three sources is produced at every sample period of the analysis filterbank output, and each location estimate is shaded based on the score, where a darker shade is an intersection point with a larger score. The true source locations are shown as the red stars, and the two sensor arrays are shown as red circles at the bottom of the figure.

Due to erroneously selected bearing line pairs, some locations are found to be false. These are mostly visible between the sensor arrays and the true

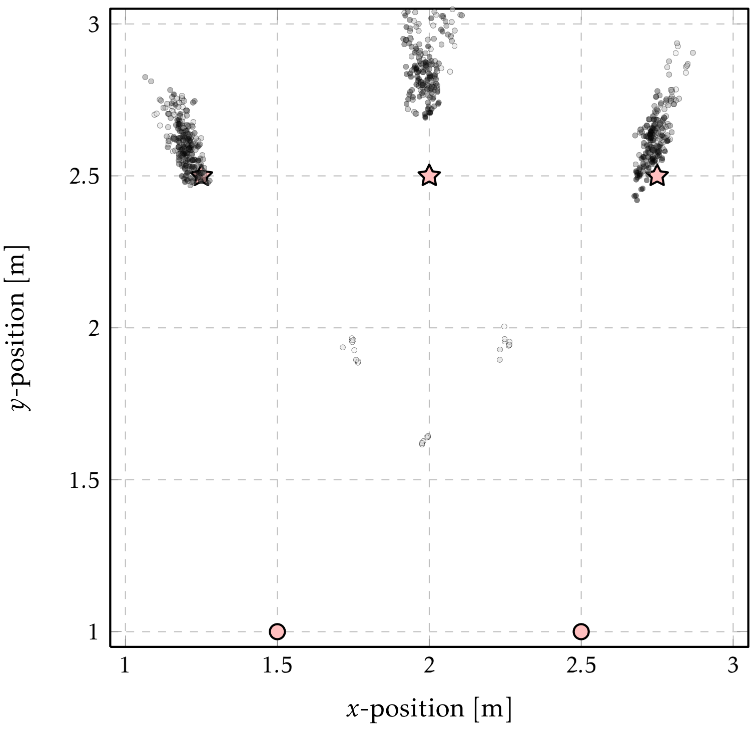


Figure 6.2: An overhead view of the room with three sources, where each dot is a location estimate at each time instant. Location estimates are scored according to the correlation of the binary masks in (6.2). Source direction-of-arrival estimation and bearing line pair selection are performed with the proposed methods.

source locations. These erroneous location estimates have a lighter shade and thus a lower score. The location estimates with a larger score are mainly clustered near the true source locations. Note also how the clusters, most notably the center cluster, are not centered around the true source positions as indicated by the circles. The direction-of-arrival estimates are offset towards the broadside of each sensor array, consistent with the mean estimation errors in the simulated results in table 5.1 in chapter 5.

Table 6.1 shows the percentage of correctly selected bearing line pairs for real recordings with three speech sources, and for simulated recordings with up to five sources. A bearing line pair is considered to be correctly chosen when the resulting intersection point is closer to any of the true intersection points than to any of the false intersection points. The two proposed score methods have been evaluated both in relation to the correlation of the masks, and to the correlation of the separated source signals. Furthermore, two score weighting methods have been evaluated, one which considers no weighting, which simply corresponds to the percentage of correctly matched bearing line pairs, and one which does consider score weighting, which is the percentage weighted by the score of each location estimate. It can be seen from the table that a success rate of over 95 %, with and without score weighting, is achieved in the majority of the test cases. Only the case with five sources shows a general decrease in success rate. The masks and the separated signals have been shown to be sufficiently similar between sensor arrays, and can be used to pair bearing lines that belong to the same source.

6.3 Conclusion

Two scoring methods are proposed and evaluated. One method is based on correlating the binary masks and one is based on correlating the energy of the separated source signals. The difference in the two methods is what mutual information is emphasized. The correlation of the binary masks is robust to estimation errors of the speech formants, while the correlation of the energy of the separated source signals is robust to mask noise. Both methods work well in low-noise environments, where more than 95 % or all location estimates are situated near the real physical source locations.

Table 6.1: The percentage of correctly matched bearing line pairs for three to five sources. The "no weight" column is the percentage of correctly matched pairs, and the "score weight" column is the percentage when weighted by the corresponding score in (6.2).

(a) Score by Mask.

	No weight	Score weight
3 sources, recorded	97 %	98 %
3 sources, simulated	95 %	98 %
4 sources, simulated	94 %	98 %
5 sources, simulated	90 %	92 %

(b) Score by Signal.

	No weight	Score weight
3 sources, recorded	89 %	95 %
3 sources, simulated	96 %	99 %
4 sources, simulated	96 %	99 %
5 sources, simulated	81 %	84 %

Chapter 7

Calibration Errors of Uniform Linear Sensor Arrays

an Analysis of the SRP-PHAT Method

This chapter is based on the following article:

Mikael Swartling and Nedelko Grbić. Calibration Errors of Uniform Linear Sensor Arrays for DOA Estimation: an Analysis with SRP-PHAT. *Signal Processing*, 91(4):1071–1075, April 2011.

When used in source localization, a geometrical calibration is often needed for sensor arrays on top of the phase calibration of individual sensors. Geometrical calibration is necessary in order to correctly map the modeled source position to a real physical location of the source. An important question here is to what degree is it necessary to calibrate the sensor array geometry for a sufficiently accurate mapping. This chapter presents an analysis of the time-delay errors caused by the geometrical errors when using the SRP-PHAT method in a uniform linear sensor array employed in far-field speech source localization.

A similar analysis, which presents a sensitivity analysis of localization and sensor calibration errors, has previously been undertaken in relation to the MUSIC method [46]. In other contexts, the problem with uncertain sensor arrays has been studied in relation to, for example, the effect on beamformers [47, 48], source location estimation [49, 50] and speech recognition [51]. Various approaches to handle uncertain sensor arrays have also been proposed, for example self-calibrating and partly calibrated sensor arrays [52, 53].

This chapter presents a model that approximates the SRP-PHAT method for small sensor displacements, as well as providing the theory and simulations for how calibration errors affect the time-difference-of-arrival estimation and, consequently, the direction-of-arrival estimation. This chapter also explains why, and under what conditions, the SRP-PHAT method fails to estimate the direction of arrival for large sensor displacements as indicated by the presented simulation results.

7.1 Sensor Location Calibration

This chapter focuses on two variants of the time-difference-of-arrival and the angle-of-arrival estimates: the true and the estimated values. In this context, the true time-difference of arrival, denoted τ, is the theoretical value expected to be obtained if the array is perfectly calibrated according to the assumed sensor array geometry. The estimated time-difference of arrival, denoted $\hat{\tau}$, is the actual value obtained from the array. It is potentially affected by erroneously located sensors, introducing an offset to the estimated value compared to the true value. The true and the estimated angles of arrival, denoted α and $\hat{\alpha}$, are defined accordingly.

It should be emphasized that the estimated values, in this context, do not refer to the values being estimated from real sensor data, but to the ideal geometrical values produced by a sensor array and a source location. The difference is whether the sensor array geometry is perfectly calibrated or not. The equations (2.16) and (2.17) in chapter 2 show the relationship between the time-difference of arrival and the angle of arrival for a uniform linear sensor array.

7.1.1 Two-Sensor Array Displacements

The effects on the estimated time-difference of arrival and angle of arrival for a two-sensor array can be trivially analyzed. An arbitrary displacement of the sensors can be modeled by a changed sensor spacing and a rotation of the sensor array baseline that connects the two sensors. Both the sensor spacing and the baseline orientation are independent parameters that can be analyzed separately. The sensor displacement is modeled as in figure 7.1 where $\mathbf{v}$ is the direction-of-arrival vector pointing towards the source, Δ_d is the change in the sensor spacing and Δ_α is the rotation of the sensor array baseline. The ideal sensor array assumes a perfectly calibrated sensor array, so that $\Delta_d = 0\,\mathrm{m}$ and

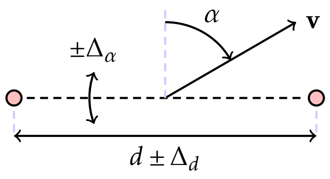


Figure 7.1: Two-sensor model.

$\Delta_\alpha = 0°$. The analysis will focus on how these parameters affect the time-difference-of-arrival and the angle-of-arrival estimates.

The estimated time-difference of arrival for $\Delta_\alpha = 0°$ is

$$\hat{\tau} = \frac{d + \Delta_d}{c} \sin(\alpha) \tag{7.1}$$

and the corresponding estimated angle of arrival is

$$\hat{\alpha} = \arcsin\left(\sin(\alpha) + \frac{\Delta_d}{d} \sin(\alpha)\right). \tag{7.2}$$

A change in the sensor spacing linearly translates into a corresponding change in the estimated time-difference of arrival. The effect on the angle of arrival, however, is non-linear, and the effect of an erroneously calibrated sensor spacing increases for sources near the endfire. Near the broadside, the time-difference-of-arrival estimate error and, consequently, the angle-of-arrival estimate error, approach zero.

Besides the actual estimation error, another problem with an erroneously calibrated sensor spacing is that if the sensor spacing is larger than the assumed model, the parameter to the arcsin function may fall outside its domain. In this scenario, the source is effectively located outside the beamformer's visible region. A time-difference of arrival can still be estimated, but there is no corresponding physical angle of arrival.

Similarly, the estimated time-difference of arrival for $\Delta_d = 0\,\mathrm{m}$ is

$$\hat{\tau} = \frac{d}{c} \cdot \sin(\alpha + \Delta_\alpha) \tag{7.3}$$

and the corresponding estimated angle of arrival is

$$\hat{\alpha} = \alpha + \Delta_\alpha. \tag{7.4}$$

A change in the sensor array baseline orientation linearly translates into a corresponding change of the estimated angle of arrival. There are no issues with domains, but the reflective symmetry in the sensor array baseline of the time-difference-of-arrival estimates may cause an additional aliasing effect if the sensor array baseline is rotated past the source location.

Uncertainties in the sensor spacing and the orientation of the sensor array baseline thus directly affect the time-difference-of-arrival and angle-of-arrival estimates. Calibration of a two-sensor array is necessary, and the accuracy of the calibration affects the accuracy of the time-difference-of-arrival and angle-of-arrival estimates.

7.1.2 Multi-Sensor Uniform Linear Array Approximation

The GCC integral in (3.44) is the inverse Fourier transform of $\Psi(\omega)G(\omega)$, where τ is the point at which the inverse Fourier transform is evaluated. Given the assumed anechoic signal model, the model of the cross-power spectrum becomes

$$\begin{aligned} G_{p,q}(\omega) &= X_p(\omega)\,X_q^{\star}(\omega) && (7.5)\\ &= |S(\omega)|^2\,\mathrm{e}^{-\mathrm{j}\omega\hat{\tau}_{p,q}}. && (7.6)\end{aligned}$$

Combined with the PHAT processor

$$\Psi_{p,q}(\omega) = \frac{1}{\left|G_{p,q}(\omega)\right|} \tag{7.7}$$

the amplitude is normalized and the generalized cross-power spectrum becomes

$$\Psi_{p,q}(\omega)G_{p,q}(\omega) = \frac{G_{p,q}(\omega)}{\left|G_{p,q}(\omega)\right|} = \mathrm{e}^{-\mathrm{j}\omega\hat{\tau}_{p,q}} \tag{7.8}$$

which is a unit amplitude linear phase signal which inverse Fourier transform is

$$\mathcal{F}^{-1}\left\{\mathrm{e}^{\mathrm{j}\omega\hat{\tau}_{p,q}}\right\}\Big|_{\tau=\hat{\tau}_{p,q}} = \operatorname{sinc}(\tau - \hat{\tau}_{p,q}). \tag{7.9}$$

The sinc function is here approximated by a first-order Taylor series expansion around the origin:

$$\operatorname{sinc}(\tau) = \frac{\sin(\pi\tau)}{\pi\tau} \tag{7.10}$$

$$= \sum_{n=0}^{\infty} (-1)^n \cdot \frac{(\pi\tau)^{2n}}{(2n+1)!} \tag{7.11}$$

$$\approx 1 - \frac{\pi^2}{6} \cdot \tau^2. \tag{7.12}$$

The SRP is the sum of the GCC for all sensor pairs, so the SRP maximum is located at the maximum of the sum of the GCC for all sensor pairs. The SRP-PHAT is thus approximated by the maximization of the sum of second degree polynomial functions, which has a trivial analytical solution.

Assume that the set of sensor pairs $\mathbf{P}$ contains the sensor pairs $\{p,q\}$. The SRP-PHAT method is then approximated by

$$\hat{\tau} = \arg\max_{\tau} \sum_{\{p,q\}} \int \Psi_{p,q}(\omega)\, G_{p,q}(\omega)\, e^{j\omega\tau(q-p)} d\omega \tag{7.13}$$

$$= \arg\max_{\tau} \sum_{\{p,q\}} \operatorname{sinc}\left[\tau(q-p) - \hat{\tau}_{p,q}\right] \tag{7.14}$$

$$\approx \arg\max_{\tau} \sum_{\{p,q\}} 1 - \frac{\pi^2}{6}\left[\tau(q-p) - \hat{\tau}_{p,q}\right]^2. \tag{7.15}$$

The maximum of the first-order Taylor series expansion approximation is solved analytically, and is

$$\hat{\tau} \approx \frac{1}{W_{\mathbf{P}}} \sum_{\{p,q\}} \hat{\tau}_{p,q}(q-p) \tag{7.16}$$

where

$$W_{\mathbf{P}} = \sum_{\{p,q\}} (q-p)^2. \tag{7.17}$$

The time-difference of arrival for the SRP-PHAT method is approximated by the average of the time-difference of arrival for all individual sensor pairs, weighted by the sensor spacing factor. A derivation of this analytical solution is presented in appendix B.

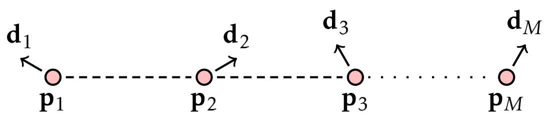


Figure 7.2: Multi-sensor model.

7.1.3 Multi-Sensor Uniform Linear Array Displacements

The sensor displacement of a uniform linear sensor array with three or more sensors is modeled as in figure 7.2, where $\mathbf{p}_m$ is the position of the sensor m, and $\mathbf{d}_m$ is a random sensor displacement vector. The direction-of-arrival vector $\mathbf{v}$ is a vector pointing from the sensor array towards the source and is defined as

$$\mathbf{v} = \begin{bmatrix} \sin(\alpha) & \cos(\alpha) \end{bmatrix}^{\mathrm{T}}. \tag{7.18}$$

Due to the far-field assumption, $\mathbf{p}_m$ can be projected onto the direction-of-arrival vector (or, in general, arbitrarily displaced perpendicularly to the direction-of-arrival vector) without affecting the estimated time-difference of arrival.

The time-difference of arrival for a sensor pair $\{p, q\}$ can be calculated from the difference in sensor position as projected onto the direction-of-arrival vector:

$$\hat{\tau}_{p,q} = \frac{(\mathbf{p}_q - \mathbf{p}_p)^{\mathrm{T}} \mathbf{v}}{c}. \tag{7.19}$$

A displaced position $\tilde{\mathbf{p}}_m$ of the sensor is

$$\tilde{\mathbf{p}}_m = \mathbf{p}_m + \mathbf{d}_m. \tag{7.20}$$

The estimated time-difference of arrival for the array with displaced sensors is, according to the approximation in (7.16), the average of the time-differences of arrival for all sensor pairs:

$$\hat{\tau} = \frac{1}{W_{\mathbf{P}}} \sum_{\{p,q\}} \frac{\left(\tilde{\mathbf{p}}_q - \tilde{\mathbf{p}}_p\right)^{\mathrm{T}} \mathbf{v}}{c} \cdot (q-p) \tag{7.21}$$

$$= \frac{1}{W_{\mathbf{P}}} \sum_{\{p,q\}} \frac{\left(\mathbf{p}_q + \mathbf{d}_q - \mathbf{p}_p - \mathbf{d}_p\right)^{\mathrm{T}} \mathbf{v}}{c} \cdot (q-p) \tag{7.22}$$

$$= \frac{1}{W_{\mathbf{P}}} \sum_{\{p,q\}} \frac{\left(\mathbf{p}_q - \mathbf{p}_p\right)^{\mathrm{T}} \mathbf{v}}{c} \cdot (q-p) + \frac{1}{W_{\mathbf{P}}} \sum_{\{p,q\}} \frac{\left(\mathbf{d}_q - \mathbf{d}_p\right)^{\mathrm{T}} \mathbf{v}}{c} \cdot (q-p). \tag{7.23}$$

The estimated time-difference of arrival is expressed in terms of the displaced sensor positions $\tilde{\mathbf{p}}$, which can be separated into the modeled sensor positions $\mathbf{p}$ and the displacement terms $\mathbf{d}$. An important observation here is that the estimated time-difference of arrival for the modeled sensor positions becomes the modeled time-difference of arrival:

$$\hat{\tau} = \tau + \frac{1}{W_{\mathbf{P}}} \sum_{\{p,q\}} \frac{\left(\mathbf{d}_q - \mathbf{d}_p\right)^{\mathrm{T}} \mathbf{v}}{c} \cdot (q-p). \tag{7.24}$$

This means that the estimation error for the time-difference of arrival can be calculated solely from the sensor displacements terms $\mathbf{d}$.

Assume, as an example, that the set of sensor pairs consists only of pairs of adjacent sensors, and that there are at least three sensors within the sensor array. The time-difference-of-arrival estimate is then

$$\hat{\tau} = \tau + \sum_{j=0}^{J-2} \frac{\left(\mathbf{d}_{j+1} - \mathbf{d}_j\right)^{\mathrm{T}} \mathbf{v}}{c} \tag{7.25}$$

$$= \tau + \sum_{j=0}^{J-2} \frac{\mathbf{d}_{j+1}^{\mathrm{T}} \mathbf{v}}{c} - \sum_{j=0}^{J-2} \frac{\mathbf{d}_j^{\mathrm{T}} \mathbf{v}}{c}. \tag{7.26}$$

Furthermore, assume that the two outer sensors are not displaced such that

$\mathbf{d}_0 = \mathbf{0}$ and $\mathbf{d}_{J-1} = \mathbf{0}$. The error term then becomes

$$\hat{\tau} = \tau + \sum_{j=0}^{J-3} \frac{\mathbf{d}_{j+1}^{\mathrm{T}} \mathbf{v}}{c} - \sum_{j=1}^{J-2} \frac{\mathbf{d}_{j}^{\mathrm{T}} \mathbf{v}}{c} \tag{7.27}$$

$$= \tau + 0 = \tau. \tag{7.28}$$

The average of the estimated time-difference of arrival of the displaced sensor pairs is reduced to the true time-difference of arrival, independently of the displacement vectors $\mathbf{d}_j$. Aside from the errors of the approximation in (7.16), variations in inner sensor locations affect neither the estimated time-difference of arrival nor the angle of arrival.

By displacing only the inner sensors in the sensor array, there are sensor pairs that counteract the time-difference-of-arrival offsets. Each inner sensor exists in two sensor pairs in $\mathbf{P}$, and each sensor pair counteracts the time-difference-of-arrival offset of the other. Because only one sensor pair has one of the end-point sensors, there are no other sensor pairs that counteract their time-difference-of-arrival offset when the end-point sensors are displaced. The effect of displacing an end-point sensor is analyzed by assuming that the end-point sensors are not displaced. Instead, a uniform change in the sensor spacing and a rotation of the sensor array baseline are assumed, as described in section 7.1.1.

7.1.4 Simulations

The presented theory is evaluated using real speech signals in a simulated anechoic environment. The middle sensor of a three-sensor uniform linear sensor array with a sensor spacing d of 0.04 m is displaced around its true location. The simulated sensor signals are sampled at 8 kHz, and decomposed into the time-frequency domain for the cross-power spectrum estimation using a uniform DFT filterbank [8] with 256 subbands.

Figure 7.3 shows the average angle-of-arrival estimation error for a varying sensor location displacement distance and direction. The displacement direction is defined relative to the true source direction-of-arrival vector:

$$\Delta_\alpha = \arccos\left(\frac{\mathbf{d}^{\mathrm{T}} \cdot \mathbf{v}}{\|\mathbf{d}\|}\right). \tag{7.29}$$

A displacement direction of 0° corresponds to a sensor displacement along the direction-of-arrival vector, while a 90° displacement direction corresponds to

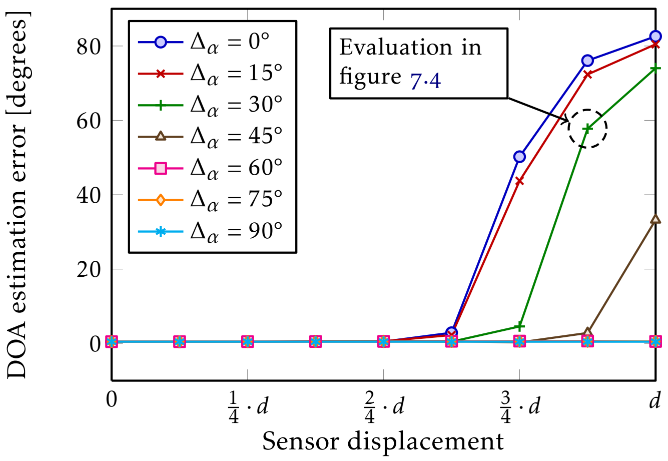


Figure 7.3: The inner sensor of an array is displaced. The graph shows the errors in the angle-of-arrival estimation as the difference between the estimated angle of arrival and the true angle of arrival, for a varying sensor displacement direction and distance.

a sensor displacement perpendicular to the direction-of-arrival vector. The simulations also show that the angle-of-arrival estimation error is symmetric with respect to positive and negative displacement angles and distances, so only displacement angles between 0° and 90° are shown.

The angle-of-arrival estimation error is small for most of the test cases, especially for displacements near-perpendicular to the direction-of-arrival vector. However, the error increases significantly for some cases when the sensor is significantly displaced along the direction-of-arrival vector. These large estimation errors are geometrical errors introduced by the destructive addition of shifted sinc functions, as shown later in section 7.2.

7.2 The Geometrical Errors

7.2.1 Modeling the Error

Figure 7.4 shows the cause of the estimation error in figure 7.3 for the case corresponding to the encircled data point. The solid lines are the sinc functions for the two sensor pairs and their peaks correspond to the pair's time-difference of arrival. The peaks are separated because the inner sensor is displaced, causing the difference in the estimated time-differences of arrival for the sensor pairs. The dashed line is the sum of the two delay functions and corresponds to the beamformer output power. Because the sinc functions are separated to such an extent that they no longer are able to constructively produce a single global maximum, there are two peaks in the output power, although one of the peaks is outside the visible region of the beamformer as indicated by the vertical dashed lines. Thus, the beamformer's response becomes a local minimum at the correct source location.

The maximum allowed separation, which occurs before the global maximum turns into a local minimum, can be calculated with the second derivative of the sum of the two sinc functions. If the two sinc functions are separated by τ, then

$$f(t,\tau) = \operatorname{sinc}(t+\tau) + \operatorname{sinc}(t-\tau). \tag{7.30}$$

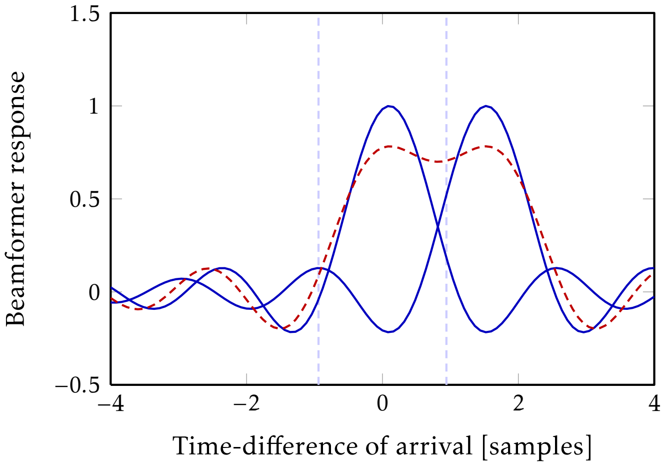
By letting $t = 0$, and recognizing the even symmetry of the sinc function, the maximum separation $\tau_{\max}$ is

$$\frac{\partial^2}{\partial \tau^2} f(t=0,\tau) = 0 \Rightarrow \tau_{\max} = \tau. \tag{7.31}$$

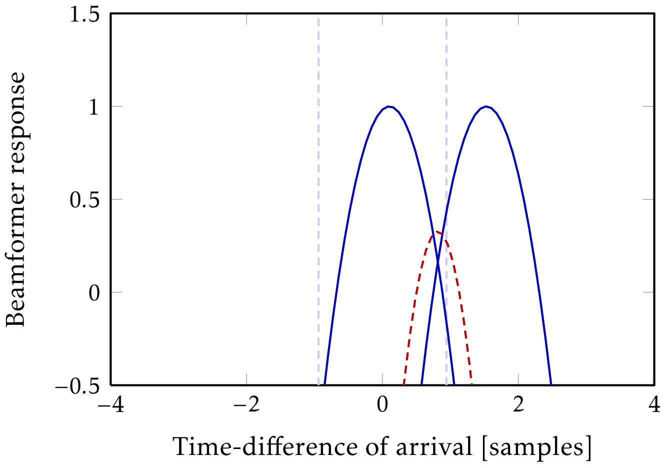
A numerical search reveals that the maximum separation is approximately 0.66 samples. As long as a sensor displacement results in a time-difference-of-arrival displacement less that $\tau_{\max}$ samples, the sum of the two sinc functions maintains the global maximum given by the SRP-PHAT method.

7.2.2 Simulations

Figure 7.5 shows the average time-difference-of-arrival estimation error for a sensor array with displaced inner sensors for 10 000 simulations. The sensors are randomly displaced with a uniform distribution up to a distance corresponding to 1 sample at a sample rate of 8 kHz. The error is estimated for different numbers of sensors within the sensor array and for a varying maximum random displacements of the sensors. The simulations for the three-sensor array is consistent with the presented theory: the error is close to zero up to the vicinity of the maximum separation $\tau_{\max}$, indicated by the vertical dashed line. Beyond this limit, this method effectively breaks down. Simulations for sensor arrays with more than three sensors are also consistent with the theory. The error is small for displacements where the second-degree polynomial



(a)



(b)

Figure 7.4: The geometrical effect causing the estimation error in figure 7.3 for a case corresponding to the dashed circle. The sinc functions no longer add constructively, causing erroneous global peaks in the beamformer response.

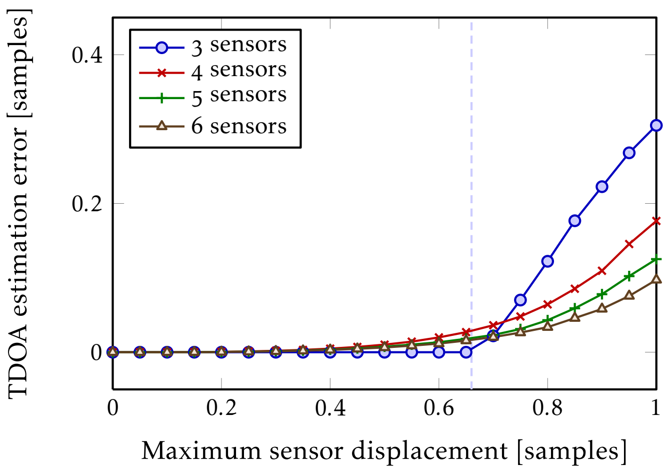


Figure 7.5: The time-difference-of-arrival estimation errors for multiple sensors. The inner sensors of the sensor array are randomly displaced, and the time-difference-of-arrival estimation error is the mean error of the estimated time-difference of arrival for many random displacements.

sufficiently approximates the sinc function, and where the averaging of sensor displacement errors decreases the time-difference-of-arrival error in larger arrays.

7.3 Conclusion

The chapter presents an analysis of the geometrical errors when performing time-difference-of-arrival and angle-of-arrival estimations with the SRP-PHAT method, using a uniform linear sensor array and a far-field source. Both the theory and the simulations show that the SRP-PHAT method is robust in relation to small sensor placement errors. The direction-of-arrival estimation breaks down when the time-difference-of-arrival estimates are significantly offset due to extreme sensor displacement. The derived maximum allowed sensor displacement $\tau_{\max}$ for a three-sensor array corresponds to a 2.82 cm location error at a sample rate of 8 kHz, or 0.47 cm at 48 kHz. The sensor location error should be related to the initially assumed 4 cm sensor spacing of the sensor array model, which is a significant relative sensor location error. To achieve sufficient direction-of-arrival estimation accuracy, a rough calibration of the array may thus be sufficient.

Chapter 8

Higher-Order Steered Beamformer Response

This chapter is based on the following article:

Mikael Swartling, Benny Sällberg, and Nedelko Grbić. Direction of Arrival Estimation for Speech Sources using Fourth Order Cross Cumulants. In *Proceedings of IEEE International Symposium on Circuits and Systems*, pages 1696–1699, May 2008.

In many applications, an accurate estimate of the location of an acoustic speech source is desired. An example of where speech source localization can be used is video conferencing systems where the camera automatically tracks the person speaking. For reliable camera tracking, an accurate estimate of the speaker's location is needed. In speech enhancement, accurate estimates may also be necessary, as in [54] where time-difference-of-arrival estimates are used to focus a beamformer towards an active speaker. Speech enhancement systems target for example hands-free sets for cars, mobile devices, laptop computers and conference telephony systems. A popular method used lately for direction-of-arrival estimation is the SRP-PHAT [15] which uses second-order statistics through the use of cross-power spectra to steer a beamformer, searching for a maximum in the power output from the beamformer. A peak in the power output means the beamformer is aimed towards the acoustic source with the highest power.

The use of fourth-order statistics (the kurtosis measure) can provide a way to distinguish speech from noise. A Gaussian random process exhibits a cumulant with a zero value for any cumulant order higher than two, so estimation of higher-order cumulants will in theory eliminate Gaussian noise [55].

Although not entirely Gaussian, most real noise sources have a probability density function close to the Gaussian distribution, whereas speech can be modeled more accurately as a Laplacian distribution [56]. This a-priori knowledge can be exploited in order to separate and enhance speech sources by, for example, maximizing the kurtosis measure of the output of a beamformer [57].

In this chapter, a new method for localizing speech sources is presented. The method is based on the SRP-PHAT method where a new optimization target is proposed. Instead of focusing on second-order statistics (the cross-power spectrum), fourth-order statistics are used. The rationale behind choosing fourth-order statistics when estimating the direction of arrival for speech sources is that speech sources exhibit larger fourth-order cumulant values than most common noise sources. By comparison, second-order statistics relate only to the power of the sources, and not specifically to their probability density function. Other direction-of-arrival estimation methods based on higher-order statistics have also been proposed, but these are eigenstructure-based methods for narrow-band sources [58, 59, 60].

8.1 Fourth-Order Cross Cumulant

8.1.1 Kurtosis

Kurtosis, a fourth-order statistical measure, can be viewed as a measure of the Gaussianity of a random process. A larger kurtosis measure corresponds to a sharper probability density function compared to the Gaussian distribution. A process that has a sharper probability density function than a Gaussian process is called super-Gaussian. Speech signals have been shown to exhibit a probability density function closer to a Laplacian distribution, a super-Gaussian distribution, while noise sources generally exhibit a probability density function closer to a Gaussian distribution [56].

Figure 8.1 shows the estimated probability density function for the speech and noise signals used to evaluate the proposed method in comparison to Gaussian and Laplacian noise. The probability density functions are estimated using a histogram, and the signals are generated Gaussian and Laplacian noise, factory noise and speech, where all signals have a unit variance. The probability density function for factory noise is very similar to the Gaussian distribution, while the speech is closer to the Laplacian distribution.

Applications using the kurtosis measure of speech includes blind speech separation [57]. In [61] the subband kurtosis is maximized in order to extract a speech source from a noisy environment, and in [62], a modification of the SRP-PHAT method is presented where each subband is additionally weighted according to the subband kurtosis measure to emphasize subbands with speech present.

8.1.2 Cross Cumulants

A second-order complex valued cross cumulant is the cross-power spectrum [55], expressed as

$$C_2(X, Y^\star) = \mathrm{E}[X\,Y^\star] - \mathrm{E}[X]\,\mathrm{E}[Y^\star]. \tag{8.1}$$

For the zero-mean processes X_p and X_q, the second-order complex valued cross cumulant $C_2(X_p, X_q^\star)$ becomes the cross-power spectrum $G_{p,q}$ used as the optimization target by the SRP-based methods. A fourth-order complex valued cross cumulant for two zero-mean processes X and Y can be expressed as

$$C_4(X, X, Y^\star, Y^\star) = \mathrm{E}[X^2\,Y^{\star 2}] - \mathrm{E}[X^2]\,\mathrm{E}[Y^2]^\star - 2\mathrm{E}[X\,Y^\star]^2. \tag{8.2}$$

It should be noted that there are many variants of cross cumulants for complex valued random processes. For a complex valued cross cumulant of the order N, there are 2^N ways to configure the complex conjugate on the parameters to C_N, resulting in different definitions [55].

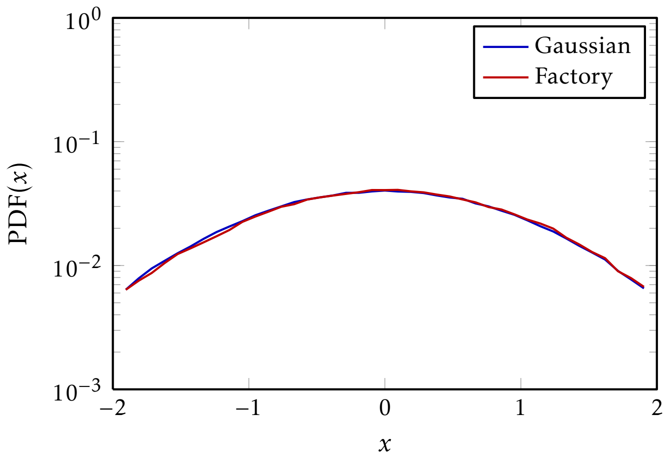
8.2 Proposed Method

Instead of using the SRP method, which is based on a second-order complex valued cross cumulant to search for a maximum in the power response, the proposed method utilizes a fourth-order complex valued cross cumulant. In the discrete-time time-frequency domain, the fourth-order complex valued cross cumulant used as optimization target is

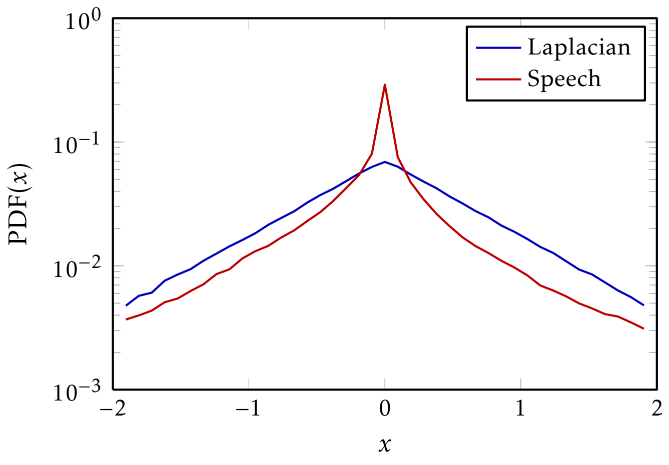
$$\begin{aligned} K_{p,q}[k] &= C_4\Big[X_p[k], X_p[k], X_q^\star[k]\,\mathrm{e}^{\mathrm{j}\omega_k\tau}, X_q^\star[k]\,\mathrm{e}^{\mathrm{j}\omega_k\tau}\Big] && (8.3)\\ &= C_4\Big[X_p[k], X_p[k], X_q^\star[k], X_q^\star[k]\Big]\,\mathrm{e}^{\mathrm{j}2\omega_k\tau}. && (8.4)\end{aligned}$$

The process of estimating the time-difference of arrival for a speech source, using the phase transform processor, is therefore stated as

$$\hat{\tau} = \arg\max_{\tau} \sum_{\{p,q\}} \sum_{k=0}^{K-1} \frac{K_{p,q}[k]}{\big|K_{p,q}[k]\big|}\,\mathrm{e}^{\mathrm{j}2\omega_k\tau}. \tag{8.5}$$



(a) Gaussian noise and factory noise.



(b) Laplacian noise and speech.

Figure 8.1: Probability density functions for (a) Gaussian noise and factory noise, and (b) Laplacian noise and speech.

Due to the similarities with SRP, where a cross-power measure is used instead of a cross-kurtosis measure, the method is herein denoted as Steered Response Kurtosis (SRK).

8.3 Evaluation

The proposed method for direction-of-arrival estimation is evaluated and the performance is compared to the SRP-PHAT method. The difference between the two methods is only the target for optimization: the fourth-order cross cumulant for SRK-PHAT and the second-order cross cumulant for the SRP-PHAT.

8.3.1 Setup

To record the evaluation signals, speech signals are played through a loudspeaker placed at different angles relative a four-microphone uniform linear sensor array with a 4 cm sensor spacing. The recordings are performed at a sample rate of 16 kHz in a room with the dimensions $4\,\text{m} \times 5\,\text{m} \times 2.5\,\text{m}$. The expectation values when calculating the second and fourth-order cross cumulants are estimated from a window starting 250 ms prior to the time of the time-difference-of-arrival estimation.

8.3.2 Speech

The clean speech test is used to compare the SRK-PHAT and the SRP-PHAT methods for a speech signal with no added noise other than the environment noise together with reverberation created in the recording environment. Figure 8.2 shows the standard deviation of the estimated angle of arrival using the SRK-PHAT and the SRP-PHAT methods. From the figure it is apparent that the variance of the direction-of-arrival estimates increases as the angle approaches the endfire, which is consistent with (2.25) for the second-order cumulant. The two methods perform similar under the no-noise condition.

8.3.3 Speech and Noise

The same speech signals are used as in the previous test with an added noise source at different signal-to-noise ratios. The noise is a prerecorded sequence of omnidirectional factory noise with machinery and moving vehicles. Fig-

Table 8.1: Time spent in parts of the SRK-PHAT and the SRP-PHAT angle-of-arrival estimation methods.

Operation	SRK	SRP
Cross cumulant estimation	15 s	2 s
Cross cumulant optimization	10 s	10 s
Total running time	28 s	15 s

ure 8.3 shows the standard deviation at a signal-to-noise ratio of 0 dB. In a noisy environment, the SRK-PHAT performs better than the SRP-PHAT.

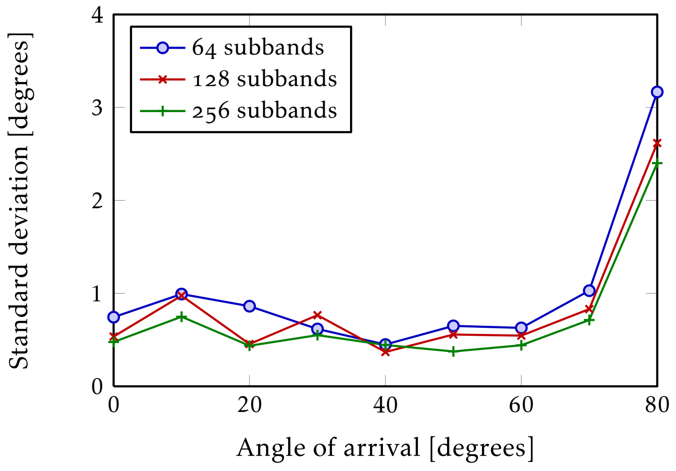
8.3.4 Computational Complexity

In this section, a brief analysis of the computational complexity is presented. Table 8.1 shows the time in seconds spent in various parts of the angle-of-arrival estimator as reported by the Matlab profiler. The cross cumulant estimation time is the time spent estimating the fourth and second-order cross cumulants. The cross cumulant optimization is the time spent in the optimizing function to search for a beamformer output maxima. Finally, the total running time is the total time spent in the program.

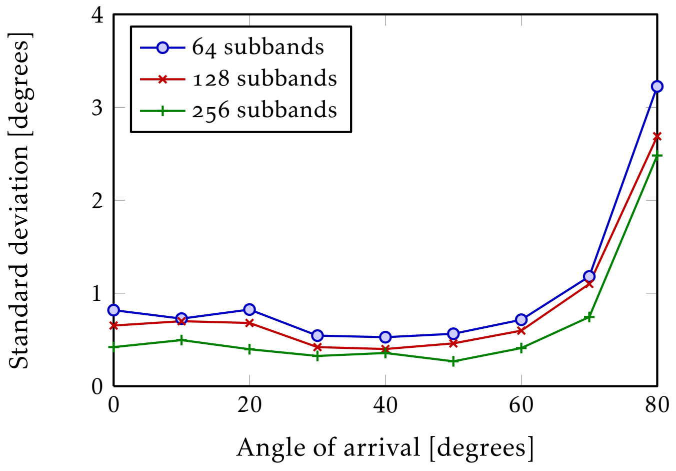
The total running time is approximately doubled when fourth-order cross cumulants are introduced. Optimization time is unchanged as the process of phase alteration and summing in the SRP and the SRK integral is the same. What contributes to the increased processing time is the estimation of fourth-order cross cumulants over second-order cross cumulants. The total running time includes subband transformation of the input signals and various one-time calculations before the actual direction-of-arrival estimation methods are executed.

8.4 Conclusion

The proposed method to use fourth-order statistics over second-order statistics was shown to improve performance of the direction-of-arrival estimator for speech sources under noisy conditions. Under a no-noise condition, the two methods yield similar results, which is reasonable since the SRK method

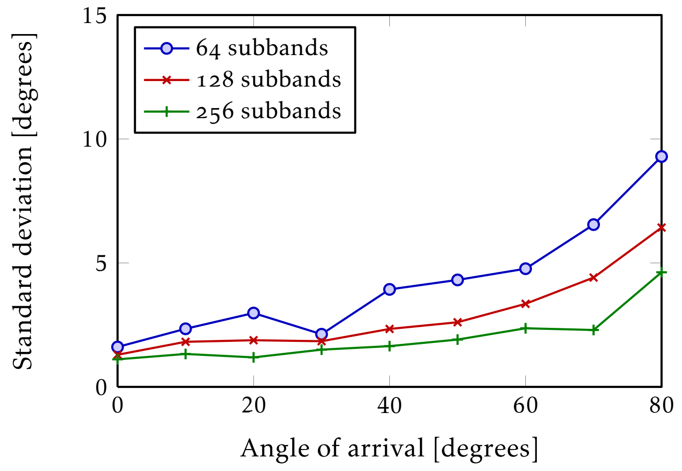


(a) SRK-PHAT.

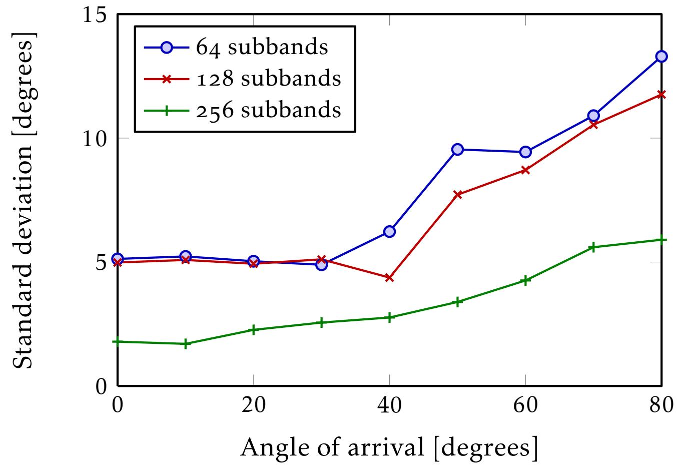


(b) SRP-PHAT.

Figure 8.2: The standard deviation of the angle-of-arrival estimate for a clean speech source. The graphs compare the SRK-PHAT and the SRP-PHAT methods for filterbanks with 64, 128 and 256 subbands.



(a) SRK-PHAT.



(b) SRP-PHAT.

Figure 8.3: The standard deviation of the angle-of-arrival estimate for a speech source and environment noise with 0 dB SNR. The graphs compare the SRK-PHAT and the SRP-PHAT methods for filterbanks with 64, 128 and 256 subbands.

was expected to be less sensitive to low-kurtosis Gaussian noise than the SRP method. At the same time, a brief analysis of the computational complexity shows that the complexity of the SRK method is approximately double that of the SRP method for the direction-of-arrival estimation procedure as a whole.

Chapter 9

An Alarm Signal Detection Framework

This chapter is based on the following article:

Mikael Swartling, Mikael Nilsson, and Nedelko Grbić. Detection of Vehicle Mounted Auditory Reverse Alarm using Hidden Markov Model. In *Proceedings of ELMAR-2007*, pages 163–166, September 2007.

An important safety issue at many industrial workplaces is the interaction between workers and moving vehicles. Due to limited visibility from the driver's seat of many vehicles, the risk of an accident increases when driving the vehicle in reverse. Different kinds of warning systems exist to prevent accidents and an auditory reverse alarm is one such system. The auditory reverse alarm consists of a buzzer that emits a single-frequency pulsating tone. The auditory reverse alarm is automatically activated when, for example, the vehicle is in reverse gear.

At a workplace with high noise levels, hearing protection devices are often required by safety regulations. However, such devices degrade the worker's auditory perception of his or her environment, making it difficult to both detect and localize the reverse alarm [63]. In addition, hearing-impaired workers may have even greater difficulties detecting and locating the alarm.

This chapter presents a method to automatically detect a general single frequency tone with a pulsating pattern. It it based on the hidden Markov model and pattern matching techniques, and designed for real-time platforms. The purpose of the method is to enable active hearing protection devices to aid the users in both detecting and locating auditory alarms such as reverse alarms. The method is computationally simple and suitable for real-time platforms such as the active hearing protection devices that exist today.

9.1 Signal Model and Matching Features

The received sensor signal is modeled as

$$x[n] = x_{\mathrm{a}}[n] + v[n] \tag{9.1}$$

where $x_{\mathrm{a}}[n]$ is the desired alarm signal that should be detected and $v[n]$ is an arbitrary interfering signal component representing the other sounds present in the environment. The alarm signal is assumed to be a narrow-band amplitude modulated signal, and as such is modeled as

$$\begin{aligned} x_{\mathrm{a}}[n] &= \cos(\omega_{\mathrm{a}} n) x_{\mathrm{a}}^{\shortparallel}[n] && (9.2) \\ &= \frac{1}{2}\left[\mathrm{e}^{\mathrm{j}\omega_{\mathrm{a}} n} + \mathrm{e}^{-\mathrm{j}\omega_{\mathrm{a}} n}\right] x_{\mathrm{a}}^{\shortparallel}[n] && (9.3) \end{aligned}$$

where ω_{a} is the center frequency of the narrow band signal, and $x_{\mathrm{a}}^{\shortparallel}[n]$ is the alarm envelope signal to be modulated.

9.1.1 Alarm Envelope Estimation

The alarm envelope signal, which contains the signature of the alarm signal, can be estimated by performing pass-band sampling around the center frequency ω_{a}. Pass-band sampling is similar to the procedure used in the subbands of a modulated uniform DFT analysis filterbank, with the exception that only a single subband is of interest, and that the center frequency of that subband can be chosen arbitrarily. A filter $h_{\omega_{\mathrm{a}}}[n]$ is created as

$$h_{\mathrm{a}}[n] = h_{\mathrm{LP}}[n] \mathrm{e}^{\mathrm{j}\omega_{\mathrm{a}} n} \tag{9.4}$$

where the prototype filter $h_{\mathrm{LP}}[n]$ is a low-pass prototype FIR filter with $P \cdot K$ filter coefficients and a cut-off frequency of $\omega_{\mathrm{c-o}} = 1/K$.

The full band signal is filtered by the modulated prototype filter to get the modulated alarm signal

$$x_{\mathrm{a}}[n] = \sum_{m=0}^{P \cdot K-1} x[n] h_{\mathrm{a}}[n-m] \tag{9.5}$$

and the alarm envelope signal is estimated as

$$\hat{x}_{\mathrm{a}}^{\shortparallel}[n] = |x_{\mathrm{a}}[n]|. \tag{9.6}$$

Because the upper frequency content of the demodulated baseband signal is limited in frequency by the low-pass prototype filter to a bandwidth of $1/K$, the baseband signal can be decimated by a factor K. The bandwidth of the modulated prototype filter is assumed to be small, and the alarm signal is therefore assumed to be the dominant component within the baseband signal.

9.1.2 Normalization

The alarm envelope signal is the base of the feature extraction for the hidden Markov model and the pattern matching. However, the actual amplitude of the alarm envelope signal varies depending on parameters such as the physical properties of the reverse alarm, the sensor properties, the recording or amplifier settings, and the distance of acoustic propagation. In order to make the alarm envelope signal amplitude independent, an amplitude normalization stage is performed.

A simple way to normalize the alarm envelope signal is to normalize its mean value. Given that the assumed alarm envelope signal has the duty cycle d, the peaks of the on-cycle of the alarm envelope signal duty cycle will be distributed around $1/d$. In practice, however, a real recorded alarm envelope signal is not stationary over a longer time period. This can be addressed by normalizing the alarm envelope signal over shorter time frames. The individual frames can be considered stationary and be normalized in amplitude.

As outlined in figure 9.1, a framing and overlap-add reconstruction approach is taken to normalize an alarm envelope signal that can vary over time. The alarm envelope signal is broken into overlapping and windowed signal frames:

$$x_{\mathrm{f}}[k,m] = \begin{cases} x_{\mathrm{a}}^{\shortparallel}[kD+m]\,w[m] & 0 \leq m < L \\ 0 & \text{otherwise} \end{cases} \tag{9.7}$$

where L is the frame duration, D is the frame step and w is a window of the length L. The frame duration should be chosen to correspond to a whole number of pulse periods of the alarm envelope signal to ensure that the normalization does not depend on the phase of the signal window with respect to the alarm envelope signal. The frames are then normalized:

$$\hat{x}_{\mathrm{n}}[k,m] = d \cdot L \cdot U \cdot \frac{x_{\mathrm{f}}[k,m]}{\sum_{m=0}^{L-1} x_{\mathrm{f}}[k,m]} \tag{9.8}$$

where

$$U = \frac{1}{L}\sum_{0}^{L-1} w[n] \tag{9.9}$$

is a factor that compensates for the effect the window has on the mean value of the alarm envelope signal block. The duty cycle factor d normalizes the amplitude of the on-cycle of the alarm envelope signal to unity. The last step is to reconstruct the framed and windowed signal into a continuous time signal by the overlap-add procedure:

$$\hat{x}_{\mathrm{a}}^{\mathrm{II}}[n] = \sum_{k=-\infty}^{\infty} \hat{x}_{\mathrm{n}}[k, n-kD]. \tag{9.10}$$

Figure 9.2 shows an example of the feature extraction and normalization method. A spectrogram of a full-band training signal is shown in figure 9.2(a). Figures 9.2(b) and 9.2(c) show the demodulated baseband envelope signal and the normalized baseband envelope signal, respectively. Note that the spectrogram shows the full time frame of the training signal used in the evaluation section, but the subband envelope signals show only a slice of the full time frame to emphasize how local details are normalized.

The proposed normalization method will only normalize the amplitude of the on-cycle to unity as long as the overlap-add of the window is unity. That is, as long as

$$\sum_{k=-\infty}^{\infty} w[n-kD] = 1 \tag{9.11}$$

for all n. This is the case for the typical window and frame steps configurations, such as a rectangular window with $D = L$, or a Hanning window with $D = L/2$. Other configurations, such as a Hanning window with $D = L/4$ do not overlap-add to unity. The overlap-add is, however, constant for all n in the latter example, and fundamental to the method proposed in this chapter. The fact that the alarm envelope signal's on-cycle values are normalized to unity is only a convention, and any constant scale factor thereof does not affect the proposed method as long as the normalization is consequent for both the training and the execution stages.

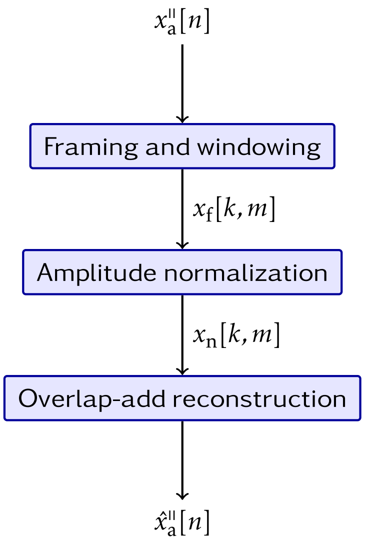


Figure 9.1: Block diagrams of the alarm envelope normalization stage.

9.2 The Hidden Markov Model

9.2.1 Training

The training of the hidden Markov model can be stated as [64, 65, 66]

$$\lambda^\star = \arg\max_{\lambda} P(\mathbf{O}|\lambda) \tag{9.12}$$

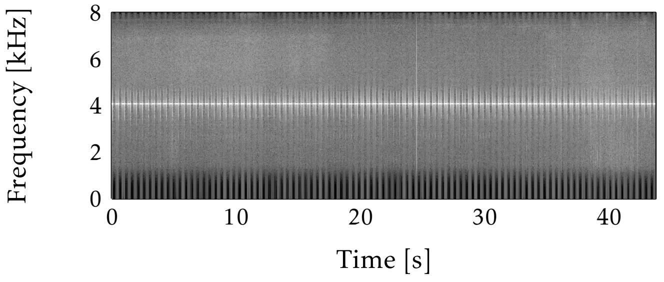
where

$$\mathbf{O} = \left\{\mathbf{o}[0] \quad \mathbf{o}[1] \quad \mathbf{o}[2] \quad \cdots\right\} \tag{9.13}$$

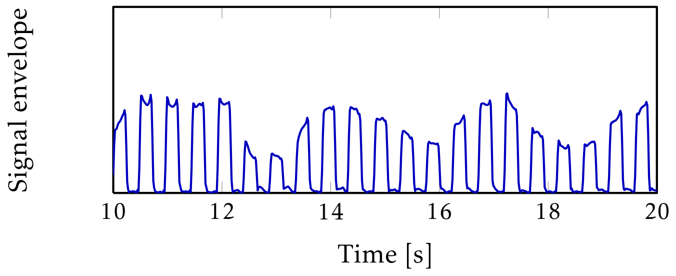
is a set of training feature vectors and

$$\lambda = \left\{\mathbf{A} \quad \mathbf{B} \quad \boldsymbol{\pi}\right\} \tag{9.14}$$

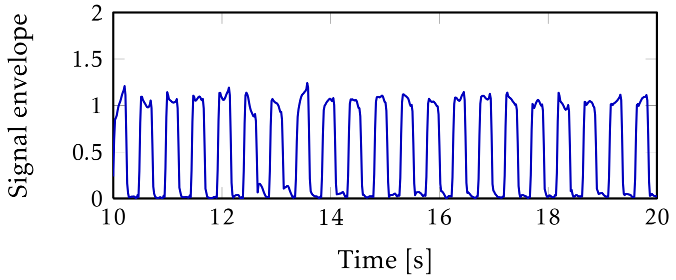
are the parameters describing the hidden Markov model. The hidden Markov model parameters comprise the state transition probability distribution **A**, the



(a) Spectrogram of the full band signal.



(b) Decimated subband envelope signal.



(c) Decimated and normalized subband envelope signal.

Figure 9.2: Alarm envelope signal extraction and normalization: (a) spectrogram of the full band signal, (b) the subband envelope signal, and (c) the normalized subband envelope signal.

observation symbol probability distribution $\mathbf{B}$ and the initial state distribution $\boldsymbol{\pi}$. The feature vectors $\mathbf{o}[n]$ used by the hidden Markov model are created by framing the normalized alarm envelope signal into blocks of N samples, so that

$$\mathbf{o}[n] = \begin{bmatrix} \hat{x}_{\mathrm{a}}^{\shortparallel}[n] & \hat{x}_{\mathrm{a}}^{\shortparallel}[n-1] & \cdots & \hat{x}_{\mathrm{a}}^{\shortparallel}[n-N+1] \end{bmatrix}^{\mathrm{T}}. \tag{9.15}$$

The length N of the feature vectors are here chosen to be 4.

The hidden Markov model is trained using the Baum-Welch re-estimation method with two states as an ergodic model. Since the alarm envelope signal was normalized, the two states are expected to settle near the feature vectors with values of all zeros and all ones. From here on the two expected states are denoted s^0 and s^1, respectively.

After training the hidden Markov model, a reference pattern for matching is calculated. The same training sequence is passed through the hidden Markov model: on a transition from state s^1 to s^0 at the time t, $\hat{\mathbf{x}}_M(t)$ is saved for the calculation of the reference pattern. The proposed method in figure 9.3 shows how the reference pattern is calculated. The value of M determines the length of the reference pattern, and is chosen to be $M = 2.5 \cdot T$ so that it contains three on-cycles and two off-cycles.

The function ScaledForward, given a hidden Markov model and an observation vector, returns the most likely current state. After processing the training sequence, the set $\mathbf{P}$ contains all the detected matching patterns, and the final reference pattern $\mathbf{p}$ is the average of all patterns in $\mathbf{P}$.

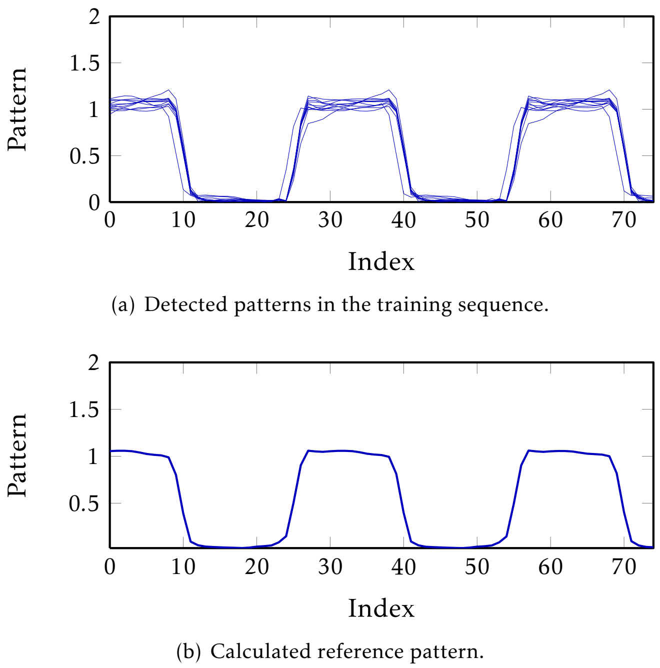
Figure 9.4 shows the detected patterns and the average pattern for the training sequence used in the evaluation section. The sequence is approximately 60 seconds long and the reverse alarm is active for the entire sequence. Figure 9.4(a) shows the set of all patterns detected during training overlapping in a single graph. Figure 9.4(b) shows the average of all detected patterns. This average is subsequently used as the reference pattern in the matching stage.

9.2.2 Detection and Matching

The procedure for detection and matching is similar to the one for the reference pattern calculation. The recorded sequence, after passing through the feature extraction and the normalization, is passed through the hidden Markov model. On a transition from state s^1 to s^0 at the time t, the reference pattern $\mathbf{p}$ is compared to $\hat{\mathbf{x}}_M(t)$. A threshold $\theta_{\mathbf{p}}$ is used to determine

```
 1: procedure ReferencePattern($\lambda, \hat{x}[n]$)
 2:     $\mathbf{P} \leftarrow \emptyset$
 3:     for $k = \{0, \ldots, K-1\}$ do
 4:         $s_k \leftarrow$ ScaledForward($\lambda, \mathbf{o}_k$)
 5:         if $s_k = \mathrm{s}^0$ and $s_k \neq s_{k-1}$ then
 6:             $\mathbf{P} \leftarrow \mathbf{P} \cup \{\hat{\mathbf{x}}_M(t)\}$
 7:         end if
 8:     end for
 9:     $\mathbf{p} \leftarrow \bar{\mathbf{P}}$
10:     return $\mathbf{p}$
11: end procedure
```

Figure 9.3: Proposed algorithm for calculating the reference pattern.



(a) Detected patterns in the training sequence.

(b) Calculated reference pattern.

Figure 9.4: Calculation of the alarm envelope signal reference pattern: (a) detected patterns in the training sequence, and (b) the reference pattern.

```
1: procedure EVALUATE($\lambda, \mathbf{p}, \hat{x}[n]$)
2:     for $k = \{0, \dots\}$ do
3:         $s_k \leftarrow$ SCALEDFORWARD($\lambda, \mathbf{o}(t)$)
4:         if $s_k = \mathrm{s}^0$ and $s_k \neq s_{k-1}$ and $\|\mathbf{p} - \hat{\mathbf{x}}_M[k]\| < \theta_{\mathbf{p}}$ then
5:             Alarm envelope signal detected
6:         end if
7:     end for
8: end procedure
```

Figure 9.5: Proposed method to determine the presence of an alarm signal.

whether the state transition should be attributed to noise or to the presence of a signal that resembles the reference pattern. Figure 9.5 shows the proposed algorithm for evaluating a signal in order to determine if and where a reverse alarm is present.

9.3 Evaluation

To evaluate the proposed method, real recordings of a reverse alarm are used. The reverse alarm used in the evaluation is a J. Auer ESM Panel Mount Buzzer. The frequency is 3900 Hz, the pulse period time T is 0.48 s and the duty cycle factor d is 0.5. The signals were sampled at 16 kHz and a Hanning window with a 50 % overlap is used in the framing, windowing and normalization stages. The prototype low-pass filter is designed with $K = 256$ and $P = 4$.

Figure 9.6 shows the recording, and the subsequent detection, of a reverse alarm in a noisy environment. The spectrogram of the recorded signal in figure 9.6(a) shows the reverse alarm at 3900 Hz during three time periods. The normalized subband envelope signal in figure 9.6(b) shows three temporal sections which clearly consist of a pattern similar to the calculated reference pattern. Finally, figure 9.6(c) shows the detected and estimated active-state of the reverse alarm.

All three temporal sections are detected, and there are neither false negative nor false positive detections. However, there are two notable delays, one at the beginning and one at the end of a temporal section where the reverse alarm is truly active. The reasons for these delays can be explained in the following ways:

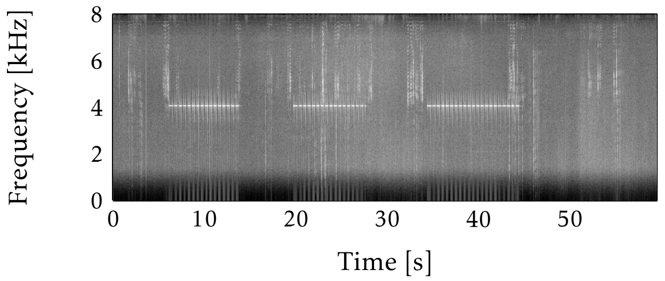
- The delay at the beginning of the temporal section is due to the length of the matching pattern. The longer the reference pattern, the longer the delay before a complete pattern match is achieved to decide that the alarm signal has started.

- The delay at the end of the temporal section is due to the heuristics determining whether the alarm is active or not between consecutive state transitions of the hidden Markov model. A pattern is only successfully matched when the hidden Markov model transitions into the state s^0, which happens every 0.48 s in this particular evaluation. In between these state transition times, the alarm must be assumed to still be active. To protect from a possible loss of a state transition, the heuristics used here allows an additional time where the alarm signal is assumed to be active without a state transition and a reference pattern match. State transition losses can be produced by an incorrectly determined state transition causing the reference pattern not to match, such losses may simply be the result of noisy measurements. The longer the additional time, the longer the delay before the heuristics decide that the alarm signal has ended.

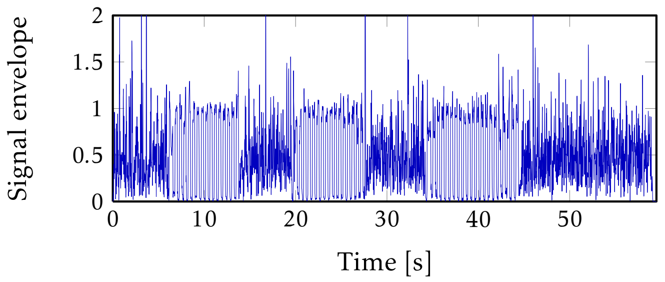
The length of the reference pattern and the additional time allowed by the heuristics is a tradeoff between detection delays and detection success. The threshold is a tradeoff between detection success and the false positive and false negative detection rates.

9.4 Conclusion

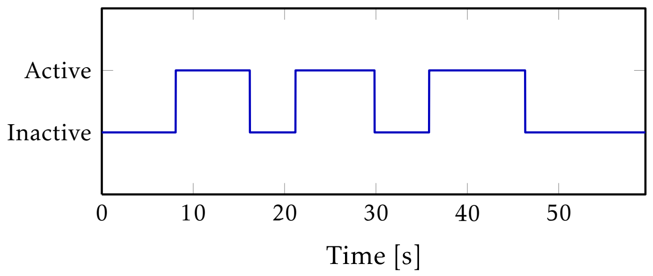
This chapter presented a framework able to detect a narrow-band alarm envelope signal. A general approach to subband envelope signal normalization, pattern generation and pattern matching was presented. The method was then evaluated for a common type of alarm signal consisting of a carrier modulated by a square pulse train alarm envelope signal. The method is suitable for execution in an embedded real-time environment, while the passband sampling filter and the matching pattern can be computed as a separate preparation stage.



(a) Spectrogram of a noisy recording with an occasional reverse alarm.



(b) Normalized subband envelope signal.



(c) Detected and estimated active-state of the reverse alarm.

Figure 9.6: Evaluation of the proposed method: (a) the spectrogram of a noisy recording of a reverser alarm, (b) the subband envelope signal, and (c) the detected and estimated active-state of the reverse alarm in the noisy recording.

Chapter 10

Tone Detection and Cancelation in the Modulated Envelope Domain

This chapter is based on the following article:

Mikael Swartling, Benny Sällberg, and Nedelko Grbić. Tone Detection and Cancelation in the Modulated Envelope Domain. *IEEE Signal Processing Letters*, December 2011. Submitted.

The problem of tone detection has been studied for many years in applications ranging from radar and sonar systems to dial tone multiple frequency (DTMF) detection in telecommunication systems. By canceling disturbing tones, significant speech enhancement can be accomplished in communications systems and hearing aid equipment. In the latter case, the cancelation provides an important protective shield towards further damaging the hearing.

Many tone detection methods rely on an energy estimate of a pre-processed signal. The signal may contain one or more narrow band tonal components embedded in a useful signal. By detecting energy above a certain threshold, a tone is assumed. Similarly, energy below the threshold is assumed to result from the useful signal. The pre-processing may consist of an overlapped FFT operation or of windowed periodograms [67, 68]. While these techniques circumvent problems inherent in adaptive notch filtering, they provide little or no separation of the measured energy between the tone(s) and the useful signal. In the case of two-tone signals with constant frequency separation as in DTMF, it has been shown that an energy operator can be used to increase accuracy when estimating the difference in frequency between the tones [69].

Adaptive notch filtering techniques rely on the fact that if the useful signal's correlation function drops more rapidly than the correlation function of the tone, an adaptive filter with delayed input can be used to subtract the tone [70, 71]. The adaptive notch filtering can provide much better performance than the energy-based methods when it comes to accurate frequency estimation and cancelation. Drawbacks encountered with adaptive notch filtering include instability problems, uncontrolled resonance effects, and that multiple simultaneously active tones are difficult to track.

Tone detection by zero-crossing measures relies on the fact that the distances in time between zero-crossings of a tonal signal are constant [72, 73]. The presence of a dominant tonal component in a subband can thus be detected by a low variance of zero-crossing distances, and the corresponding subband can be attenuated to cancel the tone.

The problem encountered in hearing aids includes both narrow band oscillation cancelation and the reduction of acoustic feedback [74, 75, 76]. Conventional techniques of echo and howling cancelation include identification of the acoustic feedback path and subtraction of the feedback signal. A major problem in this context is the correlation between the input to the echo and howling canceler and the desired response signal [76]. It has been shown that a subband implementation reduces these problems by allowing adaptation only in bands where oscillations normally occur [77].

This chapter proposes a new concept of tone detection which takes advantage of the benefits of the energy-based methods. The method comprises a pre-processing step that utilizes a complex valued FIR filterbank and that performs simple envelope tracking separately in each subband. Since a narrow-band signal with a constant amplitude will have a constant envelope when filtered by a complex valued band-pass filter, each tone will appear as a DC component in the envelope domain. Furthermore, it has been shown that speech signals do not contain any significant information below approximately 4 Hz in the envelope domain [78]. These facts allow for the use of a simple low-pass filtering of the envelope in the subbands where a tone is detected. The proposed structure is suitable to incorporate with existing subband echo and howling techniques. The main properties of the proposed method are:

- Any number of oscillation frequencies are detected and adaptively tracked for the purpose of acoustic subband feedback detection and cancelation.
- The bandwidth of the resulting notch filters is significantly smaller than the bandwidth of a single subband.
- The cancelation frequency of the notch filters is continuous, i.e. it is not restricted to discrete frequency points.

10.1 Signal Model

The aim of the proposed structure is to detect and cancel any narrow-band signal with a slowly varying amplitude, while passing a wide-band speech signal through the filtering process. Consider a narrow-band signal given by

$$s[n] = A\cos(\omega_c n - \phi), \quad -\pi \le \omega_c \le \pi \tag{10.1}$$

with an arbitrary phase factor ϕ that is filtered by a filter $h[n]$ with the Fourier-transform $H(\omega)$. The resulting filtered signal becomes

$$s_f[n] = \frac{A}{2}\int_{-\pi}^{\pi} H(\omega)\Big[\delta(\omega-\omega_c)\,\mathrm{e}^{-\mathrm{j}\phi} + \delta(\omega+\omega_c)\,\mathrm{e}^{\mathrm{j}\phi}\Big]\,\mathrm{e}^{\mathrm{j}\omega n}\,\mathrm{d}\omega \tag{10.2}$$

$$= \frac{A}{2}\,|H(\omega_c)|\,\mathrm{e}^{\mathrm{j}(\omega_c n - \phi + \angle H(\omega_c))} + \frac{A}{2}\,|H(-\omega_c)|\,\mathrm{e}^{-\mathrm{j}(\omega_c n - \phi + \angle H(-\omega_c))}. \tag{10.3}$$

If the filter $H(\omega)$ is chosen such that

$$\begin{aligned} |H(\omega_c)| &= 1 \\ |H(-\omega_c)| &= 0 \end{aligned} \tag{10.4}$$

the resulting magnitude envelope of $s_f[n]$ will be constant for all n, and independent of the phase factors ϕ and $\angle H(\pm\omega_c)$. A requirement on the filter $H(\omega)$ is that the impulse response $h[n]$ must be complex valued, since a real valued impulse response results in a symmetric Fourier transform magnitude around the zero-frequency.

A modulated uniform DFT filterbank is used to decompose the signal into several narrow-band signals. The condition given in (10.4) can only hold for a finite number of frequency points when the filterbank is restricted to finite length filters. Following the methodology given by de Haan et al. [79, 80, 81], a good approximation of the filterbank can be found that minimizes leakage from the side lobes and imaging effects across the entire frequency range.

10.2 Subband Algorithm

The subband signals are independently processed in order to detect any signal component with a constant, or slowly varying, envelope. Once such a signal is detected, a simple high-pass filter is used to filter the envelope and the phase of the signal. It should be noted that an exact estimate of each tone's amplitude and frequency is not of interest in this application. Rather, the concern here is the detection and removal of multiple tones without causing severe distortion to the useful signal.

10.2.1 Detection Method

The input signal $x[n]$, which contains the useful speech signal and possibly several tonal disturbances, is first decomposed by the analysis filterbank, creating a set of K subband signals. The detection process is based on tracking a subband weight to each subband envelope's DC component according to

$$w_{k,\text{opt}} = \arg\min_{w_k} \sum_{n=-\infty}^{\infty} \lambda^n \left(\mathbf{h}_\text{e}^\text{H} \mathbf{x}_k^{\shortparallel}[n] - w_k\right)^2 \tag{10.5}$$

where w_k is the weight of the subband k, $\mathbf{h}_\text{e}$ is the subband envelope low-pass filter used to exclude the speech frequency range from the optimization process, $\mathbf{x}_k^{\shortparallel}[n]$ is the subband envelop signal vector and $0 < \lambda < 1$ is an exponential weighting factor. The subband envelope signal vector is

$$\mathbf{x}_k^{\shortparallel}[n] = \Big[|x_k[n]| \quad |x_k[n-1]| \quad \cdots \quad |x_k[n-L+1]|\Big]^\text{T} \tag{10.6}$$

where $x_k[n]$ is the subband signal from the subband k. The optimization criterion in (10.5) provides the DC component of the signal's envelope and it is theoretically given by

$$w_{k,\text{opt}} = A\,|H(\omega_c)| \tag{10.7}$$

if the subband signal contains the signal given in (10.1). The weight will be (close to) zero if the subband signal only contains a non-stationary useful signal, e.g. a speech signal. The decision is then simply based on a threshold in the following way:

$$\begin{cases} w_{k,\text{opt}} > \alpha & \text{a tone is detected in subband } k \\ w_{k,\text{opt}} \le \alpha & \text{a tone is not detected in subband } k. \end{cases} \tag{10.8}$$

Here, α is a small constant chosen such that tonal amplitudes exceeding the value are detected and canceled.

The practical optimization of (10.5) is performed using the weighted recursive least squares algorithm [82]. This enables a sample-by-sample update of the envelope tracking. Since the optimization is restricted to a single individual parameter in each subband, no matrix invertibility problems will

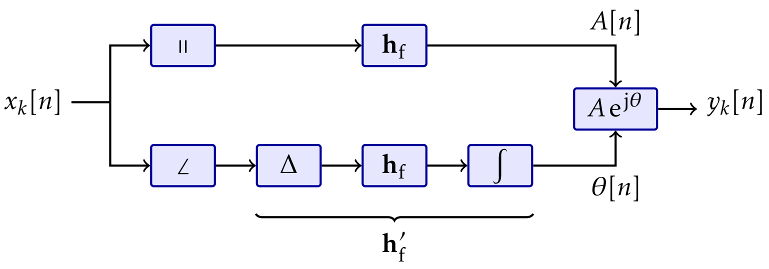


Figure 10.1: The filtering process to cancel the envelope and phase of a tone.

occur. Consequently, the computational complexity of the method is very modest.

10.2.2 Filtering Algorithm

Once a tone is detected, a simple high-pass filtering is performed to remove the DC component from the envelope and the phase of the signal. The resulting signal is then reconstructed by a synthesis filterbank to form a full band time-domain output signal. The output in each subband is given by

$$y_k[n] = \begin{cases} \mathbf{h}_\mathrm{f}^\mathrm{H}\mathbf{x}_k^{|\cdot|}[n] \cdot \exp\left\{\mathrm{j}\mathbf{h}_\mathrm{f}'^\mathrm{H}\mathbf{x}_k^{\angle}[n]\right\} & w_{k,\mathrm{opt}} > \alpha \\ x_k[n] & \text{otherwise} \end{cases} \tag{10.9}$$

where $\mathbf{h}_\mathrm{f}$ is the subband envelope high-pass filter used to remove the slowly varying components of the envelope and the phase of the signal, and $\mathbf{x}_k^{\angle}[n]$ is the subband phase signal vector. The subband phase signal vector is

$$\mathbf{x}_k^{\angle}[n] = \begin{bmatrix} \angle x_k[n] & \angle x_k[n-1] & \cdots & \angle x_k[n-L+1] \end{bmatrix}^\mathrm{T}. \tag{10.10}$$

Since the phase is constant in its derivative, the phase filter $\mathbf{h}_\mathrm{f}'$ is actually a cascade of three filters: a differentiator, the actual envelope high-pass filter $\mathbf{h}_\mathrm{f}$ and an integrator. The filtering process is visualized in figure 10.1.

10.3 Evaluation

The evaluation includes a speech signal disturbed by a number of tones, both pure and amplitude modulated, starting at different times. Two tones are

Table 10.1: Properties of the three tones used in the evaluation.

Frequency	Start time	Modulation
500 Hz	0.5 s	—
1015.625 Hz	1.0 s	—
1500 Hz	1.5 s	10 Hz

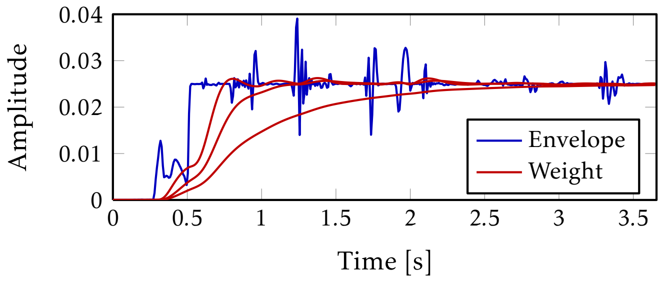
pure, located at both whole and fractional subband indices, and one tone is amplitude modulated. The setup illustrates several of the useful aspects of the proposed method: multiple tones can be detected and canceled, the number of tones can be dynamic over time, the tones can be of arbitrary frequencies, and the amplitudes need not be long-term stationary. Table 10.1 describes the three disturbing tones. The first and third tones are located at the center of a filterbank subband. The second tone is located in the middle between two filterbank subbands, at 1000 Hz plus one half subband width. The third tone is amplitude modulated to demonstrate the effect on a speech component with a rapidly varying envelope. A modulated DFT analysis filterbank with 512 subbands is used to process the resulting signal [8]. The prototype filter is designed such that there are four polyphase subband filter taps, and the oversampling ratio is four times [79]. The envelope low-pass filter $\mathbf{h}_e$ and the envelope high-pass filter $\mathbf{h}_f$ are both implemented as third-order Butterworth IIR filters with cut-off frequencies at 4 Hz.

The three tones are added to a sampled speech signal of approximately four seconds duration. Figure 10.2 shows the envelope signal and weights resulting from optimizing (10.5) using the weighted recursive least squares method with the weighting factor λ chosen as 0.75, 0.95 and 0.99, respectively. The corresponding equivalent integration times are 32 ms, 160 ms and 800 ms, respectively. Note that the amplitude modulation of the third tone at 1500 Hz is 10 Hz and is thus not tracked by the weights since it is within the assumed frequency range of the speech in the envelope domain. Based on the subband weights w_k, a decision according to (10.8) is made to determine the presence of a tone, and the subband signal is filtered according to (10.9). The output from each subband is used to reconstruct the equivalent full band signal using a modulated DFT synthesis filterbank.

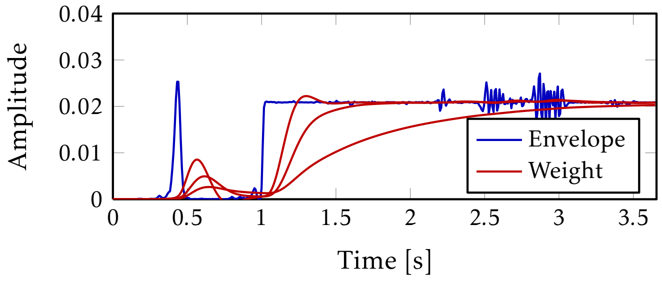
Figure 10.3(a) shows the time-frequency spectrogram of the disturbed signal, and figure 10.3(b) shows the spectrogram of the processed signal for the weighting factor λ chosen as 0.95. Note that the amplitude modulation component of the third tone is not canceled from the input signal, which is expected. Varying the integration time is a trade-off between the responsiveness of the tone detection and potential cancelation of short, near-tonal speech components. It should also be noted that even though the threshold α is chosen to be 0.001, which is significantly smaller than the contribution from speech to the decision statistic w_k in some time periods in some subbands, no significant distortion to the processed output signal can be noticed. This comes from the fact that even though the method occasionally assumes tones being present at times where there are no tones active, the filtering process of (10.9) only cancels signals that are close to the DC envelope.

10.4 Conclusion

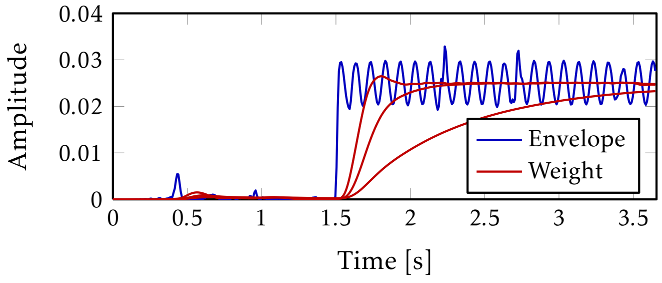
This chapter proposes a new subband method for the purpose of multiple tone detection and cancelation when the useful signal is a non-stationary speech signal. Tones are detected by envelope tracking performed individually in each subband. To isolate the useful signal, this is followed by a high-pass filtering in the envelope domain. The resulting system acts as multiple adaptive notch filters and is thus not in need of any reference signal.



(a) The envelope signal and weights for the subband at 500 Hz.

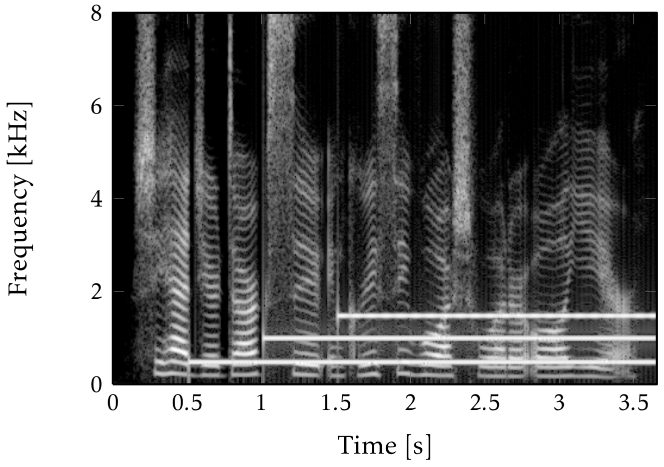


(b) The envelope signal and weights for the subband at 1000 Hz.

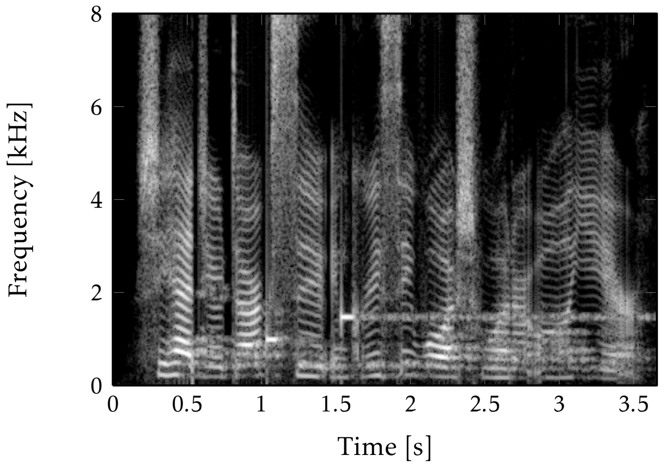


(c) The envelope signal and weights for the subband at 1500 Hz.

Figure 10.2: The envelope signal and the corresponding subband weight w_k over time for the weighting factors 0.75, 0.95 and 0.99. The subband envelope signal and the subband weight are shown for the subbands at 500 Hz, 1000 Hz and 1500 Hz. The tones in (a) and (b) are pure and have a constant amplitude envelope. The tone in (c) is amplitude modulated and have a time-varying amplitude envelope.



(a) Input speech signal disturbed by three tonal components.



(b) Processed output signal with $\lambda = 0.95$.

Figure 10.3: The spectrogram of (a) the input signal and (b) the processed signal with $\lambda = 0.95$.

Chapter 11

A Real-Time Audio Processing Framework for Matlab

A necessary step when developing algorithms for real-time signal processing is to evaluate these algorithms in a (quasi-) real-time environment. An implementation for the final target environment, for example a DSP platform, may be the best evaluation environment when evaluating computational performance. Here, sound quality and design issues can be evaluated properly in the correct environment. The drawback, however, is the need for a separate platform, and the necessity to implement the algorithm for this platform as well.

Another approach is a quasi-real-time environment, where an off-line processing environment simulates a real-time environment. This can be achieved in Matlab by, for example, framing a prerecorded input signal and passing individual blocks individually to a processing function. The advantage of this is that an algorithm can be designed and evaluated in a single environment, eliminating the need of porting and maintaining the algorithms on several platforms. However, such evaluations are usually limited to prerecorded sets of signals, or to a procedure where signals are first recorded and then processed.

An early attempt to make a real-time framework for a uniform environment utilized the Matlab Compiler toolbox to translate the Matlab scripts into a platform-native executable file. This translation incorporated the framework into the final application to provide the audio interface. The framework provided a compromise in which the same development environment was used, but the code had to be compiled into a separate executable application. The compilation and deployment process was cumbersome and not as easy as one would wish it could be, but the overall process was still a long way ahead of porting the algorithms to a second platform or working with prerecorded signals.

A second attempt to make the framework made use of another approach. Instead of moving the user code to an external framework, the framework was designed to be operational directly within the Matlab environment. As a result, a lightweight real-time signal processing framework for Matlab was developed. The framework operates entirely within the Matlab workspace and acts as a layer between the audio drivers and the user script for processing the input signals to the output signals. This allows for rapid algorithm development, evaluation and demonstration within a single development environment. The framework is intended to operate on computers having audio hardware supporting the Audio Stream Input/Output (ASIO) standard [83].

11.1 Design Approaches

When designing the internal structure of the new framework, three design and implementation parts had to be considered:

- the framework core and its interface towards the audio driver,
- the framework core and its interface towards Matlab, and
- the clients and their interface towards the framework.

Typically, there are also three different approaches to designing the interface exposed to the users. The interface can thus be designed as

- an asynchronous callback-based interface,
- a blocking poll-based interface, or
- a non-blocking poll-based interface.

The three approaches all have their advantages and disadvantages in various situations. For example, the advantage of the asynchronous callback-based interface is that the application is automatically notified when an event occurs and, in contrast to the poll-based approaches, the application does not have to manually query if an event has occurred. The disadvantage is that the asynchronous callback must typically come from a secondary thread to be able to interrupt and notify the application. This can cause synchronization issues when transferring the notification from the notifying thread to the application thread.

When using a poll-based interface, it is the application that is responsible for querying whether an event has occurred. The synchronization issues that complicate the asynchronous design approach are thus eliminated. However, the responsibility to check for events is instead transferred to the application. Furthermore, an important difference between the blocking and non-blocking approaches is if the querying for events will block the application and wait for the event to actually occur before returning, or if the query is returned immediately with a status that indicates whether an event has occurred or not.

The same interface design is not necessarily chosen for all three framework parts. Some parts impose restrictions on which interface design approach that has to be taken, while others allow some freedom of choice. For example, the ASIO standard exclusively decides how the framework core has to behave when interfacing with the audio driver, while at the same time giving full freedom when designing the interface between the clients, their code, and the framework.

11.1.1 The Framework and the Audio Driver

The design of the framework core and its interface towards the audio driver was limited by the ASIO standard. The ASIO standard provides an asynchronous callback-based interface for recording and playback, where the application (the framework core in this case) operates independently of the audio driver. The audio driver then notifies the application by a callback function when a new input sample buffer is available to, and when a new output sample buffer is required from, the application. Since input and output buffers are synchronized, a single common notification is used for both input and output notifications.

11.1.2 The Framework and the Matlab Interface

When designing the interface towards the audio driver, there are two options that need to be considered when handling the asynchronous callback. Either processing is performed within the asynchronous callback function called by the audio driver, or the callback is only responsible for forwarding the notification to the application thread. The limiting factor here is Matlab, which requires the application to interface with it using Matlab's own thread. Processing cannot, therefore, be performed within the audio driver's callback, and it must be forwarded to the application thread. The framework is then responsible for assembling the input signal matrix and make the call to the user-defined script where the processing is performed. Likewise, the returned signal matrix is disassembled and forwarded to the audio driver thread as the next output signal block.

The two main issues that must be handled are:

- synchronization of the callback thread and the application thread to safely deliver the data between the two, and
- conversion of the sample data to and from the signal matrices.

The synchronization issues can be handled by double buffering and the use of proper synchronization mechanisms provided by the operating system.

Double buffering is an approach where two pairs of buffers are utilized for recording and playback. One pair is for recording or playback, and the other pair is for processing. The benefit of this approach is that buffers are not accessed by both the audio hardware and the processing functions at the same time, a process where the two risk corrupting each other's sample data. When the audio hardware is recording or playing back, samples are written to, or read from, one buffer pair. At the same time, the application processes the other buffer pair. The ASIO standard already utilizes a double buffer mechanism, so the framework only has to focus on thread synchronization and the management of the signal matrices.

Synchronization of the callback and application threads is done with the synchronization mechanisms provided by the operating system. The operating system typically provides a variety of ways to approach synchronization. A mutex object can be used to get exclusive access to a resource (for example, the sample buffers), and an event object can be used to signal events between threads (for example, a buffer is ready for processing, or the framework must stop processing and exit).

Sample data conversion to and from the signal matrices is necessary since the sample buffers are not compatible with the layout of Matlab matrices. The signal matrices require each column to be stored in contiguous memory, while the sample buffers are allocated separately for each channel by the audio driver. Also, the binary data format depends on the audio hardware and must be converted to a uniform format for processing in Matlab. The ASIO protocol has a predefined set of formats that the audio hardware may support. The protocol allow many combinations of 16 to 64 bits per sample, possibly packed within a larger type, using integer or floating point representation, and stored in MSB or LSB formats. The framework converts between these formats and the 64-bit floating point value used by default in Matlab. The samples are converted such that the sample value 0 is mapped to 0, and the smallest and the largest sample values are mapped to ±1. Thus, in the signal matrices, the range −1 to 1 covers the full dynamic range of the audio hardware.

11.1.3 The Interface to the Framework

The two major approaches to consider when the user script interfaces with the framework are callback-based or poll-based. Similar to the audio driver, the callback approach allows the framework to call the script when a signal matrix is ready for processing, and it also has the responsibility of controlling when processing is performed on the framework. In this approach, the framework is located between the Matlab workspace and the user script, controlling the program flow to and from the user script.

In a poll-based approach, the user script is responsible for querying the framework whether processing is necessary or not, either by blocking or non-blocking queries. A blocking poll could be a function call that returns only when processing is necessary, while a non-blocking poll could be a function that returns the processing status. This status could be either true or false if processing is necessary or not, or it may consist of the reading of a status flag. The non-blocking poll approach is useful if other tasks are to be performed in parallel with the signal processing. In the presence of a threading environment (which is not present in Matlab), threads can be used to separate tasks instead. In this approach, the user script is located above the Matlab workspace, and the framework is executed in parallel with the user script, communicating between the two via the polling mechanism.

The advantage of a callback-based approach is the control the framework has of its resources and the status of the user script. In an environment like Matlab, it is not easy for a poll-based approach to detect situations when, for example, the user exits the script without stopping the framework because the user simply forgot to issue a stop-command, or there was an error in the script which caused Matlab to break the execution of the script. In a C++ environment, for example, this can easily be achieved with automatic resource management. With a poll-based approach, the user is in control of the framework. This is problematic since the framework must manage its own resources despite the fact that it is not in control of its own state with respect to the user code.

In the callback-based approach, on the other hand, the framework is in control of the script. The framework can safely allocate and release resources, and monitor the calls to the user script. There is no way for the script to leak the framework's resources, and the framework only has to assume that the framework itself is correct. However, the main disadvantage with a callback-based approach is the communication between the caller and the script. In the poll-based approach, parameters can be passed directly to the script from the caller as function parameters or simply by having the processing steps in the same scope as the main application and defining parameters and states as normal variables. This can be done because the framework is not interfering between the caller and the script. In the callback-based approach, these parameters and states need to be accessible through the framework layer. This, however, is not a problem in Matlab, where it is possible to define global and persistent variables.

The approach used by the framework is the callback-based approach. The reason was that the advantages, as far as the framework itself is concerned, were in favor of the callback-based approach. There were, naturally, disadvantages, but these were easily bypassed by the scripts. In later releases of Matlab, however, new object-oriented programming features that void some of the disadvantages of the blocking poll-based approach have been introduced, allowing for a more convenient interface for the user.

11.1.4 The User Interface

So far, the design concerns discussed have been related to the software interfaces for the framework towards the audio hardware and Matlab, and to the interface towards the framework. However, the user of the application must also be able to interface with the framework. Since the framework is used for continuous processing, the user must be able to terminate the processing and end the framework. Two mechanisms are implemented:

- run the framework for a specified amount of time, and
- run the framework indefinitely, and have the user manually terminate the processing.

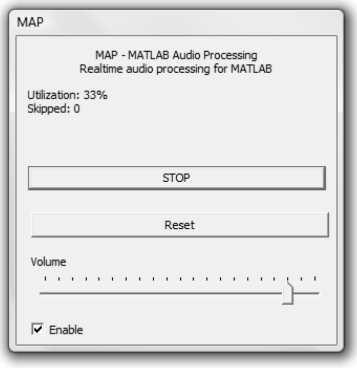
The first mechanism, to run the framework for a specified amount of time, can be used for batch processing of short sequences. The user has an exact control over the processing time for each batch, which can be useful when, for example, storing input, output or intermediate signals for comparisons between batches. Since each batch is identical lengthwise, and playback of external signals can be performed using the framework at the same time as the processing, batches can be considered to be synchronized in time for easy comparison between batches.

The second mechanism, to have the user manually terminate the processing, requires user interaction. For that reason, the framework presents a control panel whenever the framework is active, where the user can get statistical feedback and the process can also be terminated. Figure 11.1 shows an example of the control panel. From the control panel, statistical information about utilization and dropped blocks can be viewed, and the processing can be terminated. The utilization is the percentage of the time spent calling Matlab and the user-defined processing function. When the user script is taking too long to process the input data such that a new block of samples is ready before the processing of the previous block is finished, blocks are lost (the input samples are dropped and zero-samples are output when new ones are not ready in time). This happens when the utilization increases above 100%, and the control panel reflects the number of dropped blocks.

The framework also provides the ability to add user interface controls to the control panel which the user can use to, for example, control internal algorithm parameters. The framework provides support to add slide bars, check boxes and buttons. The slide bars provide a continuous parameter over a range, for example a volume slide. The check boxes provide a binary parameter, for example by turning algorithms on and off. The buttons provide a one-time set parameter which is set for one frame after the button is pressed. This can be a button to reset an algorithm to its initial state.

11.2 Implementation

A schematic view of the framework is shown in figure 11.2. The framework is initiated and started from the Matlab environment, and is then responsible for initiating the audio drivers for input and output signals, as well as for using the Matlab environment to process the signals with a user defined script. Since the framework is operating entirely within the Matlab environment, the user script has access to everything provided by Matlab during processing, including toolboxes and graph plotting.


MAP
MAP - MATLAB Audio Processing
Realtime audio processing for MATLAB
STOP
Reset
Volume
Enable

Utilization: 33%
Skipped: 0

Figure 11.1: An example of the control panel for the real-time framework.

11.2.1 The Audio Driver

The audio driver protocol utilized by the framework to access sound hardware is the ASIO standard. The ASIO standard was developed by Steinberg Media Technologies to provide applications with a low-latency, high quality, multichannel interface for sound recording and playback equipment [83]. The availability of ASIO drivers is generally good for high-end sound equipments, but may be limited in the lower-end range. With the existence of wrapper drivers that provide a generic ASIO driver for standard Windows audio drivers, there is no need for the framework to include support for other driver protocols [84].

11.2.2 The Framework Interface

The framework operates on a block basis, where individual non-overlapping blocks of samples, denoted as signal vectors, are collected, processed by the user defined scripts and played back. When recording or playing back multiple channels, the input or output signal vectors become matrices which columns each represents one signal vector of the input or the output channels.

Given an input signal vector

$$\mathbf{x}_m[n] = \begin{bmatrix} x_m[n] & x_m[n-1] & \cdots & x_m[n-L-1] \end{bmatrix}^{\mathrm{T}} \tag{11.1}$$

where L is the block size, the input signal matrix is

$$\mathbf{X}[k] = \begin{bmatrix}\mathbf{x}_1[kL] & \mathbf{x}_2[kL] & \cdots & \mathbf{x}_M[kL]\end{bmatrix} \tag{11.2}$$

where k is the block index. The output signal matrix $\mathbf{Y}[k]$ is defined accordingly, consisting of N output signal vectors of the length L. The user defined scripts, represented by $f(\cdot)$, then process the input signal matrix as

$$\mathbf{Y}[k] = f(\mathbf{X}[k]). \tag{11.3}$$

In a trivial script, the function $f(\cdot)$ may depend only on the input. For a non-trivial script, it is often necessary to keep states between function calls. Consider, for example, an FIR filter or an AR-process for short or long-term averaging of a signal parameter. The FIR filter requires samples from the previous signal matrix to properly initiate or continue the filtering of the current signal matrix, and the AR-process updates the estimate of the average value based on the previous average value. Internal states within the scripts can be easily handled in Matlab using global variables, persistent variables, or local functions and closures.

The signal recording parameters, the sampling rate f_s, the number of input channels M, the number of output channels N and the block size L, are only limited by the audio hardware. The framework does not impose any further limitations. The input matrix $\mathbf{X}[k]$ and the output matrix $\mathbf{Y}[k]$ are $L \times M$ and $L \times N$ real-valued matrices, respectively. Low-end audio hardware is usually limited to two input and output channels, sampling at up to 48 kHz. High-end multi-channel audio hardware may support eight or more input and output channels, with sampling rates of up to 96 kHz, or even as high as 192 kHz. The block rate at which the framework and the user defined scripts have to process the input signal matrices is f_s/L blocks per second.

Latency is an inherent problem in block-based signal processing where blocks must be buffered before processing can begin. Aside from the buffering performed by the audio hardware and its driver, the framework adds an additional buffer worth of latency. The audio hardware and its driver typically have one or two buffers, and with the additional buffer imposed by the framework, the latency is typically in the order of $(2 \text{ to } 3) \cdot L/f_s$ seconds. The latency is affected mostly by the block size L and the sample rate f_s. There can also be additional fixed latencies in the complete audio processing chain that cannot be controlled by the user, for example in the sampler circuits in the audio hardware. Thus, the latency can be limited by reducing the block

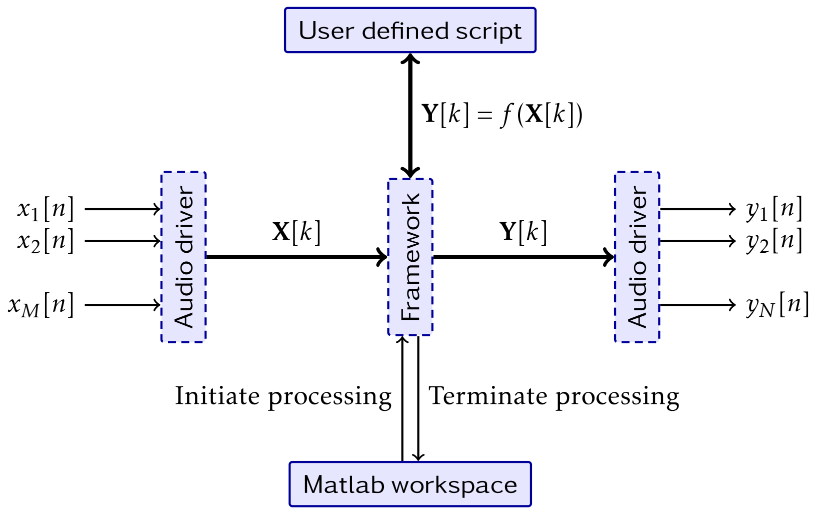


Figure 11.2: A schematic view of the framework for real-time signal processing in Matlab.

size or increasing the sample rate, both at the expense of an increased block processing rate. As the block rate increases, the static overhead per block (as introduced by the framework and by making calls into Matlab) increases in relation to the processing of the block data.

11.3 The Framework Core

The core of the framework is the part managing all interfaces including the interface from the core to the audio driver and Matlab, as well as the interface from the application to the core. The core is responsible for managing the audio driver, processing the audio data via the Matlab interface, and managing the control panel. Since the audio driver operates in a different thread, proper multi-thread synchronization is necessary to ensure a safe interaction between the audio driver and the framework core. It was decided that the control panel should be executed in a separate thread as well, which also requires proper multi-thread synchronization for safe interaction with the framework core.

11.3.1 Multi-Thread Synchronization

The core of the framework consists of a loop waiting for events to be signalled by the audio driver interface and the control panel. Events objects, which are synchronization objects typically provided by the operating system, provide a way to signal events to threads waiting for the event object to enter a set state. A thread can enter a waiting state to wait for the event object to be set. This provides a way to synchronize the execution of parts of code between threads. The events that can be triggered are events that process new sample data and stop the framework core loop.

To protect data from simultaneous access, especially when write operations are involved, a mutex object is used to protect and give a thread exclusive access to a resource. In the framework, the signal matrices that are passed to and from the user script must be accessed by both the framework core and the audio driver interface. Since the framework core and the audio driver interface reside in different threads and require read and write access to the signal matrices, they must be protected and access must be controlled to ensure that the audio driver does not interfere with the framework core, or vice versa. By creating a single mutex object shared by all threads and associating its access to a resource, a thread can request to lock the mutex object in order to gain exclusive access to the resource. Other threads can also request to lock the mutex, but, depending on how this access is requested, they either fail if the mutex is already locked or they will be forced to wait until the mutex is released and ready to be locked again.

The reasons for using the synchronization objects provided by the operating system instead of manual flag variables to indicate events or locked resources are:

- the handling of the synchronization object is thread-safe and atomic, and
- threads can idle while waiting for the synchronization object.

Thread-safe and atomic handling of the synchronization object is important to ensure proper handling. An event object must ensure that all waiting threads are signalled before being reset, and a mutex object must ensure that several threads cannot be granted exclusive access to a resource.

Consider, as an example, a manual mutex object implemented as a flag variable, which state indicates whether the mutex object is locked or unlocked. A typical implementation can be as in table 11.1, which demonstrates both the problem with non-thread safe mutex management, and two ways to check or wait for the mutex object. The first thread, Thread1, checks the locked or unlocked state of the mutex and takes an action depending on its state. If it is unlocked, no other thread has locked it and exclusive access is granted. If it was locked, some other action may be taken. The threads then lock the mutex and processing of the shared resource can commence. Once the processing is completed, the mutex is unlocked. The second thread, Thread2, takes an alternate approach. In a busy-wait loop, the threads wait until the resource is free, and is then granted access to it.

From a single thread's point of view, this mutex management seems to be reasonable. However, in a concurrent, multi-threaded environment, this manual approach is not safe. If Thread1 is executing and is interrupted by Thread2 between checking the state of the mutex object on line 2 and setting the locked state on line 3, both threads can be given exclusive access to the resource. What happens is that Thread1 determines that the mutex is unlocked, but is interrupted before marking it as locked. Meanwhile, Thread2 also determines that the mutex object is unlocked, and both threads proceed to mark it as locked and begin to access the resource. The problem is that comparing and setting the state must be an atomic operation that cannot be interrupted midway. A mutex object and its associated functions guarantee atomic operation.

A manual mutex object can be implemented provided that there is an atomic compare-and-exchange operation. However, this does not solve the second benefit of using the synchronization objects. In Thread2, the loop at line 11 will spin until the mutex can be locked which consumes processing power by needlessly looping and checking the state of the mutex. Alternatively, the mutex object and its associated functions can idle the thread, and the thread is not resumed until the mutex is unlocked by the other thread.

11.3.2 Initialization of the Core Loop

Before starting the core loop, the framework environment is set up. Due to the design choices made, the framework is entirely passive when not in use, and fully initialized and executed only when needed for processing. This gives the framework full control over its resources and when to release them.

The initialization phase consists of parsing the parameters passed to the framework (which audio driver to use and what sampling parameters to set, et cetera). The audio driver is initialized and prepared for sampling and play-

Table 11.1: Demonstration of a manual non-thread safe approach to mutex management, both as a direct flag check and a busy-wait loop.

```
Require: $m \leftarrow$ unlocked
1: procedure Thread1
2:     if $m$ = unlocked then
3:         $m \leftarrow$ locked
4:         Access resource exclusively.
5:         $m \leftarrow$ unlocked
6:     else
7:         Exclusive access was not granted.
8:     end if
9: end procedure
10: procedure Thread2
11:     while $m$ = locked do
12:         Wait
13:     end while
14:     $m \leftarrow$ locked
15:     Access resource exclusively.
16:     $m \leftarrow$ unlocked
17: end procedure
```

back, synchronization objects are initialized, signal matrices are prepared to be processed by the user script, and the control panel is created.

11.3.3 The Core Loop

The framework core loop is the center of processing, connecting all interfaces. The loop waits for either the process or the stop event to set, at which point the action is taken to either process recently recorded data and prepare the output for playback, or to abort the loop and exit the framework.

The process event is set in the audio driver's asynchronous callback which indicates that a block of samples is ready for processing. The framework core loop then proceeds to call the user interface to process the input signal matrix and to generate the output signal matrix. The processing requires access to resources that are shared among multiple threads (the signal matrices) and access to them is guarded by a mutex.

The mutex not only prevents multiple access to a resource, but it also serves to indicate whether the user script has processed the signal matrices in time. If the mutex is still locked upon the asynchronous callback from the audio driver, the user script has taken too long to process the signal matrices. This prevents the audio driver from reading the output matrix for playback and from storing the new samples in the input signal matrix. In this case, the new samples are discarded, and a zero-matrix acts as a no-output source, introducing a short period of silence in the output signal. Dropped blocks may introduce audible artifacts, but the source of the problem is the fact that the user script simply cannot cope with the rate at which blocks need to be processed.

If the script takes too long to process the signal matrices, there is not much that can be done; the processing just is not efficient enough. However, occasional spikes in processing time can also cause some blocks to be dropped, even though the script is sufficiently efficient. These spikes can be outside the control of both the user and the framework. This is sometimes the case if another application requires some processing time and the operating system's task scheduler decides to take processing time from the framework. The framework and the user script may be able to handle the amount of processing needed, but each block of signal data must also be processed before the next block. Thus, the response time is also important, not just the required processing time.

11.4 Usage

With the user interface, the application can query the framework for lists of available audio drivers, manage device and settings structures and initiate processing. The device and settings structures are collections of values that describe the audio driver and its capabilities (the number of available input and output channels, restrictions on block size, etc.), acquisition parameters (sample rate, block size, input and output channel configurations, etc.) and other settings (control panel controls, etc.). The processing is initiated by specifying which settings structure to use to setup the framework, and which function to call for each sample block.

11.4.1 The Device and Settings Structure

The device structure describes the audio driver and its capabilities, contains the number of available input and output channels, and the supported buffer sizes. The ASIO protocol only provides means to query if a specific sample rate is supported, but does not provide a list of supported sample rates. Therefore, the audio driver's capabilities do not include any information about the supported sample rates, as such information is not available.

```
>> device = mapdevice(0)
device =
                  KeyName: 'ASIO for TerraTec FW Series'
               DeviceName: 'ASIO for TerraTec FW Series'
         NumInputChannels: 10
        NumOutputChannels: 10
      BufferSizePreferred: 768
            BufferSizeMin: 768
            BufferSizeMax: 768
    BufferSizeGranularity: 768
```

The argument can either specify the name of a device or the index of the device in the list of devices. Calling the function without argument selects the first device.

From the device structure, a settings structure is created. The settings structure contains the input and output channel configurations, and the desired sample rate and buffer size. The channel configurations and the buffer size are validated with the device structure.

```
>> settings = mapsettings(device, 1:4, 1:4, 48000, 768)
settings =
            Device: [1x1 struct]
     InputChannels: [1 2 3 4]
    OutputChannels: [1 2 3 4]
        SampleRate: 48000
        BufferSize: 768
```

The channel configuration is such that the *n*th signal vector of the input signal matrix, `X(:, n)`, corresponds to the input channel `InputChannels(n)` from the audio hardware. The output channel configuration is defined in accordance with `Y(:, n)` and corresponds with the output channel `OutputChannels(n)`.

11.4.2 Initiating Processing

The processing is initiated by specifying what settings structure to use to setup the framework, and what function to call for each sample block.

```
>> device = mapdevice(0);
>> settings = mapsettings(device, 1:4, 1:4, 48000, 768);
>> mapstart(settings, @(x) x);
```

The example uses an anonymous function which returns its input. Thus, the processing function simply copies the input to the output without any processing and creating a pass-through for the channels 1 to 4 of the audio device.

Internal states for the callback function can be handled by:

- global variables that can be accessed across function borders,
- persistent variables that remain between function calls, and
- a local callback function and closures.

Variables can be marked as global in Matlab using the `global` keyword, and can then be accessed from anywhere. The application can declare state variables as global and then access them from the callback function. The global variables retain their values between calls and can be used to store states between blocks.

```
function maptest
    device = mapdevice(0);
    settings = mapsettings(device, 1:4, 1:4, 48000, 768);
    global h z;
    h = fir1(10, 0.25);
    z = [];
    mapstart(settings, @callback);
    clear global h z;
end

function Y = callback(X)
    global h z;
    [Y, z] = filter(h, 1, X, z);
end
```

The filter h and the filter states z are marked as global variables, and can then be accessed from the callback by introducing the global variables into their scope. Persistent variables, marked by the persistent keyword, work similarly, but are local only to the function where they are defined.

The callback function can also be defined as a local function, and can then access variables from within the outer function's scope directly.

```
function maptest
    device = mapdevice(0);
    settings = mapsettings(device, 1:4, 1:4, 48000, 768);
    h = fir1(10, 0.25);
    z = [];
    mapstart(settings, @callback);

    function Y = callback(X)
        [Y, z] = filter(h, 1, X, z);
    end
end
```

The callback function is here defined as a local function within the outer function, and can access the outer function's variables. The filter h and the filter states z can be accessed directly from the application's scope.

11.5 Performance Evaluation

The purpose of this performance evaluation section is to give a brief description of what can be done in real-time within the framework. A dedicated application written in a lower-level language like C or C++ for the native hardware platform environment may be orders of magnitude more efficient than scripts running within the Matlab environment. However, when taking the actual platforms into account, the performance difference may not be so obvious. If the efficient hardware-native code is to be executed on a small, embedded, low-power digital signal processor, the sheer computing power of a modern desktop computer running the Matlab environment and the framework will compensate for the reduced efficiency of the code.

11.5.1 Results

Two algorithms are implemented and run in real-time in Matlab with the framework. The algorithms are a tone cancelation method, and the SRP-PHAT source localization method using a linear sensor array. The tone cancelation algorithm processes a single input channel, and the output is played back for listening through a speaker or headphones. This is the typical and intended setup for the real-time framework with a set of input channels, a processing stage, and a set of output channels. The source localization algorithm, on the other hand, has to visually present the estimated direction of arrival instead, and there are no output signals produced. Fortunately, since the scripts are running entirely within the Matlab environment, the complete set of functions for drawing graphics is available. The only drawback is that the drawing performance in Matlab is poor in comparison to the speed the framework needs to maintain. As a comparison, when plotting a fairly simple graph, each callback adds an additional 50 to 60 percentage units to tables 11.2 with the setup in this evaluation.

Table 11.2 shows the CPU usage of the tone cancelation method and the source localization algorithm (excluding visualization). A sample rate of 32 kHz and a filterbank with the number of subbands M and the oversampling ratio O as described in the table, and with a fixed polyphase subband filter length N of 4 taps, are used. The decimation factor $D = M/O$ is the factor by which the sample rate of the filterbank subband signals can be reduced, and this thus defines the rate at which the subband signals must be processed. At each time instant of the filterbank subband sample period, the filterbank produces a vector containing a sample of each subband signal, which can be processed in parallel.

Varying the number of subbands and the oversampling ratio shows some interesting properties with regard to the performance in Matlab:

- For a constant number of subbands M, the utilization is increased at a super-linear rate with the increased subband sample rate as the oversampling factor O increases. In this case, a constant amount of data needs to be processed per subband sample period, but the subband sample period is decreased. Thus, the total required processing increases at a linear rate, but the time to perform it increases at a super-linear rate.

- For a constant decimation factor D (the constant decimation factor can be read from the table along the top-left to bottom-right diagonals), the utilization is increased at a sub-linear rate with the increased number of subbands M. In this case, the subband sample rate is constant, but the amount of data that needs to be processed per subband are increased. Thus, the total required processing increases linearly, but the time to perform it increases at a sub-linear rate.

- For a constant oversampling ratio O, the utilization is decreased as the number of subbands M increases. In this case, a constant total amount of data needs to be processed. Thus, the total required processing is constant, but the time to perform it decreases as the size of the blocks of data increase.

In the first two cases, the utilization, or equivalently the time taken to perform the computations, increases at super or sublinear rates in comparison to the required amount of processing. The difference between the two cases is what is constant and what is changing. In the first case, smaller blocks (number of subbands) are processed more often (the subband sample period), and in the second case, larger blocks are processed less often. The third case shows how a constant amount of data takes different times to process, with lower times for larger blocks at a lower processing rate. This clearly shows how Matlab is more suitable for vector processing than sequential processing. Even a relatively simple algorithm, as the tone canceler on a modern desktop computer, still requires a significant CPU utilization. Computational performance can be gained by changing the filterbank parameters, but that also has side effects in that audio quality, delay and general algorithm performance may change.

Table 11.2(b) shows similar results with regard to performance, but they are not as prominent as in table 11.2(a), and the changes in utilization is at a more linear rate. An explanation for this is that the source localization algorithm is computationally more expensive, making the effects of any overhead in Matlab less prominent.

11.6 Conclusion

A lightweight framework for real-time multichannel audio signal processing within Matlab was successfully developed. The framework makes the development, evaluation and demonstration of real-time signal processing algorithms easy and uniform to maintain, as there is only a single platform. Since its development, the framework has proved to be a useful tool. It has been used in research, demonstrations and in both introductory and research-level courses on signal processing.

Table 11.2: Evaluation of the CPU utilization when running different algorithms within Matlab and the real-time framework.

(a) Tone cancelation algorithm.

Subbands	Oversampling		
	2	4	8
64	13 %	30 %	78 %
128	7 %	17 %	44 %
256	4 %	11 %	26 %
512	3 %	7 %	16 %

(b) Source localization algorithm.

Subbands	Oversampling		
	2	4	8
128	45 %	94 %	196 %
256	24 %	53 %	111 %
512	16 %	31 %	63 %
1024	10 %	22 %	45 %

Three possible design choices were initially considered when designing the interfaces between various parts of the framework. The first choice was the interface towards the audio driver. Because of the design of the ASIO driver protocol, there was not much room for any other choice as far as the framework was concerned. The ASIO driver protocol provided a callback-based approach only. The second choice was the interface towards the Matlab environment. Here, too, no choice needed to be made, as the Matlab environment already had its requirements. The signal matrices had to be properly constructed in the right format and from the right thread to work with Matlab. For the third design choice, the user interface towards the framework, there were multiple approaches with their own advantages and disadvantages that had to be considered. A callback-based approach for the user script was finally selected. Both the callback-based and the poll-based approaches had their advantages. The disadvantages, however, were easily circumvented in the callback-based approach, setting the final design choice.

The framework is not intended as a target environment for a final product, but an environment where algorithm development, evaluation and demonstrations can be made in a single unified environment. The environment provided by Matlab and by the framework provides a sufficient replacement for evaluation and demonstration on secondary platforms in terms of performance compared to embedded digital signal processor platforms.

Appendix A

The Least Squares Point-to-Line Distance Intersection Point

The minimum distance d between the point $\mathbf{s}$ and a point along the bearing line originating at the point $\mathbf{p}$ and with the unit-length direction vector $\mathbf{v}$ is

$$d(\mathbf{s},\mathbf{p},\mathbf{v}) = \left\|(\mathbf{s}-\mathbf{p}) - \mathbf{v}\mathbf{v}^{\mathrm{T}}(\mathbf{s}-\mathbf{p})\right\| \tag{A.1}$$

$$= \|\mathbf{P}(\mathbf{s}-\mathbf{p})\|. \tag{A.2}$$

Here

$$\mathbf{P} = \mathbf{I} - \mathbf{v}\mathbf{v}^{\mathrm{T}} \tag{A.3}$$

is the perpendicular projection matrix of a projection onto the vector $\mathbf{v}$. Given an arbitrary number of bearing lines, the least squares estimate of the intersection point can be defined as the point that minimizes the distance to all bearing lines. A least squares cost function is defined as

$$J(\mathbf{s}) = \sum_n \|\mathbf{P}_n(\mathbf{s}-\mathbf{p}_n)\|^2 \tag{A.4}$$

$$= \sum_n (\mathbf{s}-\mathbf{p}_n)^{\mathrm{T}}\mathbf{P}_n^{\mathrm{T}}\mathbf{P}_n(\mathbf{s}-\mathbf{p}_n) \tag{A.5}$$

$$= \mathbf{s}^{\mathrm{T}}\left[\sum_n \mathbf{P}_n^{\mathrm{T}}\mathbf{P}_n\right]\mathbf{s} - 2\mathbf{s}^{\mathrm{T}}\left[\sum_n \mathbf{P}_n^{\mathrm{T}}\mathbf{P}_n\mathbf{p}_n\right] + \sum_n \mathbf{p}_n^{\mathrm{T}}\mathbf{P}_n^{\mathrm{T}}\mathbf{P}_n\mathbf{p}_n. \tag{A.6}$$

Optimization of the cost function is done by finding the extreme point via the first partial derivative with respect to $\mathbf{s}$ so that

$$\frac{\partial}{\partial\mathbf{s}} J(\mathbf{s}) = 2\left[\sum_n \mathbf{P}_n^{\mathrm{T}}\mathbf{P}_n\right]\mathbf{s} - 2\left[\sum_n \mathbf{P}_n^{\mathrm{T}}\mathbf{P}_n\mathbf{p}\right] = 0. \tag{A.7}$$

This yields the solution

$$\hat{\mathbf{s}} = \left[\sum_n \mathbf{P}_n^{\mathrm{T}} \mathbf{P}_n\right]^{-1} \left[\sum_n \mathbf{P}_n^{\mathrm{T}} \mathbf{P}_n \mathbf{p}_n\right]. \tag{A.8}$$

Furthermore, the product $\mathbf{P}^{\mathrm{T}}\mathbf{P}$ can be simplified as

$$\begin{aligned} \mathbf{P}^{\mathrm{T}}\mathbf{P} &= (\mathbf{I} - \mathbf{v}\mathbf{v}^{\mathrm{T}})^{\mathrm{T}}(\mathbf{I} - \mathbf{v}\mathbf{v}^{\mathrm{T}}) && \text{(A.9)} \\ &= \mathbf{I} - \mathbf{v}\mathbf{v}^{\mathrm{T}} - \mathbf{v}\mathbf{v}^{\mathrm{T}} + \mathbf{v}\mathbf{v}^{\mathrm{T}}\mathbf{v}\mathbf{v}^{\mathrm{T}} && \text{(A.10)} \end{aligned}$$

and since $\mathbf{v}^{\mathrm{T}}\mathbf{v} = 1$, it can be simplified to

$$\begin{aligned} \mathbf{P}^{\mathrm{T}}\mathbf{P} &= \mathbf{I} - \mathbf{v}\mathbf{v}^{\mathrm{T}} && \text{(A.11)} \\ &= \mathbf{P}. && \text{(A.12)} \end{aligned}$$

The least squares point-to-line distance solution can be expressed as

$$\hat{\mathbf{s}} = \left[\sum_n \left(\mathbf{I} - \mathbf{v}_n \mathbf{v}_n^{\mathrm{T}}\right)\right]^{-1} \left[\sum_n \left(\mathbf{I} - \mathbf{v}_n \mathbf{v}_n^{\mathrm{T}}\right) \mathbf{p}_n\right]. \tag{A.13}$$

Appendix B

Optimization of Sum of Sinc Function Approximations

The objective is to find an analytical solution for

$$\hat{\tau} = \arg\max_{\tau} \sum_{\{p,q\}} \operatorname{sinc}\left[\tau\,(q-p) - \hat{\tau}_{p,q}\right] \tag{B.1}$$

where $\{p,q\} \in \mathbf{P}$, given the second-order polynomial expansion of the sinc function

$$\operatorname{sinc}(\tau) \approx 1 - \frac{\pi^2}{6} \cdot \tau^2. \tag{B.2}$$

The sinc function is expanded around $\tau = 0$, so it is assumed that $\hat{\tau}_{p,q} \approx \tau$. The optimization is thus to find the maximum of a sum of second-order polynomial functions:

$$\begin{aligned} \hat{\tau} &\approx \arg\max_{\tau} \sum_{\{p,q\}} 1 - \frac{\pi^2}{6}\left[\tau\,(q-p) - \hat{\tau}_{p,q}\right]^2 && \text{(B.3)} \\ &= \arg\max_{\tau} f(\tau). && \text{(B.4)} \end{aligned}$$

and the extreme point is found by solving for τ in

$$\frac{\mathrm{d}}{\mathrm{d}\tau} f(\tau) = 0. \tag{B.5}$$

Differentiation yields

$$\frac{\mathrm{d}}{\mathrm{d}\tau} f(\tau) = \sum_{\{p,q\}} -\frac{2\pi^2}{6} \cdot \left[\tau(q-p) - \hat{\tau}_{p,q}\right](q-p) \tag{B.6}$$

$$= -\frac{\pi^2}{3} \sum_{\{p,q\}} \left[\tau(q-p)^2 - \hat{\tau}_{p,q}(q-p)\right] \tag{B.7}$$

$$= -\frac{\pi^2}{3} \sum_{\{p,q\}} \tau(q-p)^2 + \frac{\pi^2}{3} \sum_{\{p,q\}} \hat{\tau}_{p,q}(q-p). \tag{B.8}$$

Finally, solving for τ yields the location of the extreme point:

$$\tau \sum_{\{p,q\}} (q-p)^2 = \sum_{\{p,q\}} \hat{\tau}_{p,q}(q-p) \tag{B.9}$$

$$\tau = \frac{1}{W_{\mathbf{P}}} \sum_{\{p,q\}} \hat{\tau}_{p,q}(q-p) \tag{B.10}$$

where

$$W_{\mathbf{P}} = \sum_{\{p,q\}} (q-p)^2. \tag{B.11}$$

The second derivative yields

$$\frac{\mathrm{d}^2}{\mathrm{d}\tau^2} f(\tau) = -\frac{\pi^2}{3} \sum_{\{p,q\}} (q-p)^2 \leq 0. \tag{B.12}$$

Equality holds if and only if $p = q$ for all $\{p, q\} \in \mathbf{P}$. In the context presented in this thesis, equality holds if and only if the set of sensor pairs are selected to only contain self-pairs. Since it is assumed that the set contains at least one non-self pair, which is a requirement to achieve any spatial resolution, equality is assumed not to hold. The second derivative is then assumed to be strictly negative, and the extreme point is a maximum point.

References

[1] Michael S. Brandstein. *A Framework for Speech Source Localization Using Sensor Arrays*. PhD thesis, Brown University, May 1995.

[2] Patrick A. Naylor and Nikolay D. Gaubitch, editors. *Speech Dereverberation*. Springer, 2010.

[3] Lauri Savioja. *Modeling Techniques for Virtual Acoustics*. PhD thesis, Helsinki University of Technology, 1999.

[4] Finn Jacobsen. The Sound Field in a Reverberation Room. Technical Report 31261, Technical University of Denmark, February 2006.

[5] Jont B. Allen and David A. Berkley. Image Method for Efficiently Simulating Small-Room Acoustics. *The Journal of the Acoustical Society of America*, 65(4):943–950, April 1979.

[6] Firas Jabloun and Benoit Champagne. A Fast Subband Room Response Simulator. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages II925–II928, June 2000.

[7] Nedelko Grbić, Sven Nordholm, and Antonio Cantoni. Optimal FIR subband beamforming for speech enhancement in multipath environments. *Signal Processing Letters*, 10(11):335–338, November 2003.

[8] P. P. Vaidyanathan. *Multirate Systems and Filter Banks*. Signal Processing Series. Prentice-Hall, 1993.

[9] Maurice G. Bellanger, Georges Bonnerot, and Michel Coudreuse. Digital Filtering by Polyphase Network: Application to Sample-Rate Alteration and Filter Banks. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-24(2):109–114, April 1976.

[10] Nedelko Grbić. *Optimal and Adaptive Subband Beamforming: Principles and Applications*. PhD thesis, Blekinge Tekniska Högskola, 2001.

[11] Benny Sällberg. *Applied Methods for Blind Speech Enhancement*. PhD thesis, Blekninge Tekniska Högskola, 2006.

[12] Özgür Yılmaz and Scott Rickard. Blind Separation of Speech Mixtures via Time-Frequency Masking. *IEEE Transactions on Signal Processing*, 52(7):1830–1847, July 2004.

[13] John S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathan G. Fiscus, David S. Pallett, Nancy L. Dahlgren, and Victor Zue. *TIMIT Acoustic-Phonetic Continuous Speech Corpus*. Linguistic Data Consortium, Philadelphia, 1993.

[14] Michael S. Brandstein and Harvey F. Silverman. A Practical Methodology for Speech Source Localization with Microphone Arrays. *Computer Speech & Language*, 11(2):91–126, April 1997.

[15] Michael Brandstein and Darren Ward, editors. *Microphone Arrays: Signal Processing Techniques and Applications*. Springer, 2001.

[16] Alex B. Gershman, Michael Rübsamen, and Marius Pesavento. One- and Two-Dimensional Direction-of-Arrival Estimation: An Overview of Search-Free Techniques. *Signal Processing*, 90(5):1338–1349, May 2010. Special Section on Statistical Signal & Array Processing.

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

[18] Arthur J. Barabell. Improving the Resolution Performance of Eigenstructure-Based Direction-Finding Algorithms. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 8, pages 336–339, April 1983.

[19] Hamid Krim and Mats Viberg. Two Decades of Array Signal Processing Research: the Parametric Approach. *IEEE Signal Processing Magazine*, 13(4):67–94, July 1996.

[20] Yung-Dar Huang and Mourad Barkat. Near-Field Multiple Source Localization by Passive Sensor Array. *IEEE Transactions on Antennas and Propagation*, 39(7):968–975, July 1991.

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

[22] Anders Johansson, Greg Cook, and Sven Nordholm. Acoustic Direction of Arrival Estimation, a Comparison between Root-MUSIC and SRP-PHAT. In *Proceedings of IEEE Region 10 Conference on Computer, Communications, Control and Power Engineering*, November 2004.

[23] G. Clifford Carter. Time Delay Estimation for Passive Sonar Signal Processing. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 29(3):463–470, June 1981.

[24] Charles H. Knapp and G. Clifford Carter. The Generalized Cross Correlation Method for Estimation of Time Delay. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-24(4):320–327, August 1976.

[25] Maurizio Omologo and Piergiorgio Svaizer. Acoustic Event Localization using a Crosspower-Spectrum Phase Based Technique. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume ii, pages II/273–II/276, April 1994.

[26] Maurizio Omologo and Piergiorgio Svaizer. Acoustic Source Location in Noisy and Reverberant Environment using CSP Analysis. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 921–924, May 1996.

[27] Cha Zhang, Dinei Florêncio, and Zhengyou Zhang. Why Does PHAT Work Well in Low Noise, Reverberative Environments? In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2565–2568, March 2008.

[28] Michael S. Brandstein, John E. Adcock, and Harvey F. Silverman. A Closed-Form Method for Finding Source Locations from Microphone-Array Time-Delay Estimates. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 3019–3022, May 1995.

[29] Michael S. Brandstein, John E. Adcock, and Harvey F. Silverman. A Closed-Form Location Estimator for Use with Room Environment Microphone Arrays. *IEEE Transactions on Speech and Audio Processing*, 5(1):45–50, January 1997.

[30] Jean-François Cardoso. Blind Signal Separation: Statistical Principles. In *Proceedings of the IEEE, Special issue on blind identification and estimation*, volume 86, pages 2009–2025, October 1998.

[31] Alexander Jourjine, Scott Rickard, and Özgür Yılmaz. Blind Separation of Disjoint Orthogonal Signals: Demixing N Sources From 2 Mixtures. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 5, pages 2985–2988, June 2000.

[32] Scott Rickard, Radu Balan, and Justinian Rosca. Real-Time Time-Frequency Based Blind Source Separation. In *Proceedings of International Workshop on Independent Component Analysis and Blind Signal Separation*, pages 651–656, December 2001.

[33] Shoko Araki, Hiroshi Sawada, Ryo Mukai, and Shoji Makino. A Novel Blind Source Separation Method with Observation Vector Clustering. In *Proceedings of International Workshop on Acoustic Echo and Noise Control*, September 2005.

[34] Jan Cermak, Shoko Araki, Hiroshi Sawada, and Shoji Makino. Blind Speech Separation by Combining Beamformers and a Time Frequency Binary Mask. In *Proceedings of International Workshop on Acoustic Echo and Noise Control*, September 2006.

[35] Shoko Araki, Shoji Makino, Audrey Blin, Ryo Mukai, and Hiroshi Sawada. Blind Separation of More Speech than Sensors with Less Distortion by Combining Sparseness and ICA. In *Proceedings of International Workshop on Acoustic Echo and Noise Control*, pages 271–274, September 2003.

[36] Shoko Araki, Shoji Makino, Audrey Blin, Ryo Mukai, and Hiroshi Sawada. Underdetermined Blind Separation for Speech in Real Environments with Sparseness and ICA. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 3, pages iii/881–iii/884, May 2004.

[37] Hiroshi Sawada, Shoko Araki, Ryo Mukai, and Shoji Makino. Blind Extraction of Dominant Target Sources Using ICA and Time-Frequency Masking. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(6):2165–2173, November 2006.

[38] Shoko Araki, Shoji Makino, Hiroshi Sawada, and Ryo Mukai. Reducing Musical Noise by a Fine-Shift Overlap-Add Method Applied to Source Separation Using a Time-Frequency Mask. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 3, pages iii/81–iii/84, March 2005.

[39] Shoko Araki, Hiroshi Sawada, Ryo Mukai, and Shoji Makino. Blind Sparse Source Separation with Spatially Smoothed Time-frequency Masking. In *Proceedings of International Workshop on Acoustic Echo and Noise Control*, September 2006.

[40] Jan Cermak, Shoko Araki, Hiroshi Sawada, and Shoji Makino. Blind Source Separation Based on a Beamformer Array and Time Frequency Binary Masking. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages I–145–I–148, April 2007.

[41] Radu Balan, Justinian Rosca, Scott Rickard, and Joseph O'Ruanaidh. The Influence of Windowing of Time Delay Estimates. In *Proceedings of Conference on Information Sciences and Systems*, volume 1, pages 15–17, March 2000.

[42] Ka Fai Cedric Yiu, Nedelko Grbić, Sven Nordholm, and Kok Lay Teo. Multi-Criteria Design of Oversampled Uniform DFT Filter Banks. *IEEE Signal Processing Letters*, 11(6):541–544, June 2004.

[43] George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler. *Computer Methods for Mathematical Computations*. Prentice Hall Professional Technical Reference, 1977.

[44] Elio D. Di Claudio, Raffaele Parisi, and Gianni Orlandi. Multi-Source Localization in Reverberant Environments by ROOT-MUSIC and Clustering. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 921–924, June 2000.

[45] Takanobu Nishiura, Takeshi Yamada, Satoshi Nakamura, and Kiyohiro Shikano. Localization of Multiple Sound Sources Based on a CSP Analysis with a Microphone Array. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 1053–1056, June 2000.

[46] Benjamin Friedlander. A Sensitivity Analysis of the MUSIC Algorithm. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 38(10):1740–1752, October 1990.

[47] Mati Wax and Yosef Anu. Performance Analysis of the Minimum Variance Beamformer in the Presence of Steering Vector Errors. *IEEE Transactions on Signal Processing*, 44(4):928–937, April 1996.

[48] Olivier Besson and Francois Vincent. Performance Analysis of Beamformers Using Generalized Loading of the Covariance Matrix in the Presence of Random Steering Vector Errors. *IEEE Transactions on Signal Processing*, 53(2):452–459, February 2005.

[49] Michael S. Brandstein, John E. Adcock, and Harvey F. Silverman. Microphone-Array Localization Error Estimation with Application to Sensor Placement. *Journal of Acoustical Society of America*, 99(6):3807–3816, June 1996.

[50] Brent Gold, Michael J. Roan, Marty Johnson, and Elizabeth Hoppe. Source Localization Performance of a Multi-Array Network under Sensor Orientation and Position Uncertainty. In *Proceedings of Annual Conference on Information Sciences and Systems*, pages 146–149, March 2008.

[51] Ivan Himawan, Sridha Sridharan, and Iain McCowan. Dealing with Uncertainty in Microphone Placement in a Microphone Array Speech Recognition System. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1565–1568, April 2008.

[52] Marius Pesavento, Alex B. Gershman, and Kon Max Wong. Direction Finding in Partly Calibrated Sensor Arrays Composed of Multiple Subarrays. *IEEE Transactions on Signal Processing*, 50(9):2103–2115, September 2002.

[53] Chong Meng Samson See and Alex B. Gershman. Direction-of-Arrival Estimation in Partly Calibrated Subarray-Based Sensor Arrays. *IEEE Transactions on Signal Processing*, 53(2):329–338, February 2004.

[54] Zohra Yermeche, Nedelko Grbić, and Ingvar Claesson. Beamforming for Moving Source Speech Enhancement. In *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 25–28, October 2005.

[55] Chrysostomos L. Nikias and Athina P. Petropulu. *Higher-Order Spectral Analysis—A Nonlinear Signal Processing Framework*. Prentice-Hall, 1993.

[56] Wei Zhang and Saeed Gazor. Statistical Modelling of Speech Signals. In *Proceedings of International Conference on Signal Processing*, volume 1, pages 480–483, August 2002.

[57] James P. LeBlanc and Phillip L. De Leòn. Speech Separation by Kurtosis Maximization. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 1029–1032, May 1998.

[58] Wen-Jun Zeng, Xi-Lin Li, and Xian-Da Zhang. Direction-of-Arrival Estimation Based on the Joint Diagonalization Structure of Multiple Fourth-Order Cumulant Matrices. *IEEE Signal Processing Letters*, 16(3):164–167, March 2009.

[59] Jean-François Cardoso and Éric Moulines. Asymptotic Performance Analysis of Direction-Finding Algorithms based on Fourth-Order Cumulants. *IEEE Transactions on Signal Processing*, 43(1):214–224, January 1995.

[60] Boaz Porat and Benjamin Friedlander. Direction Finding Algorithms Based on High-Order Statistics. *IEEE Transactions on Signal Processing*, 39(9):2016–2024, September 1991.

[61] Benny Sällberg, Nedelko Grbić, and Ingvar Claesson. Online Maximization of Subband Kurtosis for Blind Adaptive Beamforming in Realtime Speech Extraction. In *Proceedings of IEEE Digital Signal Processing*, pages 603–606, July 2007.

[62] Zohra Yermeche, Nedelko Grbić, and Ingvar Claesson. Blind Subband Beamforming with Time-Delay Constraints for Moving Source Speech Enhancement. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(8):2360–2372, November 2007.

[63] Brian D. Simpson, Robert S. Bolia, Richard L. McKinley, and Douglas S. Brungart. The Impact of Hearing Protection on Sound Localization and Orienting Behavior. *Human Factors*, 47(1):188–198, June 2005.

[64] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

[65] Mikael Nilsson. First Order Hidden Markov Model: Theory and Implementation Issues. Technical report, Department of Signal Processing, Blekinge Institute of Technology, 2005.

[66] Mikael Nilsson. *On Feature Extraction and Classification in Speech and Image Processing*. PhD thesis, Blekinge Tekniska Högskola, 2007.

[67] Brian H. Maranda. On the False Alarm Probability for an Overlapped FFT Processor. *IEEE Transactions on Aerospace and Electronic Systems*, 32(4):1452–1456, October 1996.

[68] H. C. So, Y. T. Chan, Q. Ma, and P. C. Ching. Comparison of Various Periodograms for Single Tone Detection and Frequency Estimation. In *Proceedings of IEEE International Symposium on Circuits and Systems*, volume 4, pages 2529–2532, June 1997.

[69] Edgar F. Velez. Detection of Multi-tone Signals Based on Energy Operators. In *Proceedings of IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, pages 229–232, October 1994.

[70] S. M. Kuo and J. Chen. New Adaptive IIR Notch Filter and its Application to Howling Control in Speakerphone System. *Electronics Letters*, 28(8):764–766, April 1992.

[71] Robert Beck, Andrew G. Dempster, and Izzet Kale. Finite-Precision Goertzel Filters used for Signal Tone Detection. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 48(6):691–700, June 2001.

[72] Nils Westerlund, Mattias Dahl, and Ingvar Claesson. Speech Enhancement by Non-stationary Tonal Disturbance Cancellation using Subband Zero Crossing Measures. In *Proceedings of International Symposium on Intelligent Multimedia, Video and Speech Processing*, pages 13–16, October 2004.

[73] Nils Westerlund, Mattias Dahl, and Nedelko Grbić. Detection and Attenuation of Feedback Induced Howling in Hearing Aids using Subband Zero-Crossing Measures. In *Proceedings of International Symposium on Signal Processing and Its Applications*, pages 751–754, August 2005.

[74] Jinhui Chao and Shigeo Tsujii. A Stable and Distortion-Free IIR Echo and Howling Canceler. *IEEE Transactions on Signal Processing*, 39(8):1812–1821, August 1991.

[75] Joseph A. Maxwell and Patrick M. Zurek. Reducing Acoustic Feedback in Hearing Aids. *IEEE Transactions on Speech and Audio Processing*, 3(4):304–313, July 1995.

[76] Marcio G. Siqueira and Abeer Alwan. Steady-State Analysis of Continuous Adaptation in Acoustic Feedback Reduction Systems for Hearing-Aids. *IEEE Transactions on Speech and Audio Processing*, 8(4):443–453, July 2000.

[77] Hsiang-Feng Chi, Shawn X. Gao, and Sigfrid D. Soli. A Novel Approach of Adaptive Feedback Cancellation for Hearing Aids. In *Proceedings of IEEE International Symposium on Circuits and Systems*, volume 3, pages 195–198, May 1999.

[78] Rob Drullman, Joost M. Festen, and Reinier Plomp. Effect of Temporal Envelope Smearing on Speech Reception. *The Journal of the Acoustical Society of America*, 95(2):1053–1064, February 1994.

[79] Jan Mark de Haan, Nedelko Grbić, Ingvar Claesson, and Sven Nordholm. Design of Oversampled Uniform DFT Filter Banks with Delay Specifications using Quadratic Optimization. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 6, pages 3633–3636, May 2001.

[80] Jan Mark de Haan, Nedelko Grbić, Ingvar Claesson, and Sven Nordholm. Filter Bank Design for Subband Adaptive Microphone Arrays. *IEEE Transaction on Speech and Audio Processing*, 11(1):14–23, January 2003.

[81] Jan Mark de Haan. *Filter Bank Design for Digital Speech Signal Processing: Methods and Applications*. PhD thesis, Blekinge Tekniska Högskola, 2004.

[82] Simon Haykin. *Adaptive Filter Theory*. Information and System Sciences. Prentice-Hall, fourth edition, 2002.

[83] Steinberg Media Technologies GmbH. Steinberg Audio Streaming Input Output Specification. Part of the ASIO 2.2 Software Development Kit.

[84] ASIO4ALL. http://www.asio4all.com.

ABSTRACT

Speech communication is gaining in popularity in many different contexts as technology evolves. With the introduction of mobile electronic devices such as cell phones and laptops, and fixed electronic devices such as video and teleconferencing systems, more people are communicating which leads to an increasing demand for new services and better speech quality. Methods to enhance speech recorded by microphones often operate blindly without prior knowledge of the signals. With the addition of multiple microphones to allow for spatial filtering, many blind speech enhancement methods have to operate blindly also in the spatial domain. When attempting to improve the quality of spoken communication it is often necessary to be able to reliably determine the location of the speakers. A dedicated source localization method on top of the speech enhancement methods can assist the speech enhancement method by providing the spatial information about the sources.

This thesis addresses the problem of speech-source localization, with a focus on the problem of localization in the presence of multiple concurrent speech sources. The primary work consists of methods to estimate the direction of arrival of multiple concurrent speech sources from an array of sensors and a method to correct the ambiguities when estimating the spatial locations of multiple speech sources from multiple arrays of sensors. The thesis also improves the well-known SRP-based methods with higher-order statistics, and presents an analysis of how the SRP-PHAT performs when the sensor array geometry is not fully calibrated. The thesis is concluded by two envelope-domain-based methods for tonal pattern detection and tonal disturbance detection and cancelation which can be useful to further increase the usability of the proposed localization methods.

The main contribution of the thesis is a complete methodology to spatially locate multiple speech sources in enclosed environments. New methods and improvements to the combined solution are presented for the direction-of-arrival estimation, the location estimation and the location ambiguity correction, as well as a sensor array calibration sensitivity analysis.

BLEKINGE TEKNISKA HÖGSKOLA · BTH ·

2012:03

ISSN 1653-2090
ISBN: 978-91-7295-226-3