Object-Based Audio Capture: Separating Acoustically-Mixed Sounds

Alexander George Westner

Bachelor of Science
Rutgers University, 1996

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences
at the Massachusetts Institute of Technology

February 1999

© Massachusetts Institute of Technology, 1999
All Rights Reserved

Author
Alexander George Westner
Program in Media Arts and Sciences
October 9, 1998

Certified by
V. Michael Bove, Jr.
Principal Research Scientist
MIT Media Laboratory
Thesis Supervisor

Accepted by
Stephen A. Benton
Chair, Departmental Committee on Graduate Students
Program in Media Arts and Sciences
Object-Based Audio Capture: Separating Acoustically-Mixed Sounds

Alexander George Westner

Submitted to the Program in Media Arts and Sciences, School of Architecture and Planning, on October 9, 1998 in partial fulfillment of the requirements for the degree of Master of Science in Media Arts and Sciences

Abstract

This thesis investigates how a digital system can recognize and isolate individual sound sources, or audio objects, from an environment containing several sounds. The main contribution of this work is the application of object-based audio capture to unconstrained real-world environments. Several potential applications for object-based audio capture are outlined, and current blind source separation and deconvolution (BSSD) algorithms that have been applied to acoustically-mixed sounds are reviewed. An explanation of the acoustics issues in object-based audio capture is provided, including an argument for using overdetermined mixtures to yield better source separation. A thorough discussion of the difficulties imposed by a real-world environment is offered, followed by several experiments which compare how different filter configurations and filter lengths, as well as reverberant environments, all have an impact on the performance of object-based audio capture. A real-world implementation of object-based audio capture in a conference room with two people speaking is also discussed. This thesis concludes with future directions for research in object-based audio capture.

Thesis Supervisor:
V. Michael Bove, Jr.
Principal Research Scientist
MIT Media Laboratory

This work was supported by the Digital Life Consortium at the Media Laboratory.
Object-Based Audio Capture:  
Separating Acoustically-Mixed Sounds

Alexander George Westner

Thesis Reader

Barry Vercoe
Professor
Program in Media Arts and Sciences

Thesis Reader

James L. Flanagan
Vice President of Research
Rutgers University, New Brunswick, New Jersey
Acknowledgments

I'd like to first thank the Professors who have helped me along in my academic life: Mike Bove for bringing me to the Media Lab, for being patient with me while I was here, and for supporting me when I did things right; Mark Kahrs for giving me sound advice (pun intended) throughout my undergraduate life at Rutgers; Jim Flanagan for giving me the opportunity to work at CAIP, helping me get into MIT, and serving as a reader for this thesis; and Barry Vercoe for advice, encouragement and for serving as a reader, as well.

Loudest gratitude to those who helped me put this thesis project together: Bill Gardner, Keith Martin, Joe Pompei and Paris Smaragdis from the Machine Listening Group for letting me bombard them with questions and concerns about things audio; Dan Rabinkin from CAIP's Microphone Array group; Bill and Wad from my group; Te-Won Lee from the Salk Institute; Russ Lambert from TRW; Geoff Martin from McGill; and Freedom for excellent advice and copy-editing.

At the Media Lab, for the first time in my life, I've felt comfortable with almost everyone around me. Quoting Freedom: "I've found my people!" So, thanks people: arjan, baird, cahn, cdodge, dakss, daniel, hannes, jebara, joey, kkarakhal, mikeb, nelson, nickm, nitin, pompeii, pwise, richlach, slowe, softball, stefan, swaldman, thursdays, vemuri, verp, and wojal. Individually, however, I want to thank Stefan Agamanolis for being a fun person to work with; Tom Slowe for being a great roommate and drinking buddy; Joey for listening and encouraging; and all my officemates for putting up with Reflection of Presence demos!

Big up's to my friends back in New Jersey for letting me do my thing: DJ, Flavio, Tom and Vin. Many thanks to the Media Lab staff: Deirdre, Felice, Henry, Laurie, Lena, Linda, Pat, Rebecca, Santana, and Theresa.


Finally, I thank Mom and Dad for supporting me in everything I ever wanted to do; and I am humbly grateful to Freedom for her love; this thesis is dedicated to them.
# Table of Contents

Abstract 3

Acknowledgments 7

## Chapter 1 Introduction 13

1.1 The Cocktail-Party Effect 13
   1.1.1 The music studio 14
   1.1.2 Film studio 15
   1.1.3 Telepresence 16
   1.1.4 Wearable Audio Computing 18
   1.1.5 Smart Rooms 19
1.2 Object-Based Media 21
1.3 Blind Source Separation and Deconvolution 22
   1.3.1 Blind source separation 22
   1.3.2 Blind deconvolution 23
1.4 Thesis Outline 25

## Chapter 2 Multi-Channel Blind Source Separation and Deconvolution 27

2.1 Approaching the Problem 27
2.2 Multi-Sensor Techniques 28
   2.2.1 Herault and Jutten 28
   2.2.2 Independent Component Analysis (ICA) 30
   2.2.3 Separating acoustically-transmitted sources 30
   2.2.4 What's phase got to do with it? 32
2.3 Multi-Channel Blind Least-Mean-Square 33
2.4 Time-Delayed Decorrelation 35
2.5 Information Maximization 37
   2.5.1 Information theory and blind source separation 37
   2.5.2 Modern extensions 38
2.6 Microphone Arrays 40

## Chapter 3 Acoustic Mixtures 41

3.1 Room Impulse Responses 41
   3.1.1 Acquiring room impulse responses 42
   3.1.2 The Garden Conference Room 44
   3.1.3 Analyzing the impulse responses 48
3.2 Inverting the Room 49
3.3 Filter Configuration 51
   3.3.1 What do the separating filters look like? 51
   3.3.2 Using ideal separating filters to separate acoustically-mixed sounds 53

Object-based Audio Capture 9
Table of Contents

Chapter 4 Implementation and Results 57

4.1 The Real World 57
  4.1.1 Microphones 58
  4.1.2 Sound source-microphone distances 59
  4.1.3 Room noise 62
4.2 The Algorithm 64
  4.2.1 The procedure 65
  4.2.2 Parameters 66
4.3 Varechoic Chamber Data 68
  4.3.1 Data Acquisition 69
  4.3.2 Some characteristics of the data 69
4.4 Experimentation 71
  4.4.1 SNR improvement 71
  4.4.2 A word about unmixing filter lengths 72
4.5 Reverberation 72
4.6 Overdetermined Blind Separation 73
  4.6.1 A Simple experiment 73
  4.6.2 Direct filters and cross filters 75
  4.6.3 Experiments with the Varechoic Chamber data 78
4.7 Square Configurations 81
4.8 A Conference Room 85

Chapter 5 Conclusions and Future Directions 89

5.1 Conclusions 89
  5.1.1 Overdetermined mixtures 90
  5.1.2 Acoustics and noise 90
  5.1.3 Are we better off than just doing nothing with these mixtures? 91
5.2 Future Directions 91
  5.2.1 Revisiting microphones 91
  5.2.2 Room noise 92
  5.2.3 Picking inputs and outputs 93
  5.2.4 Initialization 94
  5.2.5 Real-time? 94
  5.2.6 Other potential BSSD algorithms 95
5.3 Final Thought 95

Appendix A Information Theory and Blind Source Separation 97

A.1 Mutual Information 97
A.2 Sigmoidal Function 98
A.3 Entropy 98
A.4 Stochastic Gradient Ascent 99
Appendix B  Tabulated SNR Results  101
B.1  Reverberation Experiment  102
B.2  Overdetermined Configuration Experiments  103
B.3  Square Configuration Experiments  107

References  111
Chapter 1   Introduction

1.1 The Cocktail-Party Effect

Digital systems are commonly used today to record and analyze audio in such environments as modern music and film studios, as well as in telepresence applications. At the MIT Media Lab, projects like Reflection of Presence (Aga- manolis, et. al., 1997), Wearable Audio Computing (Roy, et. al., 1997), and Smart Rooms (http://vismod.www.media.mit.edu/vismod/demos/smartroom/) explore more complex digital audio applications for use in future technologies. Recording audio in these environments often requires contextual information about the soundscape, audio engineering knowledge and skilled miking techniques.

Digital systems that process audio signals typically treat their input as a single sound source, yet many of these systems monitor environments that are filled with several different sources. While humans can focus their attention on any one sound source out of a mixture of many in real-time (a phenomenon termed by E. Colin Cherry in 1953 as the “cocktail-party effect”), current digital audio systems lack this ability (Figure 1.1).
Chapter 1: Introduction

This thesis describes how we are heading towards the goal of enabling a digital system to recognize and isolate individual sound sources, or audio objects, from an environment which contains several different sound sources. The main contribution of this project is the application of object-based audio capture to unconstrained real-world environments.

In the following sections, I describe several applications in which object-based audio capture would improve the quality of audio acquisition, by enabling access to the individual sound sources in a mixture of sounds. When a system has access to the individual audio objects in its acoustic environment, it can determine some useful features, such as how many different sounds are present, what these sounds might be, and how each of these sounds should be processed and recorded.

1.1.1 The music studio

In a music studio, an engineer often wants to maximize the isolation between acoustic instruments to gain more control over the individual sound sources. According to Peter Tattersall (1980), however, “it is good practice to group the musicians as close together as possible, as they play with more feeling if they
are together, instead of being shut off in separate booths or heavily screened" (p. 327). The studio engineer is, therefore, faced with a trade-off: the closer the musicians are to each other, the more difficult it is to isolate their individual sounds. To overcome this problem, he/she uses special microphone patterns along with careful microphone placement to reduce the amount of leakage from unwanted sounds during recording (Moylan, 1992, pp. 98-99). A system that could automatically capture individual audio objects — even when the musicians are playing near each other — would certainly be of great assistance to the recording engineer.

Research in the field of object-based audio capture has not yet produced a system that can completely isolate the sounds of these different instruments as they're being recorded. This thesis will explore how close we are to this goal right now. Skillful miking techniques used in conjunction with an object-based audio capture system can enhance current recording methods in the music studio.

1.1.2 Film studio

Tim Amyes (1990) describes the audio content in a typical scene of a film production (p. 80):

A simple scene may, for example, consist of two people walking, talking to each other and then driving away in separate cars. The scene will consist of many separate sounds — the feet of the two characters meeting; the talk between the two, consisting of a separate recording of each; the doors slamming and the noise of the two cars starting. In addition, the general background noise of the complete scene is needed. This may be cars going by, birds singing, street vendors or activity in a harbour. The sound scene is more than just one picture following another; it consists of sounds that geographically, and in time, knit together the whole discontinuous scene.

To achieve this sort of sonic continuity, dialogue among characters is often recorded in a manner that can best isolate it from other external sounds, using hidden wireless microphones, for example. Background sounds, like car noises and ambient environmental sounds, are recorded later and added into the scene in post-production.
Chapter 1: Introduction

The sounds required for such a scene must be recorded cleanly so that they can easily be edited in post-production. Although it is very difficult to overdub audio dialogue, filmmakers and sound recordists often do so to gain more control over the film’s sound. A lot of extra time and money is routinely spent at post-production facilities to compensate for less-than-ideal recording conditions (Amyes, 1990, pp. 84-85).

In a manner similar to that previously described for the music studio, an object-based audio capture system could be very valuable to a film sound recordist by automatically isolating the different sounds that he/she is trying to capture. In addition, whereas the members of a music group play their instruments for the duration of a recording session, the different foreground sounds on the set of a film scene might not overlap as much, making it easier to isolate individual sound sources. On the other hand, background sounds on a film set will tend to be shorter (e.g. a door slamming, footsteps), thus giving the system less information about how to separate them from the total mixture.

1.3 Telepresence

Most telepresence applications rely on the participants’ often untrained assessment of acoustics when setting up microphones to record the different audio sources to be transmitted. While audio quality is essential in music and film productions, time is the most important factor in telepresence applications; as soon as audio is captured, it needs to be transmitted to remote locations. Efficiency and hands-off operation of audio capture are also important in these situations.

The most basic videoconferencing systems provide a simple speakerphone-like device placed in the middle of the room to capture as much audio as possible; more advanced systems provide multiple microphone inputs for better coverage of all of the potential speakers in the room. For example, PictureTel’s LimeLight™ technology uses audio input from multiple sensors to locate the position of a particular speaker in the room, using this information to automatically steer a video camera towards the speaker. LimeLight™ was not
Reflection of Presence is a telepresence application, developed at the MIT Media Lab, that explores the use of object-based techniques to enhance remote communication and collaboration (Agamanolis, et. al., 1997). The motivation behind the application is that, by using object-based audio and video, cross-media interactions are simplified. For example, the audio from each participant is analyzed to determine for how long and how frequently each participant speaks; these features are then used to aid in the video composition by varying the opacity and superposition of each participant. (See Figure 1.2)

The audio from each participant in Reflection of Presence is recorded using a headset microphone to insure that the only sound captured by the system is the participant's voice; this consequently limits the application to only one participant for each site. An object-based audio capture system would allow an application like Reflection of Presence to expand, allowing for multiple participants from each site. Using object-based audio capture, the participants would not need to wear microphones, yet each participant's voice would be recorded and analyzed individually.
As demonstrated in *Reflection of Presence*, videoconferencing and telepresence applications can benefit from using more individualized audio objects instead of one complex stream of audio data.

1.1.4 Wearable Audio Computing

Real-time hands-off audio capture is also important in a Wearable Audio Computer (WAC). Deb Roy, et. al. (1997) outline several audio capture applications for a WAC: easy recording of conversations, lectures, meetings, etc.; a speech-recognition interface to the wearable computer; and augmented audio memories (recordings of the user’s audio environment for future reference).

The interfaces to a WAC should be unobtrusive and self-sufficient, meaning that not only should the user be unencumbered by the device, he/she should not have to worry about how the device is acquiring data. Nitin Sawhney is modifying the *Soundbeam Neckset*, an audio I/O research prototype device developed by Nortel (shown in Figure 1.3) for use with a WAC. The Neckset has “two directional speakers, mounted on the user’s shoulders, and a directional microphone placed on the chest” (Sawhney, 1998).

Since a WAC may be used in noisy environments, or environments with several significant simultaneous sound sources (is that a tongue twister, or what?) present, it would be beneficial to use object-based audio capture to give the WAC a clearer description of the scene.
The Cocktail-Party Effect

1.1.5 Smart Rooms

The MIT Media Lab’s Vision and Modeling Group describes Smart Rooms as “invisible butlers.” Using cameras, microphones and other sensors, they try to interpret what people are doing in order to help them.4

The ALIVE (Artificial Life Interactive Video Environment) project is a Smart Room application that explores “remote-sensing full-body interactive interfaces for virtual environments” (Casey, et. al., 1995). In one application of the ALIVE space, the user interacts with “Silas,” a virtual dog, by using gestures or spoken commands to affect the dog’s behavior. The ALIVE system tracks the user and records gestures with a video camera mounted atop the large projection screen (see Figure 1.4). A vision-steered beamforming microphone

5. The ALIVE system uses computer vision to find the location of the speaker, passing this information along to a beamforming microphone array. The beamformer, then, uses the speaker location and the known geometry of the array to time-align the signals received by the microphones so that the desired sounds add coherently, and the undesired noise adds incoherently. Thus, a beamformer can assist in the quality of the audio capture.

array captures the audio commands.

The array was designed to focus on one sound source, eliminating ambient room noise from the recorded signal. The array is certainly unobtrusive and requires no hands-on attention, but it relies on computer vision to assist in tracking the sound sources. The system described in this thesis tracks objects solely in the audio domain, and can provide ALIVE-type spaces with the potential to record multiple individual sounds.

Figure 1.4 The ALIVE space


The KidsRoom, designed and built by the Media Lab's High Level Vision and Interactive Cinema groups, is a "fully-automated, interactive playspace for children." Set in a child's magical bedroom, the KidsRoom uses images, lighting, sound, and computer vision technology to generate a fantasy story with which children can interact (see Figure 1.5). Using computer vision primarily, the room reacts to the children's actions, guiding them through an adventure story, and engaging them with playful characters. Please refer to Aaron Bobick, et. al.'s (1996) article and the KidsRoom website for a detailed explanation, pictures and audio files.
At two different moments in the narrative, the children are asked by the room to shout a specific magic phrase. A single microphone placed in the room measures the collective loudness of the children's responses, then directs the narrative appropriately. With an object-based audio capture system, the *KidsRoom* would be able to carefully "listen" to what each child in the room is saying — though it would not *understand* them, the system could nevertheless increase the interactivity of the environment as well as broaden its narrative possibilities.

### 1.2 Object-Based Media

The theoretical motivation for the research discussed in this thesis stems from the core principles of the Object-Based Media research group, led by V. Michael Bove, Jr. at the MIT Media Laboratory. The group focuses on the development of object-based representations of audio and video. As an example of this fundamental idea, we describe video "not as sequences of frames but rather as collections of modeled objects that are encoded by machine-vision algorithms and decoded according to scripting information" (Bove,
1996). Instead of blindly passing a digitized video stream to an output device, we can analyze the stream and extract salient objects from it. We can then deduce information about the content of the video, enabling more intelligent decision-making during the processing, transmission, and composition of the video.

The target of this research effort, therefore, is to provide digital systems with the ability to segment audio and video information “in a manner that is content-driven rather than arbitrary (e.g. localized, modeled sound sources and acoustical environments rather than speaker channels; coherent objects rather than blocks of pixels)” (Bove, 1996).

Bove has outlined some potential benefits from this approach:

- New production and post-production methods
- Intelligent database search
- Easier authoring of interactive or personalized content
- Better support for distributed storage
- Assembly of content “on-the-fly” from disparate elements

Thus far, the primary focus of the group has been on capturing and rendering video objects. Recent work, however, explores the audio equivalent: in 1995, Araz Inguilizian wrote his master’s thesis on the rendering of 3-D audio objects. This thesis explores object-based audio capture.

1.3 Blind Source Separation and Deconvolution

Before continuing my discussion of this thesis project, I need to define some fundamental concepts which underlie my description of “object-based audio capture” — specifically, blind source separation and blind deconvolution.

1.3.1 Blind source separation

In the signal processing research community, there is currently a strong interest in source separation, where the goal is to extract a desired signal from a mixture of many other signals. In practice, however, we often don’t know how the sources were mixed together. The problem then becomes to extract the
original sources given only the mixtures themselves (see Figure 1.6). This is commonly referred to as blind source separation (Hérault and Jutten, 1986).

![Diagram of source separation](image)

**Figure 1.6** The goal in blind source separation is to use only the mixture of sources to find the originals.

### 1.3.2 Blind deconvolution

Researchers who use source separation algorithms make the fundamental assumption that the signals are mixed together *instantaneously*, meaning that all of the signals are time-aligned so that they enter the mixture simultaneously. Consider, however, the way in which a sound is recorded in a typical room using one microphone (see Figure 1.7). The microphone will receive a direct copy of the sound source (at some propagation delay based on the location of both the source and the microphone) as well as several reflected and modified copies of the sound source (as the sound waves bounce off walls and objects in the room).
Figure 1.7 A microphone picks up the direct sound source as well as several delayed copies of it.

The distortion of the recorded signal is dependent upon the reverberation and absorption characteristics of the room, as well as the objects within the room, and can be modeled as an impulse response in a linear system (Orfanidis, 1996):

\[
\text{mic signal} = \text{room impulse response} * \text{sound source}
\]  

Equation 1.1

The room impulse response models all possible paths the sound source takes to arrive at the microphone. A unique impulse response exists for every possible location of both the source and the microphone! The "*" symbol in Equation 1.1 denotes convolution. To find the original sound source that was recorded with a microphone in a room, we must cancel out, or deconvolve, the room impulse response from the original sound source. Since we have no prior knowledge of what this room impulse response is we call this process blind deconvolution (Haykin, 1994).

Object-based audio capture combines blind source separation and blind deconvolution to find the original sound sources that are recorded by microphones in a room.
1.4 Thesis Outline

In the next chapter, I will discuss the current trends in blind source separation and deconvolution research, concluding with three approaches that I feel are best suited for object-based audio capture. Chapter 3 follows with an explanation of the acoustics related to object-based audio capture, and includes a quantitative comparison of several different unmixing filter configurations, arguing that overdetermined mixtures yield better source separations. Implementations of object-based audio capture are presented in Chapter 4, including a thorough discussion of the difficulties that a real-world environment imposes. Chapter 5 follows with conclusions and future directions.
I will use the following notation throughout this thesis (following Lambert, 1996): a lowercase bold variable, \( \mathbf{a} \), is a vector and an uppercase bold variable, \( \mathbf{A} \), is a matrix. Any underlined bold variable, \( \underline{\mathbf{a}} \) or \( \underline{\mathbf{A}} \), is a vector or matrix where each element is a time sequence, or an FIR polynomial or filter.

2.1 Approaching the Problem

One important difference between the way humans and machines process sound is that humans listen binaurally (with two ears, each receiving complex localization cues) whereas most machines are configured to receive one or more discrete channels of microphone-transduced audio. We can use a binaural microphone as an audio input to a machine to account for the acoustic transform of human ears, but we still don't know enough about how our brains interpret this binaural data to be able to program a machine to do likewise.

Microphone arrays, on the other hand, provide data that is easier for us to interpret and process: the use of microphone arrays has been investigated for over 35 years, and has proven to be effective in blind source separation and
deconvolution (BSSD) research. I feel, therefore, that it is of sound reasoning (pun intended) to focus on multi-sensor (microphone array-based) techniques, as opposed to binaural or single-input techniques, to study object-based audio capture.

In the following section, I will describe the fundamental concepts behind multi-sensor approaches. I will then discuss three important threads of current research in BSSD algorithms as they apply to object-based audio capture.

2.2 Multi-Sensor Techniques

Consider an array of microphones recording sound from a space which contains several different sources (as in the examples presented in section 1.1 — a music or film studio, a teleconferencing session, etc.). Since each of the microphones in the array has a unique impulse response pattern, and is placed at some distance away from the other microphones, each will record a different mix of the sound sources. With multi-sensor BSSD techniques, we use subtractive cancellation between the different microphone inputs to undo some of the acoustics of the space, suppressing unwanted sources to inaudible levels. Over the past decade, researchers have proposed many different multi-sensor BSSD algorithms based on these assumptions.

2.2.1 Hérault and Jutten

Jeanny Hérault and Christian Jutten (1986) formalized the algebra behind blind source separation, summarized below (see Figure 2.1). Please keep in mind that this work is a general treatment of blind source separation — it applies to a variety of applications, including sound source separation:
1. Assume that we have $N$ statistically independent sources, $s_i(t)$, $i=1...N$, recorded by $M$ sensors, where $N=M$.

2. The signal measured by each sensor, $x_j(t)$, $j=1...M$, contains a copy of each of the sources, linearly mixed together by a matrix $A$, the mixing matrix.

3. The objective of blind source separation is to find the inverse of $A$, which is the unmixing matrix, $W$.

4. By multiplying $W$ with $x(t)$, we obtain approximations, $u(t)$, of the original sources.

![Image of a 4-channel network architecture](image)

**Figure 2.1** A 4-channel network architecture (Bell and Sejnowski, 1995a)

The problem that researchers must face in blind source separation is just how to determine the unmixing matrix, given only the mixtures, $x(t)$. Hérault and Jutten (1991) used an “adaptive neuromimetic network,” shown in Figure 2.2.

The neuromimetic network uses higher-order statistics to determine the weights, $w_{ij}$, that will result in the separation of the two input sources, which emerge as the outputs, $u(t)$.

---

9. A neuromimetic network, known more commonly today as an artificial neural network (ANN), is a system of interconnected weights that adapt to information entering the system. The word “neural” is used because the inspiration for these networks comes from the field of neuroscience. Understand that we are not purporting to model the human brain with these networks!
2.2.2 Independent Component Analysis (ICA)

Hérault and Jutten's work led to the development of Independent Component Analysis (ICA). Formally defined by Pierre Comon (1994), ICA is a mathematical technique based on higher-order statistics used to find a linear transformation that minimizes the statistical dependence between the components of a given vector. Comon assembled previous ICA work into a comprehensive mathematical basis which "served as a springboard for further research," resulting in more efficient ICA-based source separation algorithms (Smaragdis, 1997, p. 23).

Strict ICA algorithms, however, only perform source separation for instantaneous source mixtures; they do not address the problem of delayed and convolved source mixtures, such as real-world audio signals.

2.2.3 Separating acoustically-transmitted sources

I described the nature of a real-world audio signal earlier in section 1.3.2. Now, I would like to expand the discussion to include multiple sound sources. Consider two sounds recorded in a room using two microphones. Each microphone receives a direct copy of both sound sources (at different propagation delays between each source and microphone) as well as several reflected and modified copies of both sound sources (see Figure 2.3). The distortion of the recorded signals is dependent upon the reverberation and absorption charac-
The acoustic paths of 2 sources in a room with 2 microphones. Referring back to Figure 1.7, note how many more paths are generated as the number of sources and microphones increases.

Characteristics of the room and the objects within the room and can be modeled as impulse responses in a linear system (Orfanidis, 1996):

\[ x_1(t) = a_{11} * s_1(t) + a_{12} * s_2(t) \] (2.1)

\[ x_2(t) = a_{21} * s_1(t) + a_{22} * s_2(t) \] (2.2)

where \( x_i \) are the microphone signals, \( a_{ij} \) are the room impulse responses between each source and microphone, and \( s_j \) are the sound sources. Consolidating the two above equations into matrix form yields

\[ x(t) = A * s(t) \] (2.3)

Given only \( x(t) \), we must determine a set of filters that will deconvolve the room impulse responses from the original sound sources. Hérault and Jutten’s blind source separation problem statement, outlined in section 2.2.1, can be expanded to account for convolved mixtures of acoustically-mixed sounds:

1. Assume that we have \( N \) statistically independent sound sources, \( s_i(t) \ i=1...N \), recorded by \( M \) microphones, where \( N=M \).

2. The signal measured by each microphone, \( x_j(t) \ j=1...M \), contains a copy of each of the sound sources, convolved by a matrix \( A \), the mix-
Chapter 2: Multi-Channel Blind Source Separation and Deconvolution

...ing matrix. Each element in $A$ is an impulse response between a sound source and a microphone.

3. The objective of blind source separation and deconvolution is to find the inverse of $A$, which is the unmixing matrix, $W$. Each element in $W$ is a finite impulse response (FIR) filter.

4. By convolving $W$ with $x(t)$, we obtain approximations, $u(t)$, of the original sources.

2.2.4 What's phase got to do with it?

Kari Torkkola (1996) designed a BSSD algorithm using a full feedback network architecture. This feedback network takes the same form as the Hérault-Jutten network (Figure 2.2), except that the weights, $w_{ij}$, in Torkkola’s network correspond to two filters instead of two numbers.

Torkkola was among the first to apply blind source separation to convolved sound sources.

One constraint of the feedback architecture, however, is that it is only able to learn the inverses to minimum phase filters (Haykin, 1996). In general, for an impulse response filter to be minimum phase, the first sample should be larger than all other samples, and the response should decay rapidly (Neely and Allen, 1979). Visual inspection of a typical room impulse response (in the time domain) shows that the response is not minimum phase (see Figure 2.4).

Therefore, we will need to implement a feedforward network architecture to successfully learn the inverses to room impulse responses. I will discuss room impulse response characteristics further in Chapter 3.

In the next three sections, I will describe three of the more popular approaches to object-based audio capture. They are: multi-channel blind least-mean-square (MBLMS), time-delayed decorrelation (TDD), and information maximization (infomax).
2.3 Multi-Channel Blind Least-Mean-Square

The least-mean-square (LMS) algorithm is one of the simplest adaptive filter algorithms in use today. It consists of three main steps, illustrated in Figure 2.5 (Haykin, 1996):

1. Filter an input signal, $x(t)$, through the adaptive filter, $W$.
2. Generate an error signal, $e(t)$, based on a comparison of the filtered signal, $u(t)$, and a desired response, $d(t)$; in the case of blind deconvolution, the desired response is the original source signal, $s(t)$.
3. Adjust the filter taps as a function of this error signal.

Intuitively, the goal of LMS is to minimize (in a mean-squared sense) the error signal generated in step 2. Researchers in the field of blind deconvolution refer to minimizing a cost function, $J$, in much the same way as minimizing the mean-squared error (Lambert, 1996):

$$ J = E|u - s|^2 $$  \hspace{1cm} (2.4)
where \( E[\cdot] \) is the mean of the argument.

Without access to the original source, however, we do not explicitly know what our desired signal is. Instead, we use the filter output, \( u \), along with a Bussgang nonlinear function, \( g(\cdot) \), to produce the cost function

\[
J = E[u - g(u)]^2
\]

(2.5)

The Bussgang nonlinearity is a function of the probability density function (pdf) of the estimated input sources (Haykin, 1994):

\[
g(u) = \frac{-E|u|^2 p'_u(u)}{p_u(u)}
\]

(2.6)

We minimize \( J \) to get a function that will iteratively adapt the weights, or taps, of the deconvolution filter, \( W \). Lambert (1996) derives a blind LMS cost
function that can be extended to multiple inputs, which he refers to as multi-channel blind least-mean-squares (MBLMS):

\[ J = \text{trace } E\{(u - g(\xi))(u - g(\xi))^H\} \]  

(2.7)

We can determine the weight update equation by minimizing this cost function:

\[ \frac{\partial J}{\partial W} = \frac{\partial J}{\partial u} \frac{\partial u}{\partial W} = \frac{\partial J}{\partial u} \tilde{x}^* \]  

(2.8)

\[ \frac{\partial J}{\partial u} = u - g(\xi) \]  

(2.9)

\[ \Delta W \propto u - g(\xi)\tilde{x}^* \]  

(2.10)

Equation 2.10 tells us exactly how to change the weights of the unmixing filters, given only the mixtures and the pdf's of the original sources.

2.4 Time-Delayed Decorrelation

In 1994, Lutz Molgedey and Heinz Schuster incorporated more information about the time structure of the sources into the adaptation process introduced by Hérault and Jutten (1986), whose correlation-based algorithm only decorrelated instantaneously mixed sources. I refer to Ikeda and Murata (1998) to help me summarize Molgedey and Schuster's seminal work.

The correlation matrix, \( R_{xx}(\tau) \), of the observable mixtures is written as

\[ R_{xx}(\tau) = E[x(t)x(t + \tau)^T] \]  

(2.11)
We can relate this correlation matrix with that of the sources, $s(t)$, transformed by the mixing matrix, $\mathbf{A}$:

$$R_{xx}(\tau) = \mathbf{A} R_{s\!s}(\tau) \mathbf{A}^T = \mathbf{A} E[s(t)s(t + \tau)^T] \mathbf{A}^T$$  \hspace{1cm} (2.12)

Since each $s_i(t)$ is independent from the others, $R_{s\!s}(\tau)$ is diagonal for any $\tau$. This, derived by Molgedey and Schuster (1994), reduces the blind source separation problem of finding $\mathbf{W}$ to solving the eigenvalue problem

$$(R_{xx}(\tau_1) R_{xx}(\tau_2)^{-1}) \mathbf{W} = \mathbf{W}(\Lambda_1 \Lambda_2^{-1})$$  \hspace{1cm} (2.13)

Ikeda and Murata (1998) show that this problem can also be solved by the simultaneous diagonalization of several time-delayed matrices:

$$\mathbf{WR}_{xx}(\tau_i)\mathbf{W}^T = \Lambda_i, \quad i = 1 \ldots r$$  \hspace{1cm} (2.14)

They obtain $\mathbf{W}$ in a two-step procedure, consisting of sphering and rotation. Sphering essentially performs Principle Component Analysis (PCA). It is used to find a matrix $\mathbf{V}$ which yields the following relation:

$$\mathbf{VR}_{xx}(0)\mathbf{V}^T = \mathbf{I}$$  \hspace{1cm} (2.15)

Rotation is an orthogonal transform that removes the off-diagonal elements of a correlation matrix. The objective is to find an orthogonal matrix $\mathbf{C}$ that minimizes the following:

$$\sum_{i=1}^{r} \sum_{i \neq k} \left| (\mathbf{CVR}_{xx}(\tau_i)\mathbf{V}^T \mathbf{C}^T)_{ik} \right|^2$$  \hspace{1cm} (2.16)

Finally, $\mathbf{W}$ is determined by

$$\mathbf{W} = \mathbf{C} \sqrt{\mathbf{V}^{-1}}$$  \hspace{1cm} (2.17)
Ikeda and Murata (1998) have obtained excellent separation results of real-world mixtures, using time-delayed decorrelation as the foundation for their algorithm.

## 2.5 Information Maximization

In 1995, Anthony Bell and Terrence Sejnowski laid out a theoretical framework for using an information-theoretic approach to object-based audio capture. They describe a neural network that adapts toward maximizing the information that is transmitted through an output sigmoid function. By maximizing this information transfer, the redundancy between output units is reduced. This redundancy reduction is, effectively, independent component analysis, or blind source separation. Using their “infomax” approach, Bell and Sejnowski have successfully separated unknown instantaneous mixtures of up to ten sound sources. In the following section, I will outline the information theory perspective on blind source separation.11

### 2.5.1 Information theory and blind source separation

As with Hérault and Jutten’s approach (see section 2.2.1), we begin with the assumption that the original sources are statistically independent from one another. This means that there is no mutual information between any two sources. This is written mathematically as:

\[ I(s_i, s_j) = 0 \]  

(2.18)

The mixtures, \( x(t) \), however, are statistically dependent on one another, and thus there is mutual information between each mixture. In terms of information theory, the objective of blind source separation is to find an unmixing matrix, \( W \), that re-establishes the mutual information condition, \( I(u_i, u_j) = 0 \), for the estimated sources, \( u(t) = Wx(t) \).
In order to make the estimated sources, \( u_j \), independent, we need to operate on non-linearly transformed output variables, \( y_i = g(u_i) \), where \( g(\cdot) \) is a sigmoidal function. The purpose of the sigmoidal function is to provide the higher-order statistics needed to establish independence. Please refer to Appendix A for a more detailed explanation of how information theory has been applied to blind source separation. The derived infomax weight update equation is:

\[
\Delta W \propto \left[ W^T \right]^{-1} \cdot \tilde{y}(t) x(t)^T
\]  

(2.19)

where

\[
\tilde{y}_i = \frac{\partial}{\partial y_i} \frac{\partial y_i}{\partial u_i}
\]

(2.20)

The two most common sigmoid functions used are the “logistic” function,

\[
y(t) = \frac{1}{1 + e^{-u(t)}}^{-1}
\]

(2.21)

and the hyperbolic tangent function,

\[
y(t) = \tanh(u(t))
\]

(2.22)

For these two functions, we derive \( \tilde{y}(t) \) to be \( (1 - 2y(t)) \) for the logistic function and \( (-2y(t)) \) for the hyperbolic tangent function.

2.5.2 Modern extensions

Bell and Sejnowski’s work (1995a) only considers instantaneous source mixtures; they do not derive a weight update rule for delayed and convolved sources, such as real-world audio signals.

Kari Torkkola (1996) expanded Bell and Sejnowski’s infomax algorithm to handle delayed and convolved sources. As discussed in section 2.2.4 of this

Object-Based Audio Capture
thesis, however, his choice of the feedback filter architecture inhibits the algorithm from learning non-minimum phase mixtures.

Shun-ichi Amari, et. al. (1996) proposed a more efficient "natural gradient" update equation than Equation 2.19 — one that speeds convergence:

$$\Delta W \propto [I + \hat{y}(t)u(t)^T]W$$  

(2.23)

Te-Won Lee, et. al. (1996) extended Torkkola's work and applied these update equations to a feedforward network architecture capable of learning unmixing filters that are non-minimum phase. They achieved good results, partially separating two sound sources recorded in a room at close-range (the sources were a little more than a half-meter away from the microphones).

Paris Smaragdis (1998) implemented Amari, et. al.'s natural gradient update equation (Equation 2.23) to run purely in the frequency domain, simplifying computation and speeding convergence.

Lee, et. al., in 1998, applied time-delayed decorrelation (described in section 2.4) as a pre-processor to their 1996 work and achieved better results, partially separating two people speaking 120cm away from two microphones in a conference room.

After experimenting with all of the algorithms discussed in this chapter, I feel that the infomax method will best suit my purposes. While the MBLMS algorithm performs well with gamma-distributed (speech-like) random noise as sound source inputs (discussed later in section 4.6.1), it fails for real speech signals. Ikeda and Murata's (1998) implementation of time-delayed decorrelation has produced excellent results for them, but I found it inflexible when trying to extend it to allow for more microphones or longer filter lengths; in the following section, I'll introduce my theory as to why more microphones will assist in object-based audio capture of a real-world environment.
2.6 Microphone Arrays

There has been a great deal of research into the use of beamforming microphone arrays for recording sounds in a reverberant environment (Flanagan, et. al., 1985). Beamforming arrays have been shown to improve the quality of sound capture by focusing on sounds emanating from a particular area of a room. The standard "delay and sum" beamforming algorithm time-aligns the signals recorded by each sensor in the array and then adds them together. The result is that signal components emanating from a desired location in the room combine coherently, while components from other locations combine incoherently. This effectively increases the signal-to-noise ratio (SNR) — i.e., the energy of the desired signals over the undesired noise — of the sound capture. The SNR is a monotonically increasing function of the number of sensors (Rabinkin, et. al., 1997), meaning that more sensors leads to clearer sound capture.

In an effort to take advantage of the SNR gains that microphone arrays can achieve, I will, in Chapter 4 of this thesis, extend these blind source separation and deconvolution algorithms to that of the overdetermined case, where there are more microphones than sources. In the next chapter, I'll begin with a study of room impulse responses and their inverses, and continue with experiments using ideal solutions of various unmixing filter configurations.
Chapter 3 Acoustic Mixtures

For the sake of simplicity and due to the fact that separating sound sources in a real-world environment is very difficult (see Chapter 4), I've limited all of my experiments to separating just two sound sources. Everything discussed in this thesis can scale to situations with more than two sources, as long as one uses at least as many microphones as there are sources to separate.

3.1 Room Impulse Responses

The most efficient way to experiment with real world signals is to generate them by taking a clean sound source (i.e. someone talking in a dry room\textsuperscript{13} with a microphone close to his/her mouth) and convolving it with a known impulse response of a reverberant room. The perceived output of this convolution is as though the sound source were actually recorded in the reverberant room. These signals are still artificial, but they're more useful to us in the early experimentation and design phase of this research.

There are two important reasons for experimenting with artificially generated signals. One of the biggest considerations in designing a blind source separa-

\textsuperscript{13}A dry room is a room that has negligible reverberation which enables one to record only the direct signal from the sound source.
Acoustic Mixtures

The BSSD algorithm is the length of the separating filters — shorter filters require less computation, and therefore less time is needed to perform the separation and deconvolution. By recording and analyzing room impulse responses, we can determine how long the separating filters need to be to achieve good results. The second advantage to using artificially generated signals is that, since the mixing filters are known, we can use them to determine what the ideal unmixing filters should be. Knowing precisely what the output of the system should be helps us debug the system, since we can easily compare our estimated solution with the ideal solution. In addition to that, we can perform a quantitative analysis on our results to objectively compare the effectiveness of the different algorithms that we design.

### 3.1.1 Acquiring room impulse responses

Using the system designed by Bill Gardner and Keith Martin (1994), I took impulse response measurements of a large conference room in our lab. Gardner and Martin’s software runs on a Macintosh computer equipped with an AudioMedia II DSP card (sampling at 44.1kHz, 16 bits/sample), and they use a maximum length sequence\(^{14}\) as the excitation signal. The impulse response is obtained by comparing the excitation signal that is sent out of the computer to the instantaneous response that comes back in (please see their technical report for more details about the algorithm).

The impulse response of a room is essentially a measure of all the paths a source signal takes to arrive at a specific location in the room. As a result, the impulse response is dependent upon the location of both the sound source and the receiving microphone. This means that several impulse response measurements are needed to cover the many possible source locations of the room.

The impulse responses are acquired in the following manner (see Figure 3.1 for a simplified illustration):

\(^{14}\)A random signal with a distribution similar to that of pink noise, often used for impulse response measurements. See Rife and Vanderkooy, 1989.
A room impulse response is determined through a comparison of an output excitation signal with the acoustically-transformed input response.

1. A maximum length sequence is generated on the Macintosh and then converted by the AudioMedia II DSP card into an analog audio signal.
2. The audio signal is sent out of the card into an NEC A-640 amplifier, driving an Electro-Voice Sentry 100A Studio Monitor loudspeaker, placed in various positions around the room.
3. As the impulse plays out of the loudspeaker, the AudioMedia card records the response of one pair of microphones (Audio Technica ATM-10a omnidirectional condensers), which hang from the lighting grid attached to the ceiling of the room. Since the AudioMedia card only accepts line-level inputs, I used a Mackie CR1604 mixer to amplify the microphone signals.
4. After the responses from the microphones are recorded, the impulse responses are computed on the Macintosh and saved locally to two separate files.
5. Since the AudioMedia card can only record two inputs at a time, I had to repeat each source location four times to get an impulse response from all eight microphones.

For better and for worse, the response of each component in the acquisition system is also incorporated into the impulse response. Since the microphones and the mixer used to acquire the impulse response data are the same used in recording real audio signals in the room, it is beneficial to have their responses be considered part of the acoustic effects of the room. However, the
AudioMedia card, the amplifier, and the loudspeaker are all components that are not used in the system that records the real audio signals — I certainly would not want the responses from these components to skew the true impulse response of the room. Since all of these components are of high quality and have reasonably flat frequency responses, I have proceeded with the assumption that their influences on the impulse responses are negligible.

3.1.2 The Garden Conference Room

The room I'm using for all my experiments is a 12'×24'×10' conference room. There's a large table in the middle of the room, and a bunch of A/V equipment in one corner. Two and a half walls of the room are covered with whiteboards and one wall is covered with a projection screen. A lighting grid and a projector hang from the ceiling. So, yes, it's not your ideal anechoic chamber! See Figures 3.2-3.4 for photos and a diagram (drawn to scale).

One important constraint for object-based audio capture is that the microphones should be inconspicuously placed in the room — hanging from the ceiling, for example; we want the microphones to be inobtrusive to people as they
freely move about the space. The large table in the middle of the conference room roughly divides the room into two areas from which people might speak, so I wanted to make sure the microphones would cover those areas. The hanging apparatus in the conference room enabled me to construct two linear microphone arrays, each with four elements (see Figure 3.3). Within each array, I spaced the microphones about a half-meter apart from each other, as suggested by Rabinkin, et. al. (1997) in optimum sensor placement. I chose
24 positions in the room that seemed like typical locations from which a sound might originate (see Figure 3.4 for a diagram of these locations):

- There are 16 positions around the conference table.
- Half of these are "seated" positions, where I placed the loudspeaker on the edge of the conference table, approximating the location of a person's mouth when he/she is sitting at the table.
- The other half are "standing" positions, where I elevated the speaker about five feet off the ground, and moved it about a foot and a half away from the table.
- The remaining 8 positions are in the corners of the room, where we typically place our loudspeakers, or where there might be some other ambient noises. For half of these positions, I placed the loudspeaker on the floor; for the other half, I raised the loudspeaker to the "standing" position.
Figure 3.4  Garden conference room (to scale). The solid black circles represent the position of the microphones, and the letters indicate the various positions of the loudspeaker during the capture of the room impulse responses. Two sets of measurements were acquired for the "Z" locations (one set on the floor, the other in the "standing" position).
3.1.3 Analyzing the impulse responses

To insure that I would capture the full response of the room, I set the acquisition software to compute responses of approximately 750ms. After downsampling the data to 11.025kHz, this equates to a 8,192-point response. A typical impulse response is shown in Figure 3.5.

![Typical impulse response from the Garden Conference Room](image)

**Figure 3.5** Typical impulse response from the Garden Conference Room

After convolving a few of these impulse responses with “clean” sources, I observed that the sources indeed sound as if they were recorded in the conference room. What dominates this convolved signal, however, is a strong characteristic low frequency murmur, which is an artifact of the room configuration. Rabinkin, et. al. (1996) use a high-pass filter with a 200Hz cutoff to remove this “room mode noise.” Following their example, I applied a 200Hz high-pass filter to the impulse responses before convolving them with
the sources. The perceived output is much clearer, and visually, the impulse response itself looks cleaner — see Figure 3.6.

![Graph](image)

**Figure 3.6** High-pass filtered impulse response

Although this post-processing might seem to corrupt the true impulse response of the room, it has the same effect as applying the high-pass filter to source signals recorded directly in the room (which one would do in a real-time system). Therefore, as long as I filter the incoming source signals with the same high-pass filter used in these impulse response experiments, the high-pass filter is simply considered a part of the impulse response.

### 3.2 Inverting the Room

In section 2.2.3, I described the acoustic-mixture model for object-based audio capture. Please recall Equation 2.3, where $s(t)$ is the vector of sound sources, $\mathbf{A}$ is the FIR polynomial mixing matrix, and $\mathbf{x}(t)$ is the vector of microphone signals:

$$\mathbf{x}(t) = \mathbf{A} \ast s(t)$$  \hspace{1cm} (3.1)
Chapter 3: Acoustic Mixtures

The goal of object-based audio capture is to find the matrix, \( W \), that, when convolved with the microphone mixtures, \( x(t) \), will produce estimates of the original sound sources, \( u(t) \):

\[
\begin{align*}
\mathbf{u}(t) &= \mathbf{W} \ast \mathbf{x}(t) \\
\end{align*}
\]  

The ideal solution for \( \mathbf{W} \) is simply the inverse of \( \mathbf{A} \). In his Ph.D. Thesis (1996), Russell Lambert described how standard scalar matrix manipulations apply to FIR polynomial matrices. For example, the following equations show how to invert a 2x2 matrix \( \mathbf{A} \), where each \( a_{ij} \) is an FIR polynomial:

\[
\begin{align*}
\mathbf{A} &= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \\
\mathbf{W} &= \mathbf{A}^{-1} = \frac{1}{a_{11} \cdot a_{22} - a_{12} \cdot a_{21}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}
\end{align*}
\]  

In the overdetermined case, however, \( \mathbf{A} \) is not a square matrix. Therefore, to find \( \mathbf{W} \), we need to find the pseudoinverse of \( \mathbf{A} \), as defined in the following equation:

\[
\begin{align*}
\mathbf{W} &= \text{inv} (\mathbf{A}^H \mathbf{A}) \ast \mathbf{A}^H
\end{align*}
\]  

Figure 3.7 shows a block diagram of how to obtain \( \mathbf{W} \) from \( \mathbf{A} \). The following is a step-by-step procedure:

1. First, to speed computation, transform \( \mathbf{A} \) into the frequency domain by applying a Fast Fourier transform (FFT) to each filter in the matrix, since convolution in the time domain translates to multiplication in the frequency domain.
2. Compute the pseudoinverse of the frequency domain matrix, \( \mathbf{A} \), using Equation 3.5.
3. Convert the pseudoinverse back into time domain by applying an inverse FFT to each filter in the matrix.
4. Since \( \mathbf{A} \) contains non-minimum phase filters (see section 2.2.4), its inverse will be anti-causal — rotate the leading weights of the time-domain inverse to the middle of the filters.
5. Finally, to “clean-up” the edges of the filters, apply a Hanning window to the shifted, time-domain inverse.
3.3 Filter Configuration

I conducted simple experiments to determine the answers to two fundamental questions about finding an optimal separating filter configuration: how short can the separating filters be to achieve an adequate separation; and is having more microphones than sources helpful to the blind separation and deconvolution of acoustically-mixed sounds. Naturally, the first place to look for answers to these questions is in the separating filters themselves.

3.3.1 What do the separating filters look like?

In a simple experiment, I used room impulse responses as mixing filters to directly determine what the corresponding ideal separating filters would look like. Specifically, I formed four different mixing matrices, one matrix for each...
Chapter 3: Acoustic Mixtures

of the following configurations: 2×2, 4×2, 6×2, and 8×2. Each matrix was comprised of the room impulse responses discussed in section 3.1. Given only these mixing matrices, I then used the inversion technique described in section 3.2 to determine the ideal separating matrices. Figure 3.8 shows one of the separating filters from each configuration.

![Typical separating filters](image)

**Figure 3.8** Typical separating filters for a 2×2, 4×2, 6×2 and 8×2 configuration.

Notice the characteristics that are similar in all of the filters: a sharp spike in the middle with decaying energy in both directions moving away from the spike. As for the differences, note that the 2×2 configuration is more dense than the other three configurations — each 2×2 separating filter clearly requires more information to separate the mixtures than the other three configurations.
Also observe that the range in amplitude of the separating filters decreases as the number of sensors in the configuration increases. This can be explained by the fact that each unmixing filter in an $M \times N$ configuration adds $M$ modified copies of a mixed signal to produce the output. Therefore, the more copies that are added together, the lower the amplitude for each copy.

An important corollary of this observation is as follows: in general, when using a blind deconvolution algorithm, (most of) the weights of the separating filter are initialized to zero. It is, therefore, beneficial if these slowly-adapting filters do not need to reach such high amplitudes to converge upon a solution.

### 3.3.2 Using ideal separating filters to separate acoustically-mixed sounds

Our ears are always the final judge, of course. The purpose of the following experiment is to learn more about how different filter configurations and filter lengths affect the quality of sound separation and deconvolution. For each configuration and filter length tested, the experiment proceeded as follows:

1. A mixing matrix of room impulse responses was created.

2. Acoustically-mixed sounds were generated by multiplying clean sounds\(^\text{15}\) by the mixing matrix.

   Using the appropriate impulse responses and sources, I created two sets of mixtures: for one set, I put a source in channel 1 and nothing in channel 2; for the other set, I put a source in channel 2 and nothing in channel 1. By processing the two mixtures in parallel, it was easier to determine the resultant SNR's (explained later).

3. Using the inversion technique described in section 3.2, I then used the mixing matrix to determine the ideal separating matrix.

   To vary the filterlengths, I applied $L$-point Hanning windows (with the windows centered around the peak of each filter) to the 8,192 tap separating filters, where $L$ is the desired filter length for each experiment.

4. Finally, I convolved the separating matrix with the mixed sounds to get the separated sources.

---

15. I used the speech and music samples from Dominic Chan's "Blind signal separation audio demonstrations" WWW page: http://www2.eng.cam.ac.uk/dbcc1/research/demo.html
Chapter 3: Acoustic Mixtures

Not only did I listen to the quality of the outputs, I quantitatively determined the signal-to-noise ratios (SNR) of the separated outputs, using the following formula:

$$SNR = 10 \log \left( \frac{E[s(t)^2]}{E[n(t)^2]} \right)$$  \hspace{1cm} (3.6)

where $E[\cdot]$ is the mean of the argument, $s(t)$ is the desired signal, and $n(t)$ is the undesired signal (the noise). In this particular experiment, the SNR will simply show how much louder the desired source is than the undesired source. The SNR results are shown in Table 3.1. During the experiment, I tried multiple source locations and used different combinations of sound sources (i.e. speech + music, speech + speech). There was no particular bias for any combination of source locations or the types of sounds used, so, for each filter configuration, I simply took an average of the SNR's that I obtained from each filter length.

<table>
<thead>
<tr>
<th>filter length (taps)</th>
<th>8192</th>
<th>4096</th>
<th>2048</th>
<th>1024</th>
<th>512</th>
<th>256</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>8x2</td>
<td>18.5</td>
<td>16.7</td>
<td>12.3</td>
<td>8.2</td>
<td>4.3</td>
<td>1.7</td>
<td>2.6</td>
</tr>
<tr>
<td>6x2</td>
<td>16.5</td>
<td>14.2</td>
<td>10.9</td>
<td>6.5</td>
<td>1.7</td>
<td>-1.5</td>
<td>0.6</td>
</tr>
<tr>
<td>4x2</td>
<td>14.5</td>
<td>12.5</td>
<td>6.7</td>
<td>2.1</td>
<td>1.0</td>
<td>-1.2</td>
<td>3.4</td>
</tr>
<tr>
<td>2x2</td>
<td>7.9</td>
<td>6.9</td>
<td>4.1</td>
<td>2.0</td>
<td>0.3</td>
<td>-0.6</td>
<td>-1.7</td>
</tr>
</tbody>
</table>

Table 3.1 SNR Measurements: shaded areas show good, consistent separation; italicized SNR values are invalid due to signal distortion.

As expected, longer filter lengths and more microphones yield better separation. With filter lengths of 1,024 taps, for example, using 8 microphones instead of 2 or 4 provides an additional 6dB of separation. Since it is generally more difficult for a BSSD algorithm to adapt to longer filters, these results encourage us to use overdetermined mixtures whenever possible. As I shortened the filters to 256 and 128 taps, the separating filters began to distort the signals, thereby making my SNR measurements invalid.
Aurally, 15dB of separation sounds as if the sound sources were *perfectly* separated; at about 6dB of separation, one can hear the other sound source, but it is *significantly* quieter; at 3dB of separation, one can hear both sound sources, the desired source being perceivably louder than the undesired source. At 6dB of separation, I consider the sources to be effectively separated. If we perform object-based audio capture in an environment similar to that of the Garden Conference Room, we will need to use the filter configurations and filter lengths that yield at least 6dB of separation, indicated by the shaded areas in Table 3.1.
Chapter 4 Implementation and Results

The main contribution of this thesis project is the application of blind source separation and deconvolution (BSSD) algorithms to unconstrained real-world environments. For the first part of this chapter, I will discuss experiments in and realizations about the “real world” which have helped refine my implementation of object-based audio capture. The remainder of the chapter consists of a thorough quantitative comparison of different filter architectures as applied to the separation of two speech sources in an acoustically-controlled room, concluding with an object-based audio capture experiment conducted in a typical conference room.

4.1 The Real World

While applying BSSD algorithms to real-world sounds and environments, I discovered how sensitive they are to the less-than-ideal conditions of such environments. In the following sections I will discuss how microphone choice, the distances between the sound sources and the microphones, the directivity of the sources, and room noise can all affect the performance of the algorithms.
4.1.1 Microphones

Although I had hoped to use the room impulse responses described in Chapter 3 for my experiments with object-based audio capture, initial attempts at using BSSD algorithms to find their inverses proved futile. The room impulse responses I collected simply had too many reflections and were too rich in energy — the algorithms couldn’t pick out the sound objects which had been mixed with these impulse responses. Since these algorithms are currently the most capable of implementing object-based audio capture, the best way to proceed, then, was to reduce the amount of reflections in the microphone signals, with the hope that the BSSD algorithms would perform better.

The simplest way to reduce reflections (while recording the audio signals) was to use less sensitive and more directional microphones. When choosing a microphone for any application, one must consider two important variables: the directional pattern and the transducer. The directional pattern of a microphone describes how sensitive it is to the sounds that surround it. The most common patterns are: omni, cardiod, and directional (see Figure 4.1). Omnidirectional microphones, for example, pick up sounds from all directions, whereas directional microphones only capture sounds from the direction in which the microphone is pointed. The cardiod microphone falls in-between omni and directional — it is sensitive to sounds that are in front of and to its sides, but not behind it. The two most common types of transducers used in microphones are dynamic and condenser. The difference between them, most relevant to this thesis, is that condenser microphones tend to be more sensitive, and have flatter frequency responses than dynamic microphones.

As part of my earlier experiments, I collected room impulse responses using omnidirectional condenser microphones; the omnidirectional pattern is preferred for acquiring as many reflections as possible, and the condenser transducer picks up more sound with its flatter frequency response. After realizing, however, that I needed to acquire less rich room responses, it became clear that dynamic cardiod microphones would be more suited to recording audio that is to be processed by a BSSD algorithm; the cardiod pattern significantly
cuts down on the amount of reflections the microphone receives, and dynamic microphones, specifically those designed for recording vocals, tend to have a shorter effective distance range — i.e. they tend to only pick up sounds that are near the microphone. Ultimately, I decided to replace my omnidirectional condensers with Shure SM57 cardioid dynamic microphones.

4.1.2 Sound source-microphone distances
During the process of programming and testing the BSSD algorithms described in Chapter 2, I ran tests on the same data that Te-Won Lee had used for his experiments. That way, I could compare my results with his to see if my programs were functioning properly. After confirming that I had a working set of BSSD functions, I then collected my own data to experiment with, using an array of cardioid dynamic microphones to record two sounds in a conference room, varying the types of sound sources used: two people speaking, one person speaking over a radio, and two people playing instruments.
I did not have much success, however, getting any of the BSSD algorithms to find the original sound objects from my recordings. Russ Lambert and Te-Won Lee offered me some advice on how to position the sound sources in the room so that the algorithms would perform better. In their experience, the algorithms tend to fix on the signal with the most energy from each microphone. In order to separate two sound sources with two microphones, for example, sound source A will need to have a stronger presence in microphone 1, and sound source B must have a stronger presence in microphone 2. The easiest way to create this effect is to put source A closer to microphone 1, and source B closer to microphone 2.

To my knowledge, this shortcoming has not been reported in any of the literature concerning blind source separation and deconvolution, yet it is clear that the distances between microphones and sound sources have a very strong influence on the performance of BSSD algorithms. The following are two critical consequences of this distance constraint.

**A really loud source**

Consider two sound sources: if one sound source is significantly louder than the other, dominating both microphone signals, the algorithm will not be able to find and extract the quieter sound from the mix. This could be advantageous, however, if the quieter sound is not important to the audio acquisition, i.e. if the quieter sound is just undesired background noise.

**A non-directive source**

When a person speaks, he/she sends his/her speech in a particular direction. His/her head, in fact, acts as a baffle, partially impeding the soundwaves from traveling in other directions. If an array of directional microphones is being used to record the speaker, the microphones that lie in the path of the speaker's voice will acquire the direct soundwaves, while the other microphones will acquire the reflected soundwaves at a lower amplitude.
Now consider a non-speech source — a musical instrument, such as a marimba (see Figure 4.2). When struck with a mallet, the “carefully selected Honduras Rosewood bars” (http://www.pearldrums.com) of this marimba resonate and project outward in all directions. If placed in a room with an array of microphones, this type of source would permeate the space, sending sound-waves of equal amplitude to each microphone, in addition to creating a great many reflections. Since each microphone in the array will pick up a substantial amount of sound from such a non-directional sound source, a BSSD algorithm would be inhibited from finding any other source that might also be in the room.

**Geometry**
Since the locations of the sound sources relative to the microphones have such a strong impact on the performance of BSSD algorithms, a suitable array geometry is vital for obtaining good results. The microphones must be placed in such a way that each desired source can be distinctly recognized within at least one microphone signal (see Figure 4.3).
Chapter 4: Implementation and Results

4.1.3 Room noise

Armed with a better understanding of the spatial constraints of BSSD algorithms, I recorded a new set of data in the conference room, but still had difficulties getting a BSSD algorithm to separate two sound sources. In one example, I recorded two males speaking simultaneously, with each one standing underneath a microphone in opposite corners of the array, and one speaking louder than the other. The BSSD algorithms seemed to converge on noise from a computer disk drive instead of the quieter speaker. There are two important reasons why this happened. First, the disk drive noise is a non-directive source, as described in the previous section; therefore, it has a presence in each of the microphone signals. The second reason, exemplified by the following experiment, is that the persistence and the invariance of the noise leads the algorithm to fixate upon it, neglecting the desired sounds.

Figure 4.3 An example arrangement of sources and microphones that would please a BSSD algorithm.

4.1.3 Room noise

Armed with a better understanding of the spatial constraints of BSSD algorithms, I recorded a new set of data in the conference room, but still had difficulties getting a BSSD algorithm to separate two sound sources. In one example, I recorded two males speaking simultaneously, with each one standing underneath a microphone in opposite corners of the array, and one speaking louder than the other. The BSSD algorithms seemed to converge on noise from a computer disk drive instead of the quieter speaker. There are two important reasons why this happened. First, the disk drive noise is a non-directive source, as described in the previous section; therefore, it has a presence in each of the microphone signals. The second reason, exemplified by the following experiment, is that the persistence and the invariance of the noise leads the algorithm to fixate upon it, neglecting the desired sounds.

Figure 4.4 shows a spectrogram\(^\text{16}\) of one of the microphone signals from the recording described above, and a spectrogram of the corresponding output of a BSSD algorithm. First, notice the faint horizontal energy bands, at about 7,100Hz and 6,500Hz, in the output of the BSSD algorithm. These energy

16. A spectrogram shows how the frequency content of a signal changes over time. Darker areas represent more energy.
bands are not apparent in the original microphone signal on the left. Evidently, the BSSD algorithm has strengthened the energy bands of an unwanted noise source — the computer disk drive that I heard while listening to this output.

In order to gain a better understanding of what the noise in the room “looks” like, I then recorded five seconds of room noise in the same space with no speakers present; the spectrogram of this recording is shown in Figure 4.5. Notice that the horizontal energy bands are much more apparent in this spectrogram, since the noise is shown in full-scale. Given that the BSSD algorithm moves through the mixture data in small chunks (about 0.25 seconds long), it is almost inevitable that the noise will appear in any given chunk. Since the algorithm receives more data from the noise than the desired sounds, it considers the noise to be more important, and, therefore, erroneously converges toward it.
Most of the object-based audio capture environments described in Chapter 1 are permeated by one or more steady state noises. In a typical office, for example, one might hear the constant whir of a computer disk drive, an air conditioner, or a fluorescent light; outdoor noises include automobile traffic and wind. All of these sounds are detrimental to the performance of an object-based audio capture system, and, therefore, must be suppressed as much as possible.

4.2 The Algorithm

In Chapter 2, I discussed three algorithms which I feel have the most potential for performing object-based audio capture: multi-channel blind least-mean-squares (MBLMS), time-delayed decorrelation (TDD), and information maximization (infomax). After experimenting with these algorithms, the infomax method emerged as the clear choice for my purposes. While the MBLMS algorithm performs well with gamma-distributed (speech-like) random noise
as sound source inputs (discussed later in section 4.6.1), it barfs on real speech signals. Ikeda and Murata's (1998) implementation of time-delayed decorrelation has produced excellent results for them, but I found it inflexible when trying to extend it to allow for more microphones or longer filter lengths; in Chapter 3, I showed that these extensions are necessary for implementing object-based audio capture in a real-world environment.

### 4.2.1 The procedure

Continuing, then, with the infomax method, I chose to use a slightly modified version of Paris Smaragdis's (1997, 1998) frequency domain implementation. The algorithm runs off-line and proceeds as follows (see section 2.5 and Appendix A for more information about the infomax method, and see Figure 4.6 for a block diagram):

1. Pre-process the time-domain input signals, \( x(t) \): filter out the ground hum and the room mode noise; subtract the mean from each signal.
2. Initialize the frequency domain unmixing filters, \( W \).
3. Take a block of input data and convert it into the frequency domain using the Fast Fourier Transform (FFT).
4. Filter the frequency domain input block, \( \hat{x} \), through \( W \) to get the estimated sources, \( \hat{u} \).
5. Pass \( \hat{u} \) through the frequency domain nonlinearity, \( y = \tanh(\text{real}(\hat{u})) + \tanh(\text{imag}(\hat{u})) \).
6. For square filter configurations, use \( W \), \( \hat{u} \) and \( y \) along with the natural gradient extension (Amari, et. al., 1996) to compute the change in the unmixing filter, \( \Delta W \).
7. For rectangular filter configurations, use \( W \), \( \hat{x} \) and \( y \) with the standard infomax update rule to compute \( \Delta W \).
8. Take the next block of input data, convert it into the frequency domain, and proceed from step 4. Repeat this process until the unmixing filters have converged upon a solution, passing several times through the data as necessary.
9. Normalize \( W \) and convert it back into the time domain, using the Inverse Fast Fourier Transform (IFFT).
10. Conolve the time domain unmixing filters, \( W(t) \), with \( x(t) \) to get the estimated sources.
Figure 4.6 Block diagram of steps 1-8 of my BSSD algorithm. The solid lines represent inputs into a mathematical operation. The dotted lines show which inputs are fed into the update rule.

4.2.2 Parameters

This algorithm contains a number of parameters that need to be optimized, including the FFT size, the input block length, the step size, the learning rate, and the momentum.

FFT Size

The Fourier transform is an orthogonal representation of a signal — that is, the Fourier coefficients are independent of one another. By implementing this algorithm in the frequency domain, each filter coefficient is updated independently. This property can speed the convergence of an adaptive filter (Haykin, 1996), but it can also lead to an erratic frequency spectrum. As suggested by Smaragdis (1998), and through conversations with Russ Lambert
and Te-Won Lee, I learned that using an FFT size that is much larger than the input block length will result in a smoother spectrum.

Input block length and step size
When using an FFT to convert a block of data into the frequency domain, it is assumed that the signal within the block is stationary. By this, I mean that the frequency content of the signal under a short time window is invariant. Real world audio signals, however, generally have strong non-stationary behavior; natural sounds are arguably considered to be strictly stationary for only under a short window length of about 10ms (Steiglitz, 1996). At a sampling rate of 16kHz, this corresponds to about 160 samples — much too short for any BSSD algorithm to get a sense of all of the delays and reflections embedded in the signals. By using a much longer block of data (250ms, for example), the frequency changes within the block average out over the duration to form an ensemble of data, which can statistically correspond to a wide-sense stationary signal — stable enough to pass through an FFT (Haykin, 1996).

Therefore, a large input block length is desirable, but still must be kept shorter than the FFT size to insure smooth unmixing filter spectra. One final note about input blocks: it is important to have the next input block overlap the current by at least 50%, to preserve the signal continuity (Steiglitz, 1996).

Learning rate and momentum
The output of the weight update equation is multiplied by the learning rate, $\mu$, which controls how fast the weights are updated. A high learning rate increases the amount by which the weights can change after each block of data has been processed, allowing the unmixing filters to update more rapidly. A learning rate that is too high, however, will allow the weights to change too much, causing the system to constantly diverge from the correct solution. A learning rate that is too low, on the other hand, can take a very long time (days, even!) to converge on a solution.
Chapter 4: Implementation and Results

One simple technique for stabilizing convergence is to use momentum:

\[ W(\tau + 1) = \mu \Delta W(\tau) + \eta W(\tau) \]  

The left-hand term in the above equation is the next state of the unmixing filter matrix. The first term on the right-hand side is the output from the weight update equation \( \Delta W \), multiplied by the learning rate, \( \mu \). The second term on the right-hand side is the current state of \( W \), multiplied by the momentum factor (sometimes called the forgetting factor), \( \eta \). By using momentum, the weight update equation (discussed in Chapter 2) additionally becomes a direct function of the current state of the weights. This, in effect, gives the adaptive filter some inertia towards arriving at a solution.

For my experiments, I used an FFT size of 16,384 points, an input block length of 4,096 samples (about 250ms at a sampling rate of 16kHz), a step size of 1,024 samples, a learning rate of 0.00025 and a momentum of 0.9. Furthermore, I allowed the algorithm to process at least 1 minute of data, using only a short sample (about 5 seconds long), but also to make several passes through it.

4.3 Varechoic Chamber Data

Room reverberation and noise both make object-based audio capture more difficult than it already is: sound sources in a large room with hard surfaces produce more reflections, creating more copies of the sound, which then have to be cancelled out; in section 4.1.3, I discussed the adverse effects of room noise on a BSSD algorithm. Since the data that I had been using from the Garden Conference Room was both noisy and reverberant, I had been getting very poor BSSD results. To continue with my experimentation on the algorithm, then, I chose to use data from a more controlled environment — the Varechoic Chamber described in the following section. Researchers can vary the reverberation characteristics of the chamber, from relatively “dead” to very “live.” By experimenting with data from this room, I would then be able to test the BSSD algorithm in a variety of conditions.
4.3.1 Data Acquisition

Last year, the CAIP Center’s Microphone Array group at Rutgers University made a series of multichannel speech recordings at Lucent Technologies’ Varechoic Chamber, in Murray Hill, New Jersey (Ward, et. al., 1994). The Varechoic Chamber is a $6.7 \times 6.1 \times 2.9$ meter room whose walls are comprised of sliding panels that are computer controlled to vary the reverberation time of the room between 0.1 and 0.9 seconds.

The CAIP group acquired the data as follows. Two sets of four omnidirectional condenser microphones (mounted in acoustically absorvent material) were placed on orthogonal walls of the room. The microphones were arranged in rectangles 30cm high and 26cm across, and were connected to a computer with a Signalogic DSP card, capable of simultaneously recording eight channels of audio, while playing out eight channels at a sampling rate of 16kHz. A loudspeaker, connected to an analog audio output of the DSP card, was placed in four different positions around the room. They used two speech signals, one male and one female, as sound sources for their data acquisition. It is apparent from the data that the male speech sample was generally louder than the female speech sample. For each loudspeaker position, they acquired speech signals in four different reverberation settings: 0.1, 0.25, 0.5 and 0.9 seconds. See Figure 4.7 for the physical layout of the experiment (Renomeron, 1997).

4.3.2 Some characteristics of the data

Unfortunately, two of the microphone channels of their data are corrupted with a loud 60Hz ground hum. This noise was easily removed with a comb filter tuned to attenuate the 60Hz frequency band and all its harmonics. The data was then sent through a high-pass filter to remove any room mode noise (as discussed in section 3.1.3).

Keeping in mind the importance of the distances between the sound sources and the microphones, I chose to work with signals from positions 1 and 2, which are about 3.2m and 2.6m, respectively, from their nearest microphone.
As mentioned earlier, the male speech sample is louder than the female speech sample. A consequence of this, as discussed in section 4.1, is that the female sample is more difficult to separate out from the mixture. Table 4.1 shows the signal-to-noise ratios (SNR's) of the four mixtures used in this experiment. The SNR's, calculated using Equation 3.6, quantitatively show (in dB) how much louder one source is than the other.
Experimentation

<table>
<thead>
<tr>
<th>reverb time (seconds)</th>
<th>male/female SNR</th>
<th>female/male SNR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mic 3</td>
<td>mic 8</td>
</tr>
<tr>
<td>0.1</td>
<td>4.5dB</td>
<td>-0.7dB</td>
</tr>
<tr>
<td>0.25</td>
<td>3.5dB</td>
<td>-1.5dB</td>
</tr>
<tr>
<td>0.5</td>
<td>6.1dB</td>
<td>-6.1dB</td>
</tr>
<tr>
<td>0.9</td>
<td>5.1dB</td>
<td>-3.5dB</td>
</tr>
</tbody>
</table>

Table 4.1 This table shows, for each reverberation setting of the Varechoic Chamber, the comparative signal power of one source to another. For example, in the 0.1 second mixture: in microphone 3, the male source is 4.5dB louder than the female source; in microphone 8, the female source is 0.7dB quieter than the male source.

4.4 Experimentation

In section 3.3.2, I showed that we can get better performance out of an object-based audio capture system if we increase the unmixing filter lengths and/or the number of microphones. In the following sections, I will discuss experiments that I ran using the BSSD algorithm described in section 4.2 along with the Varechoic Chamber data described in the previous section. These experiments will compare how well the BSSD algorithm performs under varying filter configurations and filter lengths. In addition, by using data from the Varechoic Chamber, I can also show the effects of reverberation on object-based audio capture.

4.4.1 SNR improvement

Since the mixture data is biased toward the male speaker, I am reporting the average SNR improvement in the distinction between sources, which is an accurate representation of how well the BSSD algorithm is performing. To arrive at the SNR improvement, I compare the output SNR's of the BSSD algorithm with the input SNR's listed in Table 4.1, to determine how much further the sources have been separated after being processed. For example, looking at the data set with a 0.1 second reverberation time, if the SNR of the
male sample after processing is 8.5dB and that of the female sample is 6.7dB, then the overall SNR improvement will be the average of the two differences:

\[
\frac{(8.5 - 4.5) + (6.7 - (-0.7))}{2} = 5.7 \text{dB}
\] (4.2)

4.4.2 A word about unmixing filter lengths

It is important to note two restrictions on the unmixing filter lengths. The nature of the BSSD algorithm encourages using lengths of even powers of 2 — i.e. 512, 1,024, 2,048, 4,096, etc. When running the algorithm with unmixing filter lengths of 8,192 taps, however, the filters began to distort the signals to the extent of rendering them unlistenable. Therefore, my experiments do not include filters lengths longer than 4,096 taps.

4.5 Reverberation

Because I was having so much difficulty getting a BSSD algorithm to find the sound objects in the data that I had recorded from the Garden Conference Room, I was very interested in exploring how reverberation impairs an object-based audio capture system.

For this experiment, I used a simple 2x2 filter configuration with data from four different reverberation settings of the Varechoic Chamber. To generate more data for comparison, and to observe how the length of the unmixing filters affects object-based audio capture, I also varied the filter lengths for each trial run.

The results of this experiment (as well as the experiments described in the remainder of this chapter) are tabulated in Appendix B. Figure 4.8 shows a bar graph of the results. As expected, the separation is generally better when longer unmixing filter lengths are used. More importantly, the results show that even a small amount of reverberation severely degrades the performance of the BSSD algorithm.
4.6 Overdetermined Blind Separation

The most significant hypothesis to come out of the experiments I discussed in Chapter 3 is that we can greatly improve the performance of object-based audio capture by simply adding more microphones to the system. Such a system, with more sensors than sources, is classified as overdetermined, since there are more than the minimum number of data channels needed to solve the BSSD problem. In the following sections I will discuss some experiments that use overdetermined mixtures for object-based audio capture, comparing the performance of BSSD algorithms with various filter configurations.

4.6.1 A Simple experiment

I will begin with a simple experiment using truncated room impulse responses as mixing filters (see Figure 4.9). Out of the three algorithms discussed in Chapter 2, only the MBLMS algorithm had been implemented with overdeter-
mined mixtures (Lambert, 1996). Therefore, for the sake of simplicity, I used an MBLMS algorithm with gamma-distributed (speech-like) random sources. For each of the four filter configurations tested in Chapter 3 — 2x2, 4x2, 6x2 and 8x2 — I set the MBLMS algorithm to learn unmixing filters of 512 taps.

![Figure 4.9](image)

**Figure 4.9** A shortened room impulse response. The sampling rate for this filter is 11.025kHz.

Since, in this experiment, I know what the mixing filters are, I can use the Multichannel Intersymbol Interference (ISI) performance metric (Lambert, 1996) to quantitatively compare how well the algorithm separates mixtures of different filter configurations. The Multichannel ISI shows how close the learned unmixing filters are to the ideal solution:

$$\text{ISI}_i = \frac{\sum_j \sum_k |s_{ij}(k)|^2 - \max_{j,k} |s_{ij}(k)|^2}{\max_{j,k} |s_{ij}(k)|^2}$$  \hspace{1cm} (4.3)
where \( s_{ij} \) are the filter elements of the mixing matrix, \( W \), convolved with the separating matrix, \( A \). The ISI for each source, \( i \), converges to zero for a perfectly learned unmixing matrix. Figure 4.10 shows a comparison plot of the ISI measurements from each of the four filter configurations tested.

![Figure 4.10 ISI measurements from an MBLMS algorithm for four different filter configurations.](image)

I ran the MBLMS algorithm for 200,000 samples for each of the four filter configurations. The plots show the way in which each configuration converged on the solution. It should be clear from these plots that the MBLMS algorithm performs significantly better when the number of sensors is increased. These results encouraged me to continue working with overdetermined mixtures.

### 4.6.2 Direct filters and cross filters

One fundamental problem with the BSSD algorithms discussed in Chapter 2 lies within the model used to describe the notion of object-based audio capture. These algorithms are all based on the Hérault-Jutten network, shown in Figure 2.2, which assumes that there are direct filters, and cross filters. The direct filters are initially assigned to unity, and in some cases
(Torkkola, 1996), are fixed there. The cross filters are initially set to zero, and then adapt to cancel out the unwanted sources from the direct channels.

In an \(N \times N\) configuration, each microphone has assigned to it one and only one direct-filtered mixture. In an overdetermined case, however, where there are more microphones than sound sources, how do we decide which microphone signals should be assigned to direct filters, when there can only be as many direct filters as there are sources? (see Figure 4.11)

![Diagram showing microphone signals and estimated sources.](image)

**Figure 4.11** In this figure, there are three microphone signals and two sources to estimate. Notice that each source is determined by a filtered combination of three microphone signals. Since there can't be two microphone signals directly filtered to any one source, how do we determine which microphone signals should be assigned to direct filters?

Ideally, this assignment should be arbitrary — all of the filters should work together and adapt accordingly. Experimentation with BSSD algorithms proves otherwise — these algorithms tend to adapt the cross filters to cancel out the unwanted sounds from the direct-filtered channels, leaving the direct filters relatively unaltered. As an example, Figure 4.12 shows the unmixing filters for a 2x2 configuration, as determined by the BSSD algorithm described in section 4.2. Observe that the direct filters very closely resemble the delayed impulse filters that they were initialized to. The cross filters have been adapted to cancel the undesired sources from the direct-filtered signals.
This observation suggests that, for an overdetermined case, the assignment of the direct filters is important — it involves picking the microphone signals which all of the other signals will reference when adapting their cross filters. Intuitively, then, each direct filter should be assigned to the microphone signal which contains the cleanest copy of the desired source. This notion is an extension of the discussion from section 4.1.2, where I stated how vital it is to a BSSD algorithm that each sound source have a distinct presence in at least one microphone signal.
4.6.3 Experiments with the Varechoic Chamber data

I will now discuss experiments with overdetermined mixtures using data from the Varechoic Chamber. Most of the modern BSSD algorithms in use today are restricted to working with $N \times N$ unmixing filter configurations. The three algorithms discussed in Chapter 2 allow for overdetermined, non-square filter configurations, but the more recent development work has unfortunately been steered towards using square configurations. As a consequence, I must use the older infomax update rule, instead of the optimized rule proposed by Amari, et. al. (1996), in my BSSD algorithm. The original infomax equation does not converge as well, and it is dreadfully slow (due to the weight matrix inversion).

Adding more unmixing filters significantly slows down computation — in an $8 \times 2$ configuration, the BSSD algorithm needs to adapt 16 filters, while in a $2 \times 2$ configuration, there are only 4 filters to adapt. Due to this extra load and the slow update rule, I had to decrease the number of times the algorithm passed through the data, to the point where the system had not fully converged on a solution. One further note: in each of the following experiments, I assigned the direct filters to the microphones closest to the sources.

In the first experiment, I used data with a reverberation time of 0.1 seconds. As before, all of the results from the experiments discussed in this chapter are tabulated in Appendix B. The SNR improvements in Figure 4.13 show that the overall performance of the BSSD algorithm drops when more microphones are added. However, looking at the separation SNR's, the separation of the female voice improves with additional microphones, while the separation of the male voice worsens.

I had expected the overall performance to increase with additional microphones. As these initial results seemed contradictory, I proceeded to try the other reverberation settings from the Varechoic Chamber data. Figure 4.14 shows the separation results for a reverberation time of 0.25 seconds.
Figure 4.13  SNR's (in dB) for a reverberation time of 0.1 seconds.

Figure 4.14  SNR's (in dB) for a reverberation time of 0.25 seconds.
These results show different trends than that of the previous data set. The importance of using longer unmixing filter lengths becomes more apparent in this data set, where there is more reverberation in the mixtures. Comparing the results across the different filter configurations from this data set, the 4x2 configuration slightly outperforms the 2x2 configuration, while the 6x2 and 8x2 configurations similarly don’t perform as well as either of the smaller configurations.

Since the results from the last two data sets were inconsistent I decided to run the BSSD algorithm on the final two sets, with reverberation settings of 0.5 seconds and 0.9 seconds. Figures 4.15 and Figure 4.16 show the separation results for a reverberation time of 0.5 seconds and 0.9 seconds, respectively. The SNR improvement results from the last two data sets agree with the trends of the first data set: increasing the unmixing filter lengths improves performance, while adding more microphones does not seem to help (in Chapter 5, I will discuss my thoughts on why).

![Figure 4.15 SNR's (in dB) for a reverberation time of 0.5 seconds.](image)
4.7 Square Configurations

Since the results from the experiments described in Chapter 3 seemed to strongly demonstrate the benefits of using more microphones than sources, I didn’t want to give up on the hypothesis just yet. In this section, I will discuss experiments I did with N×N unmixing filter configurations. In an N×N configuration, we use N microphones to find N sources. In the following experiments, however, I’ll use N×N configurations to find only two sources, where, for configurations with more than two microphones, each of the N outputs will contain estimates of either of the two sound objects.
Chapter 4: Implementation and Results

The following results are presented in the same format as in the last section — for each reverberation setting, I'll show the separation SNR's between the male and female voices, followed by the separation improvement SNR's. For each configuration, I chose the best-separated outputs from the BSSD algorithm. To compute the improvement SNR's, I used the corresponding microphones to the best-separated outputs as the initial SNR's of the signals.

Figures 4.17 and 4.18 show the results for a reverberation setting of 0.1 seconds and 0.25 seconds, respectively. Looking at the SNR improvement results, it is clear that, for these particular sets of data, adding more microphones does improve the performance of the BSSD algorithm. For the data from the 0.1 second reverberation setting (Figure 4.17), the 4×4, 6×6, and 8×8 configurations all produced slightly better results than the 2×2 configuration, the 4×4 yielding the best performance of the three. In the more reverberant 0.25 second environment (Figure 4.18), the SNR improvement results show that the performance of the BSSD algorithm steadily increased with the addition of more microphones.

For the 0.5 second and the 0.9 second environments, however, the results for the varied unmixing filter configurations are somewhat inconsistent. Looking at Figures 4.19 and 4.20 (showing the results for the 0.5 second data and the 0.9 second, respectively), it appears that adding more microphones improves the separation of the female voice, but either produces little change, or worse, reduces the separation of the male voice. The average SNR improvements reflect this disparity — there doesn't seem to be any consistent trend in performance as more microphones are added.

In spite of these inconsistencies, looking at all of the results presented in this section, the separation of the female voice is always improved by adding more microphones. This observation shows the strong potential for square unmixing filter configurations in an object-based audio capture system.
Square Configurations

Figure 4.17 SNR's (in dB) for a reverberation time of 0.1 seconds.

Figure 4.18 SNR's (in dB) for a reverberation time of 0.25 seconds.
Figure 4.19  SNR’s (in dB) for a reverberation time of 0.5 seconds.

Figure 4.20  SNR’s (in dB) for a reverberation time of 0.9 seconds.
Perhaps there are more advantages than disadvantages to using a square configuration over an overdetermined configuration. Some advantages are: we don't have to make a decision about which channels to assign a direct filter; we can use the natural gradient extension to the infomax algorithm; and the algorithm can find up to $N$ sound objects in the mixture. One disadvantage is that we don't have an automatic method for choosing what the best outputs are — we have to listen to each output individually.

4.8 A Conference Room

In this chapter, I have:

- discussed a number of different experiments that address the challenges of performing object-based audio capture in a real world environment;
- outlined a BSSD algorithm that I feel is aptly suited for object-based audio capture;
- thoroughly experimented with how reverberation, and the configuration and length of the unmixing filters affect a BSSD algorithm.

My goal for the final experiment of this thesis was to use the knowledge acquired thus far to apply object-based audio capture in a real-world setting — namely, the Garden Conference Room described in Chapter 3 and section 4.1. I did not run a comprehensive set of tests, but simply wanted to show that the algorithm can separate sounds in an unconstrained environment.

I made three, 15 second recordings of two people speaking simultaneously from different positions in the conference room. For each recording, I made sure that each person was standing near one of the eight microphones that I had hanging from the ceiling of the room, to insure that at least one of the microphones would have a strong direct signal to work with. The reverberation time of the room is about 0.2 seconds, so there were a significant number of reflections present in the microphone signals. As discussed in section 4.1.3, there was also a marked amount of steady noise that permeated the room.
After listening to the signals obtained in these recordings, it seemed that, in all cases, the microphone that was positioned above each person captured his/her speech so well, that the other person was barely audible in that signal — the signals were already adequately separated! The purpose of these experiments was to test the ability of a BSSD algorithm to perform object-based audio capture when given a mixture of sound sources. For the sake of experimentation, then, I chose to exclude these already-separated microphone signals — instead, I used the next-nearest microphones to each person, where there was more of a balance of both sources in each signal.

I then applied the BSSD algorithm to each of the three sets of data, using a 2x2 unmixing filter configuration, with filter lengths of 4,096 taps, and I let the algorithm make 100 passes through the mixture data. After comparing the quality of separation in the outputs, I chose to work with the data set that yielded the best separation. Please refer to Figure 4.21 for an accurate layout of these microphones and sound sources.

Using this data set, I experimented with 2x2 and 4x4 unmixing filter configurations with filter lengths of 1,024, 2,048 and 4,096 taps. Because there was so much noise present in the room, the additional microphones in the 4x4 configurations seemed to add more noise than the desired sounds. Unmixing filters of 1,024 taps were barely long enough to capture the delays between the sources, while 4,096-tap unmixing filters slightly distorted the desired sound sources. The 2x2 configuration with filter lengths of 2,048 taps worked best.

One can certainly appreciate the detrimental effects of room noise and reverberation by listening to the microphone signals from the conference room and the sound objects that the BSSD algorithm found. In the output containing the female voice, the male voice isn't attenuated much, but it is perceptually “blurred” to the point of making it unintelligible in comparison to the female voice. In the output containing the male voice, the female voice is signifi-
Figure 4.21 The sources and microphones that I used for my final experiment in the Garden Conference Room. The microphones are represented by black circles. Source A is a female speaker and source B is a male speaker. The arrows indicate the directions that they were facing.
cantly attenuated, but the reverberation in the signal seems to be stronger. While not perfect, the algorithm does a decent job of finding the important sound sources in the room, in spite of the noise and reverberation. See my WWW page for an audio demonstration:
http://www.media.mit.edu/~westner/sep.html
Only in the last decade of blind source separation and deconvolution research has the implementation of object-based audio capture seemed feasible. Beginning with the seminal work of Jeanny Hérault and Christian Jutten (1986) on blind source separation, we have advanced to using FIR filter configurations capable of separating acoustically-mixed sources (Torkkola, 1996). This thesis has explored just how far along we have come towards a reliable object-based audio capture system.

5.1 Conclusions

As researchers in blind sound separation and deconvolution, we work with relatively large blocks of data, which complicates our analyses — it is a cumbersome process to break down a 16,384-point FFT, or scrutinize a few filter taps in a 4,096-tap anti-causal unmixing filter. As we increase the number of microphone inputs, and consequently, the number of unmixing filters, we generate an abhorrently large amount of data to examine! My point here is that the research presented in this thesis, particularly on the use of overdetermined mixtures, is in its earliest stage. Further experimentation is necessary,
beginning with a thorough analysis of where and why the overdetermined configurations faltered in the experiments discussed in Chapter 4.

5.1.1 Overdetermined mixtures
At the conclusion of Chapter 3, I showed that by using overdetermined mixtures, we can gain a significant performance improvement in the extraction of acoustically-mixed sounds. In my experiments with object-based audio capture, however, overdetermined mixtures often degraded the system. These results are disappointing, but also preliminary. To my knowledge, there has not yet been any other published work concerning the use of overdetermined mixtures in BSSD algorithms. (Russ Lambert coded a Matlab implementation of a 3x2 configuration in an appendix to his Ph.D. Thesis (1996) but he did not discuss any results.)

At this point, I can merely speculate as to why overdetermined configurations proved detrimental to the performance of the BSSD algorithm. Perhaps the added microphone signals were ill-conditioned (as per the discussion in section 4.1) in such a way that they added an ambiguity into the system that led it to diverge from the proper solution. Another possible point of failure is in the frequency domain nonlinearity. According to Lee and Sejnowski (1997), "the form of the nonlinearity plays an essential role in the success of the algorithm." The addition of more microphone signals will change the shape of the inputs into the nonlinearity, something I did not account for in my BSSD algorithm. Perhaps a frequency domain contextual ICA (Pearlmutter and Parra, 1997) needs to be developed, where the nonlinearity is in a parametric form that adapts to the input data. As mentioned in section A.4, the information theoretic BSSD algorithms are not guaranteed to find statistically independent sources unless the input pdf's are perfectly aligned with the slope of the sigmoidal function.

5.1.2 Acoustics and noise
As reflected (pun intended) in the experiments discussed in section 4.5, reverberation has a large negative impact on the performance of object-based audio capture. While BSSD algorithms can, to some degree, find and cancel out the
direct waves of the undesired sounds in a mixture, they have great deal of trouble removing the reflected copies of the all sounds. In an outdoor, or free-field environment, there are very few reflections, if any; therefore, reverberation ceases to be an issue.

Neither indoor nor outdoor environments, however, are immune to unwanted noise. As explained in section 4.1.3, constant noises such as computer fans, air conditioners, automobile traffic, and wind can influence a BSSD algorithm a great deal more than the sound objects we wish to find. Since current BSSD algorithms are so sensitive to the environments in which they are used, an object-based audio capture system will only perform reliably in an acoustically-treated environment devoid of constant noises.

5.1.3 Are we better off than just doing nothing with these mixtures?
Yes!

In Chapter 4, I ran a total of 160 trials of the BSSD algorithm, on various unmixing filter configurations and lengths, and reverberation settings, and reported a separation improvement metric for each trial. In only 2 of these trials, the separation improvement SNR was negative, meaning that the algorithm actually worsened the isolation between sounds. That's a success ratio of just under 99%!

5.2 Future Directions
In section 5.1.1, I mentioned a few specific details of the BSSD algorithm that need to be examined more closely. In this section, I'd like to propose higher level issues that need to be addressed in the short-term future of object-based audio capture research.

5.2.1 Revisiting microphones
After taking some time to carefully listen to the recordings I obtained from the conference room, I realized one drawback of using directional microphones: if a sound source is not positioned in the direction of the microphone, then the
microphone will only pick up the reflections of that sound source, and none of the direct sound. Given that BSSD algorithms have a great deal of trouble dealing with reflections, perhaps directional microphones are not the right choice for object-based audio capture. Perhaps it is more important to the algorithm that the microphones get the direct sound from each source, in lieu of picking up more reflections.

More experimentation needs to be done with different types of microphones. The next type of microphone I would try would be a “boundary” microphone. Shown in Figure 5.1, a boundary mic mounts flat on the surface of a wall, ceiling, floor, etc. Boundary microphones will acquire the direct path from a sound source in a room, and will eliminate some reflections — when the microphone is mounted on the wall, for example, it does not pick up any reflections off of that wall.

![Figure 5.1 An Audio-Technica AT841a boundary microphone. Its actual diameter is 2.56 inches (65mm).](image)

### 5.2.2 Room noise

In section 4.1.3, I addressed the adverse affects of room noise on a BSSD algorithm. To find the steady state noises in a room, one can simply record a few seconds of data (without any desired sound sources present) then compute...
the room's spectrogram, like the one shown in Figure 4.5. The steady state noise frequencies should be clear in the spectrogram. One possible way of removing these noises is to design several notch filters that will cancel out these specific frequency bands. This may, however, necessitate a long setup time for object-based audio capture.

Going one step further, we should be able to program the BSSD algorithm to automatically detect, and then ignore, steady-state frequency components. The frequency components of speech, for example, will smoothly change over just a small duration of, say, 500ms, while the noise bands shown in Figure 4.5 persist for as much as five seconds.

### 5.2.3 Picking inputs and outputs

When there are more microphones than sound sources, we have two options as far as what type of unmixing filter configuration to use. Both options currently require some human intervention: in an overdetermined configuration, we need to determine which microphone signals should be assigned to direct filters (see section 4.6.2), and in a square configuration we need to decide which are the best-isolated sound objects. It would be useful to us if these choices were automatically determined by the BSSD algorithm, enabling the object-based audio capture system to run more autonomously.

**Source location — choosing the best inputs**

In section 4.1.2, I discussed the importance of the geometric relationships between microphones and sound sources. In my experiments, I used these relationships to choose which microphones to use as direct-filtered signals. To integrate this decision into a BSSD algorithm, we could implement a simple source location algorithm (Svaizer, et. al., 1997) to find the positions of the sources in the room; then, using these positions, we could provide the algorithm with some geometric rules to determine which microphones to use as the direct signals.
So many outputs — which are the best?

Programming a BSSD algorithm to automatically choose the best output from a square configuration would be beneficial to us. At this time, however, I have not found an easy way to do this. I've only looked at some simple statistical properties of the outputs and was unable to consistently correlate any metric with the best-isolated sound object — I had to pick the best outputs by listening to each one.

5.2.4 Initialization

In all of the BSSD algorithms that I've come across, the direct unmixing filters are initialized to a unit impulse, while the cross filters are zeroed. Clearly, a better way to initialize the unmixing filters is to make them more closely resemble the ultimate solution that the BSSD algorithm is trying to find. We will get better initial estimations for these unmixing filters if we first find the time delays between the sound objects in the mixture. There is already a plethora of research on time-delay estimation (Jian, et. al., 1998), and its application to beamforming and sound source location (Rabinkin, et. al., 1996). We can use the time-delays in a manner similar to beamforming approaches, and setup initial spikes in each of the unmixing filters that would, in effect, cohere the time-shifted sound sources. In addition to giving the BSSD a better initial guess at the solution, this would also eliminate the distinction between direct filters and cross filters, which I've found to be a very restrictive way to model the source separation problem.

5.2.5 Real-time?

Of course, the ultimate goal for an object-based audio capture system is to record individual audio objects as they arrive into the digital audio system. Real-time source separation has, thus far, effectively been done only with simple mixtures (Murata and Ikeda, 1998; Smaragdis, 1997, 1998). When working with more complex, real-world mixtures, we must generally use a slower learning rate, and feed the algorithm minutes of data before it will converge on an acceptable solution. Current algorithms need to be made parallel and run on faster computers to function in real-time.
5.2.6 Other potential BSSD algorithms
Interest in blind source separation and deconvolution has snowballed over the past few years. A quick glance through recent conference proceedings and journals turns up several different BSSD algorithms and approaches. For this thesis project, I chose the few that have already been applied to acoustically-mixed sound sources.

I feel that Ikeda and Murata’s (1998) implementation of time-delayed decorrelation (TDD) shows great potential. Their online demos are convincing, and they have already discussed a real-time adaptation of their algorithm (Murata and Ikeda, 1998). Their algorithm, however, is relatively involved, and I was unable to expand it to allow for more microphone inputs or longer unmixing filter lengths (though theoretically it will scale to larger configurations).

5.3 Final Thought
In controlled “laboratory” environments, BSSD algorithms are very adept at isolating sound objects. When placed in the real world, however, they are almost completely lost. Through the experiments I discussed in Chapter 4, I showed that BSSD algorithms perform extremely well for a room with very little reverberation and noise. I feel, therefore, that we need to focus more of our attention on the practical acoustics of object-based audio capture: i.e., which types of microphones should be used and how should they be placed; and how can we effectively cancel out constant room noises. If we put more effort into acquiring signals without reflections and noise, then our current BSSD algorithms will succeed.
In this appendix, I detail the steps that Bell and Sejnowski (1995a, 1995b) used to derive a blind source separation weight update equation based on information theory.

A.1 Mutual Information

Sources in a mixture are statistically independent of one another. From an information theory standpoint, this means that there is no mutual information between any two sources, $s_i$ and $s_j$. This is written mathematically as $I(s_i, s_j) = 0$. Mixtures of sources, $x(t)$, on the other hand, are statistically dependent upon one another and thus there is mutual information between them.

In blind source separation, the goal is to find an unmixing matrix, $W$, such that when multiplied with the mixtures, $x(t)$, will yield outputs, $u(t)$, that satisfy the condition $I(u_i, u_j) = 0$, where $u(t)$ is essentially the best guess at the original sources. By satisfying the above condition, these estimated sources will be statistically independent of one another.
A.2 Sigmoidal Function

To make the estimated sources independent of one another, we need to operate on non-linearly transformed output variables, \( y_i = g(u_i) \), where \( g(\cdot) \) is a sigmoidal function. Bell and Sejnowski (1995b) generally define a sigmoidal function as "an invertible twice-differentiable function mapping the real line into some interval, often the unit interval: \( \mathbb{R} \rightarrow [0,1] \)." The sigmoidal function provides the higher-order statistics needed to establish independence, and, more importantly, it helps maximize the information transfer from input to output.

A.3 Entropy

Entropy is a measure of the uncertainty of a random variable. Maximum entropy is at zero, where we are absolutely certain of the outcome of the random variable. The entropy of the sigmoidally transformed output, \( H(y(t)) \), is maximum when the sigmoid function matches the distribution of the input, \( u(t) \). Thus, by maximizing the information transfer from input to output, we are also maximizing the entropy.

The joint entropy of two sigmoidally transformed outputs, \( H(y_1, y_2) \), can be defined in the following relation:

\[
H(y_1, y_2) = H(y_1) + H(y_2) - I(y_1, y_2)
\]  
(A.1)

The joint entropy is maximized when each sigmoid function matches the distribution of each \( u_i \), and \( y_1 \) and \( y_2 \) are independent from one another. This maximum value occurs when \( u_i = s_i \), which means the sources are separated. Therefore, we can do blind source separation by maximizing the joint entropy, \( H(y(t)) \), of an output which has been transformed by sigmoidal functions that match the distribution of the input sources that we're trying to find.
A.4 Stochastic Gradient Ascent

We can maximize the joint entropy, $H(y(t))$, through a standard stochastic gradient ascent algorithm, as described below. By definition, the joint entropy is a function of the probability density function (pdf) of its arguments:

$$H(y(t)) = -E[\log f_y(y)] \tag{A.2}$$

We can write the pdf of the output, $y$, as a function of the pdf of the input, $x$:

$$f_y(y) = \frac{f_x(x)}{|\frac{\partial y}{\partial x}|} \tag{A.3}$$

Substituting Equation A.3 into Equation A.2, gives

$$H(y(t)) = E\left[\log \frac{\partial y}{\partial x}\right] + H(x(t)) \tag{A.4}$$

The infomax weight update equation changes $W$ by maximizing $H(y(t))$. The second term in Equation A.4 is unaffected by any change in $W$, so we can ignore it in the weight update equation. In gradient ascent, we change $W$ over time proportionally to the entropy gradient:

$$\Delta W \propto \frac{\partial H(y)}{\partial W} = E\left[\frac{\partial \log (\frac{\partial y}{\partial x})}{\partial W}\right] \tag{A.5}$$

In stochastic gradient ascent, we remove the expected value from Equation A.5, simplifying the weight update equation to

$$\Delta W \propto \frac{\partial}{\partial W} \log \frac{\partial y}{\partial x} = \left(\frac{\partial y}{\partial x}\right)^{-1} \frac{\partial}{\partial W} \frac{\partial y}{\partial x} \tag{A.6}$$

which simplifies to

$$\Delta W \propto [W^T]^{-1} + \hat{y}(t)x(t)^T \tag{A.7}$$
where

\[ \hat{y}(t) = \frac{\frac{\partial}{\partial y} \frac{\partial y}{\partial u}}{ \frac{\partial y}{\partial u} } \]  

(A.8)

For the logistic sigmoid function, \( y(t) = \frac{1}{1 + e^{-u(t)}} \), \( \hat{y}(t) = 1 - 2y(t) \), and for the hyperbolic tangent sigmoid function, \( y(t) = \tanh(u(t)) \), \( \hat{y}(t) = (-2y(t)) \).

Referring back to Equation A.1, the gradient ascent algorithm, however, is not guaranteed to reach the absolute minimum of \( I(Y_1, Y_2) \), because of the interference from the other entropy terms. When the input pdf's are not perfectly aligned with the slope of the sigmoidal function, the algorithm may have trouble finding statistically independent sources.
Appendix B  Tabulated SNR Results

This appendix contains the tabulated SNR results from the experiments discussed in Chapter 4. As mentioned in section 3.3.2, 15dB of separation sounds as if the sources were perfectly separated; at about 6dB of separation, one can hear the other source, but it is significantly quieter; at 3dB of separation, one can hear both sources, the desired source being perceivably louder than the undesired source. For a relatively "dead" room with a reverberation time of 0.1 seconds, the BSSD algorithm was able to effectively separate the male sample from the female sample.

Each experiment produced two sets of tables: the first shows, for each entry, how much the male sample is separated from the female sample, and vice versa; for example, the upper left entry in Table B.1 indicates that, for a reverb time of 0.1 seconds and a filter length of 512 taps, the algorithm was able to extract the male source with an SNR of 7.4dB and the female source with an SNR of 2.5dB. The second table of the pair shows the improvement in separation as represented by SNR's. See section 4.4.1 for a description of the separation improvement metric.
B.1 Reverberation Experiment

<table>
<thead>
<tr>
<th>reverb/voice</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 sec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>7.4</td>
<td>7.1</td>
<td>7.8</td>
<td>7.4</td>
</tr>
<tr>
<td>female</td>
<td>2.5</td>
<td>6.6</td>
<td>6.6</td>
<td>6.7</td>
</tr>
<tr>
<td>0.25 sec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>2.2</td>
<td>1.8</td>
<td>2.0</td>
<td>1.7</td>
</tr>
<tr>
<td>female</td>
<td>1.5</td>
<td>2.3</td>
<td>3.2</td>
<td>3.7</td>
</tr>
<tr>
<td>0.5 sec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>6.1</td>
<td>6.3</td>
<td>5.6</td>
<td>5.6</td>
</tr>
<tr>
<td>female</td>
<td>-4.8</td>
<td>-3.5</td>
<td>-1.8</td>
<td>-1.2</td>
</tr>
<tr>
<td>0.9 sec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>4.7</td>
<td>5.1</td>
<td>4.3</td>
<td>4.0</td>
</tr>
<tr>
<td>female</td>
<td>-2.0</td>
<td>-1.2</td>
<td>-0.4</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Table B.1 SNR's (in dB) of each separated source for different reverb times and different filter lengths, given a 2x2 filter configuration.

<table>
<thead>
<tr>
<th>reverb</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 sec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.8</td>
<td>4.7</td>
<td>5.1</td>
<td>4.9</td>
<td></td>
</tr>
<tr>
<td>0.25 sec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>1.0</td>
<td>1.6</td>
<td>1.8</td>
<td></td>
</tr>
<tr>
<td>0.5 sec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.7</td>
<td>1.5</td>
<td>1.9</td>
<td>2.3</td>
<td></td>
</tr>
<tr>
<td>0.9 sec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.6</td>
<td>1.2</td>
<td>1.2</td>
<td>1.5</td>
<td></td>
</tr>
</tbody>
</table>

Table B.2 Separation improvement SNR's (in dB) for a 2x2 filter configuration for different reverb times and different filter lengths.
B.2 Overdetermined Configuration Experiments

Reverberation time: 0.1 seconds

<table>
<thead>
<tr>
<th>config/voice</th>
<th>unmixing filter lengths</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
</tr>
</thead>
<tbody>
<tr>
<td>2x2 male</td>
<td>3.7</td>
<td>3.6</td>
<td>3.5</td>
<td>3.6</td>
<td></td>
</tr>
<tr>
<td>female</td>
<td>1.1</td>
<td>1.8</td>
<td>1.9</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>4x2 male</td>
<td>1.3</td>
<td>1.1</td>
<td>1.1</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>female</td>
<td>3.2</td>
<td>3.8</td>
<td>3.9</td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td>6x2 male</td>
<td>0.3</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>female</td>
<td>3.5</td>
<td>4.1</td>
<td>4.2</td>
<td>4.2</td>
<td></td>
</tr>
<tr>
<td>8x2 male</td>
<td>0.5</td>
<td>0.6</td>
<td>0.6</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>female</td>
<td>3.5</td>
<td>4.0</td>
<td>4.1</td>
<td>4.1</td>
<td></td>
</tr>
</tbody>
</table>

Table B.3 SNR's (in dB) of each separated source for different filter configurations and different filter lengths, given a reverb time of 0.1 seconds

<table>
<thead>
<tr>
<th>config</th>
<th>unmixing filter lengths</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
</tr>
</thead>
<tbody>
<tr>
<td>2x2</td>
<td>0.8</td>
<td>1.1</td>
<td>1.1</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>4x2</td>
<td>0.6</td>
<td>0.8</td>
<td>0.9</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td>6x2</td>
<td>0.3</td>
<td>0.5</td>
<td>0.6</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>8x2</td>
<td>0.4</td>
<td>0.7</td>
<td>0.7</td>
<td>0.7</td>
<td></td>
</tr>
</tbody>
</table>

Table B.4 Separation improvement SNR's (in dB) from the results in Table B.3.
Appendix B: Tabulated SNR Results

Reverberation time: 0.25 seconds

<table>
<thead>
<tr>
<th>config/voice</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2 male</td>
<td>0.0</td>
</tr>
<tr>
<td>female</td>
<td>0.4</td>
</tr>
<tr>
<td>4x2 male</td>
<td>1.4</td>
</tr>
<tr>
<td>female</td>
<td>2.5</td>
</tr>
<tr>
<td>6x2 male</td>
<td>0.8</td>
</tr>
<tr>
<td>female</td>
<td>2.0</td>
</tr>
<tr>
<td>8x2 male</td>
<td>0.7</td>
</tr>
<tr>
<td>female</td>
<td>1.9</td>
</tr>
</tbody>
</table>

**Table B.5** SNR's (in dB) of each separated source for different filter configurations and different filter lengths, given a reverb time of 0.25 seconds

<table>
<thead>
<tr>
<th>config</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2</td>
<td>-0.1</td>
</tr>
<tr>
<td>4x2</td>
<td>1.8</td>
</tr>
<tr>
<td>6x2</td>
<td>1.9</td>
</tr>
<tr>
<td>8x2</td>
<td>1.2</td>
</tr>
</tbody>
</table>

**Table B.6** Separation improvement SNR's (in dB) from the results in Table B.5.
Overdetermined Configuration Experiments

Reverberation time: 0.5 seconds

<table>
<thead>
<tr>
<th>config/voice</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2</td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>5.5</td>
</tr>
<tr>
<td>female</td>
<td>-4.2</td>
</tr>
<tr>
<td>4x2</td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>4.3</td>
</tr>
<tr>
<td>female</td>
<td>-3.4</td>
</tr>
<tr>
<td>6x2</td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>4.5</td>
</tr>
<tr>
<td>female</td>
<td>-3.4</td>
</tr>
<tr>
<td>8x2</td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>4.8</td>
</tr>
<tr>
<td>female</td>
<td>-3.7</td>
</tr>
</tbody>
</table>

Table B.7 SNR's (in dB) of each separated source for different filter configurations and different filter lengths, given a reverb time of 0.5 seconds

<table>
<thead>
<tr>
<th>config</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2</td>
<td>0.7</td>
</tr>
<tr>
<td>4x2</td>
<td>0.5</td>
</tr>
<tr>
<td>6x2</td>
<td>0.5</td>
</tr>
<tr>
<td>8x2</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table B.8 Separation improvement SNR's (in dB) from the results in Table B.7.
Reverberation time: 0.9 seconds

<table>
<thead>
<tr>
<th>config/voice</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2 male</td>
<td>3.0</td>
</tr>
<tr>
<td>female</td>
<td>-2.2</td>
</tr>
<tr>
<td>4x2 male</td>
<td>1.4</td>
</tr>
<tr>
<td>female</td>
<td>-1.0</td>
</tr>
<tr>
<td>6x2 male</td>
<td>1.8</td>
</tr>
<tr>
<td>female</td>
<td>-1.4</td>
</tr>
<tr>
<td>8x2 male</td>
<td>1.4</td>
</tr>
<tr>
<td>female</td>
<td>-1.7</td>
</tr>
</tbody>
</table>

Table B.9 SNR's (in dB) of each separated source for different filter configurations and different filter lengths, given a reverb time of 0.9 seconds

<table>
<thead>
<tr>
<th>config</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2</td>
<td>0.2</td>
</tr>
<tr>
<td>4x2</td>
<td>-0.1</td>
</tr>
<tr>
<td>6x2</td>
<td>0.1</td>
</tr>
<tr>
<td>8x2</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table B.10 Separation improvement SNR's (in dB) from the results in Table B.9.
Square Configuration Experiments

B.3 Square Configuration Experiments

Reverberation time: 0.1 seconds

<table>
<thead>
<tr>
<th>config/voice</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
</tr>
</thead>
<tbody>
<tr>
<td>2x2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>4.0</td>
<td>3.9</td>
<td>3.9</td>
<td>4.0</td>
</tr>
<tr>
<td>female</td>
<td>1.8</td>
<td>3.1</td>
<td>3.1</td>
<td>3.1</td>
</tr>
<tr>
<td>4x4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>4.9</td>
<td>4.7</td>
<td>4.7</td>
<td>4.7</td>
</tr>
<tr>
<td>female</td>
<td>3.0</td>
<td>4.3</td>
<td>4.4</td>
<td>4.5</td>
</tr>
<tr>
<td>6x6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>3.8</td>
<td>3.4</td>
<td>3.5</td>
<td>3.5</td>
</tr>
<tr>
<td>female</td>
<td>3.4</td>
<td>4.6</td>
<td>4.9</td>
<td>4.9</td>
</tr>
<tr>
<td>8x8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>3.8</td>
<td>3.4</td>
<td>3.5</td>
<td>3.5</td>
</tr>
<tr>
<td>female</td>
<td>3.3</td>
<td>4.4</td>
<td>4.7</td>
<td>4.9</td>
</tr>
</tbody>
</table>

Table B.1 SNR's (in dB) of each separated source for different filter configurations and different filter lengths, given a reverb time of 0.1 seconds

<table>
<thead>
<tr>
<th>config</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
</tr>
</thead>
<tbody>
<tr>
<td>2x2</td>
<td>0.8</td>
<td>1.4</td>
<td>1.4</td>
<td>1.4</td>
</tr>
<tr>
<td>4x4</td>
<td>1.5</td>
<td>2.0</td>
<td>2.1</td>
<td>2.1</td>
</tr>
<tr>
<td>6x6</td>
<td>1.3</td>
<td>1.7</td>
<td>1.9</td>
<td>1.9</td>
</tr>
<tr>
<td>8x8</td>
<td>1.3</td>
<td>1.6</td>
<td>1.8</td>
<td>1.9</td>
</tr>
</tbody>
</table>

Table B.12 Separation improvement SNR's (in dB) from the results in Table B.11.
Appendix B: Tabulated SNR Results

Reverberation time: 0.25 seconds

<table>
<thead>
<tr>
<th>config/voice</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2 male</td>
<td>2.8</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
</tr>
<tr>
<td>4x4 male</td>
<td>1.4</td>
</tr>
<tr>
<td></td>
<td>1.9</td>
</tr>
<tr>
<td>6x6 male</td>
<td>1.3</td>
</tr>
<tr>
<td></td>
<td>2.2</td>
</tr>
<tr>
<td>8x8 male</td>
<td>1.1</td>
</tr>
<tr>
<td></td>
<td>2.0</td>
</tr>
</tbody>
</table>

Table B.13 SNR's (in dB) of each separated source for different filter configurations and different filter lengths, given a reverb time of 0.25 seconds

<table>
<thead>
<tr>
<th>config</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2</td>
<td>1.0</td>
</tr>
<tr>
<td>4x4</td>
<td>0.7</td>
</tr>
<tr>
<td>6x6</td>
<td>0.8</td>
</tr>
<tr>
<td>8x8</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Table B.14 Separation improvement SNR's (in dB) from the results in Table B.13.
Square Configuration Experiments

Reverberation time: 0.5 seconds

<table>
<thead>
<tr>
<th>config/voice</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2 male</td>
<td>5.7</td>
</tr>
<tr>
<td>female</td>
<td>-4.9</td>
</tr>
<tr>
<td>4x4 male</td>
<td>7.1</td>
</tr>
<tr>
<td>female</td>
<td>-3.5</td>
</tr>
<tr>
<td>6x6 male</td>
<td>6.6</td>
</tr>
<tr>
<td>female</td>
<td>-3.9</td>
</tr>
<tr>
<td>8x8 male</td>
<td>6.5</td>
</tr>
<tr>
<td>female</td>
<td>-2.3</td>
</tr>
</tbody>
</table>

Table B.15 SNR's (in dB) of each separated source for different filter configurations and different filter lengths, given a reverb time of 0.5 seconds

<table>
<thead>
<tr>
<th>config</th>
<th>unmixing filter lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>512</td>
</tr>
<tr>
<td>2x2</td>
<td>0.4</td>
</tr>
<tr>
<td>4x4</td>
<td>0.7</td>
</tr>
<tr>
<td>6x6</td>
<td>0.2</td>
</tr>
<tr>
<td>8x8</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Table B.16 Separation improvement SNR's (in dB) from the results in Table B.15.
Reverberation time: 0.9 seconds

<table>
<thead>
<tr>
<th></th>
<th>unmixing filter lengths</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>config/voice</strong></td>
<td><strong>512</strong></td>
<td><strong>1024</strong></td>
<td><strong>2048</strong></td>
<td><strong>4096</strong></td>
</tr>
<tr>
<td>2x2 male</td>
<td>4.6</td>
<td>4.7</td>
<td>4.1</td>
<td>3.8</td>
</tr>
<tr>
<td></td>
<td>-2.5</td>
<td>-1.6</td>
<td>-1.0</td>
<td>-0.4</td>
</tr>
<tr>
<td>2x2 female</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4x4 male</td>
<td>4.3</td>
<td>3.8</td>
<td>3.3</td>
<td>3.1</td>
</tr>
<tr>
<td></td>
<td>-1.7</td>
<td>-0.6</td>
<td>0.2</td>
<td>0.9</td>
</tr>
<tr>
<td>4x4 female</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6x6 male</td>
<td>3.8</td>
<td>3.5</td>
<td>2.9</td>
<td>2.8</td>
</tr>
<tr>
<td></td>
<td>-1.8</td>
<td>-0.8</td>
<td>0.1</td>
<td>1.1</td>
</tr>
<tr>
<td>6x6 female</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8x8 male</td>
<td>3.7</td>
<td>3.3</td>
<td>2.7</td>
<td>2.6</td>
</tr>
<tr>
<td></td>
<td>-1.8</td>
<td>0.6</td>
<td>0.5</td>
<td>1.6</td>
</tr>
<tr>
<td>8x8 female</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table B.17** SNR's (in dB) of each separated source for different filter configurations and different filter lengths, given a reverb time of 0.9 seconds

<table>
<thead>
<tr>
<th></th>
<th>unmixing filter lengths</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>config</strong></td>
<td><strong>512</strong></td>
<td><strong>1024</strong></td>
<td><strong>2048</strong></td>
<td><strong>4096</strong></td>
</tr>
<tr>
<td>2x2</td>
<td>0.3</td>
<td>0.8</td>
<td>0.8</td>
<td>0.9</td>
</tr>
<tr>
<td>4x4</td>
<td>0.5</td>
<td>0.8</td>
<td>1.0</td>
<td>1.2</td>
</tr>
<tr>
<td>6x6</td>
<td>0.2</td>
<td>0.6</td>
<td>0.7</td>
<td>1.2</td>
</tr>
<tr>
<td>8x8</td>
<td>0.2</td>
<td>1.2</td>
<td>0.8</td>
<td>1.3</td>
</tr>
</tbody>
</table>

**Table B.18** Separation improvement SNR's (in dB) from the results in Table B.17.
References


References


References


References


Some of the figures in this thesis contains images found on the internet through the Lycos search engine (http://www.lycos.com).