A Computational Framework for Sound Segregation in Music Signals

Luís Gustavo Pereira Marques Martins

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO
Departamento de Engenharia Electrotécnica e de Computadores

A Computational Framework for Sound Segregation in Music Signals

Luís Gustavo Pereira Marques Martins

Dissertação submetida para satisfação parcial dos
requisitos do grau de doutor
em
Engenharia Electrotécnica e de Computadores

Dissertação realizada sob a orientação do
Professor Doutor Aníbal João de Sousa Ferreira,
do Departamento de Engenharia Electrotécnica e de Computadores
da Faculdade de Engenharia da Universidade do Porto

Porto, Setembro de 2008

This thesis was partially supported by the Portuguese Science and Technology Foundation (FCT), INESC Porto, the Calouste Gulbenkian Foundation, the European Commission under the IST research network of excellence VISNET and VISNET II of the 6th Framework Programme and the COST IC0601 Action on Sonic Interaction Design (SID).

Typeset by the author with the LaTeX Documentation System.
Author email: lgustavo@fe.up.pt

DOCTORAL DISSERTATION COMMITTEE

Prof. Gaël Richard

Full Professor in Audio Signal Processing
Head of the Audio, Acoustics and Waves Research Group TELECOM Paris Tech

Prof. Isabel Maria Martins Trancoso

Full Professor at Instituto Superior Técnico da Universidade Técnica de Lisboa

Prof. Artur Pimenta Alves

Full Professor at Faculdade de Engenharia da Universidade do Porto

Prof. Diamantino Rui da Silva Freitas

Associate Professor at Faculdade de Engenharia da Universidade do Porto

Prof. Aníbal João de Sousa Ferreira

Assistant Professor at Faculdade de Engenharia da Universidade do Porto

To my grandparents Ilídio and Lucinda ("requiescat in pace") and Antonio and Ermelinda, whose lives will always be an inspiration for me.
To my beloved parents Artur e Conceição and my dear brother Edu for all their unconditional support and love.

Abstract

Music is built from sound, ultimately resulting from an elaborate interaction between the sound-generating properties of physical objects (i.e. music instruments) and the sound perception abilities of the human auditory system.

Humans, even without any kind of formal music training, are typically able to extract, almost unconsciously, a great amount of relevant information from a musical signal. Features such as the beat of a musical piece, the main melody of a complex musical arrangement, the sound sources and events occurring in a complex musical mixture, the song structure (e.g. verse, chorus, bridge) and the musical genre of a piece, are just some examples of the level of knowledge that a naive listener is commonly able to extract just from listening to a musical piece. In order to do so, the human auditory system uses a variety of cues for perceptual grouping such as similarity, proximity, harmonicity, common fate, among others.

This dissertation proposes a flexible and extensible Computational Auditory Scene Analysis framework for modeling perceptual grouping in music listening. The goal of the proposed framework is to partition a monaural acoustical mixture into a perceptually motivated topological description of the sound scene (similar to the way a naive listener would perceive it) instead of attempting to accurately separate the mixture into its original and physical sources. The presented framework takes the view that perception primarily depends on the use of low-level sensory information, and therefore requires no training or prior knowledge about the audio signals under analysis. It is however designed to be efficient and flexible enough to allow the use of prior knowledge and high-level information in the segregation procedure.

The proposed system is based on a sinusoidal modeling analysis front-end, from which spectral components are segregated into sound events using perceptually inspired grouping cues. A novel similarity cue based on harmonicity (termed "Harmonically-Wrapped Peak Similarity") is also introduced. The segregation process is based on spectral clustering methods, a technique originally proposed to model perceptual grouping tasks in the computer vision field. One of the main advantages of this approach is the ability to incorporate various perceptually-inspired grouping criteria into a single framework without requiring multiple processing stages.

Experimental validation of the perceptual grouping cues show that the novel harmonicity-based similarity cue presented in this dissertation compares favourably to other state-of-the-art harmonicity cues, and that its combination with other grouping

cues, such as frequency and amplitude similarity, improves the overall separation performance. In addition, experimental results for several Music Information Retrieval tasks, including predominant melodic source segregation, main melody pitch estimation, voicing detection and timbre identification in polyphonic music signals, are presented. The use of segregated signals in these tasks allows to achieve final results that compare or outperform typical and state-of-the-art audio content analysis systems, which traditionally represent statistically the entire polyphonic sound mixture. Although a specific implementation of the proposed framework is presented in this dissertation and made available as open source software, the proposed approach is flexible enough to be able to utilize different analysis front-ends and grouping criteria in a straightforward and efficient manner.

Keywords: Sound, Computational Auditory Scene Analysis, Audio Source Separation, Music Information Retrieval, Sinusoidal Modeling, Spectral Clustering, Normalized Cut, Predominant Melodic Source Separation, Main Melody Pitch Estimation, Timbre Recognition.

Resumo

A música é composta por som, e em última instância resulta de uma interacção complexa entre as propriedades sonoras de objectos físicos (i.e. instrumentos musicais) e as capacidades de percepção sonora do sistema auditivo humano.

Os seres humanos, mesmo indivíduos sem qualquer tipo de formação musical, tipicamente demonstram uma capacidade de extrair, de forma quase inconsciente, uma relevante quantidade de informação sobre um sinal sonoro musical. A percepção do ritmo de uma peça musical, a melodia principal de um arranjo musical complexo, as fontes e eventos sonoros que têm lugar numa mistura musical complexa, a estrutura de uma música (e.g. introdução, refrão) ou o seu género (e.g. rock, jazz, clássica) são apenas alguns exemplos do nível de conhecimento que um ouvinte, mesmo que sem formação musical, é normalmente capaz de inferir ao simplesmente escutar um excerto musical. Para este efeito, o sistema auditivo humano faz uso de uma variedade de pistas de natureza perceptiva que conduzem ao agrupamento perceptual dos estímulos acústicos em eventos ou entidades sonoras coerentes. Este agrupamento é em grande parte baseado em critérios de similaridade, proximidade harmónica e espacial, entre outras.

Esta dissertação propõe uma plataforma computacional flexível e extensível para a análise de sinais e cenas sonoras complexas, permitindo uma modelização de alguns dos processos de agrupamento perceptual do sistema auditivo humano envolvidos na percepção de sinais musicais. O objectivo desta plataforma é permitir representar uma mistura sonora monaural através de uma descrição perceptualmente relevante da cena sonora (ou seja, semelhante à forma como um ouvinte sem treino musical a percepcionaria), ao invés de tentar obter uma separação exacta das várias fontes sonoras que a constituem. Esta abordagem assume que a percepção se baseia principalmente no uso de informação sensorial de baixo nível, não carecendo por isso de treino ou conhecimento prévio acerca dos sinais sonoros em análise. Não obstante, a plataforma proposta é suficientemente flexível e eficiente, permitindo o uso de conhecimento prévio e de informação de alto nível no processo de segregação.

O sistema proposto baseia-se num modelo de análise sinusoidal, a partir do qual as componentes espectrais são segregadas em eventos sonoros através do uso de métricas baseadas em pistas de natureza perceptiva, entre as quais uma nova métrica de similaridade harmónica (chamada de "*Harmonically-Wrapped Peaks Similarity*"). O processo de segregação é baseado em métodos de agrupamento espectral ("*spectral clustering*"), uma

técnica originalmente proposta para modelar as tarefas de agrupamento na área da visão por computador. Uma das principais vantagens desta abordagem é a possibilidade de incorporar e combinar vários critérios de agrupamento inspirados em princípios perceptuais numa única plataforma, sem a necessidade de múltiplas fases de processamento.

A validação experimental destes critérios de agrupamento com relevância perceptual demonstram que a nova métrica de similaridade harmónica proposta neste trabalho se compara favoravelmente a outras técnicas estado da arte, e que a sua combinação com outros critérios de similaridade, tais como proximidade em frequência e amplitude, permite melhorar de forma global os resultados de segregação de misturas sonoras musicais. Adicionalmente, resultados experimentais obtidos para várias aplicações na área da Recuperação de Informação Musical ("*Music Information Retrieval*") – e.g. segregação da linha melódica predominante, estimação da melodia principal, detecção de segmentos vozeados e identificação e reconhecimento de instrumentos musicais em misturas musicais polifónicas – demonstram que o uso de sinais segregados nestas tarefas permitem obter resultados finais comparáveis ou melhores que os obtidos por outras abordagens estado da arte para a análise de conteúdos áudio onde os sinais polifónicos são tradicionalmente representados no seu todo de forma estatística. Apesar de esta dissertação apresentar uma implementação em software que é disponibilizada como software *open source*, a abordagem proposta é suficientemente flexível, permitindo incorporar de forma simples e eficiente diferentes metodologias de análise, novas métricas bem como outras técnicas de segregação de sinais sonoros complexos.

Palavras-chave: Som, Música, Análise Computacional de Cenas Auditivas, Separação de Fontes Sonoras, Recuperação de Informação Musical, Modelos Sinusoidais, Agrupamento Espectral, Cortes Normalizados, Separação da Linha Melódica Predominante, Estimação da Melodia Principal, Reconhecimento de Timbre.

Résumé

La musique est faite de sons et résulte d'une interaction complexe entre les propriétés sonores d'objets physiques (i.e. instruments musicaux) et les capacités perceptives du système auditif humain.

Les êtres humains, avec ou sans formation musicale, sont capables d'extraire, de manière presque inconsciente, une énorme quantité d'information pertinente à partir d'un signal sonore. Des caractéristiques comme le rythme d'une pièce de musique, la mélodie principale d'un arrangement musical complexe, la structure d'une chanson (e.g. couplet, refrain, etc.) ainsi que le genre musical sont juste quelques exemples du niveau d'information qu'un auditeur, même sans formation musicale, est normalement capable d'extraire en écoutant simplement une pièce de musique. A cette fin, le système auditif humain utilise une variété d'indices de regroupement perceptuel tels que, entre autres, des critères de similarité et de proximité harmonique et spatiale.

Cette dissertation propose une plate-forme de programmation informatique adaptative et extensible pour l'analyse computationnelle de signaux sonores complexes ("*Computational Auditory Scene Analysis*"), permettant une modélisation de processus de groupement perceptuels impliqués dans l'écoute musicale. Le but de cette plate-forme est de représenter un mélange sonore d'un unique canal sous la forme d'une description topologique de grande pertinence perceptive (c'est-à-dire similaire à celle d'un auditeur sans formation musicale), ceci sans essayer d'obtenir une séparation des signaux des sources physiques constituant ce mélange sonore. L'implémentation de cette plate-forme fait l'hypothèse que la perception dépend en première instance d'informations sensorielles de bas niveau, et ne requière donc aucune formation ni connaissance a priori des signaux musicaux écoutés. Cependant, cette plate-forme est adaptative et permet, si besoin est, l'inclusion de connaissance a priori et d'information de haut niveau.

Le système proposé se base sur un modèle d'analyse sinusoïdal, à partir duquel les composantes spectrales sont attribuées à divers événements sonores à partir d'indices de groupements perceptifs. Un nouvel indice de similarité harmonique est par ailleurs présenté (appelé "*Harmonically-Wrapped Peak Similarity*"). Ce processus d'attribution se base sur des méthodes de groupement spectral ("*spectral clustering*") dont l'origine se trouve dans la modélisation des tâches de groupement perceptif utilisée dans la recherche en vision artificielle. Un des avantages principaux de cette approche est la possibilité

d'inclure divers critères de groupements d'inspiration perceptuelle dans une même plate-forme, sans requérir une multiplication de phases de calcul successives.

La validation expérimentale des critères de groupement perceptif montre que le nouvel indice de similarité harmonique proposé dans cette dissertation est meilleur que les critères utilisés dans l'état de l'art actuel, et que sa combinaison avec d'autres indices, tels que la similarité fréquentielle et d'amplitude, améliore les résultats globaux du processus de représentation. De plus, les résultats expérimentaux pour diverses tâches de récupération d'information musicale ("*Music Information Retrieval*"), telles que la séparation de source mélodique principale, l'estimation de fréquence fondamentale de la mélodie principale, la détection de parties voisées et l'identification de timbres dans les signaux polyphoniques, sont présentés. Ces résultats sont comparables, voire meilleurs que ceux des systèmes d'analyses du contenu audio de l'état de l'art qui, eux, n'utilisent traditionnellement pas de représentation en événements sonores séparés. Cette dissertation présente une implémentation "software" de cette plate-forme qui est disponible sous une licence libre. L'approche suivie est adaptative, l'incorporation d'autres modules de software d'analyse est ainsi facilitée.

Mots-clés: Son, Analyse computationnelle des scènes auditives, Séparation de sources acoustiques, Récupération d'information musicale, Modélisation sinusoidale, Groupement spectral, Coupe normalisée, Séparation de source mélodique principale, Estimation de mélodie principale, Reconaissance de timbres.

Acknowledgments

"The secret to creativity is knowing how to hide your sources."

Albert Einstein

The work reported in this document is not the work of a single person. It is in fact a collaborative effort and, at the risk of unfair omission, I would like to express my gratitude to those who somehow, directly or indirectly, contributed to it.

First I would like to gratefully acknowledge all the encouragement, support and wise supervision of Prof. Aníbal Ferreira. He has been one of the key persons in my research career, being the one who first introduced me to the joys and challenges of audio and signal processing, back in 1997, when I was still finishing my undergrad studies. I am deeply thankful for his encouragement, inspiration and research opportunities during all these years, and for allowing me to share his passion for sound processing, which ultimately lead to this work. I must also thank him for his thorough proof-reading and revision of this dissertation, as well as for all his valuable comments and suggestions.

I am also deeply thankful to INESC Porto, where I have been extremely lucky to reside for the past ten years and interact with such brilliant and insightful researchers. In particular, I would like to express my deepest gratitude and admiration for Prof. Pimenta Alves, Prof. Pedro Guedes de Oliveira and Prof. Ruela. Together with Prof. Aníbal Ferreira, they invited me to join INESC Porto and the Telecommunications and Multimedia Unit (UTM), where I had the opportunity to develop and make public my research work and to participate in so many compelling research projects. I must thank them for their never ending encouragement and support since day one.

I truly enjoyed working with many people at INESC Porto and I would like to thank them here for all the interesting discussions we had about so many different topics and in particular about the work in this dissertation: Sílvio Macedo, Teresa Andrade, Rui Campos, Gustavo Carneiro, Jaime Dias, Ricardo Morla, Luís Teixeira, Filipe Sousa, Tânia Calçada, Paula Viana, Vítor Soares, André Rocha, Miguel Falcão, Gabriel Falcão, Prof. Corte Real and Prof. Manuel Ricardo. Thanks also to Renata Rodrigues for her availability and help on all those administrative tasks.

A special mention goes to Luís Filipe Teixeira, Pedro Carvalho, Jaime Cardoso and Fabien Gouyon my fellow office mates at UTM. Thanks to Pedro Carvalho, who together with Sílvio Macedo, started the initiative to promote the UTM's LabMeetings, where so many times I had the opportunity to present my work and get valuable feedback from the INESC Porto research community, as well as to attend so many interesting talks from colleagues and invited researchers. I must also thank him for his willingness to share his quiet and private office when presented with the proposal to create a new small group at INESC Porto focused in multimedia analysis and processing.

Jaime Cardoso was (and still is!) my ultimate and reliable resource for those tricky mathematical questions, always willing to spend a couple of hours to help me understand how such mathematical problems end up having elegant solutions after all. His work and his forward thinking will always be an inspiration to me.

Fabien Gouyon, who I had the luck to meet several times at the MTG and who later made me so honoured by joining INESC Porto (bringing his huge knowledge and experience to the group) spent a substantial part of his time proof-reading this document. I am deeply thankful to him for his incisive comments and points of view and for making me think out of the box. I must also thank him for his efforts into making my internship at McGill a reality as well as for his big help with the translation of parts of this dissertation to French. *Merci beaucoup* Fabien!

Luis Teixeira, my "Portuguese Invasion" buddy in the *Marsyas* project during our first internship in beautiful Victoria, made those long and well spent hours in the lab refactoring *Marsyas* from the inside out a pleasure (although we were always asking ourselves if we would ever be able to make it work again!). I will always remember those discussions during dinner about what has turned out to become *MarsyasX* (and yes, I still have that napkin!). Since the very first time I met Luís (several years ago, when we managed to implement some sort of a CORBA service for audio classification in the ORBIT Project) I have always been learning something new and more advanced with him. I must thank him for all I have learned with him and hope the future will allow us to keep collaborating

(if nothing else, in the scope of *MarsyasX*).

More recently, it has been a pleasure to have Carlos Guedes as a colleague at INESC Porto. His strong music background combined with his technical abilities are really unique and his works are always an inspiration. I must thank him for pointing me such valuable bibliographic references on music perception and cognition and for those discussions about such interesting topics of research.

I must also not forget all the people from the former MOG team at INESC Porto, where I had the opportunity to learn so much from such a motivated and knowledgeable team working in so many important projects. Many thanks to Pedro Cardoso for inviting me to make part of his team and for the confidence he always has put in my abilities. Thanks to Pedro Ferreira, Vitor Teixeira and Ernesto Santos, and all the MOG team for welcoming me to their group and for sharing their knowledge with me.

I would also like to express my gratitude to the School of Arts of Universidade Católica Portuguesa (UCP) for inviting me as a teacher during the past few years. This has been a truly fulfilling opportunity to meet and learn from so many brilliant people and to make part of such a great school. In particular I would like to express my greatest gratitude to Álvaro Barbosa, coordinator of the School of Arts, for his accessibility and support since the day he invited me to join the UCP. Many thanks for his support when confronted with my intentions to pursue some academic internships abroad in the scope of my research activities, and for his availability to deal with the many troubles that resulted from those decisions of mine. Thanks to Luís Teixeira for his valuable support, discussions and motivation. Thanks to all my colleagues and personnel at UCP and to all my students during the past years for keeping me so motivated with everything related to sound.

During the scope of this work I have been extremely lucky to do some research internships in some of the top research groups in my field of research.

Many thanks to Prof. Xavier Serra and to the MTG people (Fabien Gouyon, Pedro Cano, Emilia Goméz, Óscar Celma, Jordi Bonada, Xavier Amatriain, Pau Arumi, just to name a few) for making me feel so welcome during my several visits to Barcelona and for all our discussions about my work.

Many thanks to Prof. Thomas Sikora for welcoming me to his group at the Technical University of Berlin (TUB) where I had the chance to work in close collaboration with Juan José Burred (as well as Gunnar Eisenberg, in the scope of the VISNET project). I am particularly honoured to be a co-author with Juan José Burred and grateful for all his availability to collaborate with me and for sharing his knowledge and skills, even at a time when he was working hard finishing his PhD thesis. I really hope we will be able to

keep up our collaboration in the future.

Many thanks to Prof. Stephen McAdams and Prof. Philippe Depalle for welcoming me at the CIRMMT, McGill University, where I had the chance to work once again with Mathieu Lagrange and to present most of the work proposed in this dissertation to the CIRMMT team members. I was truly honoured by their comments and suggestions and I would like to thank the CIRMMT people for their support and discussions during my stay.

I would also like to thank Prof. Mark Sandler for all his availability to receive me at his group at the Queen Mary University at the beginning of my PhD activities (although sadly it ended up not being feasible to join such an excellent research group).

A very special reference should go to my three internships at the University of Victoria (UVic), where I was invited to work under the wise supervision of Prof. George Tzanetakis. I owe him my deepest gratitude for his never ending availability and support since we first exchanged our first emails in the scope of the *Marsyas* software framework, years before I even decided to enroll into a PhD program. In fact, his invitation to become involved on the development of *Marsyas* was a key aspect in my research career, having a direct impact on the work presented in this thesis.

While at UVic, I was also extremely fortunate to meet and to closely collaborate with Mathieu Lagrange, a post-doc researcher at Prof. Tzanetakis' lab at the time. In fact, a great deal of the work in this dissertation is the result of this close collaboration with Mathieu Lagrange and Prof. George Tzanetakis, as evident from several scientific publications where I have the honour to be their co-author. I will be forever grateful to Mathieu Lagrange's "best ideas ever", and for the enlightening discussions with him and Prof. Tzanetakis (even when collaborating thousands of miles apart!). I will always be grateful for the amount of time they both spent proof-reading this dissertation and for all the valuable comments and suggestions.

I would also thank to all the UVic people for their warm welcome during my three visits to Victoria, namely Graham Perceival, Jennifer Murdoch, Ajay Kapur, Adam Tindale, Manj Benning, Steven Ness, Prof. Andrew Schloss, Kirk McNally, and Isabel.

Thanks also to the *Marsyas* developers and users communities who helped improve the framework and keep my motivation towards contributing to it always high, with a positive impact on the work presented in this thesis.

Finally, I would like to thank Anssi Klapuri for kindly providing the multi pitch estimator code for some of the evaluations presented in this dissertation (as well as in one

of the published articles) and thank all the anonymous reviewers for their thorough comments and suggestions for improvement on all the papers submitted in the scope of this work.

On a more personal note, I would like to leave a warm word of thanks to my family and friends. To my parents and grand parents, for all their love and immense support. Thank you for transmitting to me the love of learning. To my brother Edu, for all his love, admiration and support, and for being my much loved best friend.

For being true friends, with whom I share so many happy memories, and for making Porto a place so great to live, a word of gratitude goes to Pedro, Johny, Meiras, Banderas, Vitor, Pedro Yoga, Lu, Elsa, Duarte and Susaninha. A very special word goes to Cris and Haneleh, for always being there for me, and for their children, my beloved goddaughter Kaki and her brother Smiguel, the most incredible kids in the world!

Du, you deserve a very special loving word, for being such an incredible and kind human being, who stood by my side and coped with my moody behavior in the last months of writing my thesis and suffered with those long absences during my stays abroad. This journey would have been much harder without you in my life.

To all my childhood friends back in Sever do Vouga (my beautiful hometown), thank you for still remembering and supporting me even after being away for so long during this endeavour. Bastos, Lala, Sara ("requiescat in pace"), Bamb, Ariana, Beckas, Sandrita, Joca, Dudu, Silvana, Fibras, Sónia, Kió, Luís dos Seguros, Bino, Lalo, Carc, Nelinho, Larachas, Guidinha, Sotto and JP, you are always in my heart. To my cousin Jorge, a special thanks for his friendship and for being my host during many of my stays in Barcelona.

Porto, Luís Gustavo Martins
September 2008

Reader Notes

This document is composed of six chapters. All chapters begin with a small introduction to its contents. In the end, bibliographic references are listed. These references are identified in the text using the authors names and year of publication within square brackets.

Not surprisingly, many acronyms are used throughout the text. To avoid cluttering the text with definitions, a list of acronyms at the beginning of the document is provided, allowing easy referencing.

A final note regarding the fonts used to typeset this document. The font used in body text is Computer Modern Roman and the blocks containing program code and class or module names use the `Adobe Courier` font. Chapter and section titles are in various sizes of **Adobe Helvetica-Narrow Bold**.

Table of Contents

List of Figures

List of Tables

Acronyms

AM Amplitude Modulation

API Application Programming Interface

ASA Auditory Scene Analysis

ASR Automatic Speech Recognition

CASA Computational Auditory Scene Analysis

CMSA Computational Music Scene Analysis

CPR Cluster Peak Ratio

DFT Discrete Fourier Transform

DWT Discrete Wavelet Transform

EM Expectation-Maximization

F0 Fundamental Frequency

FM Frequency Modulation

FFT Fast Fourier Transform

FOSS Free and Open Source Software

FP False Positive

GMM Gaussian Mixture Models

GNU "GNU is Not Unix"

GPL GNU Public License

GUI Graphical User Interface

HCI Human Computer Interaction

HESR Harmonically Enhanced Spectral Representation

HWPS Harmonically Wrapped Peak Similarity

ICA Independent Component Analysis

JND Just Noticeable Difference

KNN K-Nearest Neighbor

MARSYAS Music Analysis, Retrieval and SYnthesis for Audio Signals

MCA Music Content Analysis

MFCC Mel-Frequency Cepstral Coefficients

MIDI Musical Instrument Digital Interface

MIR Music Information Retrieval

MIREX Music Information Retrieval Evaluation eXchange

Ncut Normalized cut

NMF Non-Negative Matrix Factorization

PCA Principal Component Analysis

PRC Precision

RBF Radial Basis Functions

RCL Recall

SDR Signal-to-Distortion Ratio

SDR$_{ogf}$ Signal-to-Distortion Ratio with optimized gain factor

SMC Sound and Music Computing

SMO Sequential Minimal Optimization

SMS Spectral Modeling Synthesis

SNR Signal-to-Noise Ratio

SSDR Segmental Signal-to-Distortion Ratio

STFT Short-Time Fourier Transform

STS Short-Term Sinusoidal

SVD Singular Value Decomposition

SVM Support Vector Machine

TP True Positive

Chapter 1

Introduction

"The beautiful thing about learning is that no one can take it away from you."

B.B. King

1.1 Context and Motivation

This dissertation is framed in the general field of *Sound and Music Computing* (SMC), which focuses the analysis and understanding of sound and music, synthesis and processing of sound, music generation modeling, music interfaces, performance modeling and control, and sound interaction design [Serra et al., 2007].

Sound is the effect of the resonance of objects and materials in a medium (e.g. the air) while music can be coarsely defined as an intended organisation of sounds for particular uses in social and cultural contexts. Indeed, music is an important aspect of all human cultures, being the language of human emotions, of social bonding, of personal development and intellectual enrichment.

Sound and music are therefore intrinsically connected by a communication chain covering all aspects of the relationship between sonic energy and meaningful information, both from sound to sense (as in musical content extraction or perception), and from sense to sound (as in music composition or sound synthesis).

Finding its roots in the biology of humans and being intimately attached to their own daily existence, music took a central position in the current technological society. Over the past 50 years, music and technology have established such a strong connection that all stages of the economic chain, from production to distribution and consumption, are

now digital. As a result, music is quickly starting to become a commodity as ubiquitous as water or electricity, at least in all first world societies [Serra et al., 2007].

However, the real sound and sense for music is still identified with the extremes of this chain, where musicians play and where listeners search for and enjoy music as active consumers. All the rest is virtual, digitally encoded and difficult to access. It is therefore necessary to somehow bridge this semantic gap (i.e. the separation that currently exists between sound and sense), stimulating a constructive interaction between culture, science and industry.

Indeed, and accordingly to a recent roadmap proposed for the field of SMC [Serra et al., 2007], the next revolution is about the connection between sound and sense, that is, the connection between physical energy encoded in technology and the human subjective experience. Hence, new technology is required to close this gap and change the way people normally interact with music at the different levels of the digital chain. From early research in audio processing and synthesis during the second half of the past century, the challenge is now in moving to a position where areas such as Music Information Retrieval (MIR) [Downie, 2003], Interactive Multimedia Systems and sound- and music-aware environments are more relevant than ever. Innovative products are starting to foster social interaction among people and new opportunities will be further developed by offering new tools for expression and communication. This revolution will be leveraged by the dynamic forces that drive music itself, which in turn will lead to the creation of new cultural and business opportunities, from industrial to creative industries, from developers to artists, from creators to consumers.

1.2 Thesis Statement

The main problem this work tries to address is the identification and segregation of sound events in monaural (i.e. single-channel) "real-world" polyphonic music signals[1]. The approach presented has been inspired by the current knowledge of how listeners perceive sound events in music signals, be it music notes, chords, harmonic textures, melodic contours, instruments (i.e. sound sources) or any other type of event.

Indeed, when taking at hands the challenge of constructing a machine listening system, approaching the problem through engineering alone may not be the most suitable strategy. The perception of music is by nature intrinsically related to the perception of sound and

[1] In this text, the term *polyphonic* music, sound or mixture refers to sounds or musical pieces where several acoustic sources are simultaneously present.

therefore the principles upon which the human auditory system are based should first be understood [Scheirer, 2000, pp.14].

However, the objective is not, at least at this stage, to provide a complete music-listening framework that models the high level processes involved on how music expert listeners turn musical data into models of musical structure, or how cognitive music structures give rise to affective response or even how musicians turn intentions into music. Such problems mainly fall into the realm of Music Psychology [McAdams and Bigand, 1993, Deutsch, 1999], and are out of the scope of this work.

Instead, the work presented and discussed in this dissertation attempts to propose a sound segregation framework and some of the building blocks involved in sound and music perception, while already paving the way for the future inclusion of higher level processes involved in the cognition of acoustic and musical events in audio signals.

1.2.1 Current State

Typical sound analysis and MIR systems represent statistically the entire polyphonic or complex sound mixture (e.g. [Pachet and Cazaly, 2000, Tzanetakis and Cook, 2002]), without any attempt to first identify the different sound entities or events that may coexist in the signal. There is however some evidence that this approach has reached a "glass ceiling" [Aucouturier and Pachet, 2004] in terms of analysis and retrieval performance.

One obvious direction for further progress is to attempt to individually characterize the different sound events comprising the polyphonic mixture, and use this structured representation to improve the extraction of perceptually relevant information from complex audio and musical mixtures.

However, such sound segregation systems (including the one proposed in this thesis) face some demanding challenges, making their performance still quite limited compared to the human auditory system. Nevertheless, some of the current results already provide alternative and improved approaches to common sound analysis and MIR applications.

1.2.2 The Main Challenges

Human listeners are able to perceive individual sound events in complex mixtures, even if listening to a monaural music recording, which may even include unknown timbres or musical instruments.

Some techniques for sound separation are based on recordings done with multiple microphones, enabling the use of the spatial location of the source in the separation [Torkkola, 2000], often making the separation task easier. However, often only a

single-channel recording is available, and even the common stereo (i.e. two-channel) format usually used for music distribution may not be sufficient for spatial location based separation (except in some trivial cases).

The use of prior information about the sources or musical instruments in a recording can also help to optimize the segregation algorithm by using training signals where each instrument is present in isolation (e.g. [Burred and Sikora, 2007]). However, such systems may lack the ability to analyse generic sound signals, where the assumptions about the pre-trained sound sources do not hold.

As a result, this work takes the challenge to attempt sound event segregation in monaural "real-world" music signals (although stereo information can be easily incorporated and made useful to the segregation task), using an approach that is not necessarily based on the use of source-specific prior knowledge (i.e. it is not trained for a specific sound source or music instrument). Instead, the general properties of the sound and music signals, together with some of the known principles of human auditory system, are explored, requiring a multidisciplinary approach to the problem.

1.3 Related Research Areas

Being an intrinsically multidisciplinary field, Sound and Music Computing (SMC) is a contemporary designation encompassing disciplines historically known as Audio Processing, Music Technology, Computer Music, Computer Audition, among others, all committed to developing new and more advanced musical tools. The SMC field is generally considered to include all types of sounds and human communication processes except speech. Speech research has its own aims and methodologies (although most of them are of valuable importance for sound processing in general [Rabiner and Juang, 1993]) and is therefore outside the scope of the SMC field [Serra et al., 2007].

Specifically, the work presented in this thesis falls under the very active field of research known as *Computational Auditory Scene Analysis* (CASA) [Wang and Brown, 2006], which is in turn part of the *Sound and Music Description and Understanding* SMC subtopics [Serra et al., 2007, Ch.4]. CASA focuses on the development of computational approaches for the segregation of different sound producing objects in an acoustical scene, based on the principles underlying the human perception of complex acoustic mixtures, a discipline known as *Auditory Scene Analysis* (ASA) [Bregman, 1990].

Several areas of study relate to CASA. Audio and especially music signals are complex data-intensive, time-varying signals with unique characteristics that elicit new challenges

to these relevant research areas. Therefore, research in CASA provides a good way to test, evaluate and create new algorithms and tools in all those research areas. The following paragraphs provide short descriptions of these research areas and their connection to CASA.

Digital Signal Processing [Oppenheim and Schafer, 1975] is related to the computer manipulation of analog signals (commonly sounds or images) which have been converted to digital form (i.e. sampled). The various techniques of Time-Frequency analysis developed for processing audio signals in many cases originally developed for Speech Processing and Recognition [Rabiner and Juang, 1993], are of special importance for this work.

Machine Learning is a broad subfield of artificial intelligence and is focused on the design and development of algorithms and techniques that allow computers to improve their performance in inferring relevant information from data, based on previous results and training by humans [Duda et al., 2000, Witten and Frank, 2005]. In the particular case of the work presented in this thesis, machine learning techniques are used to implement a computer system which is able to identify sound "objects" in a complex acoustic mixture, taking into consideration grouping cues inspired in Auditory Scene Analysis (ASA) principles [Bregman, 1990] (e.g. the Spectral Clustering technique, as will be described in Chapter 3).

Computer Vision [Marr, 1982] is a term used to describe any algorithm and system that extracts information from visual signals such as image and video. The visual and acoustical perception in humans, although based on different mechanisms, often follow similar principles, hence some of the techniques and ideas developed in the realm of one discipline may prove useful and successful at the other (e.g. the Normalized Cut criterion, proposed for the segmentation of images [Shi and Malik, 2000], and proposed in this work for sound segregation – see Chapter 3).

Information Retrieval [van Rijsbergen, 1979] refers to techniques for searching and retrieving text documents based on user queries (e.g. web searches provided by Google[2]). More recently, techniques for multimedia information retrieval such as content-based retrieval systems for sound and music [Gouyon et al., 2008] have started appearing and a related field of research of specific interest to this thesis is *Music Information Retrieval* (MIR) [Downie, 2003, Orio, 2006]. MIR has been rapidly evolving over the past few years and it encompasses a wide variety of ideas, algorithms, tools, and systems that have been proposed to handle the increasingly large and varied amounts of musical data available digitally.

[2] `http://www.google.com`

Finally, *Perception* research. The human perceptual system has unique characteristics and abilities that should be taken into account to develop more effective computer systems for analysing and processing sensory data. The work on *Psychoacoustics* [Stevens, 1957, Moore, 1989, Zwicker and Fastl, 1990] and *Auditory Scene Analysis* (ASA) [Bregman, 1990, McAdams and Bigand, 1993] describe the relationship between acoustic sound signals, the physiology of the human ear and the cognitive processes involved in the perception of sound and music.

1.4 Applications

Sound mixtures present challenging problems for most audio applications where the analysis or the application of some processing only to a certain source within the mixture is virtually impossible. This presents an opportunity for the development of sound source separation solutions which first separate the mixture into its constituent sources, subsequently allowing their individual analysis and processing.

Additionally to the scientific challenge of creating a listening machine, where a model of the auditory apparatus could be experimentally tested and evaluated, there are several other fields of application for a CASA system. Some examples are enumerated in the following list.

- **Sound and music description.** The analysis of sound mixtures is difficult and one approach towards simplifying the problem is to use source separation as a pre-processing step. For instance, estimating the fundamental frequencies of concurrent sounds in a polyphonic mixture may be easier to tackle if using the separated monophonic signals coming from each instrument playing (e.g. [Lagrange et al., 2008a, Klapuri, 2008]).

- **Sound manipulation.** Separation enables the efficient manipulation of mixed sound signals. The removal or volume adjustment of the playing sources, the changing of the source location in the stereo field or the upmixing of audio recordings (e.g. [Lagrange et al., 2007b]) are just some examples.

- **Robust automatic speech and speaker recognition.** Although much progress has been made in automatic speech and speaker recognition in recent years, the performance of these systems still degrades quickly when in the presence of acoustic interference or concurrent signals, the normal situation in real-world scenarios. CASA presents some new approaches to this problem, in which

speech is regarded as just one of the many sources in a complex sound mixture. After segregation from interfering components, the separated speech signal can then be further processed in an attempt to improved speech and speaker recognition in environmental mixtures.

- **Object-based audio coding.** The current and more common audio coding methods encode a polyphonic signal as a single stream (e.g. MPEG1–Layer 3, a.k.a. MP3). However, higher coding flexibility and efficiency could be achieved if each separated element or sound "object" in the mixture could be independently encoded with specific source codecs (e.g. [Tolonen, 2000, Ferreira and Sinha, 2005, Vincent and Plumbley, 2005]. The MPEG-4 standard [Pereira and Ebrahimi, 2002] implements this approach as the "Structured Audio" component [Scheirer, 1998], and the standard includes a format for coding object-based audio scenes.

- **Automatic music transcription.** Automatic music transcription attempts to derive a musical score from an acoustical recording of a music piece (e.g. [Klapuri, 2006, Bello, 2003, Martins and Ferreira, 2002, Marolt, 2004, Klapuri and Davy, 2006, Paiva et al., 2006]). Given the specific challenges elicited by polyphonic music signals (as discussed in section 2.3.2), CASA can contribute with new and robust approaches to this problem by firstly segregating the playing instruments in a mixture before proceeding with their individual transcriptions.

- **Audio and music information retrieval.** One of the current key research challenges in the SMC field is the efficient automatic annotation and subsequent search of audio recording archives (private or on the Internet). Given the huge amounts of audio content on those archives, and the fact that most recordings contain mixtures of sound sources, the separation of the such mixtures is a prerequisite for automatic content based analysis.

- **Auditory scene reconstruction.** Audio restoration is another field of application for CASA, where unwanted sources in a sound mixture can be separated and subsequently removed or enhanced (e.g. [Esquef et al., 2002]). This area has recently started finding application in the cell-phone industry, where CASA principles are being used to enhance speech intelligibility in noisy environments.

- **Hearing prostheses.** Hearing aids that only provide amplification are of little help

on understanding speech in noisy environments since they equally amplify both signals. CASA could provide a solution to speech segregation from noisy signals, improving speech intelligibility.

1.5 Main Contributions

The main contributions of this thesis towards a sound segregation approach for music signals are summarized below.

1. Proposal and experimental validation of a flexible and efficient framework for sound segregation of "real-world" polyphonic music, inspired by ideas from *Computational Auditory Scene Analysis* (CASA).

2. Definition of a novel harmonicity cue, termed *Harmonically Wrapped Peak Similarity* (HWPS), experimentally shown as a good grouping criteria for sound segregation in polyphonic music signals.

3. Development of the proposed sound segregation approach as open source software, based on the MARSYAS framework[3].

Publications Related to the Thesis

The research work presented in this thesis has resulted in the collaborative publications listed below (sorted by relevance). The articles are available in PDF format at `http://www.fe.up.pt/~lgustavo`.

1. Lagrange, M., Martins, L. G., Murdoch, J., and Tzanetakis, G. (2008). Normalized cuts for predominant melodic source separation. IEEE Transactions on Audio, Speech, and Language Processing, 16(2). Special Issue on MIR. [Lagrange et al., 2008a]

2. Martins, L. G., Burred, J. J., Tzanetakis, G., and Lagrange, M. (2007). Polyphonic instrument recognition using spectral clustering. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria. [Martins et al., 2007]

3. Lagrange, M., Martins, L. G., and Tzanetakis, G. (2008). A computationally efficient scheme for dominant harmonic source separation. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Las Vegas, Nevada, USA. [Lagrange et al., 2008b]

[3] `http://marsyas.sf.net`

4. Tzanetakis, G., Martins, L. G., Teixeira, L. F., Castillo, C., Jones, R., and Lagrange, M. (2008). Interoperability and the marsyas 0.2 runtime. In *Proc. International Computer Music Conference (ICMC)*, Belfast, Northern Ireland. [Tzanetakis et al., 2008]

5. Lagrange, M., Martins, L. G., and Tzanetakis, G. (2007). Semi-automatic mono to stereo up-mixing using sound source formation. In *Proc. 112th Convention of the Audio Engineering Society*, Vienna, Austria. [Lagrange et al., 2007b]

1.6 Outline of the Dissertation

The remainder of the thesis is organized as follows.

Chapter 2 will start by focusing on the main aspects involved in *Auditory Scene Analysis* (ASA) and in music listening. A brief overview of the human auditory system will be presented, followed by a discussion about the most important principles involved in the perceptual organization of sound, with an emphasis on music signals. The remaining of the chapter will present some of the proposed computational approaches to model the auditory perception system, a field known as *Computational Auditory Scene Analysis* (CASA). The base architecture of a typical CASA system will be presented, including discussions about the requirements for mid-level representations for music signals, and the different computational approaches traditionally taken when attempting sound segregation. A general and preliminary introduction to the different evaluation methodologies typically used for CASA systems will also be presented, and the chapter concludes with a review of some of the most important and relevant work previously conducted in the field of CASA and sound analysis and segregation in music signals.

In Chapter 3 a computational framework for the segregation of sound events in "real-world" polyphonic music, inspired by ideas from CASA, will be proposed. Following an overview of the framework, a description of the sinusoidal modeling used as the underlying signal representation will be presented. The concept of texture windows is then introduced as a way to perform a preliminary time segmentation of the music signals, either based on fixed and manually specified texture window lengths or using the onsets detected from the input signal to dynamically perform the segmentation. The grouping of sound events is subsequently formulated as a graph partitioning problem, that is solved using a spectral clustering approach, based on the *Normalized Cut* (Ncut) criterion. Afterwards, the definition of the specific perceptual cues used as grouping criteria is discussed and a novel harmonicity cue, termed *Harmonically Wrapped Peak Similarity* (HWPS), is introduced.

The analysis of the aspects involved in the sound event resynthesis of the segregated signals conclude the chapter.

Chapter 4 presents a set of evaluation experiments and application scenarios where several aspects of the sound segregation framework proposed in Chapter 3 are tested and validated experimentally. Preliminary evaluations of the proposed perceptual grouping cues are conducted, with special emphasis on the novel HWPS cue introduced in the previous chapter. Experimental results are presented for typical MIR application scenarios, such as predominant melodic source segregation, main melody pitch estimation, voicing detection in polyphonic music signals and timbre identification in polyphonic music signals.

Chapter 5 will discuss some of the most relevant aspects of the software implementation of the sound segregation framework proposed in this thesis. The design requirements, implementation strategies and the major contributions towards the development of an open source software framework for sound segregation in music signals are put into perspective and ultimately justify the adoption of the MARSYAS[4] framework as the base software platform. The software implementation of the different processing algorithms that comprise the method proposed in this thesis are detailed and the final command line tool for sound segregation, `peakClustering`, is briefly described.

Chapter 6 closes the thesis with the final conclusions and suggests possible directions for future research.

This thesis also includes four appendixes. Appendix A includes additional and detailed information about the use of graph Laplacians in the context of spectral clustering, used for the grouping of sound events as described in Chapter 3. Appendix B describes the evaluation metrics used at the different experiments presented in Chapter 4. Appendix C provides an overview of the most important aspects of the MARSYAS software framework, used for the software implementation described in Chapter 5. Finally, Appendix D presents a computationally efficient dominant harmonic source segregation approach based around the novel HWPS cue, proposed in Chapter 3. Because it follows a different and more specialized approach than the framework proposed in this thesis, the goal of this appendix is to provide additional experimental evidence of the grouping abilities of the HWPS cues, mainly when dealing with highly harmonic or pitched music signals.

[4] `http://marsyas.sf.net`

Chapter 2

Music Listening

"My first relationship to any kind of musical situation is as a listener"

Pat Metheny

2.1 Introduction

Since the day they are born (and probably even before) humans start developing knowledge about their surrounding world. This is achieved by means of sophisticated mechanisms of perception and cognition which allow the acquisition of detailed information from the encircling environment. Among the several forms of events perceived by humans, auditory information contributes in a fundamental way to the development of knowledge.

One of the most remarkable achievements of human auditory perception is the ability to focus the attention into a single sound event in the presence of complex sound mixtures (consisting of numerous simultaneously sounding events). The capacity to detect a distant and threatening sound in a forest, the ability of focusing on a single voice deeply mixed with simultaneous conversations in a party, or the aptitude to follow the melody from a single instrument in a performing orchestra are just some examples. These scenarios constitute the classical "*cocktail party effect*", an effect first described and termed by Cherry in the early 1950's [Cherry, 1953]. When asked how they can listen to only one sound in a mixture, people typically reply that they try to just listen to *that sound* and not to be distracted by *any others*. This answer suggests that a number of separate sounds have already been unconsciously distinguished in the mixture and that the challenge is now just in deliberately focusing on one of them. Considering that the pressure waves radiated

into the air by each one of the sound-producing events in the environment end up summed together on the listener's ear drums, this becomes a strong evidence that the cognitive nature of the processes involved in auditory perception goes beyond the elementary phases of processing. They must also include some sort of higher-level processes taking place in the brain (and therefore not directly accessible) that allow to create individual descriptions based on only those components of the sound that result from the same environmental event.

In fact, human listening comprises many layers of information analysis and processing. From the transduction of low-level auditory stimuli in the ears to their organization in the brain, higher-level factors such as memory, experience, and context information take an important role in the auditory process. Listening in general is influenced by a large number of variables interacting, cooperating and competing with each other in a rather complex network.

In the context of this thesis, the following sections will skim the surface of some of the underlying principles involved in the human auditory perception mechanisms, with a special focus on music listening. These principles form the basis for the development of computational models of audition and music perception, and the later sections of this chapter will present an introductory discussion on some of the main challenges and approaches proposed in this area of work.

2.2 The Human Auditory System

Insight into the processes underlying the perceptual mechanics involved in human audition has come from several years of psychophysical research. The first stages of auditory processing are of upmost importance since they are the ones responsible for feeding the higher level perceptual processes with information about the acoustic stimulus coming from the surrounding world. It is therefore important to understand the structure and function of the very first stages of auditory processing. This section will present a brief review of the human auditory system, but the interested reader my find more detailed descriptions and discussions in the standard literature on this topic (e.g. [Moore, 1989, Beranek, 1990, Moore, 1995, Yost, 2001]).

The peripheral region of the human auditory system is responsible for converting sound vibrations into neural information which is conveyed to higher levels of the auditory system located in the brain. Besides this transduction function (mechanical energy in the air molecules into action potentials in the auditory nerve), the peripheral region also performs

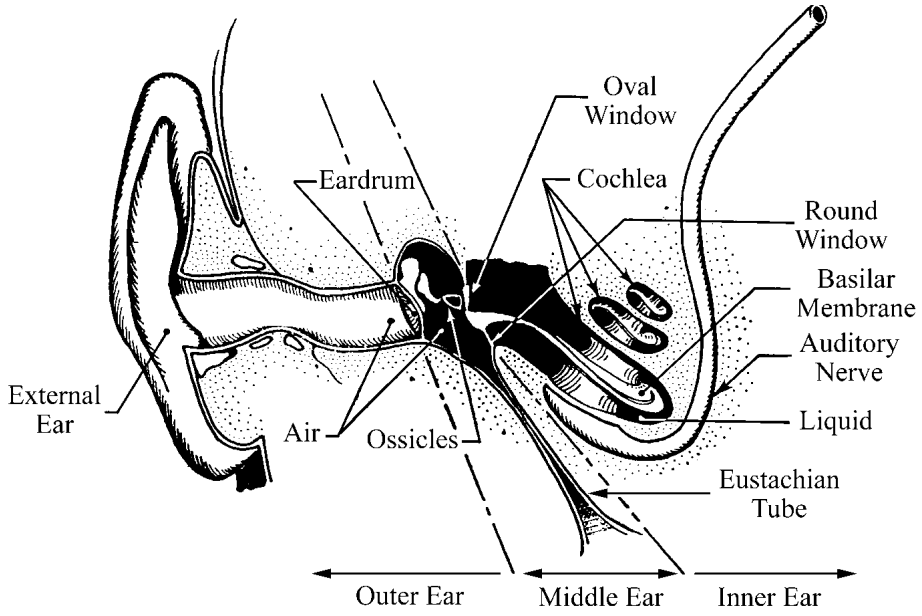


Figure 1: *A sketch of the peripheral region of the human auditory system (adapted from [Beranek, 1990]).*

a pre-processing of the acoustical signal, the most notable aspect being frequency analysis. The structure of the peripheral region is represented in Figure 1. Broadly, the periphery can be divided into three areas: the outer, the middle and the inner ear. The outer ear (pinna) collects sound energy from the surrounding environment and transmits it through the ear canal (meatus) to the eardrum (tympanic membrane), giving rise to an oscillatory motion. The pinnae play a role in spatial hearing, by imposing spectral characteristics on sound that depend on its direction of incidence with respect to the head. In the middle hear, the vibrations in the eardrum are transmitted to the fluid-filled cochlea by means of three tiny bones (the ossicles: the malleus, the incus and the stapes). They form a mechanical lever system whose main purpose is to match the impedance of air to that of the cochlear fluids and to protect the ear against aggressive sound intensities by modifying the transmission gain. The middle ear is filled with air whose pressure is equalized with that of the outside environment through the Eustachian tube. The inner ear corresponds to the cochlea, the organ of hearing. It is a tube shaped like a snail shell, filled with fluid, that is divided along its length by two membranes: Reissner's membrane and the basilar membrane. The basilar membrane varies in mass and stiffness along its length so that different regions of the membrane vibrate at different resonant frequencies. As a result, the basilar membrane exhibits a complex pattern of motion in response to sound-induced

movement of the cochlear fluids. In particular, the response of the basilar membrane to a sinusoidal stimulus is a traveling wave that moves along the length of the cochlea. The travelling wave oscillates at the frequency of the stimulus and reaches its peak amplitude at the location where the stimulus frequency matches the resonant frequency of the basilar membrane. High frequencies resonate near the oval window, while low frequencies cause the membrane to resonate near the cochlea apex (helicotrema). As a result, the basilar membrane can be seen as a wide-band filter which is able to decompose a complex sound into a number of frequency components. However, this peripheral frequency decomposition is only partial and sharper frequency analysis is performed at higher levels of the human auditory system.

In fact, although all these are interesting aspects providing valuable insight into some of the properties of the human auditory system, they only refer to a pre-processing stage. The most important processing functions, such as sound segregation or spatial perception, take place at higher centers of the human auditory system, as discussed in the next section.

2.3 Scene Analysis in Audition

The first and more traditional psychophysical approaches to auditory perception attempted to relate input (i.e. acoustic stimulus) to the output (i.e. auditory sensation), deriving simplified models of the auditory system, without getting concerned with the immense detail and entangled complexity of the physiological and psychological aspects of the auditory system. They were mostly concerned with questions such as the smallest sound manifestation the auditory system is able to sense, how far apart do frequencies of two pure tones have to be in order to be distinguished when played sequentially or simultaneously [Patterson and Moore, 1986], or how does the perceived loudness of a tone grow as its physical intensity is increased [Fletcher, 1940, Stevens, 1957, Moore, 1989]. However, most of these phenomena happen at the early stages of the human auditory system (as discussed in section 2.2) and fail to fully explain the more intricate cognitive aspects of audition, mainly when in face of complex mixtures of sounds.

Bregman, who had been studying phenomena of perceptual organization in hearing for several decades, realized in late 1960's that, despite great success in characterizing low-level detection and discrimination abilities, the sound domain had no work equivalent to the ecological (i.e. environmentally-relevant) problems that were being studied in

the field of vision [Gibson, 1979]. Researchers in the computer vision field, while studying the theoretical principles and working on basic experimental data from vision research [Marr, 1982, Pinker, 1984, Humphreys and Bruce, 1989], defined the term *"Scene Analysis"*. Their challenge was to understand how humans analyse images or scenes of varying complexity, and how the visible properties of existing elements (e.g. edges, surfaces, textures, colours, distances) were grouped together, enabling the viewer to determine the correct global shape and properties of the pictured objects. Influenced by this work, Bregman was the first to systematically explain the principles underlying the perception of complex acoustic mixtures, whereby all the auditory evidence coming over time from a single environmental source is put together as a perceptual unit. He coined this process *Auditory Scene Analysis* (ASA) [Bregman, 1990].

This cognitive approach to perception implies that the information contained in the stimuli that reach the sensory organs must be interpreted at some higher level processes, since by itself, sensory information is not always sufficient to form a consistent image of the surrounding sound environment [Bregman, 1990, McAdams and Bigand, 1993]. One of the reasons for this is the temporal nature of sound, where sound events succeed one another in time (a typical example would be music sounds). The perception of the structure of these events requires the construction of a mental representation where the relations among events (which may be separated in time by arbitrarily long intervals) can be established. Furthermore, at the presence of insufficient sensory information, the perceptual system tries to take into consideration knowledge that it has previously acquired of the surrounding sound world. This prior knowledge interacts with the current sensory data to interpret the auditory stimulation.

McAdams and Bigand [McAdams and Bigand, 1993] propose a summarized and schematic representation of this process, where the most important types of auditory processing blocks and their corresponding interactions are depicted, as in Figure 2.

Sound vibrations are received by the inner ear where they are analyzed and *transduced* into nerve impulses that are sent through the auditory nerve to the brain (as described in section 2.2).

The fusion and segregation of concurrent sound elements into sound events, as well as the temporal integration and segregation of successive sound events into auditory streams, are undertaken by *auditory grouping processes* taking place in the brain. As will be discussed later in section 2.3.1, this process is based on a number of auditory cues taken by the auditory system from the received acoustic stimulus, and known as *primitive grouping*.

From the resulting grouped sound events and inferred auditory streams, it is then

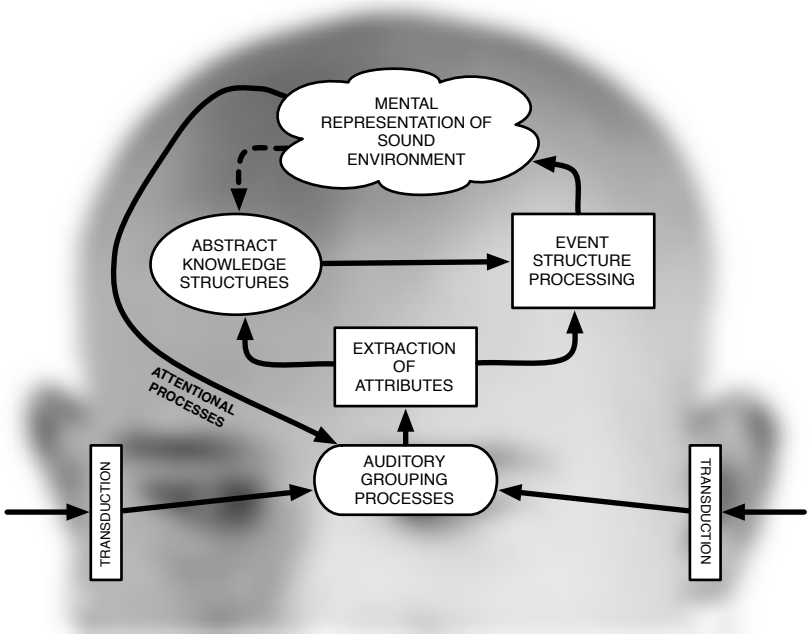


Figure 2: *The main types of auditory processing and their interactions (adapted from [McAdams and Bigand, 1993]).*

possible to *extract perceptual attributes* which provide a representation of each element in the auditory system.

These attributes can now be interpreted with respect to evoked *abstracted knowledge structures* that allow the recognition and identification of events and sequences of events, as well as the assignment of meaning and significance. All this is performed at the light of the local stimulus context and the past experience of the listener (see the discussion of *schema-based grouping* in section 2.3.1).

These perceived relations among sound events can influence the perception of subsequent events (and even revise past interpretations – e.g. the McAdams-Reynolds oboe example [McAdams, 1984] or the "old-plus-new" organization principle described in [Bregman, 1990, pp.261]). This *event structure processing* influences the establishment of larger-scale structural relations.

The resulting elements from the event structure processing give rise to the progressive creation of a *mental representation of the sound environment* involving the listener. Given the transitory nature of the acoustic stimulus, information about acoustic events must be somehow accumulated through time. This accumulated mental representation takes a crucial part on the ability to dynamically focus attention to new incoming acoustic

information and to process it taking into consideration the previous experience, allowing the listener to establish larger-scale relations between the sound events perceived along time. This is clearly visible when listening to music, where the listener may be following the various lines of musical development in a song, creating a mental representation of the musical piece since its beginning. This allows the listener to realize that the music may be at some point moving towards a recapitulation of the main theme (e.g. the chorus section of a pop song) or to the closing of the musical piece.

The interconnection between the event structure processing, abstract knowledge structures and the mental representation structures allows the listener to create expectations and to anticipate forthcoming events, setting up the listener's attention to events of a particular kind at some specific times. Furthermore, these *attentional processes* may still influence the activity at low-level organizational processes, even at the level of sensory transduction.

2.3.1 Perceptual Organization of Sound

Although the early stages of the human auditory system are an important part of the human ability to solve the ASA problem from eardrum vibrations, relatively little is known at the physiological level about how higher order mechanisms participate in the perception processes. It is intrinsically difficult to measure something as abstract and unobservable as the internal perception of sounds that belong together.

Taking on the results from carefully prepared perceptual experiments using simple stimuli (e.g. [Bregman and Ahad, 1996]), Bregman brought some light to these questions and proposed an attractive conceptual framework for ASA where he assumes an important distinction between the actual acoustic sources in a mixture and the corresponding mental representations [Bregman, 1990].

Shepard has argued that because evolution took place in a world containing a lot of examples of regularities, it is likely that the perception systems of animals evolved to take advantage of them [Shepard, 1981]. He called the match of the perceptual abilities developed by animals to the regularities exhibited in their physical environment as "psychophysical complementarity". Given so, and if taking Shepard's hypothesis as right, a good strategy for deriving the laws of auditory organization would be to try to understand the relations among the components frequently present in the mixtures of incoming sounds created by different environmental events.

Based on this assumption, and after several experiments, several authors [Bregman, 1990, McAdams and Bigand, 1993, von Noorden, 1975, Hartmaan, 1988,

Darwin, 1997, Moore, 1989, Bey and McAdams, 2002, Yost, 1997] have proposed a number of auditory cues and principles exploited by the auditory system when performing ASA, divided by Bregman into a a two-stage organization process, namely *simultaneous grouping* and *sequential grouping*.

Additionally, Bregman also proposed a distinction between two other related processes occurring in the human listener, although probably at a different level. They are known as *primitive* grouping and *schema-based* grouping. The following sections will discuss in detail each one of them.

Simultaneous versus Sequential Grouping

Bregman suggested the division of the ASA problem as a two-stage organization process. In the first stage, called *simultaneous grouping*, acoustic energy occurring concurrently in different frequency regions are fused into a single percept, which takes place in a specific time-frequency region known as a *segment*. Fusion of harmonically related sinusoid components (in the Fourier sense) into a single "rich" tone is a good example of this process. This grouping principle is usually based on auditory cues as the ones summarized next.

Simultaneous Grouping Cues

- **Harmonicity.** When a body vibrates with a periodic movement, its vibrations create an acoustic pattern whose frequency components are multiples of a common fundamental (i.e. harmonics of a fundamental frequency). Interestingly, the auditory system tends to group a set of harmonically related acoustic components into a single event.

- **Common onset and offset.** Sound components with the same onset time and, to a lesser extent, the same offset time tend to be grouped into the same unit by the auditory system. Since unrelated sounds seldom start or stop at exactly the same time, the auditory system assumes that components exhibiting a "common fate" are likely to have origin in a common single source.

- **Common modulation.** Another example of the sensibility of the auditory system to "common fate" is the segregation of components of a mixture exhibiting a common amplitude or frequency modulation. If a sound source exhibits amplitude or frequency modulation, it is expected that all of its components exhibit similar modulation manifestations.

- **Spatial proximity.** One of the best generalizations that can be made about independent sound sources is that they normally occupy distinct positions in space. As a consequence, sound source location could provide the strongest cue in the construction of an ecological representation of the surrounding sound environment[1]. However, spatial location of sound sources is, accordingly to some authors, taken as an auxiliary cue by the human auditory system [Bregman, 1990]. In fact, humans can still segregate sound mixtures when listening to monaural acoustic signals (i.e. no spatial information is included for the different sources comprising the mixture). Bregman suggests that due to the effects of reverberation and to the transparency of sound, localization cues become comparatively unreliable [Bregman, 1990, pp.83]). On the other hand, Huron does not find these arguments compelling and suggests that "the relative unimportance of localization in stream formation suggests that the ecological account may be incomplete" [Huron, 1991].

In the second stage, known as *sequential grouping*, the series of segments from the first stage are built up into one or more *streams*. Each stream is a sequence of events assumed as coming from a single source (e.g. a human listener can easily assign the notes of two melody lines played concurrently by two different instruments to the correct source). The following list presents some of the auditory cues possibly involved in sequential grouping.

Sequential Grouping Cues

- **Time and frequency proximity.** Acoustic components close in frequency tend to be interpreted by the auditory system as coming from the same sound source. Additionally, components close in time tend to be perceptually grouped into a same stream. Consequently, the closer in time and frequency components are, the stronger the tendency of being grouped into a same auditory stream.

- **Loudness proximity.** Sources with different energy levels consist of acoustic components with correspondingly distinct sound intensities. The auditory organization of physical stimuli also resorts to loudness similarity to group components into the same acoustic event.

[1] Following the argument presented by Gibson when proposing an ecological approach to visual perception [Gibson, 1979], an ecological view of sound states that it is only possible to fully explain some auditory behaviour if taking into account the environment in which that behaviour took place.

- **Timbral similarity.** Sound elements that sound alike (i.e. have similar spectral properties or similar timbres) tend to be grouped into the same auditory stream [Bregman, 1990, pp.19; 92-127]. This similarity grouping is increased by the temporal closeness of similar auditory components.

- **Smoothness of change.** A single sound tends to change its properties smoothly over time. Consequently, a sequence of sounds from a same source also tends to change its properties gradually over time. The auditory system favors the sequential grouping of sound components whose properties are stable. Abrupt discontinuities are perceived as new and unrelated events, probably with origin in a distinct source. Smooth changes in pitch contour, intensity, spatial location and spectral characteristics of a sound are usually interpreted as a continuation of an existing sound.

- **Periodicity.** A sequence of rhythmically related tones, separated in time, tends to be integrated into a same stream. This usually happens after a listener is exposed for some time to periodic repetitions of sound events, which allow the auditory system to establish a rhythmic relation between the individual sounds, forming an auditory stream.

Primitive versus Schema-based Grouping

Listening resorts to aspects dealing with prior knowledge, learning, memory and context, which govern attention, creation of expectations and the resolution of ambiguous situations. Termed as *Schema-based* grouping by Bregman [Bregman, 1990, pp.38], this top-down process is founded upon the activation of learned patterns (i.e. schemas), either as a purely automatic process (e.g. when one thinks to hear someone calling his/her name in a busy coffee shop) or as a voluntary action (e.g. when one is paying attention to the calling of his/her name while someone is announcing a list of names). In either case, automatic and voluntary recognition assume that knowledge about the structure of particular sounds or classes of sounds (which for some reason were considered relevant to the listener) have been acquired during previous exposures.

However, if this was the only process available for the human auditory system to decompose sound mixtures, it would make learning new schemas for unfamiliar but important sounds difficult, unless they had frequently occurred in isolation in the past. Consequently, some other mechanism must exist that can explain how the auditory system is capable of partitioning an incoming mixture composed of unknown sounds into separate

and corresponding acoustic events (e.g. someone listening to electronic music, where original synthesized sounds are used instead of more familiar timbres from traditional musical instruments).

In fact, there is a more general mechanism, known as *primitive* grouping, which instead of depending on acquired knowledge, is based on the intrinsic structure of environmental sounds and is regarded as an innate ability of the human auditory system. Governed by mechanisms similar to those proposed by Gestalt psychologists in relation to visual perception [Palmer, 1999], the auditory system mainly employs a bottom-up strategy that exploits general acoustic regularities in an incoming signal, known as auditory cues (as described previously in this section). These cues are based upon certain relations between the acoustic properties of a sound, being the result of general properties of sound-producing events, and consequently not specific to any particular class of sound (e.g. voices, music, noises).

Although all these organizational principles have in large part been derived from the results on the use of simple and experimental acoustic stimuli based mainly on pure sinetones or gated white noise, they still prove useful and applicable to the analysis of real-world complex sounds [McAdams and Bigand, 1993, Wang and Brown, 2006]. It is also important to note that none of these cues by themselves would guarantee a successful sound segregation from a complex mixture. The auditory system seems to dynamically adapt the priority or attention given to each type of cue depending on the specific characteristics of the signal arriving at the ears at each instant and the current mental representation of the physical surrounding world.

On the other hand, situations exist where the extracted cues may conflict with the correct perception of the actual surrounding world. Auditory illusions [Warren and Warren, 1970] occur when the inference realized on the basis of the available cues is incorrect and results in the perception of an unreal sound object. Music is a specially fascinating and illusion rich domain [Deutsch, 1975] where composers have been exploiting for centuries these perceptual characteristics in an attempt to create seductive sound figures that trick the sound perception mechanisms. Baroque composers provide remarkable examples of auditory illusions in music, where for example the listener has the clear impression of hearing two violins playing in different pitch registers, but where in reality there is just one violinist rapidly alternating notes between registers (termed as “virtual polyphony” in [Bregman, 1990, pp.464]).

2.3.2 The Specific Case of Music Perception

Music is built from sound [Bregman, 1990, pp.455], arising in all its known instantiations around the world as an elaborate interaction between the sound-generating properties of physical objects (i.e. music instruments) and the sound perception abilities of the human auditory system. The auditory capabilities humans possess and employ for survival while in their environmental world are exactly the same they use when creating, listening and enjoying music – "*the same ears, the same cochleae, the same auditory brain*" [Scheirer, 2000, pp.14]. Through a complex socio-cultural evolution, music has been used by humans to create special kinds of sounds that tickle the auditory apparatus in enjoyable and intriguing ways. Consequently some authors defend that much of the process of listening to and understanding music follows very closely the very same basic principles of ASA, as presented in the previous section [Bregman, 1990, pp.455]. This statement is mostly based on the fact that humans, even without any kind of musical background, are typically able to extract, almost unconsciously, a great amount of relevant information from a musical signal. Features such as the beat of a musical piece, the main melody of a complex musical arrangement, the sound sources present in a complex musical mixture, the song structure (e.g. verse, chorus, bridge) and the musical genre of a piece, are just some examples of the level of knowledge that a naive listener is commonly able to extract just from listening to a musical piece.

One of the main interests of this area for the cognitive sciences is that music is ubiquitous in the cultures of the world. Musical systems have attained a degree of structural (i.e. grammatical) complexity comparable to that of language, and similarly to it, humans become sensitive to the musical conventions of their culture without formal training at very early age [Handel, 1989, pp.381].

Naive and Trained Music Listeners

Although there has been a growing interest in the perception of complex auditory patterns, the origins of musical competence remain an open question [Trehub and Trainor, 1993, Bigand and Poulin-Charronnat, 2006]. The question is on when and how do humans develop the ability of understanding, appreciating and participating in musical activity.

As discussed by Trehub and Trainor [Trehub and Trainor, 1993, pp.278], one hypothesis would be to accept that music perception is almost entirely learned, consequently founded in schema-based integration processes (as discussed in section 2.3.1) necessary for acquiring musical knowledge – i.e. without such prior musical knowledge, no music

listening or perception would be possible. Another approach is to assume that basic music perception abilities could be innate and given in their entirety by nature (i.e. based on primitive grouping processes, as discussed in section 2.3.1), in which case the perception of auditory musical events would be uniform across individuals.

However, a more consensual hypothesis falls somewhere between these two extremes: innate perceptual processes could be considered to set the stage for perceiving complex patterns such as musical sequences, above which, learning processes would take place [Handel, 1989, Krumhansl, 1992, Jones, 1990].

In fact, it may not be reasonable to assume that the music processing strategies of musically untrained adults are due entirely to musical exposure, with no carry-over of primitive processing strategies from early life. Handel argues that innate abilities supply the basic perceptual structures which are then fine-tuned to the cultural context and conventions of a particular musical system [Handel, 1989]. On the other hand, composers, in creating music, intuitively capitalize on universal principles of auditory perception. Rather than a result of ages of conformation to the task of listening to music by means of evolution, the auditory system is instead on the genesis of the way music exists and evolves alongside humans.

Bigand and Poulin-Charronnat recently conducted a series of experiments where they tried to identify which musical abilities in human listeners do not depend on formal music training [Bigand and Poulin-Charronnat, 2006]. Surprisingly, and not without some controversy, they concluded that listeners without any musical training (called *naive listeners* from now on) but with sufficient exposure to music use the same principles as musically trained listeners when listening to music and structuring what they hear (even if in a more limited way), suggesting that an intensive musical training is not required to respond to music in a sophisticated way. In fact, they found that musically untrained listeners perceive musical tensions and relaxations in both melodies and harmonic sequences similarly to musicians, and show identical ability to anticipate musical events. They also concluded that both groups encountered the same type of difficulty to integrate local structures in large-scale structures, even though musicians performed usually better with explicit tasks. Musically untrained listeners showed equivalent performance when required to learn new compositional systems and both groups responded very consistently to music in an emotional (i.e. affective) way. Overall, both naive and musically trained individuals have shown to perform similarly in cognitive and emotional tasks. According to these authors,

"there seems to be an initial predisposition of the human brain for music processing that is triggered by the extensive exposition to musical stimuli in everyday life. Thanks to

both factors (predisposition and intensive exposure), non-musicians become "experienced listeners" that do not strongly differ from musically trained listeners in numerous experimental tasks".

Despite this conclusion, the authors make clear that this claim does not imply that no differences exist between both groups of individuals. Their findings indicate instead that any formal and intensive musical training advantages in musical competence remain small in light of the huge difference in training exhibited by the two groups of listeners.

At the light of the previous discussion, and as also suggested by Terhardt, it would not be therefore surprising for some of the most basic music perception abilities to be already operative in infancy [Terhardt, 1987]. As a matter of fact, at the outset of a number of experiments conducted by Trehub and Trainor [Trehub and Trainor, 1993], primitive perceptual grouping processes used for music listening (such as the ones described by Bregman [Bregman, 1990, Bregman, 1993], Bigand [Bigand, 1993] and others, and partially presented in section 2.3.1) seem indeed to be already operative in infancy.

This implicit music learning hypothesis goes into opposition to the more accepted approach that stipulates that musical abilities are mostly determined by an intensive and formal musical training, remaining rather crude in untrained listeners (see [Levinson, 1997] for a debate). While it may still be true that when compared to naive listeners musically trained listeners are believed to be in a better position for successfully resolving some of the most intricate ambiguities frequently found in polyphonic music signals (e.g. recognizing or identifying the real sources or musical instruments playing in a mixture, identifying the name and individual notes of a chord, or transcribing by ear the musical score of a music performance), it is also accepted that naive listeners are in fact able to extract a great deal of musical information from an audio signal (e.g. although not able to identify by name a chord or to recognize that such a sound consists of separate notes playing simultaneously (they tend to perceive it as a combined whole sound – a tone color), naive listeners are nevertheless able to feel the harmony and chord changes).

Physical and Perceptual Sounds in Music

One could discuss why humans are intrinsically able to segregate or identify sound "objects" or sound events in complex audio mixtures, but not necessarily the actual sound sources. In fact, there is not an unique definition to what a sound source is. One possibility is to consider each vibrating physical entity (e.g. each musical instrument) as a sound source. Alternatively, a sound source could be defined accordingly to the human

perception of a single and coherent auditory unit. Kashino, in [Klapuri and Davy, 2006, pp.302-303], refers to these two alternatives as *physical sound* and *perceptual sound*. As an example, a chord formed by the combination of simultaneously sounding notes from distinct musical instruments, may be perceptually interpreted as a single coherent auditory object, becoming hard to identify the individual instruments playing (at least by a naive listener).

Music composers frequently explore the combination of the sounds produced independently by different musical instruments into intricate harmonies and new and complex sound textures. Their intent is to often "fuse" sounds stemming from different origins into one unified but distinct perceptual element, fooling the ear into hearing a single and new "virtual" instrument timbre instead of the set of familiar musical instruments that compose the mixture. Accordingly, tonally fused sounds seem to play an important role in music perception [Scheirer, 2000, pp.30]. Bregman referred to these perceptual constructs that result from the tonal fusion of individual sound elements as *chimera* [Bregman, 1990, pp.459].

On the other hand, the *chimeric* nature of the sounds that emerge from the fusion of different simultaneously sounding notes raises the argument of whether or not musical notes, a central and atomic element in several music cultures, correspond to perceptual units. As discussed in the previous sections, humans extract auditory cues that generate percepts of the acoustical elements in a sound. And while in several circumstances such percepts are closely related to musical notes, other cases exist where such a correlation is not easy to establish (even for trained listeners). Scheirer argues that "*most stages of music perception have nothing to do with notes for most listeners*" [Scheirer, 2000, pp.69]. Furthermore, he states that "*the acoustic signal must always be considered the fundamental basis of music perception*", since "[it] *is a much better starting point than a notation invented to serve an entirely different mode of thought*" [Scheirer, 2000, pp.68].

Consequently, music is in most senses more challenging to segregate than speech or other environmental sounds, where music orchestration and composition intentionally bound together multiple, distinct, and simultaneous sounds, turning them into a single, perceptually indivisible auditory object. Huron defined music listening as "*a type of scene analysis problem, with the exception that the auditory scenes are populated by a cast of mostly fictional sources*" [Huron, 1991]. As briefly discussed in the previous section, only trained listeners are usually able to identify with some degree of success the physical sources involved in a complex polyphonic musical piece. In such situations, the information available at the level of the sensory organs may be too ambiguous or even insufficient

to give rise to a correct or unique perception of the current situation. The identification of the sources of sound becomes highly dependent on pior knowledge and can be different according to the kinds of information available to the listener. In such circumstances, a naive listener or an expert musician would probably arrive at different interpretations of the same musical sound since they do not share the same auditory knowledge base.

2.4 Computational Auditory Scene Analysis (CASA)

Bregman's work, although being a careful report of empirically-observed psychoacoustic phenomena, proposed a plausible and almost formal specification of rules by which acoustic stimuli could be converted into independent percepts [Bregman, 1990]. His work provided a suitable foundation for most of the proposed computational attempts to model the auditory perception system, including the one presented in this thesis. It paved the way to a new field of study, known today as *Computational Auditory Scene Analysis* (CASA) [Rosenthal and Okuno, 1998, Wang and Brown, 2006] and defined by Wang and Brown as

"[CASA is] *the field of computational study that aims to achieve human performance in ASA by using one or two microphone recordings of the acoustic scene*" [Wang and Brown, 2006, pp.11].

Accordingly, in a CASA system the goal of the sound processing is to extract multiple sounds from a mixture. The output sounds can be subsequently analyzed independently to compute their features so that they can be subsequently interpreted, classified or processed. Given its perceptual motivation, the output sounds of a CASA system should be similar in some perceptually important way to the sound events or "objects" that constitute the mixture. For instance, while state of the art automatic speech recognition (ASR) systems perform well on clean speech, their performance rapidly decreases in the presence of interfering sounds. An appealing idea to minimize such a degradation would be to first "clean up" the speech signal from all the background interference and then feed the separated signal into the same ASR system.

One way to approach the development of such a machine listening system is to follow a pure computational model, where systems are viewed as "black-boxes". These systems are pragmatically goal-oriented, where the devised algorithms become less important than the outcome. Consequently, in such an approach, the main concern is focused on the development of a working system and not so much on modelling the human perception

mechanisms. Although this strategy may pose some limitations to the ability to fully explain the human auditory processes, it may still provide meaningful auditory information by means of a reverse engineering exercise. A possible advantage of this approach is that for some specific and constrained conditions it can be possible to attain performances that surpass the ones delivered by the human auditory system.

On the other hand, a more psychologically and perceptually motivated approach would allow putting emphasis on the detailed study and close modelling of the way humans hear, perceive and understand sound and music, following a "clear-box" line of reasoning. This allows the development of more or less sophisticated psychoacoustic models of the human auditory system which may allow explaining some of the auditory phenomena otherwise difficult to justify. Ultimately, this methodology could allow a better theoretical insight into the human hearing abilities and consequently could pave the way to the development of better artificial hearing systems, since no other system has proved to outperform human listening abilities so far. However, several aspects of the auditory perception, deeply buried in the brain, are still just superficially understood, making them very difficult to model and simulate. Additionally, computer models of the human auditory system tend to be computationally intensive.

Wang and Brown's definition of CASA does not imply that the human auditory system must be imperatively and rigorously modeled in a computer listening system. In fact, a lot could be gained by exploiting both "black-box" analysis algorithms and the knowledge about the mechanisms involved in the human auditory perception – a so-called "gray-box" approach. Consequently, and differently from some other approaches to sound separation that are based on more mathematical or statistically oriented techniques (e.g. Independent Component Analysis (ICA) [Abdallah, 2002] and Non-negative Matrix Factorization (NMF) [Vembu and Baumann, 2005]), most of the solutions proposed in the area of CASA, including the one proposed in this thesis, follow to some extent the known principles of the inner workings of the human auditory system.

However, and as argued by Ellis in his PhD thesis, the impression that there are a set of rules waiting to be translated into computer code may be deceiving [Ellis, 1996, pp.18]. The experimental principles coming from the use of simple and highly constrained psychoacoustic stimuli such as sine-tones and gated white noise may in some cases map poorly to the messier domain of real word sounds and mixtures. Therefore, although such principles aid in the construction of more perceptually aware sound listening machines, they provide no guarantee to the successful handling of real world signals the way humans do.

2.4.1 Computational Models for Music Scene Analysis

Looking at the specificities of music signals and their perception by human listeners when compared to generic environmental sounds (see Section 2.3.2) created a particular interest in the development of computational models for music scene description. Accordingly, the goal of Computational Music Scene Analysis (CMSA) is to develop computer systems that are able to describe and understand musical audio signals similarly to the way humans do (or at least musically untrained humans, i.e. naive listeners – see section 2.3.2) [Goto, 2006, pp.252]. Even if the primary research approaches to CMSA are based on signal processing techniques, it is valuable to take on the knowledge about the human auditory processes and their music listening abilities , as well as all the CASA contributions (discussed in the previous section).

Similarly to CASA, the brain mechanisms underlying the human ability to interpret music are not yet fully understood, making those high-level cognitive processes the most difficult to model and simulate in a computer system. Furthermore, the difficulty of estimating music scene descriptions is highly dependent on the various specific properties of musical signals, which in most senses are even more challenging to analyse than non-musical sounds. Differently from non-musical sounds, and mainly resulting from the exercise of composition and orchestration, sources in a musical piece often play simultaneously and favor consonant pitch intervals (as discussed in section 2.3.2). Music signals are mostly polyphonic (i.e. signals where two or more sound sources are playing concurrently at each time instant), temporally structured, and complex, which imposes a higher complexity when compared to monophonic signals (i.e. signals where only one sound source is playing at each instant). This increases significantly their time-frequency domain overlap, turning their perception and segregation substantially harder. Furthermore, musical instruments present a wide range of sound production mechanisms, resulting in signals that exhibit a high variety of spectral and temporal characteristics.

Consequently, techniques usually employed for music signals where there is just one single sound source playing at each time instant tend to perform poorly when applied to complex mixtures. Additionally, sound sources in musical signals are highly correlated and usually exhibit a high degree of dependency among each other. This makes some of the separation criteria and assumptions traditionally used in speech and non-musical sound separation approaches (e.g. statistical independence) to become invalid or prone to failure when applied to music signals.

Yet, music signals are still well organized, presenting important and unique relationships between various simultaneous and successive sounds. Hence, CMSA can be considered as a sub-topic of the larger field of CASA, since it makes use of most of the approaches and principles used for the analysis and separation of speech and environmental sound mixtures, but dealing with the specific challenges posed by music signals. As a result, it is desirable that ideas and techniques developed in the scope of CMSA may be possible to be extended back to general audio signals, providing a good starting point for creating new frameworks for understanding audio signals in general.

Given this close two-way relation between CASA and CMSA, and for the sake of generality (although this work is focused on the analysis of music signals), CASA will be used in the scope of the remaining of this thesis to broadly refer to the study and development of computational models for generic environmental sounds as well as for acoustical music signals.

2.4.2 Base Architecture of CASA Systems

Broadly founded on Bregman's conceptual framework for ASA [Bregman, 1990], Figure 3 depicts a representative architecture of CASA systems, which is also the base structure generically followed by the sound segregation framework proposed in this thesis (see Chapter 3).

In this architecture a digitally recorded acoustic mixture is first processed by an analysis front-end which somehow models the peripheral analysis processes in the human ear. This results in some form of time-frequency representation of the input signal (e.g. an STFT spectrogram or a cochleagram [Wang and Brown, 2006, pp.15]). From this representation it is then possible to generate *mid-level representations* (e.g. spectral peaks, frequency tracks, harmonic structures), which are expected to bring out important characteristics of the acoustic signal (see Section 2.4.3). Scene organization can now take place on the basis of primitive grouping cues (see Section 2.3.1) and, if available, take into account previously trained models of individual sound sources [Ellis, 2006, Burred and Sikora, 2007], producing separated streams for the sound sources and events existing in the acoustical signal.

From each separated stream it is then usually possible to resynthesize an audio waveform. Although it is a subject of discussion whether the human brain does resynthesize or not the acoustic waveforms of each source separately (see the discussion about "understanding without separation" in [Scheirer, 2000, pp.70]) the auditory system is an useful

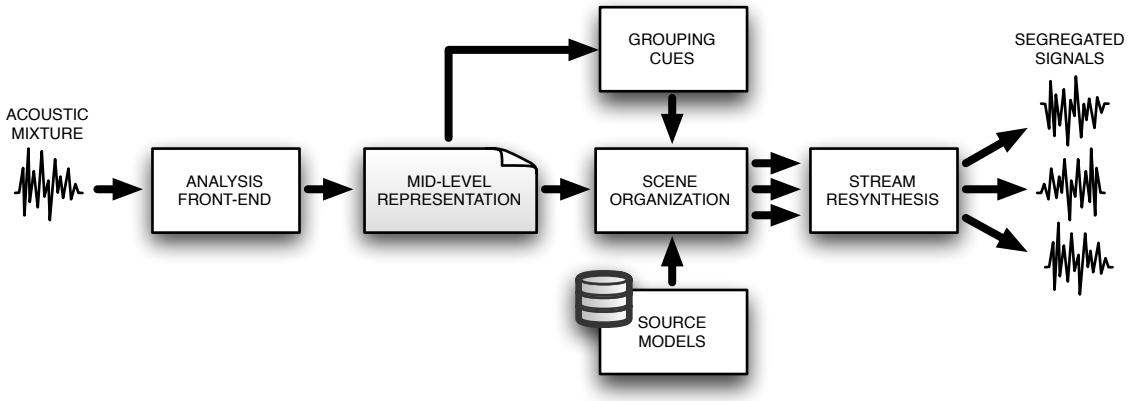


Figure 3: *System Architecture of a typical CASA system.*

reference in the development of sound source separation systems, since it is the only existing system which can robustly separate sound sources in various circumstances. As a result, the resynthesis of the separated signals allows to assess the performance of a CASA system by means of listening tests or by using objective measurements (see Section 2.4.5 for a more thorough discussion on the evaluation approaches traditionally used in CASA). The resynthesis of the segregated components in an acoustic mixture is also of great interest to most CASA applications (see Section 1.4).

2.4.3 Mid-level Representations for Music Signals

One of the most common digital representations of an acoustical signal is the sampled waveform, also known as the *time-domain* representation (see top panel of Figure 4). Each sample describes the sound pressure level of the acoustic signal at a specific time, allowing an adequate visualization of the variation of the energy of the signal along time.

Although this representation allows the efficient visual inspection of some signal features (e.g. it is easy to identify silence regions, the onsets of sound events, fade ins and outs) and is the input of many basic audio signal analysis and processing operations, it is not directly suitable for some more challenging tasks, such as the segregation of complex music mixtures.

As a result, some other type of representation is necessary to bring out more clearly the important characteristics of the signal for the application in hand, usually leading to a so-called *mid-level representation*. Mid-level representation is a concept mainly coming

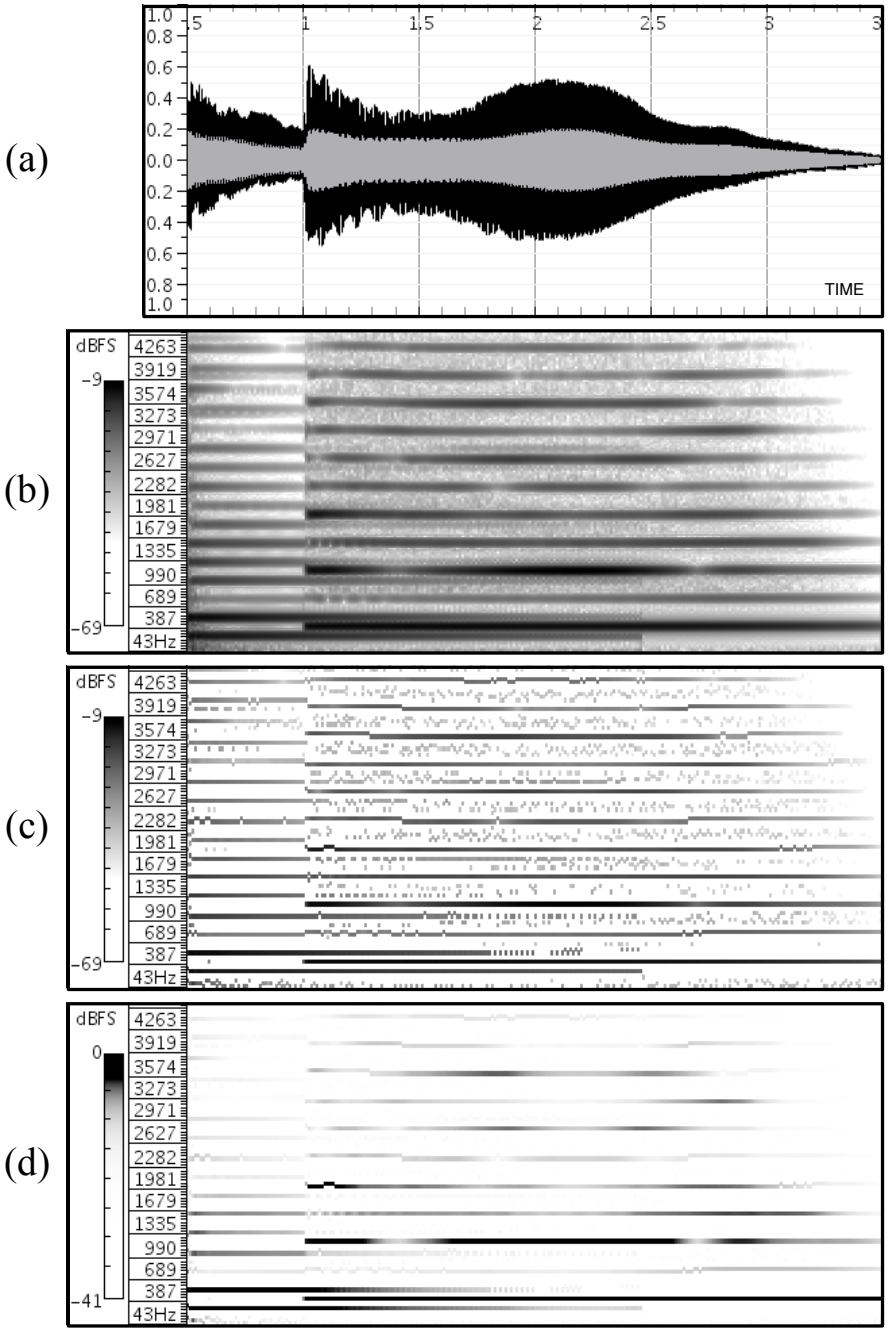


Figure 4: *Representations of an acoustic signal of a piano tone (C3) followed by an overlapping clarinet tone (G3). The upper panel (a) shows the time domain representation of the signal, followed in panel (b) by its spectrogram representation. Panel (c) shows a sparser spectrogram where only spectral peaks are considered and panel (d) shows the corresponding peak trajectories after tracking spectral peaks over time.*

from the field of computer vision research, particularly after the pioneering ideas of David Marr [Marr, 1982]. It has also become widely accepted by the computer audition community as a concept useful to model hearing [Bregman, 1990, Cooke, 1991, Ellis, 1996, Scheirer, 2000, Wang and Brown, 2006].

Auditory perception may be viewed as a hierarchy of representations from "low" to "high", where low-level representations roughly correspond to acoustic stimuli reaching the cochlea, and high-level representations relate more closely to cognitive models of sound in the brain [Ellis and Rosenthal, 1995]. Between these two levels it is admissible to presume the existence of a network of representations, labeled *mid-level*, about which little direct knowledge is available. Accordingly, the definition of such a mid-level representation must take into consideration the limitations and strengths that it imposes on the evaluation of the "meaning" of an acoustic signal during the process of reducing the number of "objects" in it. The chosen representation should readily answer the questions asked by the higher levels of processing, using the most efficient computational method possible.

As a result, CASA approaches usually start by defining an adequate representation of the acoustic signal, and among the several levels of abstraction possible, the choice should point towards the way information is supposedly represented by the human brain. The key issue is how to define the "adequacy" of such a representation.

For the specific case of music, musical scores are the most common representation of music in Western cultures. However, such a representation does not seem an appropriate mental representation of a musical scene [Goto, 2006, pp.252]. In fact, as argued in [Scheirer, 2000], and as discussed in section 2.3.2, naive listeners are able to understand music to a surprisingly great extent without mentally representing audio signals as music scores. Given so, alternative sound and music representations should be considered. Goto [Goto, 2006, pp.253] and Ellis and Rosenthal [Ellis and Rosenthal, 1995] proposed the following guidelines for the definition of an appropriate sound (musical sound in particular) representation framework:

1. The representation should decompose sound to a granularity at least as fine as the sources of interest.
2. The representation should have a reduced number of components compared with the number of original samples. While the number of objects in the representation is reduced, their perceptual relevance should increase.
3. The representation should present an intuitive description of the acoustical or music scene, based on the way a naive listener would do.
4. Although basic and intuitive, the representation should still allow trained listeners to

use it as a basis for higher-level music understanding.

5. The representation should be invertible so that an approximation of the original signals can be regenerated from it.
6. The representation should be practical enough so it can be useful for the easy development of various music analysis applications.

At the light of these guidelines, several mid-level representations have been proposed for CASA and music listening systems. The following list summarizes some of the types of representation traditionally used for the task of sound analysis and segregation.

Auditory mid-level representations. Auditory mid-level representations attempt to model the signal processing of the human auditory system. The assumption is that by mimicking the human auditory system it would be possible to achieve the sound separation ability of humans. However, the exact mechanisms of higher-level signal processing which take place in the brain are not yet known. Nevertheless, the lower-level processing already accounts for some of the separation abilities in human hearing. Given so, the signal processing of the peripheral auditory system can be modeled as a filter bank, followed by a cascade of half-wave rectification and low-pass filtering of the subband signals which accounts roughly to the mechanical-neural transduction of hair cells. Later stages often include a periodicity analysis mechanism such as autocorrelation, but the exact mechanisms of this stage are not known. This processing leads to a three-dimensional *correlogram*, which represents the signal intensity as a function of time, frequency (i.e. filter bank channel), and autocorrelation lag [Wang and Brown, 2006, pp.19]. Many algorithms sum the channel-wise autocorrelations to result in a *summary autocorrelation* function [Ellis, 1996, Scheirer, 2000], which has been used as a good basis for pitch estimation [Paiva, 2006].

Time-frequency representations. Numerous methods use a time-frequency representation, where the input signal is divided into short analysis windows, windowed, and a frequency transform of each frame is computed (typically the discrete Fourier transform, DFT). The frequency transform of a single frame is denoted by spectrum, and the magnitude of its each coefficient shows the energy at a particular frequency. The term spectrogram is used to denote the whole time-frequency representation, where the temporal locations of the frames determine the time axis, usually obtained by means of a short-time Fourier transform (STFT – see panel (b) of Figure 4). This representation reasonably follows the known ability of the human auditory system to perform frequency analysis, and many perceptually important characteristics of a sound are determined by looking at its spectrum over time. For instance, the

smoothed spectral energy distribution allows a good estimation of the formant structure of a sound, which is unique for each instrument, and therefore an important cue for its identification. On the other hand, the fine spectral structure reveals the vibration modes of the sound, which are often in harmonic relationships, resulting in perception of pitch, the basis for tonal music. However, no dimensionality reduction is attained with time-frequency representations such as the ones resulting from a STFT analysis.

Adaptive bases. Instead of using fixed and predefined basis (as is the case of the DFT), basis functions can be estimated directly from the data. Ideally, these basis functions capture redundant characteristics of the data and therefore reduce the number of required dimensions. Algorithms that have been used to estimate the basis functions include, for example, Independent Component Analysis (ICA) [Abdallah, 2002], Non-negative Matrix Factorization (NMF) [Vembu and Baumann, 2005] and Periodicity Transforms [Sethares and Staley, 1999].

Sparse representations. Time-frequency representations of ecological sounds typically present sparse distributions. This means that most of the coefficients are approximately zero, making it unlikely that two or more independent sources have a large coefficient in the same time-frequency point (see for example the representations in panels (c) and (d) in Figure 4, where only the peaks or corresponding trajectories in the spectrogram have non-zero values). As a result, sparseness may provide a good basis for sound separation. Estimating the most dominant source at each time-frequency point and then assigning all the energy at that point to the corresponding source often produces acceptable results. This is equivalent to applying a binary or soft mask for each of the detected sources to the mixture spectrogram [Wang and Brown, 2006, pp.22]. However, and as argued in Section 2.4.1, this source independence may not hold true for the case of music signals, where more instruments are more likely to have non-zero coefficients in the same time-frequency point.

Parametric models. Sparseness can also be used to develop parametric models for signals with a time-varying spectra. The active frequency components (i.e. peaks) in the time-frequency domain can be tracked into sinusoidal trajectories (see panel (d) in Figure 4), which following a sinusoidal model are then parameterized by time-varying frequencies, amplitudes and phases [McAulay and Quatieri, 1986, Ferreira, 2001, Ferreira and Sinha, 2005, Lagrange and Marchand, 2007, Lagrange et al., 2007a] (Section 3.4 will summarize sinusoidal modeling of music signals in the scope of the framework proposed in

this thesis). Serra was one of the first to make use of such a sinusoidal model for music signals in his *spectral modeling synthesis* (SMS) representation [Serra, 1989, Serra and Smith, 1990]. He additionally removed the spectral peaks (termed the *deterministic* part) from the original signal by means of spectral subtraction, subsequently modeling the remaining noise component (known as *stochastic* part) as white noise through a time-varying filter. Other parametric models for music signals include, for example, transient models [Levine, 1998, Verma and Meng, 1998, Every, 2006], instrument models (e.g. [Smith, 1992]) and abstract models (e.g. FM synthesis, which attempts to provide musically useful parameters in an abstract formula [Chowning, 1973]).

2.4.4 Computational Approaches to Scene Organization

Even if taking a suitable mid-level representation (see Figure 3), there are still many possible partitions of a sound mixture into its individual sound elements or events, depending on several conditions, as discussed in Section 2.3.2. As a result, when implementing a computer system for the segregation of complex mixtures of sound, it becomes pertinent to raise the question about how to achieve the right partition. As argued in [Shi and Malik, 2000] when considering a similar problem for image segmentation, there are at least two main aspects to take into consideration.

The first is that there may not be a single correct answer. If taking a Bayesian view, it is admissible to accept several possible interpretations in the context of prior world knowledge. This means that, as discussed in Section 2.3.2, the perception of a sound mixture strongly depends on the listener himself, his prior knowledge about the sounds and the encircling environment (i.e. the context), among other factors. As a result, the difficulty starts on the specification of the knowledge about the surrounding world. Some of it may be low-level (e.g. spatial proximity, spectral similarity, harmonicity) but it may also include important mid- or high-level prior knowledge about properties or models of already known sounds.

If taking a *bottom-up* approach (also known as a *data-driven* model), processing is assumed to start with small units of information, gathered at lower levels of the perceptual hierarchy. These small units are then progressively grouped at the different levels of the hierarchy forming more complex and larger units of information. At the top of the hierarchy it is expected to see the final units that assign meaning to the overall scene being perceived. However, the success of such a data-driven model is based on the assumption that there is enough information at the stimuli level for a comprehensive scene to be

constructed. Consequently, higher levels of comprehension cannot be accessed if lower levels are still unresolved. This is the process that is basically involved in the primitive grouping mechanism of the auditory system but, as discussed in Section 2.3.1, there are several phenomena that cannot be fully explained by this model alone. As argued by some authors [Ellis, 1996, Slaney, 1998], this suggests that more "global" information should in some way come to influence the formation of the low-level representations.

In contrast, a *top-down* approach (also known as *prediction-driven* model) proposes that comprehension occurs in a non-linear manner, starting at the top levels of the hierarchy and selectively using the lower levels to maximise the acquisition of "meaningful" stimuli. This model supports the problem-solving approach that accumulates past experience knowledge as a way to interpret what is being currently looked at on the sensory level, further allowing to build up expectations about future stimuli. This provides an explanation for the comprehension with very distorted inputs, and is on the base of the schema-based grouping mechanism in auditory perception (see Section 2.3.1).

However, pure top-down systems cannot substitute analysis based on low-level information, and therefore a satisfactory solution to the problem of automatic audition lies in the integration of both bottom-up and top-down processing approaches.

The second main aspect argued in [Shi and Malik, 2000] regards the hierarchical nature of the segregation processes in human perception. In fact, if following an attentional model of perception [Treisman, 1964, Deutsch and Deutsch, 1993, Wrigley and Brown, 2004, Kayser et al., 2005, Yost et al., 2007], it may be more appropriate to try to achieve a tree structure representation of a sound mixture, which corresponds to a hierarchical partition instead of a rigid and flat segregation of the sound scene. As a result, this suggests that a CASA system based on low-level grouping cues cannot and should not aim to produce a complete and final "correct" segregation. It should instead try to use low-level grouping cues to sequentially come up with hierarchical partitions. Mid and high-level knowledge can then be used to consolidate the resulting partitions as relevant sound "objects", or select them for further attention. This attention could result in further partitioning or grouping. The final conclusion is that sound segregation is probably better modeled when performed in a divisive manner[2] [Duda et al., 2000, pp.552], starting from the "big picture" (i.e. the mid-level representation) downward, rather like "*a painter first marking*

[2]A divisive (i.e. top-down or splitting) clustering procedure starts with all of the elements in one cluster and hierarchically and successively splits it into smaller clusters until some criteria is met. Inversely, an agglomerative approach (i.e. bottom-up or clumping) starts with all elements as singleton clusters and forms new clusters by successively merging them until some criteria is met. See [Duda et al., 2000, pp.552] for more details.

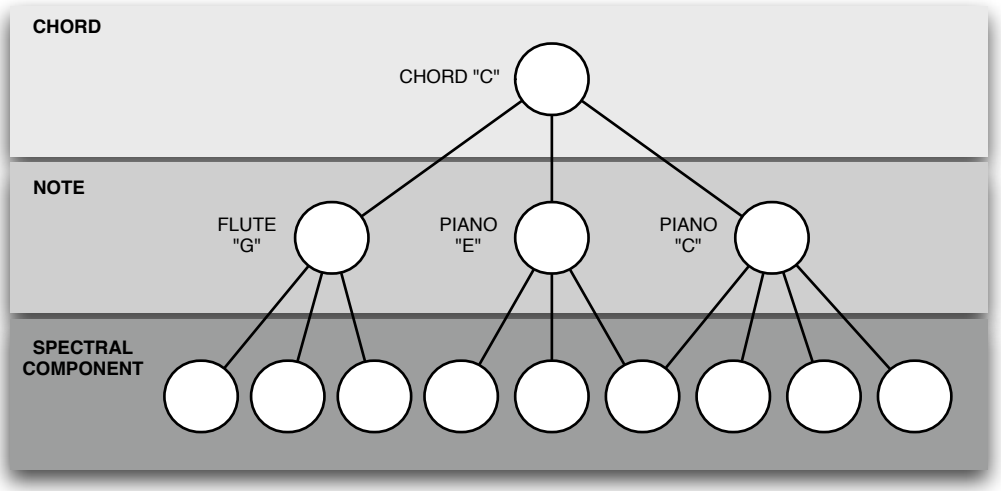


Figure 5: *Tree representation of a simple music scene (i.e. a C major chord whose notes are played by two different instruments – a flute and a piano). At each level of the tree a different perception of the same acoustic mixture is represented (the white circles depict the "perceptual" sound elements). Listeners usually start by hearing the sound as a whole chord and, depending on prior knowledge, attention and listening context, may be able to also perceive the individual notes or even the frequency components in the chord. For the sake of simplicity, the time structure of the spectral components is omitted. Figure adapted from [Kashino, 2006].*

out the major areas and then filling in the details" [Shi and Malik, 2000].

Such a hierarchical divisive approach allows to produce a tree representation of the acoustic scene. Figure 5 shows an example snapshot of the different perceived sound "objects" in a music performance (which for illustration purposes consists of a single chord). At each level of the tree it is possible to observe a different hierarchical partition of the initial complex sound mixture, which corresponds to the different perspectives taken by a same observer at different levels of attention or in different listening contexts (or by different observers perceiving a same scene, each one with different prior knowledges, listening contexts or levels of awareness). Taking the example given in the figure, when listening to music it is possible to hear multiple levels of "perceptual" sounds: sometimes chords are heard, but if the listener is attentive, has knowledge about the sound of the music instruments playing or is musically trained, he may be able to hear the independent chord notes or even the individual frequency components.

Accordingly, the framework proposed in this thesis will take the view that perception primarily depends on the use of low-level sensory information while being flexible enough to allow the use of prior knowledge to award meaning to the observations and to effectively modify the subject's focus of attention. Chapter 3 will describe and discuss in detail the

work conducted towards that goal.

2.4.5 CASA Evaluation Approaches

One of the challenges faced when developing CASA algorithms is the need to measure the goodness of the results. CASA is often known as the field that studies the problems related to the "sound source separation" in acoustic mixtures, and sound separation quality must be somehow measured in ways that correlate as much as possible to the perceptual abilities of the human hearing.

However, such an evaluation task may turn out to be an ill-posed problem. As noted in [Wang and Brown, 2006, pp.12] and discussed in Section 2.3.1, perception is a process private to the perceiver, and different observers may acquire different perceptions of the same surrounding environment. Additionally, what perception obtains at each time is *a* description of the environment and provides no guarantees about the faithfulness to the factual reality (although one could argue if the perceived description is what in fact defines reality). This creates some harsh difficulties when trying to define perceptually relevant criteria for the evaluation of the performance of CASA systems when compared to human performance. Thus, a perceptually motivated evaluation approach would be to have humans listening to the segregated signals. However, reliable listening tests require a large number of listeners, and are therefore complex, time consuming and expensive to conduct [Vincent et al., 2006b].

As a result, formal listening tests are usually replaced in the quality evaluation of sound source separation algorithms by objective computational measures [Vincent et al., 2006a], which allow an objective score to be calculated repeatedly in order to compare and measure the evolution of a system. Typically such a quantitative evaluation of the separation quality requires that reference signals (i.e. the original signals before mixing) are available, where the sound separation quality is measured by objectively comparing the separated signals with the corresponding reference sources. One important drawback of this approach is that there is no guarantee that these reference "physical sounds" actually reflect the perceptual acoustical image in the listener (i.e. the "perceptual sounds"– recall the discussion in Section 2.3.2). In fact, and depending on the evaluation measures used for the task, this approach may end up not evaluating the ability of the system to segregate sounds the way humans do. What is assessed is the capacity of the system to approximate the original sources present in the mixture signal in a mean-square sense, which is however not a guarantee of the perceptual fidelity of the segregation process [Ellis and Rosenthal, 1995].

Furthermore, and for the case of music content, this approach may pose some additional

practical difficulties. Although the production process of commercially available music content includes the recording of music instruments as individual audio tracks that could be used as references, such material is usually not delivered to the general public (although sites such as CCMixter[3] are starting to make available some of the individual recording tracks of songs published by independent artists under the Creative Commons[4] license). On top of that, the mastering stage in music production often includes nonlinear effects on the mixture signal, turning the reference audio tracks substantially different from the signals within the final stereo or monaural mixture.

Several efforts have been recently conducted for creating a common evaluation framework and large annotated datasets where researchers can test and compare the performance of their algorithms and systems for audio and music analysis and processing. The ISMIR 2004 Audio Description Contest [Cano et al., 2006] and the subsequent MIREX initiatives [5] are examples of such efforts, although an evaluation task specifically dedicated to sound segregation in music signals is yet to be proposed and conducted. More recently, the first community-based Signal Separation Evaluation Campaign (SiSEC 2008)[6] has been proposed, and it includes datasets and evaluation tasks focused on the separation of music mixtures.

In the specific case of algorithms that aim at separating individual tones of individual instruments, reference material where the tones are presented in isolation is difficult to record, and the use of synthesized material is often a less than optimal solution. Databases of audio recordings of isolated notes from various musical instruments have recently been made available (e.g. the RWC Music Database [Goto et al., 2003] or the University of Iowa Musical Instrument Sample Database[7]), allowing to artificially mix real instrument samples into polyphonic mixtures without too much loss of generality and representativeness. This allows to use the original isolated note samples as reference for the separation evaluation.

In all cases where several reference and separated signals are involved, an additional difficulty comes from the fact that it is usually not easy to know which separated signal corresponds to which reference source. There has to be some method to correctly associate the separated signals to their references so they can be compared, otherwise evaluation results will be penalized and in the limit become meaningless. Such an association can be

[3] http://ccmixter.org/
[4] http://creativecommons.org/
[5] http://www.music-ir.org/mirex/2008/index.php/Main_Page
[6] http://sisec.wiki.irisa.fr/tiki-index.php
[7] http://theremin.music.uiowa.edu/MIS.html

done with varying degrees of error using, for example, the similarity between separated signals and the references or, for the case of pitched musical signals, using pitch as a way to match signals corresponding to similar musical notes. In applications where a single source is to be segregated (e.g. the predominant melodic line, the singing voice, the bass line), this problem is non existent and usually such an approach is used as a starting strategy when developing a sound segregation system.

In order to cope with these constraints, several objective quality measures have been proposed for the evaluation of sound separation systems. The following list summarizes some commonly used evaluation metrics, here divided into three categories.

Low-Level Measures. Low-level measures are simple statistics of the separated and reference signals. One of the most common measures is the signal-to-distortion ratio (SDR), which summarizes the separation quality as the ratio of the energies of the reference signal and the error between the separated and reference signal. Appendix B presents a detailed description of this and some other related metrics used in the scope of the evaluations presented in this thesis. Additional low-level performance measures for audio source separation tasks have been discussed in [Vincent et al., 2006a], where for instance the interference from other sources was measured by the correlation of the separated signal to the other references.

Perceptual Measures. Perceptual measures in general estimate the audibility of the separation errors. Typically these approaches process the reference and separated signal using an auditory model, and the difference between signals is calculated in the auditory domain [Beerends, 1998, Beerends and Stemerdink, 1992, Beerends et al., 2002]. However, most of these perceptual measures have been developed for the quality evaluation of coded speech and audio. As a result they are usually optimized for signals where the differences between the reference and target are caused by quantization. Given that typical errors in sound source separation are deletions (where a segment of a signal is an all-zero signal) and insertions (where separated signal contains interference from another sources), these measures may produce misleading results when applied to separated signals. More recently, some novel evaluation measures that do not require reference signals and that are perceptually motivated have been proposed. Such an example is the *structural similarity index*, presented in the field of image processing [Channappayya et al., 2006]. This perceptual distortion metric, has been used to derive new linear estimators for image denoising applications. The results show that such structural similarity index optimized estimators tend to outperform the traditional linear least squared error estimators in terms

of the visual quality of the denoised images. Thus, this may as well be a promising alternative worth exploring in the future for the evaluation of the performance of sound segregation systems.

Application-specific Measures. In application-specific measures the separation accuracy is judged by the performance of the final application. In many of such applications, the separation is just one of the many algorithmic blocks in the complete system, and the separation is used as an intermediate step instead of being the final output. For example, when the separation is used as a preprocessing stage in a speech recognition system, its performance can be indirectly measured by the word-error rate. Similarly, in automatic music transcription systems, the performance can be measured by the transcription accuracy. Unlike the low-level and perceptual measures, application-specific measures do not necessarily require reference signals.

Chapter 4 will present some experiments where the segregation performance of the system proposed in this thesis is both directly assessed using low-level measures and indirectly assessed by the improvements achieved in specific tasks, such as main melody detection, voicing detection or timbre identification.

2.4.6 Previous Work on CASA

First attempts in separating sources from sound mixtures started in the area of speech enhancement (for a review of early proposals see [Lim, 1983]). However, the main objective of these approaches was to simply enhance a target voice among other concurrent voices or sounds and not to separate all the sounding sources in the mixture as independent audio signals.

Parsons was the first to propose a system that separated the voices of two talkers from a monaural mixture signal [Parsons, 1976]. His system was mainly targeted to the problem of cross-talk on communication systems, and was mostly based on signal processing techniques rather than on auditory modeling. He proposed a frequency domain approach for the segregation of concurrent voices in a single-channel signal where the separation is performed by selecting the harmonics belonging to each voice. Although this technique narrows the scope of application to mainly voiced signals (i.e. signals where a strong harmonic content is present), and consequently not being very well suited to the separation of speech signals (which comprise voiced and unvoiced parts), Parsons' work suggested some influential directions of work that proved important for research on multipitch tracking, F0-based sound segregation and the use of temporal continuity constraints.

One of the first approaches to voice segregation inspired on principles of auditory processing was proposed by Weintraub in [Weintraub, 1985]. Taking a harder challenge than Parsons, Weintraub assumes that the two speech signals can be voiced, unvoiced or silence at any given time. He uses a Markov model to represent the number of concurrent voices present and their characteristics, enabling the creation of a common framework where sequential and simultaneous grouping can be expressed. Similarly to Parsons, Weintraub also uses F0 as the basis of his voiced speech separation technique, but instead of a frequency domain front end, he employs a joint time-frequency approach – the correlogram (see [Wang and Brown, 2006, pp.19] for a revision of this technique). Weintraub also develops an evaluation framework for his system, where a waveform for each separated voice is resynthesized by refiltering the input signal, weighted by spectral estimates from the Markov model. The resulting resynthesized signal is then used as the input to an Automatic Speech Recognition (ASR) system and its performance is compared to the results obtained if using the two-voice mixture signal. Some limitations apply to Weintraub's work, as for instance, requiring that the two voices present a distinct average pitch range. This allows avoiding the complex problem of sequential grouping, since voice segments are possible to be grouped taking in consideration solely their pitch register. Weintraub's evaluations also showed that in some conditions, the ASR performance when taking the separated voice signals deteriorate, making the results somewhat inconclusive.

Shortly after Bregman's book publication [Bregman, 1990], Cooke presented his PhD thesis strongly influenced by Bregman's ASA theories [Cooke, 1991]. In fact, Cooke was one of the first to note the relation between the principles proposed by Bregman and the work developed by Marr in the field of vision [Marr, 1982]. He proposed a time-frequency representation he termed "synchrony strands", derived by applying local similarity and continuity constraints to the output of a cochlear model. For each strand, he computes the frequency, amplitude and modulation rate, building a rich representation of the input signal. From this representation he was able to separate two mixed voice signals, even with crossing F0s (a difficult scenario for previous systems). The system groups the harmonics of each source in the time-frequency plane, avoiding the need of prior knowledge of the fundamental frequency. Furthermore, Cooke's system does not require the prior specification of the number of sources in a mixture since the grouping process continues until all the synchrony strands are accounted for. The number of sources is then a by-product of the grouping process. Cooke's system does not address the problem of assigning a sequence of voiced and unvoiced speech sounds to a same source. This results from the inability of the system to sequentially group acoustic events from a same source that

are separated by silence intervals. Consequently, Cooke evaluates his system using a corpus of acoustic mixtures[8] where the target sound is always voiced. He then compares the synchrony strand representation of two signals before mixing them to the analogous representation after separated by his system.

In the same year as Cooke, Mellinger publishes his studies on the event formation and separation in musical sound [Mellinger, 1991], citing Bregman and Marr as theoretical influences. Mellinger's system takes the output of a cochlear model and processes it directly to form "feature maps", which encode acoustic onsets and frequency modulation. Harmonic components are grouped by calculating an affinity value between them. This affinity starts by taking into account the onset synchrony of the harmonic components and subsequently the coherence of their frequency modulation. An interesting aspect of Mellinger's system is its online approach where signal representations are computed and grouped as time progresses. This approach is distinct from previous works (e.g. Cooke first calculates the synchrony strand representation for the whole input and then, in a second pass, tries to group them) and seems to better follow the way the human ear works. Some of the results from Mellinger's system show that some difficulties arise under some "musically significant situations" [Mellinger, 1991, pp.183]: although harmonic relations play a significant role in musical sound, his system fails at grouping harmonic components unless they share a common onset and exhibit the same frequency modulation. Furthermore, and because in Mellinger's system the tracking of harmonics is triggered by the onset detection, failure to detect an acoustic onset (which is still today far from being a solved problem – see e.g. [Dixon, 2006]) results in missing an entire acoustic event.

Avery Wang's PhD thesis summarized several contributions to the areas of signal processing and auditory source separation [Wang, 1994]. He introduced *Frequency-Warped Signal Processing* as a means for separating the amplitude and frequency modulation contributions to the bandwidth of a complex-valued, frequency-varying sinusoid, transforming it into a signal with slowly varying parameters. This transformation was used to facilitate the removal of such sinusoids from an additive mixture while minimizing the amount of damage done to other signal components. To implement frequency tracking, a frequency-locked loop algorithm was proposed, where a complex winding error to update its frequency estimate was used. To improve tracking, Wang introduced a novel idea, termed *Harmonic-Locked Loop* tracking, which used several harmonically constrained trackers for tracking signals such as voices and certain musical instruments. The estimated fundamental frequency was computed from a maximum-likelihood weighting of the tracking estimates, as

[8] `http://www.dcs.shef.ac.uk/~martin/corpora/cookephd.tar.gz`

a way to improve robustness. His system was able to isolate harmonic signals, such as voices, from complex mixtures in the presence of other spectrally overlapping signals. Because his technique preserved phase information, he was able to remove the resynthesized harmonic signals from the original mixtures without too much damage to the residual signal.

Combining some ideas from previous systems, Brown proposes a representational approach to CASA [Brown, 1994, Brown and Cooke, 1994]. Similar to Mellinger's, his system makes use of maps of feature detectors based upon physiological grounds. These maps extract features such as onset, offset, periodicity, frequency transition information from the output of a cochlear model. Periodicity information is calculated from the correlogram (see [Wang and Brown, 2006, pp.15] for a description of this representation), being used for detecting resonances from harmonics and formants. These intermediate representations are used to construct discrete time-frequency segments. In order to combine resolved and unresolved harmonics of the same voice, a pitch contour is derived for each segment using local correlograms, a mechanism that works the same way for both cases. Brown's system is based on the correlogram, but grouping is performed in the cochleagram (see [Slaney and Lyon, 1993] for a description of the cochleagram) using pitch contours that are computed for individual segments. Consequently this has the advantage of not requiring any prior knownledge about the number of sources in the input signal. Brown evaluated his system performance using the same corpora of voiced speech used by Cooke, but using a different approach. The procedure is based on the principle of "exclusive allocation" (see [Bregman, 1990, pp.12; 595]) and consequently assumes that the energy at each point of the time-frequency space is assigned to a single source. The result of this grouping is a binary time-frequency mask, allowing to resynthesize a target source by refiltering the mask-weighted input. Although similar to the approach used by Weintraub, Brown also resynthesizes the original sources (before mixing) using the estimated mask, allowing to determine the signal-to-noise ratio in the corresponding separated target signal. This allows Brown to express the performance of his system using a well known and accepted signal-to-noise ratio (SNR) metric.

Some authors, being very critical of the "batch processing" approach of Cooke and Brown, proposed an online scheme based on a multi-agent paradigm [Nakatani et al., 1995]. Nakatani and his colleagues introduced a novel residue-driven architecture, where grouping processes are formulated as largely independent agents that interact in an attempt to explain the auditory scene. There are three types of agents: event detectors, trace generators and tracers. Event detectors subtract

the predicted input from the actual input, generating a residue, which is then evaluated by the trace generators. They determine if there is still sufficient evidence in the residue to trigger a new tracer for further analysis. Tracers are then responsible to group the features belonging to a same stream, and try to infer what the next input will be. This predicted input is then sent to the event detector, completing the processing loop. There are agents for tracing static background noise and harmonics, so that, for instance, a noisy speech signal can be analysed by a noise tracer together with a harmonic tracer. In case substantial harmonic residue is still found in the input, it may be an indication that another source appeared and a new harmonic tracer is created to track it. This system is evaluated by comparing the F0 contours output by harmonic tracers with ground-truth F0 contours, and by measuring the spectral distortion between the original and the separated sources.

In his PhD thesis, Ellis was inspired by the use of a feedback-loop in the residue-driven architecture and proposed a prediction-driven approach to CASA [Ellis, 1996]. He starts by extensively discussing previous "data-driven" approaches to CASA arguing that certain important phenomena are not easily modelled without top-down information. Phenomena such as "old-plus-new" or the interpretation of masked signals in the auditory continuity effect are presented as explicit examples. As a result, Ellis proposes a system based on an internal world model, where the auditory scene is represented using a hierarchy of sound elements. This world model is then constantly confronted with the acoustic input, following the generated hypothesis of the existing sound sources. A blackboard system was used to implement the prediction-driven CASA approach. In such a system there are independent knowledge sources that communicate via a shared pool of hypothesis (the "blackboard") providing a relatively unconstrained framework in which they can influence alternative assumptions about the input. Ellis' CASA system uses a broader signal model than previous proposals, including representations for periodic elements derived from the correlogram (termed "wefts"), aperiodic elements (called "noise clouds") and transients. Additionally, instead of following an "exclusive allocation" principle, his system allows the energy in a specific time-frequency region to be shared between sources. His evaluation framework is based on the comparison of the results of human listening tests on the same corpus used to test his CASA system.

Another system based on the blackboard architecture was later proposed by Godsmark and Brown [Godsmark and Brown, 1999, Godsmark, 2000], designed for the automatic transcription of polyphonic music signals. Similar to the approach followed by Ellis, their system used top-down predictions (concerning meter and familiar musical motifs) which

influenced the organization of the lower level of the blackboard. They also introduced the concept of an "organizational window" that slides through the input signal and inside which grouping principles interact with the objective of forming multiple hypothesis about the auditory scene, subsequently scored according to the acoustic evidence at the input. As this window slides over the input, its organization in what regards formed streams becomes fixed. This system also tries to extract features from groups of harmonics assumed to belong to single musical notes. Such features include timbre information, which is then used to achieve sequential grouping of the notes originating from a same musical instrument.

Scheirer, in his PhD thesis, presented a new computational model for the processing of complex sound scenes by the auditory system [Scheirer, 2000]. The model was based on a new principle of sound processing termed "dynamic detection of subband modulation" within an autocorrelogram representation (see e.g. [Summerfield et al., 1990] for a detailed description of the autocorrelation representation). Following an assumption about the processing of complex sound scenes by human listeners, where the formation of perceptual auditory images is governed by the discovery of groups of cochlear channels (see [Slaney and Lyon, 1993]) that exhibit common modulation behavior in the autocorrelogram domain, Scheirer proposed and implemented a computational model for sound and music listening. He showed that his model could be used to examine the predictions the theory makes about the perceived segmentation of various sound stimuli and that several well-known auditory grouping and segmentation phenomena could be qualitatively explained by this theory. He stated that this theory is different in some ways from other previous models for CASA, mainly because nearly all of the perceptual processing occurs within-band, there are no mid-level "components" to maintain an intermediate description of the auditory scene, and pitch is not used as a cue to perceptual grouping. He showed that the proposed model could be applied to the analysis of simple psychoacoustic stimuli as well as real musical sounds.

Srinivasan and Kankanhalli proposed a model for audio separation based on ASA in the STFT domain [Srinivasan and Kankanhalli, 2003]. They presented a frequency grouping algorithm based on principles of harmonicity and dynamics, where frequency components with a harmonic relation and similar dynamics are grouped as belonging to the same source. Srinivasan has later extended this work using peaks and onset times to propose a new representation scheme for "auditory blobs" [Srinivasan, 2004]. "Auditory blobs" are defined as the parts of an audio signal which have the same onset. The idea is based on using the common onset as a cue to group frequency components into the same

auditory object, using *spectral clustering* (see Section 3.6.1 and [von Luxburg, 2007] for more information about *spectral clustering*). He has shown how "auditory blobs" can be extracted, how to define harmonicity, dynamics, and onset features for each object, and how this representation can be applied to audio separation.

Bach and Jordan have also recently proposed the use of spectral clustering for an algorithm to perform blind one-microphone speech separation [Bach and Jordan, 2004, Bach and Jordan, 2006]. Their algorithm is able to separate mixtures of speech without modeling individual speakers, formulating the problem as a segmentation of the spectrogram into two or more disjoint sets. This segmentation is based on various harmonic and non-harmonic cues which are used for tuning the parameters of affinity matrices required for spectral clustering.

More recently, Mark Every has proposed in his PhD an empirical and heuristic approach to source separation from single-channel polyphonic real recordings [Every, 2006]. It has focused mainly on separating pitched notes without relying on source-specific information or prior knowledge of the recording, but MIDI information was used to provide note timing and pitch information. His work was conducted within a framework in which the music signal is considered to contain three main types of content: partials, transients and noise. He presented work towards extracting the transient attack of a note from a mixture, both when in isolation and when overlapping in time with other transient events, and a method for separating the spectral noise content of a mix of overlapping notes was also proposed. The approach relied upon the assumption that the harmonic and noise content of a note are correlated in both the temporal and spectral domains. A series of techniques for separating individual notes from a recording were used as the basis for the proposal of a system aiming at separating complete sources from a mixture, where an automatic note grouping system attempts to identify information that is generically useful for discriminating between different source types.

In his PhD work, Virtanen also focused on the separation of monaural music recordings, but where the sources or played notes to be separated are not known beforehand [Virtanen, 2006]. His proposed two separation approaches, both based on the common properties of real-world sound sources. The first separation approach uses unsupervised learning and combines non-negative matrix factorization with sparseness and temporal continuity objectives. The algorithm is based on minimizing the reconstruction error between the magnitude spectrogram of the observed signal and the model, while restricting the basis functions and their gains to non-negative values, and the gains to be sparse and continuous in time. A second proposal uses model-based inference based on

sinusoidal modeling. It minimizes the reconstruction error between the observed signal and the model, and several methods are proposed for approximating the amplitudes of overlapping overtones based on adjacent overtones.

Vincent recently proposed a family of probabilistic mixture generative models combining modified positive independent sub-space analysis, localization models, and segmental models to the problem of source separation of stereo musical mixtures using prior information about the sources (instrument names and localization) [Vincent, 2006]. He expresses source separation as a Bayesian estimation problem and proposes efficient resolution algorithms. The resulting separation methods rely on a variable number of cues including harmonicity, spectral envelope, azimuth, note duration, and monophony. He shows that these approaches outperform methods which only exploit spatial diversity and that they are robust against approximate localization of the sources.

While taking the more specific task of singing voice separation, Li and Wang proposed a CASA based system to separate singing voice from music accompaniment for monaural recordings [Li and Wang, 2006, Li and Wang, 2007]. The proposed system starts by partitioning and classifying the audio input into vocal and nonvocal segments. For vocal segments, a predominant pitch detection block estimates the pitch of the singing voice and finally a separation stage uses the detected pitch to group the time-frequency segments of the singing voice. Pitch is used as the only organizational cue, but the system is supposed to be able to incorporate other ASA cues, such as onset/offset and common frequency modulation, allowing it to be able to separate not only voiced singing but also unvoiced singing.

Approaches to CASA based on binaural or multi-channel processing have also been proposed, where the audio input is taken from two or more spatially separated microphones. Although this multi-channel information contributes to sound localization, auditory grouping and robustness to reverberation, the truth is that a human listener is still able to perform ASA even if using a single ear (or listening to monaural signals). Some references to works in this field are given in [Wang and Brown, 2006, pp.34] and [Virtanen, 2006, pp.6].

Although Bregman's studies say little about the neural mechanisms involved in ASA [Bregman, 1990], approaches based on neural models have also been proposed for CASA, and the interested reader may find references in [Wang and Brown, 2006, pp.35]).

2.5 Summary

This chapter presented a summarized overview of the main perceptual principles involved in human audition, with a special emphasis in music listening. In particular, the simultaneous and sequential grouping processes as well as the primitive and schema-based grouping principles proposed in the field of ASA have been shown to have an influential impact on the definition of computational approaches to ASA (i.e. CASA).

The typical build blocks that constitute a CASA system were introduced and a discussion about the mid-level representations and computational approaches to the problem of sound segregation, specifically in music signals, was presented.

Different methodologies can be used to evaluate a CASA system, and a summary of the most common approaches was also presented as an introduction to the experimental evaluation conducted during this work, as will be presented in Chapter 4.

The chapter concludes with an overview of some of the most important work previously conducted in the field of CASA and that has been, in different degrees, influential to the sound segregation framework proposed in the next chapter of this thesis.

Chapter 3

A Framework for Sound Segregation in Music Signals

"All models are wrong. Some are useful."

George Box

3.1 Introduction

A fundamental characteristic of the human hearing system is the ability to selectively focus on different sound elements and events in complex mixtures of sounds such as music. The goal of computational auditory scene analysis (CASA) [Rosenthal and Okuno, 1998, Wang and Brown, 2006] is to create computer systems that can take as input a mixture of sounds and form units of acoustic evidence such that each unit most likely has arisen from a single sound entity (see Chapter 2).

Humans use a variety of cues for perceptual grouping in hearing such as similarity, proximity, harmonicity and common fate (see Section 2.3.1). However, many of the computational issues of perceptual grouping for hearing are still unsolved. In particular, considering the several perceptual cues altogether is still an open issue [Roweis, 2000, Vincent, 2006], and the definition of what a sound entity should sound like when segregated is an ill-posed problem (see the discussion in Section 2.3.2).

Accordingly, this chapter will propose and discuss a flexible and extensible CASA framework for modeling the perceptual grouping problem in music listening. The goal of the proposed approach is to partition the acoustical mixture into a perceptually motivated topological description of the sound scene (similar to the way a naive listener would

perceive it) instead of attempting to accurately separate the mixture into its original constituent "sources". Thus, the term "sound source" will be used in the remaining of this thesis to mainly refer to "perceptual" sound events or "objects" in complex audio mixtures instead of limiting its meaning to the actual "physical" sources active at each time instant (see Section 2.3.2). The resulting topological representation leverages subsequent feature analysis, sound resynthesis and further transformations and manipulation of the segregated sound elements.

The framework proposed in this chapter will take the view that perception primarily depends on the use of low-level sensory information, and therefore requires no training or prior knowledge about the audio signals under analysis. It is however designed to be flexible enough to allow the use of prior knowledge and high-level information in the segregation procedure. Designed to be efficient, the resulting system can be used to segregate sound events in a complex polyphonic mixture of "real-world" music signals, paving the way to applications such as the predominant melody line detection, separation of the singing voice, instrument segregation and identification (as will be presented in Chapter 4).

3.2 System Overview

An overview of the main analysis and processing blocks that constitute the proposed sound segregation system is presented in Figure 6. A summarized description of the main blocks that constitute the proposed system will be presented in this section. More detailed discussions about each of the processing stages will appear in the subsequent sections of this chapter.

The system uses sinusoidal modeling as the underlying representation for the acoustic signals. Sinusoidal modeling is a technique for analysis and synthesis whereby sound is modeled as the summation of sine waves parameterized by time-varying amplitudes, frequencies and phases. In the classic McAulay and Quatieri method [McAulay and Quatieri, 1986], these time varying quantities are traditionally estimated by performing a short-time Fourier transform (STFT) and locating the peaks of the magnitude spectrum. Partial tracking algorithms track the sinusoidal parameters from frame to frame, and determine when new partials begin and existing ones terminate [Lagrange et al., 2007a].

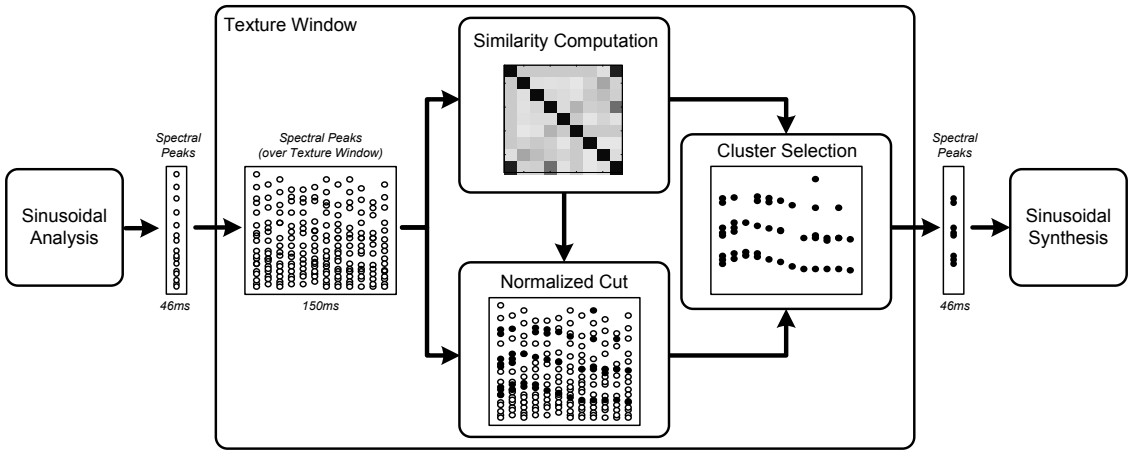


Figure 6: *Block-diagram of the proposed sound segregation framework.*

If the goal is to identify potential sound sources then a separate stage of partial grouping is needed. Typically grouping cues such as common onsets and spectral proximity are used. These two processes of connecting peaks over time to form partials (i.e. tracking) and grouping them to form potential sound sources (i.e. formation) roughly correspond to the sequential and simultaneous aspects of organization proposed by Bregman [Bregman, 1990] and discussed in Section 2.3.1. Although frequently implemented as separate stages (e.g. [Srinivasan, 2004]) these two organizational principles directly influence one another. For example, if one has knowledge that a set of peaks belongs to the same source, then their correspondence with the next frame is easier to find. Similarly, the formation of sound sources is easier if peaks can be tracked perfectly over time. However, methods that apply these two stages in a fixed order tend to be brittle as they are sensitive to errors and ambiguity.

To cope with this "circular cause and consequence" problem, the following sections try to show how both sound source tracking and formation can be jointly optimized within a unified framework using spectral clustering, and in particular the Normalized Cut criterion (Ncut) [Shi and Malik, 2000, von Luxburg, 2007].

Correspondingly, for each audio analysis frame at the input (set to have a length of about 46 ms) the system starts by computing the STFT (with a hop size of about 11 ms), where at each frame the local maxima of the magnitude spectrum are selected and stored as *peaks* characterized by their precise amplitude, frequency and phase (see Section 3.4). The current system selects the 20 highest amplitude peaks at each STFT frame, but ignores peaks whose frequencies are above 2500 Hz (empirical experiments have shown that components above 2500 Hz are often unstable, hurting the final segregation

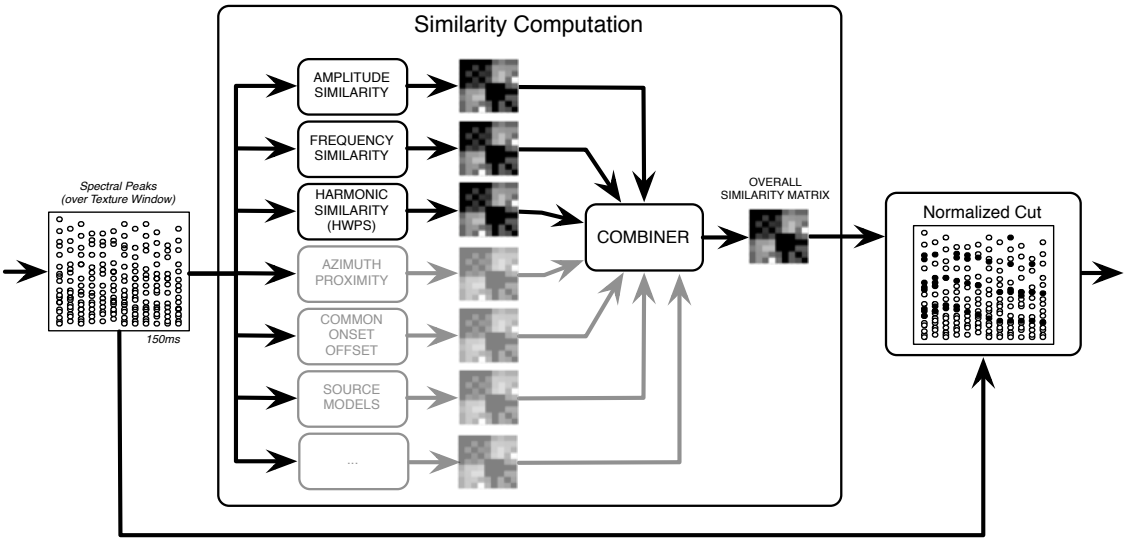


Figure 7: *Similarity cues currently implemented in the proposed sound segregation framework (black boxes). The system is flexible enough to accommodate new similarity cues (gray blocks) allowing to model an increasing number of perceptual mechanisms involved in human hearing. The individual similarity matrices from each grouping cue are combined into an overall similarity matrix that can be subsequently used by the clustering algorithm for sound event segregation (i.e. Normalized Cut).*

performance).

The challenge is then to express perceptual cues in terms of similarity between these time-frequency components in the signal. The peaks are therefore accumulated over a "texture" window (amounting to approximately 150 ms, although this value can be user or dynamically adjusted – see Section 3.5) and the problem is modeled as a weighted undirected graph [Balakrishnan, 1997, Shneiderman and Aris, 2006], where the nodes of the graph are the peaks of the magnitude spectrum and an undirected edge is formed between each pair of nodes (see Figure 10 and refer to Section 3.6.2). The edge weight is a function of the similarity between nodes and utilizes various grouping cues (recall Section 2.3.1) such as frequency and amplitude proximity, as well as a novel harmonicity criterion, termed *Harmonically Wrapped Peak Similarity* (HWPS) (see Section 3.7). The system is nevertheless flexible enough to accommodate new similarity cues allowing to model an increasing number of perceptual mechanisms involved in human hearing. Based on these grouping cues, and as depicted in Figure 7, the similarity across all the peaks is calculated and stored in similarity matrices, which are subsequently combined into an overall similarity matrix (see Section 3.7.3).

Peak clustering is then performed on this overall similarity matrix and each peak is assigned to a particular cluster, ideally grouping together components which are close in the

similarity space, and therefore have a higher chance of coming from a same sound source or event. Clustering is performed in the same way for all peaks within the "texture window" independently of whether they belong to the same frame or not, implementing some degree of time integration in the process. The peak clustering procedure, hereby formulated as a graph partitioning problem, can be efficiently solved using a spectral clustering approach, a technique used in a variety of applications including high performance computing, web mining, biological data and computer vision [Shi and Malik, 2000, von Luxburg, 2007]. Spectral clustering conveniently makes use of the computed similarity matrix to encode topological knowledge about a problem [von Luxburg, 2007] and to subsequently group together similar components, all in an unsupervised manner. In particular, the Ncut [Shi and Malik, 2000], a representative example of spectral clustering techniques, is employed in the system proposed in this thesis (see Section 3.6.1). The Ncut algorithm is computationally efficient and causal, eventually executing in real-time.

Once identified, the resulting clusters should ideally have a close perceptual correspondence to the sound events or "objects" as perceived by a human listener (at least a naive one – see the discussion about naive and trained listeners and schema and primitive grouping in Sections 2.3.1 and 2.3.2, respectively). As a result, the clusters that most likely represent the different sound events or "objects" in the mixture can be selected and resynthesized for further sound processing or evaluation (see Section 3.8).

3.3 Concepts and Terminology

For the sake of clarity, some of the parameters and terminology used when describing the audio segregation approach proposed in the remaining sections of this chapter are summarized next.

- **Frames or analysis windows** are sequences of audio samples used to estimate sinusoidal peaks from the complex spectrum computed using a Short Time Fourier Transform (STFT). For the experiments described in Chapter 4 a frame size corresponding to 46 ms and a hop size of 11 ms are used.
- **Peaks** are the output of the sinusoidal modeling stage. For each frame, a variable number of peaks corresponding to the local maxima of the spectrum are estimated. Each peak is characterized by its amplitude, frequency and phase (see Section 3.4).
- **Texture windows** correspond to an integer number of frames. Clustering of peaks across both frequency and time is performed for each texture window rather than per frame. For the experiments described in this thesis a texture window corresponding

to 10 frames ($\approx$ 150 ms) is used, but this value can be user or dynamically adjusted, as discussed in Section 3.5.2.

- **Similarity cues** express the similarity between spectral peaks belonging to the same texture window. These cues are inspired by perceptual grouping cues such as amplitude and frequency proximity, and harmonicity (see Section 3.7).
- **The similarity or affinity matrix** encodes the similarity of every peak to every peak within a texture window. Hence, the similarity matrix represents the similarity between peaks within the same frame (simultaneous integration) and across time (sequential integration) within the "texture window" (see Section 3.6.2).
- **Sound Events** are the "perceptual" sound events, "objects" or sources in complex audio mixtures, rather than the actual "physical" sources active at each time instant (see Section 2.3.2).
- **Clusters** are groups of peaks that are likely to originate from the same sound event or source. By optimizing the Normalized Cut criterion (Ncut), the overall peak similarity within a cluster is maximized and the similarity between clusters is minimized. The audio corresponding to any set of peaks (one or more clusters) can be conveniently resynthesized using a bank of sinusoidal oscillators (see Section 3.8).
- **Sound Segregation**, also known as sound separation or sound source formation in the scope of this thesis, is the process of approximately reconstructing a particular sound event from a decomposition of the polyphonic mixture (see Section 3.6).

3.4 Sinusoidal Modeling of Music Signals

Section 2.4.3 presented some guidelines proposed for the definition of a suitable music representation and a summary of the most common representations used in CASA. Accordingly, several CASA approaches consider auditory filterbanks and correlograms as their front-end [Wang and Brown, 2006]. In these approaches the number of time-frequency components is relatively small. However closely-spaced components within the same critical band[1] are hard to separate.

Other authors consider the Fourier spectrum as their front-end [Srinivasan and Kankanhalli, 2003, Bach and Jordan, 2004, Vincent, 2006]. In

[1] A critical band is said to be the frequency bandwidth or frequency separation where perceptual properties change suddenly [Hartmann, 1998, pp.256]. It may be measured using masking experiments, where a sine tone barely masked by a band of white noise around it is taken. When the noise band is narrowed until the point where the sine tone becomes audible, its width at that point is the critical bandwidth [Fletcher, 1940, Zwicker and Fastl, 1990].

these approaches, in order to obtain sufficient frequency resolution a large number of components is required. Components within the same frequency region can be pre–clustered together according to a heuristically defined stability criterion computed using statistics over the considered region. However, this approach has the drawback of introducing another clustering step, and opens the issue of choosing the right descriptors for those preliminary clusters. Furthermore, and as far as the sound segregation is concerned, considering the entire mixture will become untractable when dealing with audio streams, and as a result some complexity reduction methods should be considered.

Accordingly, in this work an alternative sinusoidal front-end is used. The sinusoidal model fulfills most of the desirable properties of a mid-level representation, as discussed in Section 2.4.3:

- it allows to provide a meaningful and precise representation of the auditory scene while considering only a limited number of components;
- it is invertible, allowing to reconstruct an approximation of the segregated sound components and it is efficient and intuitive to compute;
- it is particularly suited for sustained sounds with a well defined pitch and harmonic structure such as the vowels of a singing voice or the notes from pitched music instrument (e.g. trumpet, piano or violin);
- it allows to model consonants or non-pitched instruments (e.g. drums), although requiring a large number of sinusoidal components to accurately represented such signals [McAulay and Quatieri, 1986];
- it easily enables to use prior information about the harmonic spectral structure, making it suitable for the separation of pitched musical instruments and voiced speech and singing.

Because in traditional Western music pitched sounds tend to dominate and last for longer periods when compared to transient signals (usually originated at the attacks or onsets of notes and events, or resulting from percussion instruments such as drums), a sinusoidal model can, in practice, capture most of the pitched information that is useful for MIR applications. This includes the ability to detect musical events in a complex mixture, such as the melodic line or the timbral characteristics of the instruments playing or the main singing voice in the signal.

The following sections summarize the method used in this work for the precise estimation of the frequency and amplitude of the most prominent spectral peaks in an audio frame. The interested reader can find more detailed information about the presented technique in [Marchand and Lagrange, 2006, Lagrange and Marchand, 2007,

Lagrange et al., 2007a].

3.4.1 Short-Term Sinusoidal Analysis

Sinusoidal modeling represents a sound signal as a sum of sinusoids characterized by their main parameters: amplitudes, frequencies, and phases. It is rooted in Fourier's theorem, which states that any periodic function can be modeled as a sum of sinusoids at various amplitudes and harmonic frequencies. Since considering these parameters as constant through the whole signal duration is not perceptually relevant, a common approach is to segment the signal into successive frames of small duration. The length of each frame (N samples), and the hop size (H samples) between successive frames should be set so that the parameters can be considered constant within the frame (for a sampling frequency of 44100 Hz, N is set to 2048 samples and H is set to 512 samples, corresponding to a frame length of about 46 ms and a hop size of about 11 ms – these are typically used values in music analysis [Zölzer, 2002]). The discrete signal $x^k(n)$ at frame index k is then modeled as follows:

$$x^k(n) \;=\; \sum_{l=1}^{L^k} a_l^k \;\cos\left(\frac{2\pi}{F_s}\, f_l^k \cdot n + \phi_l^k\right) \tag{1}$$

where n is the sample index, F_s is the sampling frequency, ϕ_l^k is the phase at the beginning of the frame of the l-th component of L^k sinusoids, and f_l^k and a_l^k are respectively the frequency and the amplitude. Both are considered as constant within the frame.

3.4.2 Peak Parameters Estimation

For each frame k, a set of sinusoidal parameters $\mathcal{S}^k = \{p_1^k, \cdots, p_{L^k}^k\}$ is estimated. The system parameters of this Short-Term Sinusoidal (STS) model $\mathcal{S}^k$ are the L^k triplets $p_l^k = \{f_l^k, a_l^k, \phi_l^k\}$, often called *peaks*. These parameters can be efficiently estimated by picking some local maxima from a Short-Term Fourier Transform (STFT).

To estimate each set of peaks $\mathcal{S}^k$, the spectrum X^k is computed using a discrete Fourier transform (DFT) applied to the windowed samples of frame k. The samples are weighted by a Hann window with an even lenght N, and subsequently circularly shifted by $N/2$ samples prior to the DFT computation. This results in zero-phase windowing, a procedure that allows to obtain a robust estimation of the phase [Serra, 1989].

The precision of these estimates is further improved by using phase-based frequency estimators which utilize the relationship between phases of successive frames [Puckette and Brown, 1998, Marchand and Lagrange, 2006, Lagrange and Marchand, 2007].

Assuming that the frequencies of the pseudo-periodic components of the analysed signal are constant during the time-interval between two successive short-term spectra, with a hop size of H samples, the frequency can be estimated from the phase difference:

$$\hat{\omega} = \frac{1}{2\pi H}\Delta\phi. \tag{2}$$

The smaller the hop size the more accurate this assumption is. Two successive short-term spectra separated by one sample are considered. The frequency is estimated directly from the phase difference $\Delta\phi$ ensuring that the phase is unwrapped so that the difference is never negative. The resulting estimator, known as the difference estimator, is:

$$f_l^k = \frac{F_s}{2\pi}\left(\angle S[m_l, n_k + 1] - \angle S[m_l, n_k]\right)_{\text{unwrap}} \tag{3}$$

where $\angle S$ denotes the phase of the complex spectrum S, m_l is the frequency bin index of the peak p_l^k within the spectrum, and n_k is the index of the first sample of analysis frame k. The frame index k is omitted in the remainder of this section.

During the analysis of natural sounds, the presence of several frequency components in the same spectral bin or noise may lead to incoherent estimates. If the frequency f_l of a local maximum located at bin m_l is closer to the frequency of another bin, the local maximum should have been located at this bin. Therefore, a local maximum with an estimated frequency that does not satisfy the following condition is discarded: $|N/F_s \cdot f_l - m_l| \leq 0.5$.

Since the power spectrum of an ideal sinusoid signal has the shape of the power spectrum of the analysis window, centered around the sinusoid frequency, the improved frequency estimation can be used to estimate more precisely the amplitude:

$$a_l = 2\,\frac{|S[m_l]|}{|W_H(f_l - m_l\,F_s/N)|} \tag{4}$$

where $|S[m_l]|$ is the magnitude of the complex spectrum, $W_H(f)$ is the spectrum of the Hann window used for the STFT computation, f being the frequency in Hz.

These more precise estimates of frequency and amplitude parameters are important for calculating more accurate similarity relations between peaks. The importance of more

precise frequency estimation for the proposed method, when compared to the basic approach of directly converting the FFT frequency bin number to frequency, is explored in some of the experiments in Section 4.3.

Other frequency estimation methods can also be used, such as parabolic interpolation [Abe and Smith, 2004] or subspace methods [Badeau et al., 2006]. In particular, a technique based on the Odd-DFT filter bank that allows the accurate estimation of the frequency, phase and magnitude of stationary sinusoids proposed by [Ferreira, 2001] has been used with some promising results in a multi-pitch detection system presented in [Martins, 2002, Martins and Ferreira, 2002].

3.5 Time Segmentation of Music Signals

In order to introduce some degree of temporal integration and simultaneously optimize partial tracking and source formation, peaks should be grouped along time and frequency, into what is known in this work as a *texture window*. This allows to compute the proximity between peaks (as will be described in Section 3.7), both within a frame as well as across frames belonging to a same texture window, resulting in the construction of a similarity graph (see Section 3.6.2). Unlike approaches based on local information [McAulay and Quatieri, 1986], such a similarity graph enables the use of the global Normalized Cut criterion to partition the graph over an entire texture window (see Section 3.6.3). Thus, peaks are grouped over texture windows rather than per frame.

Two approaches for defining texture windows have been implemented: the first based on fixed length texture windows, and a second one based on the use of an onset detector to automatically define the texture window lengths.

3.5.1 Fixed Length Texture Windows

Texture windows can be obtained by simply accumulating a fixed number of frames over time (which in this work is set to correspond to a length of about 150 ms – see Figure 6). In such a case, if L^k is the number of peaks in frame k, then the number of peaks in a texture window is T:

$$T = \sum_{k=1}^{N} L^k \tag{5}$$

where N is the texture window length (in frames). As will be discussed in Section 3.6.2, this will later allow to compute the pairwise similarity between all peaks p_l^k and p_m^{k+n} in

the texture window, where l, m are the peak indices, $n \in \{0 \dots N-1\}$ is the frame offset between the peaks and $n = 0$ is used for peaks of the same frame.

Although simple and computationally efficient, this fixed length approach does not take into consideration the perceptual organization of sound as performed by the human ear. As discussed in Chapter 2, the human auditory system is highly sensitive to the time structure of a sound stream. In fact, onsets seem to be particularly important to the segregation of complex mixtures of sounds [Scheirer, 2000, Srinivasan, 2004], and the human ear tries to lock on such events as a way to perform a first low-level time segmentation of the acoustic stimuli. This allows to divide the sound into perceptually coherent time regions, where sources assume a clear presence and additional grouping principles become into play (e.g. harmonicity grouping). This fact is even more evident in the case of music signals, where in most cases a regular time structure exists, mainly resulting from the onset of the notes and events in the musical piece [Gouyon, 2005].

As a result, a more perceptually motivated way to adjust the length of the texture windows could be based on the use of an onset detector, an approach explored in this work and explained in the following section.

3.5.2 Dynamic Length Texture Windows

Fixed length texture windows blindly chop an audio signal into segments without taking into consideration the various sound events occurring over time. Additionally, by not following a perceptually inspired approach, such a method will often result in texture windows where only parts of the sound events in a signal are present – be it their attack regions (i.e. onsets), sustained sections (i.e. usually the steady-state part of the sound event) or the release sections (i.e. the offsets). Because sound event segregation will be performed at the texture window level, this may result in an attempt to segregate sound events poorly represented in the texture window and whose partial information may have very little perceptual relevance. Consequently, one may expect that a better segregation strategy could be achieved by dynamically adjusting the length of each texture window in order to enclose as much as possible "perceptually complete" sound events.

One way to achieve this goal is to use an onset detector in order to identify the occurrence of new sound events in an audio signal, and to adjust texture windows to the corresponding inter-onset audio segments. As an example, Figure 8 depicts the time domain and the corresponding spectrogram representations of a fragment of a jazz recording, below which are plotted the onsets automatically identified using the algorithm described in the following section. As it is clearly visible from this example, the audio segments

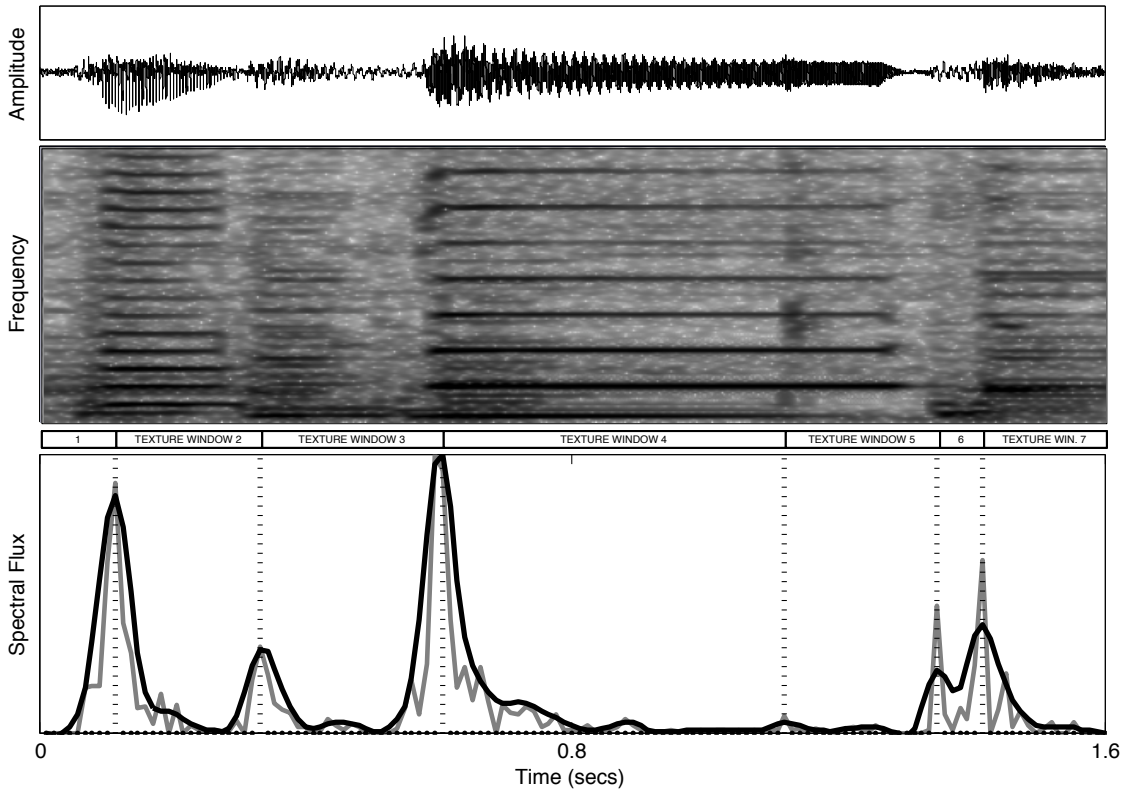


Figure 8: *Dynamic adjustment of the length of the texture windows based on the detection of onsets in the audio signal. The top panel depicts the time domain representation of a fragment of a polyphonic jazz recording, below which is displayed its corresponding spectrogram. The bottom panel plots both the onset detection function $SF(n)$ (gray line – see Equation 6), as well as its filtered version (black line – see Equation 7). The automatically identified onsets (see Section 3.5.3) are represented as vertical dotted lines.*

between consecutive onsets present a high correlation to the occurrence of sound events in the signal (visible in the spectrogram representation), be them pitched events such as musical notes, or unpitched and noisy events such as percussion hits. As a result, taking this onset information into account tends to result in more perceptually and musically meaningful texture windows.

The following section presents a summarized description of the implemented onset detector algorithm.

3.5.3 Onset Detection Algorithm

Onset detection aims at finding the starting time of musical events (e.g. notes, chords, drum events) in an audio signal. However, polyphonic music poses an increased challenge

since nominally simultaneous notes might be spread over tens of milliseconds, turning the definition of onsets ambiguous. Similarly, it is hard to define a precise onset time for sounds with slow attacks.

In a recent tutorial article, Simon Dixon revisited the problem of onset detection [Dixon, 2006], where a number of onset detection algorithms were reviewed and compared on two datasets. This study was itself based on a previous article by Bello et al. [Bello et al., 2005] where a theoretical and empirical comparison of several state-of-the-art onset detection approaches is presented. Dixon concluded that the use of a spectral flux detection function for onset detection resulted in the best performance versus complexity ratio.

Hence, and following the findings and results in [Dixon, 2006], the approach used in this work is based on the use of the spectral flux as the onset detection function, defined as:

$$SF(n) = \sum_{k=0}^{N/2} H(|X(n,k)| - |X(n-1,k)|) \tag{6}$$

where $H(x) = \frac{x+|x|}{2}$ is the half-wave rectifier function, $X(n,k)$ represents the k-th frequency bin of the n-th frame of the power magnitude (in dB) of the short time Fourier Transform, and N is the corresponding Hamming window size. For the experiments performed in this work a window size of 46 ms (i.e. $N = 2048$ at a sampling rate $f_s = 44100$ Hz) and a hop size of about 11ms (i.e. 512 samples at $f_s = 44100$ Hz) are used.

The bottom panel of Figure 8 plots the values over time of the onset detection function $SF(n)$ for an jazz excerpt example. The onsets are subsequently detected from the spectral flux values by a causal peak picking algorithm, where it attempts to find local maxima as follows. A peak at time $t = \frac{nH}{f_s}$ is selected as an onset if it satisfies the following conditions:

1. $SF(n) \geq SF(k) \quad \forall k : n - w \leq k \leq n + w$
2. $SF(n) > \frac{\sum_{k=n-mw}^{n+w} SF(k)}{mw+w+1} \times thres + \delta$

where $w = 6$ is the size of the window used to find a local maximum, $m = 4$ is a multiplier so that the mean is calculated over a larger range before the peak, $thres = 2.0$ is a threshold relative to the local mean that a peak must reach in order to be sufficiently prominent to be selected as an onset, and $\delta = 10^{-20}$ is a residual value to avoid false detections on silence regions of the signal. All these parameter values were derived from preliminary experiments using a collection of music signals with varying onset characteristics.

As a way to reduce the false detection rate, the onset detection function $SF(n)$ is smoothed using a Butterworth filter (see bottom panel of Figure 8), defined as:

$$H(z) = \frac{0.1173 + 0.2347z^{-1} + 0.1174z^{-2}}{1 - 0.8252z^{-1} + 0.2946z^{-2}}. \quad (7)$$

In order to avoid phase distortion (which would shift the detected onset time away from the $SF(n)$ peak) the input data is filtered in both the forward and reverse directions. The result has precisely zero-phase distortion, the magnitude is the square of the filter's magnitude response, and the filter order is double the order of the filter specified in Equation 7.

Additionally, a minimum and maximum texture window length was defined, so that onsets detected inside a 50 ms range from the previous one are discarded (avoiding too short texture windows) and a maximum inter-onset interval of 300 ms is used (avoiding too long texture windows, a limitation mainly due to computational constraints).

In contrast to applications related to automatic beat or rhythm analysis, the use of an onset detector for the purpose of dynamically adjusting the length of the texture windows does not necessarily need to be highly precise. Because the onsets end up being used as help guides for coarsely diving the audio stream into perceptually relevant texture windows, the occurrence of false positives and missed onsets is not expected to have a drastic negative impact on the final results of event segregation.

Considering that the described onset detection algorithm is closely based on proposals extensively evaluated in [Bello et al., 2005, Dixon, 2006], added to the difficulty to build and to find publicly available annotated datasets usable as onset ground truth, an objective and comprehensive evaluation of the performance of the described onset detector is not the central focus of this work. Nevertheless, a limited evaluation was performed in a few music audio signals, which were representative of distinct musical styles (e.g. classic piano music, choral pieces, pop/rock songs, jazz, among others)[2].

Section 4.3.3 will present an experimental evaluation and discussion on the use of the proposed onset detector for the dynamic adjustment of the length of the texture windows and compare its results to the use of the fixed length approach for the segregation of the main melody line from polyphonic music signals.

[2]Some audio examples of the results of the onset detector can be found at: `http://www.fe.up.pt/~lgustavo`

3.6 Grouping Sound Events

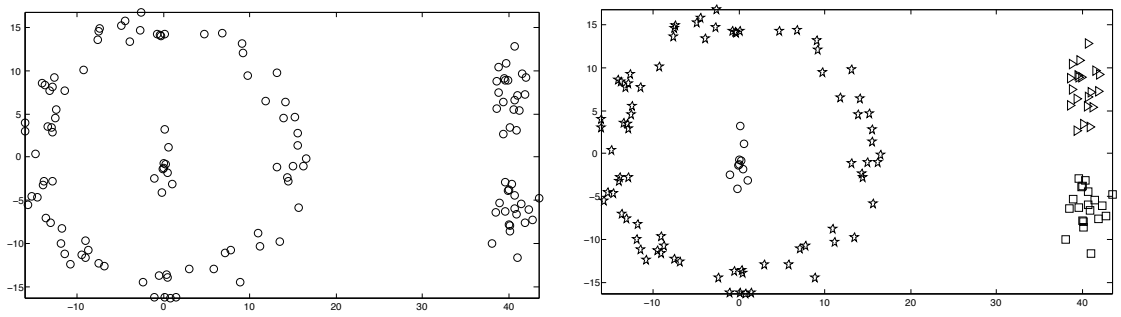
Following the sinusoidal analysis front-end and the time segmentation of a complex mixture signal into texture windows (as discussed in previous sections of this chapter), the challenge is now to find a way to cluster (i.e. to group together) the spectral peaks into entities that most likely represent sound events or "objects" in the mixture.

Clustering is the task of grouping items into different categories (i.e. clusters) according to some defined similarity measure. Clustering is a common technique for statistical data analysis and is used in many fields, including machine learning, data mining, pattern recognition, bioinformatics and image and sound analysis. Also known as unsupervised learning, where no prior information about the items is available apart from the features directly observable, it can be applied to generic data sets, such as the collection of spectral peaks extracted from a sound mixture (described in Section 3.4). Any feature from a peak, such as its frequency, amplitude or phase can potentially be used to represent the peak as a data point in a data set. However, some features may be more appropriate than others depending on the final objective of clustering (see Section 3.7). In all the cases, the partitioning of the dataset into clusters should ideally be done in a way so that data in each cluster shares some common characteristic (according to some suitable distance or proximity measure), and where each cluster is as distinct as possible from all the other clusters (similarly to Fisher's linear discriminant [Duda et al., 2000, pp.117]).

The problems of clustering and grouping is a well studied field, and the computational task of partitioning or grouping a data set into k clusters is often referred to as k-clustering. Two commonly used methods are k-means [Duda et al., 2000, pp.526] and Expectation-Maximization (EM) [Duda et al., 2000, pp.124], being the latter traditionally used for learning a mixture-model. EM and k-means in particular are based on estimating explicit models of the data, providing high quality results when the data is organized according to the assumed models. However, when data is arranged in more complex and unknown shapes (as may happen with sound data), these methods tend to poorly explain the data distribution. As a result, some other approach to clustering where the complex distribution of sound data is better modeled should be investigated. In this thesis the choice was to use a technique known as *spectral clustering*, discussed next.

3.6.1 Spectral Clustering

Spectral clustering, an alternative clustering approach that goes back to Donath and Hoffman [Donath and Hoffman, 1973], was shown to correctly handle complex and structured



(a) (b)

Figure 9: *An example dataset containing non-convex and non-Gaussian shaped distributions (a) and the resultant grouping of the data points after applying spectral clustering (b). Such a clustering result would be hard to attain if using techniques such as k-means or EM, which assume convex shaped or Gaussian distributed data sets.*

data. As a result, in the machine learning community, spectral clustering has been made popular by the works of Shi and Malik [Shi and Malik, 2000], Ng et al. [Ng et al., 2002] and Meila and Shi [Meila and Shi, 2001]. A recent and comprehensive tutorial on spectral clustering can be found in [von Luxburg, 2007].

Instead of estimating an explicit model of data distribution, spectral clustering rather performs a spectral analysis of the matrix of point-to-point similarities (i.e. the similarity or affinity matrix). The success of spectral clustering mainly results from the fact that it does not make strong assumptions on the shape of the clusters. Differently from k-means, where data is assumed to follow a convex shaped distribution, or opposed to EM, which assumes a Gaussian distribution of the data, spectral clustering can solve very general problems like the one exemplified in Figure 9.

Being based on the eigenstructure of the computed similarity matrix, spectral clustering simply needs to solve a linear problem. This avoids risks of getting stuck in local minima and consequently having to restart the algorithm for several times with different initializations (as happens with k-means). On the other hand, spectral clustering allows to extract a global impression of the data structure (e.g. the components of a sound mixture) rather than focusing on local features and their consistencies in the data.

Moreover, compared to point based clustering algorithms such as k-means, the use of a similarity matrix as the underlying representation enables expression of similarities that can not be computed as a distance function of independently calculated feature vectors. The *Harmonically Wrapped Peak Similarity* (HWPS) measure proposed in Section 3.7.2, is an example of such as a similarity measure.

Finally, spectral clustering can be computationally efficiently even for large data sets, as long as the similarity matrix is made sparse. However, and as will be discussed in the following sections, computing a suitable similarity matrix depends on the choice of a good similarity graph, an operation which is not trivial, and spectral clustering can become quite unstable under different choices of the parameters for the neighborhood graphs. Like in all machine learning approaches, the main issue is in the definition of the feature or similarity functions (i.e. the way data is abstracted) and not so much in the classifier or clustering algorithms. As a result, and as argued in [von Luxburg, 2007], spectral clustering should not be taken as a "black box algorithm" which is able to automatically detect the correct clusters in any arbitrary data set. It can, nevertheless, be used as a powerful tool which can produce interesting results if applied diligently.

Previous Use of Spectral Clustering in Sound Analysis

As far as it was possible to identify, there are few applications of spectral clustering to audio processing. It has been used for the unsupervised clustering of similar sounding segments of audio [Ellis and Lee, 2004, Cai et al., 2005]. In these approaches, each audio frame is characterized by a feature vector and a self-similarity matrix across frames is constructed and used for clustering. This approach has also been linked to the singular value decomposition (SVD) of feature matrices to form audio basis vectors [Dubnov and Appel, 2004]. These approaches characterize the overall audio mixture without using spectral clustering to form and track individual sound sources.

Spectral clustering has also been used for blind one-microphone speech separation [Bach and Jordan, 2004, Bach and Jordan, 2006]. Rather than building specific speech models, the authors show how the system can separate mixtures of two speech signals by learning the parameters of affinity matrices based on various harmonic and non-harmonic cues. The entire STFT magnitude spectrum is used as the underlying representation.

Closer to the approach proposed in this thesis, harmonicity relationships and common fate cues underlie a short-term spectra-based similarity measure presented by Srinivasan and Kankanhalli [Srinivasan and Kankanhalli, 2003]. To integrate time constraints, it is alternatively proposed in [Srinivasan, 2004] to cluster previously tracked partials to form auditory "blobs" according to onset cues. Spectral clustering (and in particular the Normalized Cut criterion – explained in a following section) is then carried out on these blobs.

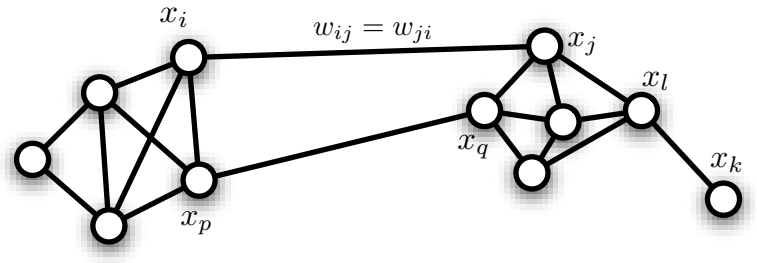


Figure 10: *Example of an undirected similarity graph (where the nodes of the graph can be the peaks of the magnitude spectrum of a sound signal), and an undirected edge is formed between each pair of nodes. The edge weight is a function of the similarity between nodes (which can be defined by means of various grouping cues, such as frequency, amplitude and harmonicity proximity, among others – see Section 3.7).*

In contrast, and as described in Section 3.4, a short-term sinusoidal modeling framework is used in the approach proposed in this chapter. It results in more accurate and robust similarity relations as well as significantly smaller affinity matrices that are computationally more tractable, allowing to consider texture windows with a high number of frames.

3.6.2 Constructing the Similarity Graph

Constructing a similarity graph appropriate for spectral clustering is not a trivial task and very little is still known about the theoretical implications of the various constructions. This results in several choices to be made and parameters to be set when attempting to get the best out of spectral clustering for a specific application domain. This section will present a discussion on the processes involved in constructing a similarity matrix that is adequate for the sound segregation task.

The use of texture windows as a way to perform time segmentation of the audio signals (see Section 3.5) will result in a dataset of T data points, corresponding to the total number of spectral peaks p_l^k over each frame k belonging to the texture window (see Equation 5). As a result, given a set of data points $x_1, \ldots, x_T$ (corresponding for e.g. to the frequency f_l^k or the amplitude a_l^k values of the peaks detected in a sound mixture – see Section 3.4) and some definition of similarity $w_{ij} \geq 0$ between all pairs x_i and x_j of data points (as will be discussed in Section 3.7), the objective of clustering is to divide the data points into k different groups (i.e. sound events or "sources" – see Section 3.6.4 for a discussion

on how this number k can be defined) so that points in the same group are similar and points in different groups are dissimilar to each other.

An elegant way of modeling the data set is to represent it as a weighted similarity graph $G = (V, E)$, where each vertex v_i in $V = \{v_1, \ldots, v_T\}$ corresponds to a data point x_i, and where two vertices i and j are connected by a weighted edge in E if the similarity w_{ij} between the corresponding data points x_i and x_j is larger than 0 (or some other defined threshold). Figure 10 depicts an example graphical representation of a similarity graph.

There are several common ways to transform a given set $x_1, \ldots, x_T$ of data points with pairwise similarities w_{ij} into a graph. When constructing these similarity graphs the goal is to model the local neighborhood relationships between the data points. At least three approaches are commonly used with spectral clustering: *ϵ-neighborhood graph*, *k-nearest neighbor graph* and the *fully connected graph*. The choice of the type of graph has a direct influence on the performance of spectral clustering and a comprehensive discussion about their properties and differences can be found in [von Luxburg, 2007].

For the work proposed in this thesis the choice came down to using a *fully connected graph*. In this approach all points in the data set end up connected: points in local neighborhoods are connected with relatively high weights, while edges between distant points have very small (but still positive and therefore non-zero) weights. This avoids the specification of a threshold for the minimum weight necessary to connect two data points. As a drawback, *fully connected graphs* result into non-sparse similarity matrices, preventing the use of more efficient computation techniques that take advantage of data sparsity.

Defining a Generic Similarity Function

Another important aspect is the definition of the similarity function to use. Since the similarity graph should represent the local neighborhood relationships, this construction is only useful if the similarity function itself models local neighborhoods. In other words, it is important to guarantee that the local neighborhoods induced by the chosen similarity function are "meaningful" – i.e. points considered as "very similar" by the similarity function should also be closely related in the specific application the data comes from. On the other hand, the global "long-range" behavior of the similarity function is not so important for spectral clustering since it does not really matter whether two data points have similarity score 0.01 or 0.001 (as just an example) – both values represent data points with negligible similarity anyway. Ultimately, the choice of the similarity function depends on the domain the data comes from, and no general advice seems to

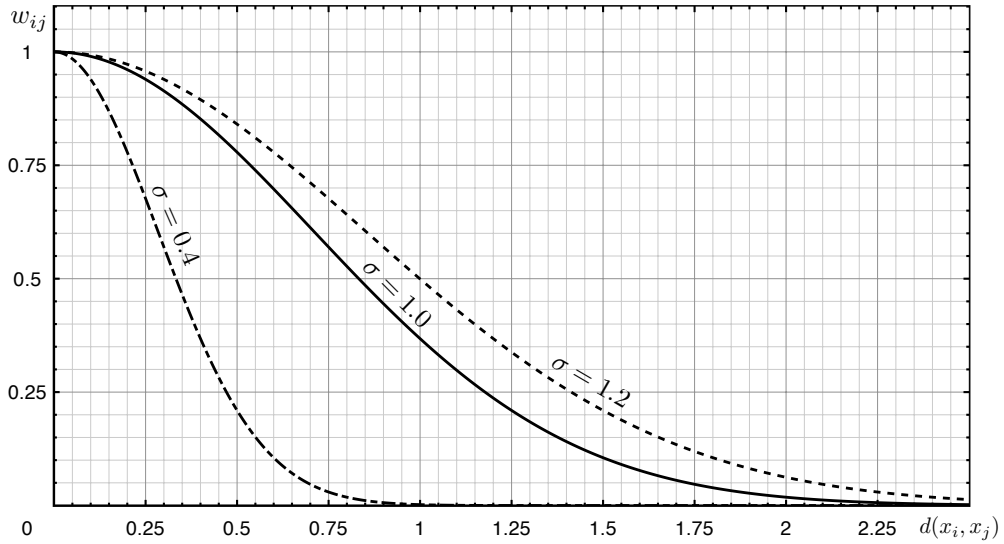


Figure 11: *Gaussian similarity function as defined in Equation 8 for three values of σ.*

be available [von Luxburg, 2007]. As a result, Section 3.7 will propose some similarity functions inspired in auditory grouping principles (as discussed in Section 2.3.1), which, as will be demonstrated, present promising results in capturing the perceptual proximity between spectral components (i.e. peaks) in complex sound mixtures. This makes the use of spectral clustering techniques feasible for the segregation of sound events.

Accordingly, the *fully connected graph* is usually used in combination with the Gaussian similarity function, defined as

$$w_{ij} = e^{-\left(\frac{d(x_i,x_j)}{\sigma}\right)^2} \tag{8}$$

where the parameter σ controls the width of the neighborhoods and $d(x_i, x_j) = \|x_i - x_j\|$ is some distance measure (e.g. the Euclidean distance). Figure 11 shows plots of the Gaussian similarity function for three values of connectivity parameter σ. Unfortunately, barely any theoretical results are known to provide guidance in the specification of σ.

As a result, some heuristics and rules of thumb have been proposed by several authors in different domains. Shi and Malik, while using spectral clustering for image segmentation, proposed to set σ to 10 or 20 percent of the total range of the feature distance function $d(x_i, x_j)$ [Shi and Malik, 2000]. On the other hand, von Luxburg proposed to use σ in the order of the mean distance of a point to its k-th nearest neighbor, with k

chosen as $k \sim log(n) + 1$ [von Luxburg, 2007]. Ng et al. suggested an iterative approach, where they select σ automatically by running their clustering algorithm repeatedly for a number of values of σ and selecting the one which resulted in a minimum distortion measure [Ng et al., 2002]. However, this approach increases significantly the computation time, and still requires the manual specification of the range of values to be tested.

In a different approach, [Zelnik-Manor and Perona, 2004] argued that when the data set includes clusters with different local statistics there may not be a single value of σ that works well for all the data. Instead, the authors propose a self-tuning approach, where they introduce an automatic selection of the local scaling value σ for each data point.

The approach used in this work is based on local statistics of the data in a texture window, as will be detailed in Section 3.7.

Encoding the Similarity Graph into a Similarity Matrix

Having the constructed graph, it can now be represented internally by a similarity matrix W of size $T \times T$ (where T is the number of peaks in a texture window, as defined in Equation 5), which encodes the pairwise similarities $W(p_l^k, p_m^{k+n})$ of all peaks p_l^k and p_m^{k+n} from the texture window with length N frames, where k is the frame index, l, m are the peak indices, $n \in \{0 \dots N-1\}$ is the frame offset between the peaks and $n = 0$ is used for peaks of the same frame (recall the definition of texture windows presented in Section 3.5). For the sake of notation simplicity, let $w_{i,j} := W(p_l^k, p_m^{k+n})$, where $i, j \in \{1, \dots, T\}$ are the single indices for the corresponding peaks p_l^k and p_m^{k+n} in the texture window. Using this notation, the similarity or affinity matrix W can be represented as:

$$W = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1T} \\ w_{21} & w_{22} & \dots & w_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ w_{T1} & w_{T2} & \dots & w_{TT} \end{bmatrix}. \tag{9}$$

Assuming G as an undirected graph implies that $w_{ij} = w_{ji}$, making W symmetric (other types of connected graphs are possible, but this symmetry leads to interesting properties explored by spectral clustering – see [von Luxburg, 2007]).

The problem of clustering can now be formulated using the computed similarity matrix W. The objective is to find a partition of the graph such that the edges between different groups have very low weights (i.e. points in different clusters are dissimilar from each other) and the edges within a group have high weights (i.e. points within the same cluster are similar to each other). There are several ways to perform clustering, and some of the

Table 1: *Frequencies and amplitudes of the peaks from two harmonic series A and B, as depicted in Figure 12.*

	A_0	A_1	A_2	A_3	A_4,B_3	B_0	B_1	B_2	B_4
f	440	880	1320	1760	2200	550	1100	1650	2750
a	.8	.8	.6	.4	.4	1	.8	.6	.4

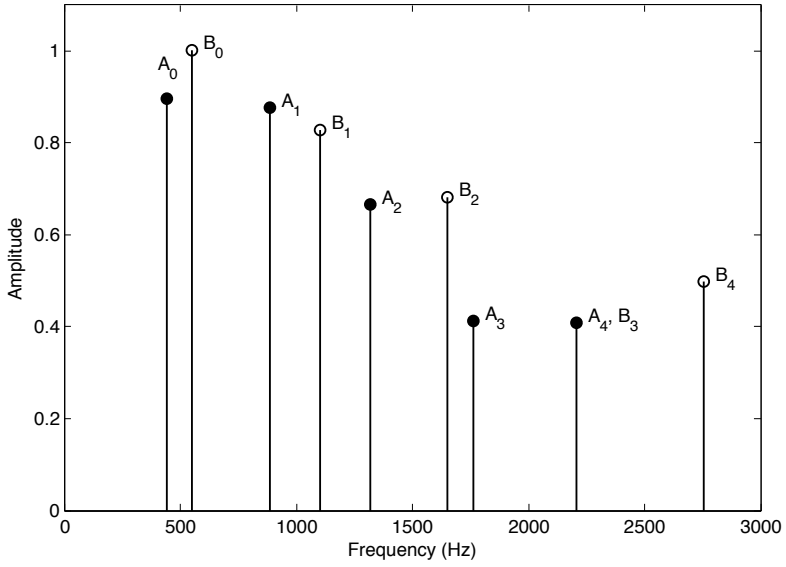


Figure 12: *Two sets A and B of harmonically related peaks, following the values specified in Table 1.*

most common approaches will be discussed in the following sections.

As a simple example, Figure 12 depicts the spectrum of two harmonic sound "sources" (i.e. harmonic series), A and B, whose spectral peaks $\{A_0, \ldots, A_4\} \in A$ and $\{B_0, \ldots, B_4\} \in B$ have the amplitudes and frequencies presented in Table 1. The peaks in each source share a harmonic relation, i.e. they are integer multiples of a same fundamental frequency f_0, which in this case is different for each source: source A has a f_0 of 440 Hz while B has f_0 equal to 550 Hz. In this particular example, the two sources share a common harmonic: A_4 has the same frequency and amplitude values as B_3.

Assume for now that, for illustrative purposes, some "ideal" harmonicity metric is used to construct a fully connected similarity graph where the similarities between each and all peaks from A and B are encoded (a detailed discussion on harmonicity metrics will be presented in Section 3.7.2). In this graph, two peaks $x_i, x_j \in A \cup B$ whose frequencies are harmonically related should have a high similarity value w_{ij}. Otherwise, they should have a low similarity w_{ij} (recall from Section 2.3.1 that harmonicity is one of the main

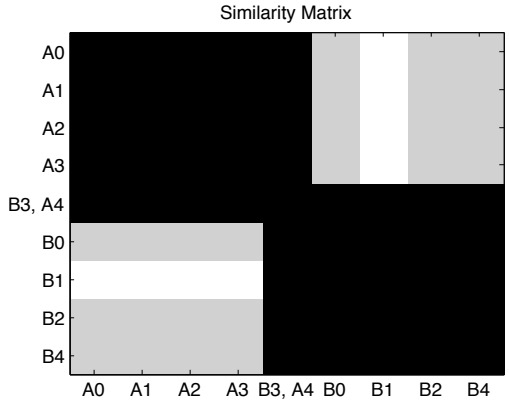


Figure 13: *Graphic representation of the similarity matrix W for two harmonic series, as defined in Table 1 and depicted in Figure 12, using some "ideal" harmonicity distance. High similarity values are mapped to black and low similarity values to white.*

perceptual cues used by the human ear to segregate sound sources).

Figure 13 presents a graphical representation of the affinity matrix W, which encodes the harmonic similarity w_{ij} between all the peaks from A and B, ordered for the sake of visualization in accordance to their corresponding source. The "ideal" harmonicity similarity measure used in this example clearly succeeds in assigning high similarity values for peaks coming from the same harmonic source (mapped to black), while attributing low similarity values for peaks from different sources (mapped to light gray or white). From this representation it is now possible to partition the peaks into clusters which will hypothetically have some correspondence to the actual sound sources in the signal (two, in this example).

Similarity Graph Notation

In order to formalize the rationale behind spectral clustering, it becomes helpful to introduce some basic graph notation at this stage (following the notation defined and used in [von Luxburg, 2007]).

Let the *degree of a vertex* $v_i \in V$ be defined as:

$$d_i = \sum_{j=1}^{T} w_{ij}. \tag{10}$$

Note that, for a fully connected graph, this sum is only meaningful for the vertices adjacent to v_i, since for all other vertices v_j the weight w_{ij} becomes negligible and in practice will end up approximated by 0 (mainly due to numerical limitations).

The *degree matrix* D can be defined as the diagonal matrix with the degrees $d_1, \ldots, d_T$ on the diagonal:

$$D = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_T \end{bmatrix}. \tag{11}$$

Given a subset of vertices $A \in V$, let its complement $V \setminus A$ be denoted as $\overline{A}$. Additionally, let the *indicator vector* $f = (f_1, \ldots, f_T)' \in {\mathbb{N}_0}^T$ be the vector with entries $f_i = 1$ if $v_i \in A$ and $f_i = 0$ otherwise (i.e. each element f_i indicates the cluster "label" assigned to each data point). For convenience let $i \in A$ be the shorthand notation for the set of indices $\{i | v_i \in A\}$, in particular when dealing with sums of the type $\sum_{i \in A} w_{ij}$. For two not necessarily disjoint sets $A, B \subset V$ define

$$cut(A, B) := \sum_{i \in A, j \in B} w_{ij}. \tag{12}$$

The *volume* of a subset $A \subset V$ can be defined as:

$$vol(A) := \sum_{i \in A} d_i. \tag{13}$$

Intuitively, $vol(A)$ measures the size of A by summing over the weights of all edges attached to vertices in A.

A subset $A \subset V$ of a graph is *connected* if any two vertices in A can be joined by a path such that all intermediate points also lie in A. A subset A is termed as a *connected component* if it is connected and if there are no connections between vertices in A and $\overline{A}$. The nonempty sets $A_1, \ldots, A_k$ form a *partition of the graph* if $A_i \cap A_j = \varnothing$ and $A_1 \cup \cdots \cup A_k = V$.

3.6.3 The Normalized Cut Criterion

When taking data in the form of a similarity graph, the task of separating points into different groups according to their similarities can be restated as follows: find a partition of the graph such that the edges between different groups have a very low weight (which means that points in different clusters are dissimilar from each other) and the edges within a group have high weight (which means that points within the same cluster are similar to each other).

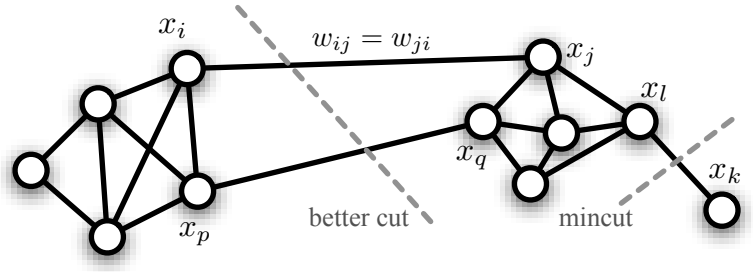


Figure 14: *Example of an undirected similarity graph, where the result of the mincut criterion defined in Equation 14 is depicted in opposition to a better and more balanced cut where the similarity between points in the same cluster and the dissimilarity between points in different clusters are maximized.*

Given a similarity graph with affinity matrix W, the simplest and most direct way to assemble a partition of the graph is to solve the *mincut* problem. Recalling the notation $\overline{A}$ for the complement of A, and the definition of $cut(A, B)$ given in Equation 12, if taking a given number of k subsets the *mincut* approach simply consists in choosing a partition $A_1, \ldots, A_k$ which minimizes:

$$\widehat{cut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} \widehat{cut}(A_i, \overline{A_i}) := \frac{1}{2} \sum_{i=1}^{k} cut(A_i, \overline{A_i}) \tag{14}$$

where the factor $1/2$ is introduced for notational consistency, otherwise each edge would be counted twice in the cut.

For the specific case of $k = 2$, the optimal bipartitioning of a graph is the one that minimizes the $cut(A, B)$ value (i.e. the mincut), a relatively easy problem that can be solved efficiently. However, in practice it seldom leads to satisfactory partitions because the solution tends to separate one individual vertex from the rest of the graph. If looking at the example graph depicted in Figure 14, mincut would cut between x_l and x_k, because the latter only has a single edge connecting it to another vertex. This goes against the desirable partition of the data, where clusters should be reasonably large groups of points. Obviously, a better cut would be between vertexes (x_i, x_j) and (x_p, x_q) which results in more balanced and denser clusters.

As a result, a way to circumvent this problem is to explicitly request that the sets $A_1, \ldots, A_k$ are "reasonably large". The two most common objective functions to encode this requirement are *RatioCut* and the *Normalized Cut* (Ncut) [Shi and Malik, 2000, von Luxburg, 2007]. Due to the need to apply a relaxation operation for computational tractability, the first leads to *unnormalized spectral clustering* while the latter leads to

normalized spectral clustering, making the Ncut a better choice when compared to RatioCut (refer to Appendix A for more details). The Ncut was therefore the approach chosen for the framework proposed in this thesis and will be described in the following.

Recalling the definition of the volume of a set A presented in Equation 13, let Ncut be defined as:

$$\mathrm{Ncut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{cut(A_i, \overline{A_i})}{vol(A_i)} = \sum_{i=1}^{k} \frac{\widehat{cut}(A_i, \overline{A_i})}{vol(A_i)} \tag{15}$$

which, differently from mincut, takes a small value if the clusters A_i are not too small. In particular, the minimum of the function $\sum_{i=1}^{k} (1/vol(A_i))$ is achieved if all $vol(A_i)$ coincide. Consequently, what Ncut tries to achieve is that clusters are "balanced", as measured by the edge weights.

However, introducing balancing conditions makes the previously simple to solve mincut problem become NP hard [Shi and Malik, 2000, von Luxburg, 2007]. Conveniently, spectral clustering appears as a way to solve relaxed versions of these problems, and a detailed discussion on the use of *graph Laplacians* for this purpose is presented in Appendix A. In summary, the solution should rely on the relaxation of the indicator vector $f = (f_1, \ldots, f_T)'$ (where, recall, each element f_i is supposed to indicate the cluster "label" assigned to each data point) to take on real values (i.e. allow $f_i \in \mathbb{R}$), making it possible to find a partition that minimizes the Ncut criterion by solving a generalized eigenvalue system, which leads to the eigenvalues λ of the *normalized graph Laplacian*, defined as follows:

$$Lf = \lambda Df \tag{16}$$

where $L = D - W$, D is the degree matrix (see Equation 11) and W is the similarity matrix (see Equation 9).

Once the eigenvectors are computed, it is possible to show that the second smallest eigenvector of this generalized eigensystem is the real valued solution to the bipartite Normalized Cut minimization [Shi and Malik, 2000, von Luxburg, 2007]. Similarly, it is possible to demonstrate that the eigenvector with the third smallest eigenvalue is the real valued solution that optimally subpartitions the first two parts. In fact, this line of argument can be extended to show that one can subdivide the existing graphs, each time using the eigenvector with the next smallest eigenvalue.

3.6.4 Partitioning Approaches

Once the eigenvectors of the generalized eigensystem defined in Equation 16 are obtained, it becomes possible to partition the graph into k pieces using the eigenvectors corresponding to the k smallest eigenvalues [von Luxburg, 2007].

In the ideal case, the eigenvectors should only take on two discrete values and their values would indicate exactly how to partition the graph. However, and because of the applied relaxation, the computed eigenvectors can now take on continuous values and it is therefore necessary to transform them into a discrete form.

Still, because in practice the approximation error from the real valued solution to the discrete valued solution accumulates with every eigenvector taken and all eigenvectors have to satisfy a global mutual orthogonality constraint, solutions based on higher eigenvectors become unreliable. Several approaches to achieve this discretization have been proposed [Shi and Malik, 2000, von Luxburg, 2007], and the following sections will discuss two alternatives solutions to obtain a final partition of the dataset.

Simultaneous k-Way Cut

The Ncut solution is obtained by solving a generalized eigensystem, as defined in Equation 16. As detailed in Appendix A, this is equivalent to finding the eigenvectors of the *normalized random walk graph Laplacian* L_{rw} (see Equation 45). Once the eigenvectors of L_{rw} are computed, it becomes possible to partition the graph into k pieces using the eigenvectors corresponding to the k smallest eigenvalues. Because of the relaxation used to solve the Ncut minimization problem, the computed eigenvectors will take real values and it is therefore necessary to transform them into a discrete form so they can be used as k dimensional indicator vectors for each data point x_i. Thus, taking these k eigenvectors of L_{rw} it is possible to change the representation of the data points x_i to points $y_i \in \mathbb{R}^k$, an operation known as *spectral embedding*.

If $u_1, \ldots, u_k$ are the first k generalized eigenvectors of the generalized eigenproblem $Lu = \lambda Du$, then let $U \in \mathbb{R}^{T \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns (where T is the number os data points, i.e. peaks, to cluster – see Equation 5):

$$U = \begin{bmatrix} u_1(1) & \ldots & u_k(1) \\ u_1(2) & \ldots & u_k(2) \\ \vdots & \ddots & \vdots \\ u_1(T) & \ldots & u_k(T) \end{bmatrix}. \tag{17}$$

Then, for $i = 1, \ldots, T$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U. The usefulness of this change of representation comes directly from the properties of the graph Laplacians (see Appendix A), which allow to enhance the cluster properties in the data and to consequently favour their detection [von Luxburg, 2007, pp.7].

Considering this spectral embedded representation, a simple clustering algorithm like k-means can then be used to extract the final partition from the real valued matrix of eigenvectors (i.e. cluster the points $(y_i)_{i=1,\ldots,T}$ in $\mathbb{R}^k$ into clusters $C_1, \ldots, C_k$ using, for e.g., k-means).

However, there is nothing principled about using the k-means algorithm in this stage. Alternative approaches to the use of k-means in this step have been proposed by several authors. For instance, Bach and Jordan take the subspace spanned by the first k eigenvectors, and try to approximate this subspace as well as possible using piecewise constant vectors [Bach and Jordan, 2004]. This also leads to minimizing certain Euclidean distances in the space $\mathbb{R}^k$, which can be done using some weighted k-means algorithm.

Shi and Malik also proposed an agglomerative approach based on "over-segmentation" followed by a pruning and merging step [Shi and Malik, 2000]. If one sets k to be higher than the maximum number of plausible clusters in the data, say $k > k'$, an over-segmentation of the graph will occur. It is then possible to come with some forms to iteratively merge segments until the desired k' number of output clusters is obtained.

In any case, this partitioning step should be very simple if the data contains well expressed clusters (e.g. in the ideal case where completely separated clusters exist, the eigenvectors of L_{rw} will be piecewise constant – see Appendix A). In such a case, all points x_i belonging to the same cluster are mapped to exactly the sample point y_i. In such a trivial case, any clustering algorithm applied to the points $y_i \in \mathbb{R}^k$ will be able to extract the correct clusters.

This means that the choice of the clustering algorithm for this last step is somewhat arbitrary and that the real question becomes whether or not the Euclidean distance between the points y_i is a meaningful quantity to look at. Several authors have studied this issue and showed that this Euclidean distance is in fact meaningful (see [von Luxburg, 2007, pp.24] and references within).

However, in all these solutions, the final number of clusters k still needs to be specified beforehand. As previously argued, and particularly relevant for the segregation of sound events from audio mixtures, this may not be desirable or even possible to do and therefore should somehow be automatically inferred from the data. The choice of the number k of output clusters is a general problem for all clustering algorithms, and a variety of more or

less successful methods have been proposed for this problem.

Estimating the Number k of Output Clusters

In model-based clustering approaches there exist well justified criteria to choose the number of clusters from the data, usually based on the log-likelihood of the data, which can then be treated in a frequentist or Bayesian way [von Luxburg, 2007, pp.22]. In approaches where few or no assumptions on the underlying model are made, a large variety of different measures can be used to select the number of clusters. Examples range from ad-hoc measures such as the ratio of within-cluster and between-cluster similarities, over information-theoretic criteria, the gap statistic, to stability approaches [von Luxburg, 2007, pp.23].

Although all these approaches could be used for spectral clustering, one tool that is intimately related to the formulation of graph Laplacians is the *eigengap heuristic* [von Luxburg, 2007, pp.23]. In this approach the goal is to choose the number of clusters k such that all eigenvalues $\lambda_1, \ldots, \lambda_k$ are very small, but where $\lambda_{k+1} > 0$ is relatively large. However, although this heuristic works well if the clusters in the data are very well pronounced, it becomes less effective for noisy or overlapping clusters. As a result, the gap between the i-th eigenvalue and the next one can be either small or large, making this criteria unreliable.

In an alternative approach, [Zelnik-Manor and Perona, 2004] propose a method based on the structure of the eigenvectors and their rotation to create a maximally sparse representation. They showed that their self-tunning spectral clustering algorithm (which additionally includes local scaling of the similarity graph, as briefly presented in Section 3.6.2) can successfully handle multi-scale data, usually problematic for other approaches.

However, and as argued in Section 2.4.4, even if it was possible to robustly estimate a number of clusters in a graph based on the structure of its eigenvalues and eigenvectors, the question about the relevance of their correspondence to the sound elements perceived by different listeners, or by a same listener in different contexts or attentional levels, remains. Consequently, a divisive clustering solution could provide a better approach to sound segregation, which intrinsically provides an hierarchical view on polyphonic music mixtures.

Recursive Two-Way Cut

In a divisive clustering approach, the low-level grouping cues used for the construction of the corresponding similarity graph can be used to come up with a hierarchical partition of the data. Therefore this approach emerges as a more suitable framework for sound

segregation. Additionally, the divisive nature of such a clustering approach does not require a priori knowledge of the number of output clusters, which, as previously discussed, is an ill-posed problem.

As described in Appendix A the second smallest eigenvector of the generalized eigensystem L_{rw} is the real valued solution to the Normalized Cut minimization. However, similarly to the k-way cut method described in the previous section, because the eigenvectors can take on continuous values, it is necessary to choose a splitting point to partition it into two parts.

There are many different ways of choosing a splitting point. One option would be to take the value 0 or the median value as the splitting point. Another option, presented in [Shi and Malik, 2000] and used in the system proposed in this thesis, is to search among l evenly spaced splitting points within the eigenvector for the one that produces a partition with the best Ncut value (i.e. smallest). After the graph is broken into two parts, the algorithm can be run recursively on the two partitioned parts until the desired number of clusters k have been extracted.

A more interesting alternative that does not need the prior specification of the number of clusters k would be to keep the partitioning recursion going on until the Ncut value exceeds a certain limit or when some stability criteria of the cut is no longer valid. Shi and Malik propose a stability criterion on the partition that is based on the observation that sometimes an eigenvector can take on the shape of a continuous function instead of a discrete indicator function being sought [Shi and Malik, 2000]. This usually means that, from the perspective of sound segregation, such an eigenvector is attempting to break an element from a sound mixture in an uncertain way. In other words, if forcing the partition of the sound element based on such an eigenvector, many different splitting points will exist (all of them with similar Ncut values) and the segregation will become highly uncertain and unstable.

The current version of the framework proposed in this work still requires the prior specification that the number of maximum clusters k, but the system is flexible enough to include a stopping criteria as proposed in [Shi and Malik, 2000]. Section 3.8.1 will provide additional details on how the partitioning and final cluster selection is currently being performed.

3.6.5 The Grouping Algorithm

Following the previous discussions about the several issues involved in the implementation of a sound segregation solution based on spectral clustering, next the final algorithm currently used in the framework proposed in this thesis is summarized. The algorithm is based on the *normalized random walk spectral clustering computation*, as proposed in [Shi and Malik, 2000] and described in Appendix A.

1. Given T data points $x_1, \ldots, x_T$, set up a weighted graph $G = (V, E)$;
2. Compute the similarity matrix $W \in \mathbb{R}^T$ and the degree matrix $D \in \mathbb{R}^T$ (the specific similarity cues used for sound segregation are described in the following Section 3.7);
3. Compute the unnormalized Laplacian $L = D - W$;
4. Calculate the first k generalized eigenvectors $u_1, \ldots, u_k$ of the generalized eigenproblem $Lu = \lambda Du$ (which correspond to the eigenvectors of L_{rw});
5. Recursively partition the dataset based on a recursive two-way Ncut approach, using the eigenvector with the second smallest eigenvalue of L_{rw} from each sub-graph;
6. Stop when the specified number of clusters k is reached, and data points are clustered into clusters $C_1, \ldots, C_k$.
7. Output clusters $A_1, \ldots, A_k$ with $A_i = \{j | y_j \in C_i\}$.

The following section will present the perceptually motivated similarity functions used in step 2, including a novel harmonicity similarity cue, known as *Harmonically Wrapped Peak Similarity* (HWPS). Later on in this chapter a discussion about the way to select the obtained clusters of peaks for resynthesis and further analysis will be presented.

3.7 Perceptual Cues as Grouping Criteria

Following the more generic considerations about the definition of a similarity function for spectral clustering presented in Section 3.6.2, this section will focus on the specification of grouping criteria that attempt to capture the perceptual proximity between sound components (i.e. peaks) in accordance to some of the perceptual grouping principles discussed in Section 2.3.1. These include frequency and amplitude proximity and a novel harmonicity criterion, termed Harmonically Wrapped Peak Similarity (HWPS). Such a harmonicity criterion is of upmost importance since amplitude and frequency cues are not enough for segregating multiple overlapping harmonic sound sources.

Still, the proposed framework is able to easily accommodate future similarity functions such as frequency or time masking [Lagrange et al., 2006], common fate, stereo location,

or other grouping principles, allowing to model an increasing number of perceptual mechanisms involved in human hearing (discussed in Section 2.3.1).

3.7.1 Amplitude and Frequency Similarity

Two of the most basic similarities explored by the auditory system are related to the frequency and amplitude features of the sound components in a sound mixture (see Section 2.3.1).

Accordingly, the edge weight connecting two peaks p_l^k and p_m^{k+n} will depend on their frequency and amplitude proximities. Following the generic considerations discussed for the definition of a similarity function for spectral clustering in Section 3.6.2, amplitude and frequency similarities, W_a and W_f respectively, are defined as follows:

$$W_a(p_l^k, p_m^{k+n}) = e^{-\left(\frac{a_l^k - a_m^{k+n}}{\sigma_a}\right)^2} \tag{18}$$

$$W_f(p_l^k, p_m^{k+n}) = e^{-\left(\frac{f_l^k - f_m^{k+n}}{\sigma_f}\right)^2} \tag{19}$$

where the Euclidean distances are modeled as two Gaussian functions, as previously defined in Equation 8. The amplitudes are measured in deciBel (dB) and the frequencies are measured in Barks (a frequency scale approximately linear below 500 Hz and logarithmic above), since these scales have shown to better model the sensitivity response of the human ear [Hartmann, 1998].

The Distances in the Similarities

Before arguing about the adequacy of the final Gaussian similarity functions W_a and W_f (which are discussed later in this section and again in Section 3.7.3 when arguing about the combination of similarity cues), it is important to first try to understand the meaning of the Euclidean distances $|a_l^k - a_m^{k+n}|$ and $|f_l^k - f_m^{k+n}|$, since they should already be able to capture relevant information about amplitude and frequency proximities between the sound elements in a complex signal.

Correspondingly, the first three rows of Figure 15 show the histograms of the amplitude and frequency distances between the peaks in four different audio examples. For the sake of clarity and illustration, frequency distances are also represented in Hz for each example.

The first audio signal, *Tone A*, with a duration of about 15 seconds, corresponds to the harmonic series A defined in Table 1 and depicted in Figure 12. Its amplitude histogram

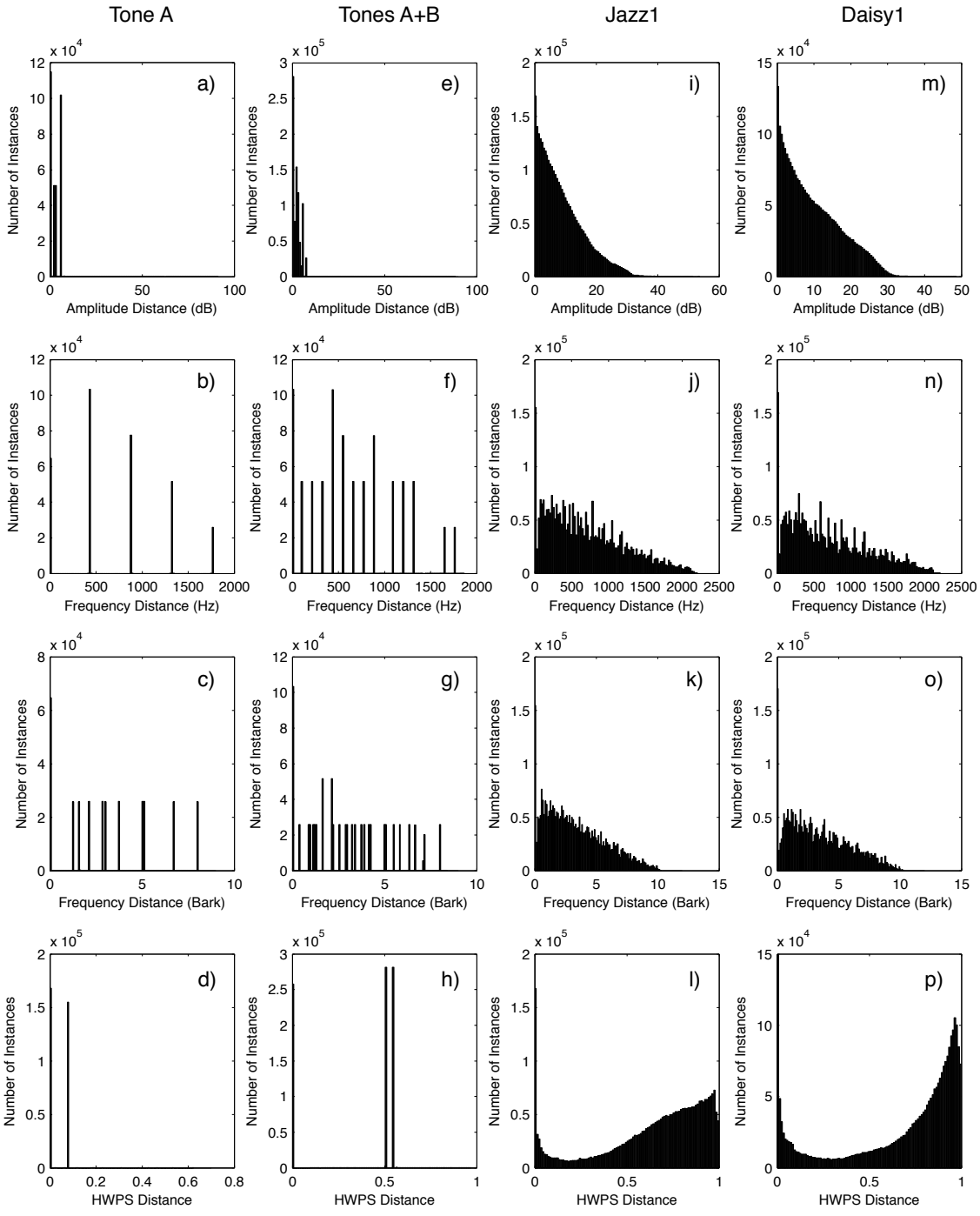


Figure 15: *Histograms for the amplitude, frequency (in Hz and Bark) and HWPS distance distributions in four different audio signals (organized in columns). See text for details.*

clearly shows three peaks (see plot *a* of Figure 15), corresponding to the pairwise absolute differences (in dB) between the three different amplitudes of its harmonics (i.e. 0.8, 0.6 and 0.4 – see Table 1). The frequency histogram (in Hz – see plot *b*) also clearly presents peaks for distances that are multiple integers of the fundamental frequency of the harmonic series *A*, i.e. 440Hz, corresponding to the pairwise distances between the frequencies of all the harmonics. Given the non-linear nature of the Bark scale, the histogram of the frequencies distances in Bark (see plot *c*) is not so easy to interpret.

The second example signal, *Tones A+B*, with a duration of about 15 seconds, is the mix of the harmonic series *A* of the previous example with the harmonic series *B*, whose complete definition is presented in Table 1 and depicted in Figure 12. In this case, the histogram of the amplitude distances becomes more complex but it is still possible to identify the main pairwise distances between the peaks in the mix as peaks in the histogram (see plot *e* in Figure 15). A similar observation can be made for the frequency distances (see plots *f, g*), where the pairwise absolute differences between the two harmonic sets of peaks are also still possible to identify in the histogram (clearer when measured in Hz – see plot *f*).

The two last columns of Figure 15 show the histograms for two 30 second real-world signals, *Jazz1* and *Daisy1*, both from the MIREX dataset[3]. Given their higher complexity, their amplitude and frequency distance distributions are equally more complex and denser, and consequently harder to interpret in detail. Nevertheless, it is interesting to observe that both signals present similarly shaped distributions for amplitude and frequency differences, even though *Jazz1* is an excerpt from a jazz piece with a saxophone playing the main melody over a piano, bass and drums background, while *Daisy1* is an excerpt from a japanese pop song, with a female singing voice.

Both amplitude distance histograms (see plots *i* and *m*) exhibit an exponentially decaying distribution, meaning that, on average, most of the spectral components in the signal are close in amplitude. In fact, this is far from surprising since the current system only selects the 20 higher amplitude spectral peaks in each frame.

The frequency distance histograms (see plots *j, n, k* and *o*) have identically shaped distributions for both signals. Also interesting to note is the similar shape of the distributions for the Hz and Bark representations. In all the cases, it is possible to observe a very high peak for zero valued distances (over 1.5×10^5) and, similarly to the amplitude histograms, their shapes indicate that there is a high number of components in the signal that are close in frequency. Additionally, in the frequency histograms expressed in Hz

[3] http://www.music-ir.org/mirex2005/index.php/Main_Page

(plots *j, n*) it is possible to observe some harmonically related local maxima (resembling the ones in plot *b*) which probably result from strong harmonic events in the signal (e.g. long and strong notes).

For the sake of presentation clarity, Figure 15 also includes in plots *d, h, l* and *p* information about the harmonicity cue proposed in this thesis, which will be discussed in Section 3.7.2.

Using Amplitude or Frequency Similarity for Sound Segregation

The definition of the amplitude and frequency similarity functions W_a and W_f given in Equations 18 and 19 takes the Euclidean distances between the amplitudes or frequency values of each pair of peaks in the signal, and computes a Gaussian function where it is necessary to specify neighbourhood width factors, σ_a and σ_f, respectively.

As discussed in Section 3.6.2, the specification of these neighbourhood parameters is not trivial. In the current system implementation, both the amplitude and frequency Euclidean distances are first normalized to values in the range $[0, 1]$ (i.e. $\overline{d(i,j)} = \frac{d(i,j)-min(d(i,j))}{max(d(i,j))-min(d(i,j))}$, where $\overline{d(i,j)}$ is the normalized distance and $d(i,j)$ can be either the amplitude or the frequency distance between peaks i and j).

Plots *a* and *c* in Figures 16 and 17 show the normalized distance histograms for the *Tones A+B* and *Jazz1* examples, respectively, where it is clear that although the range of the distances is now in the interval $[0, 1]$, their shape is unchanged (compared to plots *f, j* in Figure 15). This means that their relative distances are preserved, making no difference to the grouping algorithm. Still, this normalization allows to simplify the specification of the neighborhood parameters since now both the amplitude and the frequency distances span the same numeric range. Additionally, and as will be discussed in Section 3.7.3, this will also simplify the combination of several similarity cues into a single similarity function.

The current system uses a fixed and empirically found value of 1.2 for both σ_a and σ_f, which provided satisfactory segregation results (see Chapter 4). The gray dashed lines in plots *a* and *c* in Figures 16 and 17 present a segment of the corresponding Gaussian function (recall Figure 11) overlaid with the histograms (for visualization purposes, the Gaussian functions represented in the plots were scaled by the maximum bin value of each histogram).

However, because the proposed system works in a causal manner, the normalization operation is performed independently for each texture window and not for the full length of the signals under analysis. This may result in local normalization artifacts, since for

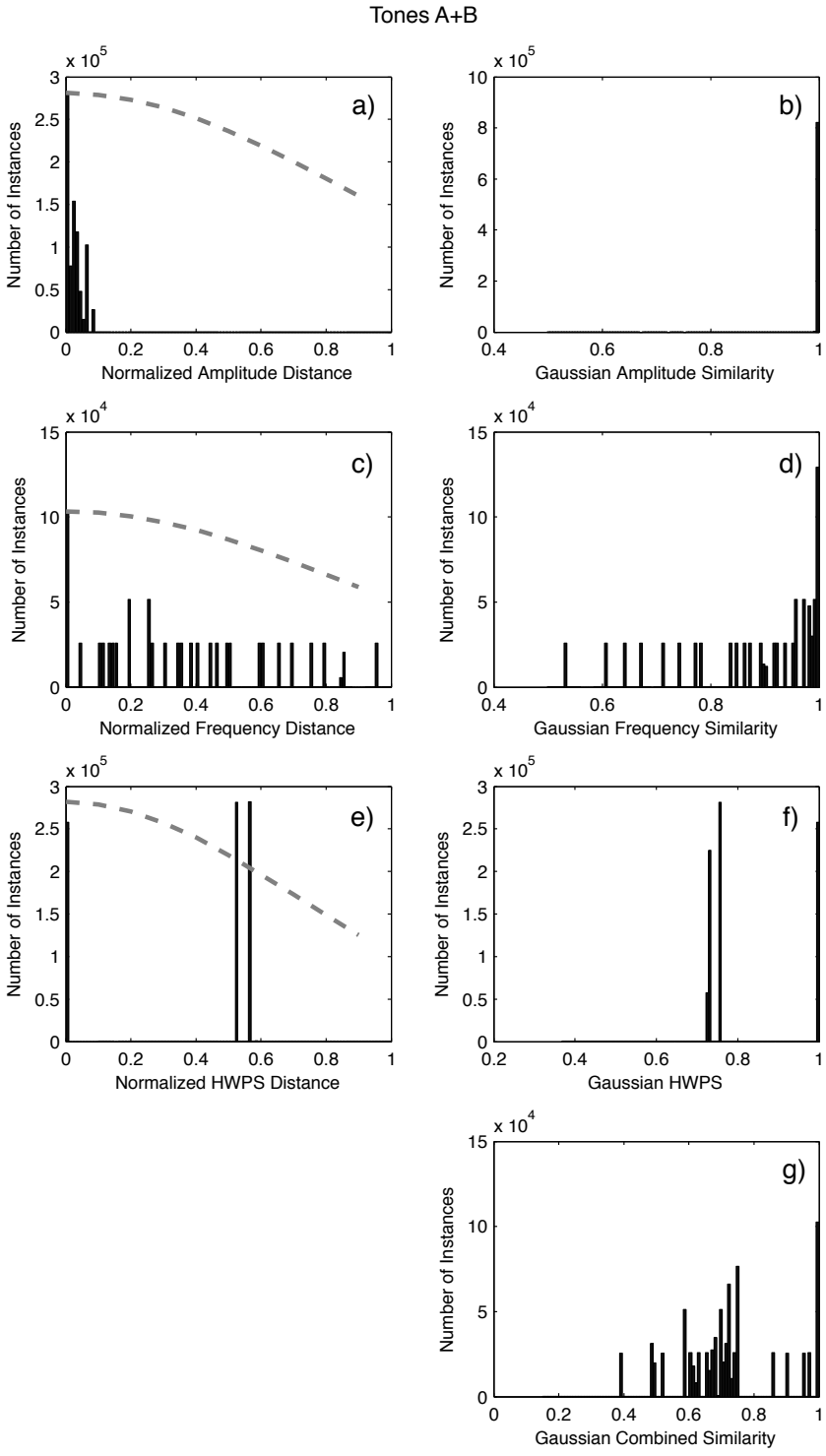


Figure 16: *Histograms of the normalized distance histograms (left column) and the corresponding histograms for the Gaussian similarity function W_a, W_f and W_h (right column) for the audio signal "Tones A+B". The gray dashed lines in the left plots represent scaled segments of the Gaussian function defined in Equation 8 using the specified σ values for each similarity cue (see text for details). Panel (g) shows the histogram of the combined Gaussian similarity function W_{afh} (defined and discussed in Section 3.7.3).*

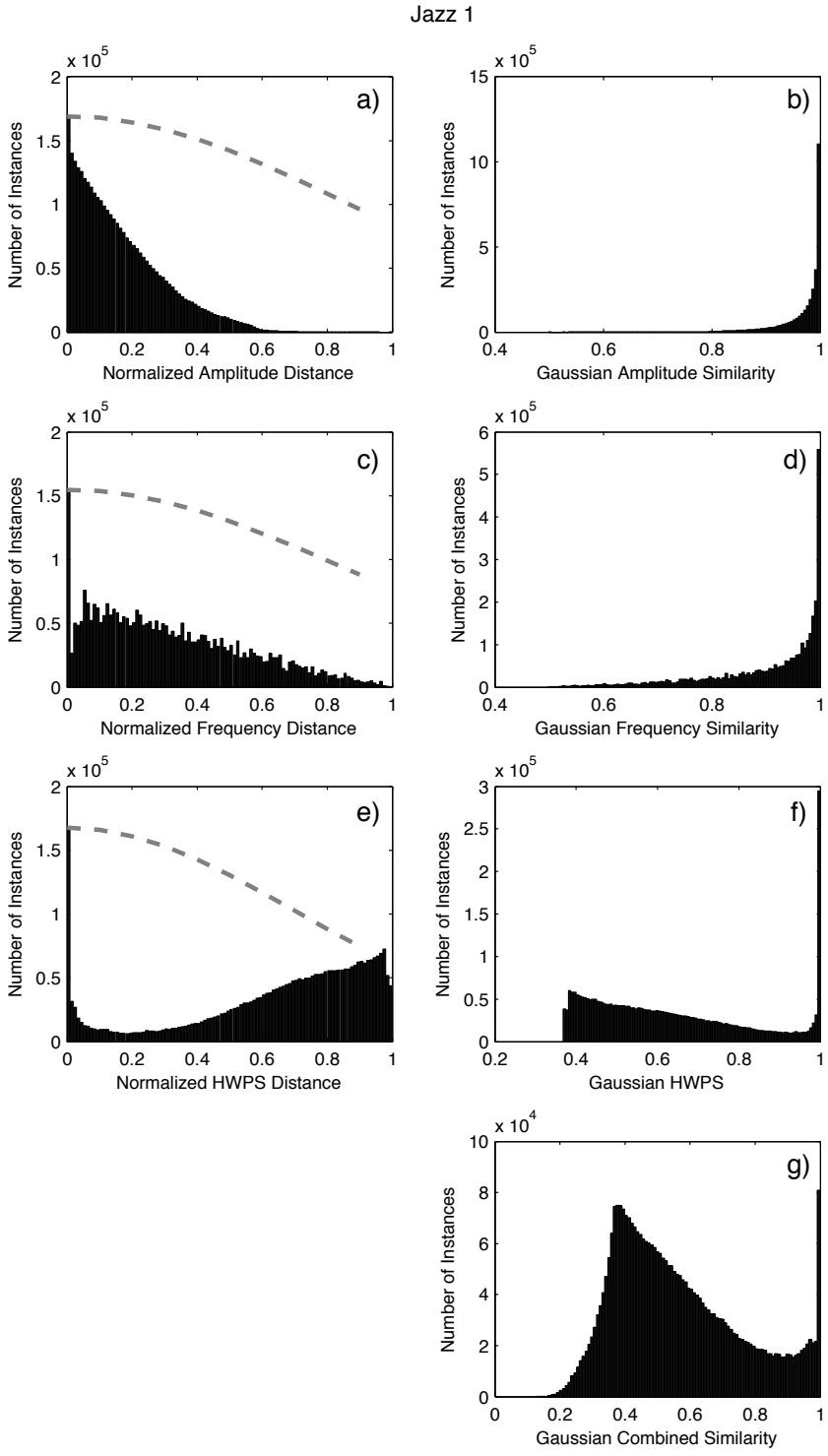


Figure 17: *Histograms of the normalized distance histograms (left column) and the corresponding histograms for the Gaussian similarity function W_a, W_f and W_h (right column) for the audio signal "Jazz1". The gray dashed lines in the left plots represent scaled segments of the Gaussian function defined in Equation 8 using the specified σ values for each similarity cue (see text for details). Panel (g) shows the histogram of the combined Gaussian similarity function W_{afh} (defined and discussed in Section 3.7.3).*

each texture window the distance ranges may be substantially different, depending on the window lengths and on the signal characteristics at that particular time interval. A possible alternative worth exploring in the future is the use of prior information to help setting up the neighborhood width for each similarity cue. The knowledge about the just noticeable differences (JND) in amplitude and frequency perception [Hartmann, 1998] may be a good criteria for defining the similarity widths, where diferences below the corresponding JND would be considered as fully similar, while all the other distances would gradually correspond to lower similarity values.

Nevertheless, if taking for now the normalized distances and using the specified fixed neighborhood width parameters, it becomes possible to compute the similarity values for amplitude and frequency between each pair of peaks, W_a and W_f. Plots *b* and *d* of Figures 16 and 17 show the corresponding Gaussian amplitude and frequency similarity histograms for the two audio examples. Notice that peaks whose features were close in the feature space (i.e. they have a distance equal or close to 0) will now have a high similarity value (i.e. equal or close to 1.0) and the specified σ parameters will make higher distances less relevant.

As an example of the segregation results obtained when using a single similarity function in step 2 of the grouping algorithm (see Section 3.6.5), Figures 18 and 19 show the bipartite segregated signals obtained by manually setting the number of clusters k to 2 and just using for each case either amplitude or frequency similarity functions (i.e. W_a in plots *b,c* and and W_f in plots *d,e*, respectively)[4].

Panel *a* in Figure 18 shows a time-frequency plot of the sinusoidal representation of the *Tones A+B* example (whose spectral components are as defined in Table 1 and depicted in Figure 12), where darker shades of gray represent higher amplitude values. Because the spectral peaks in this signal are constant both in amplitude and frequency over time, they are represented as horizontal dark lines[5].

Panels *b* and *c* show the resulting bipartite clusters of using only the amplitude similarity function W_a in the grouping algorithm. In this example it is clearly visible that the higher amplitude peaks A_0, B_0, A_1, B_1 and $A_4 + B_3$ end up grouped together in the first cluster (panel *b*), while peaks A_2, B_2 and A_3, the lower amplitude peaks, are included in the second cluster (panel *c*). In a similar way, panels *d* and *e* show the resulting bipartite clusters of using only the frequency similarity function W_f for the grouping algorithm.

[4]Audio clips of the signals plotted in Figures 18 and 19 are available at `http://www.fe.up.pt/~lgustavo`

[5]Recall that, because the current system implementation ignores frequency components above 2500 Hz (see Section 3.2), peak B_4, whose frequency is 2750 Hz, is absent from the plots.

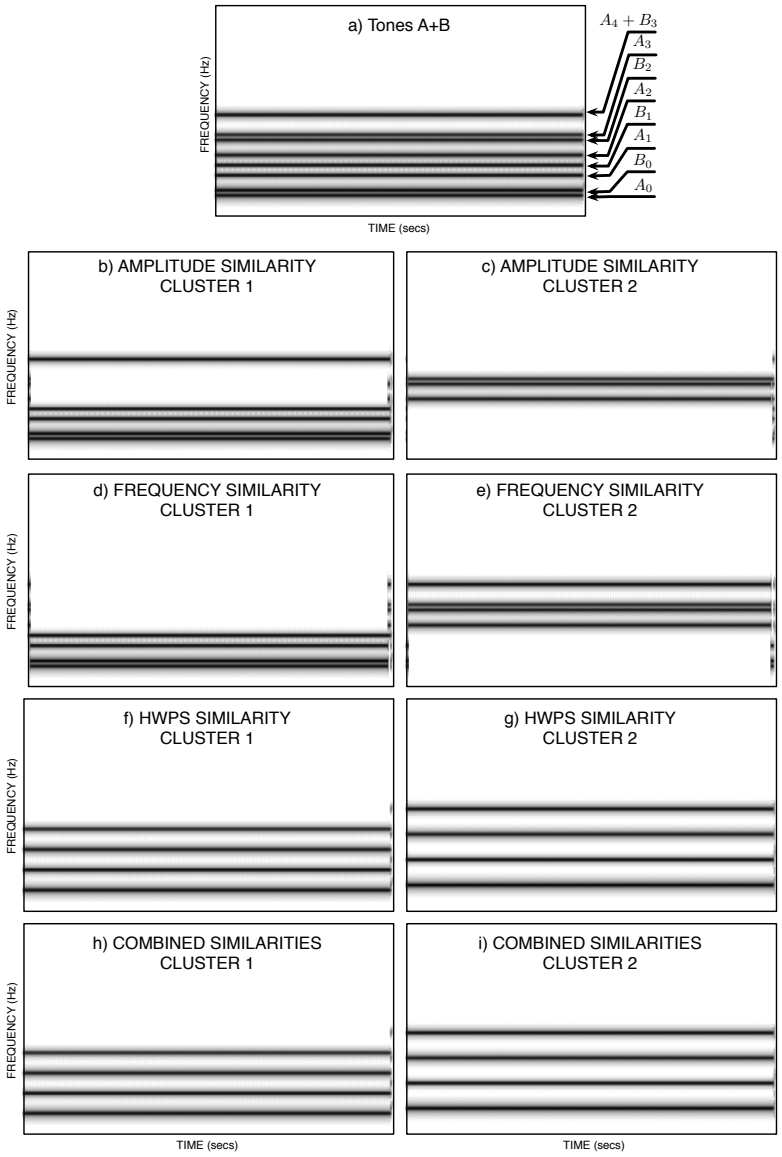


Figure 18: *Bipartite segregation results of the signal "Tones A+B" (see Table 1 and Figure 12) using different similarity cues. Panel (a) shows the time-frequency plot of the sinusoidal representation of the mixed signal, where darker shades of gray represent higher amplitude peak values. The first three rows in the figure (plots (b) to (g)) represent the pairwise clusters obtained by using in each case a single similarity cue, while the last row (plots (h), (i)) show the clusters obtained by combining all the previous similarity cues (as discussed in Section 3.7.3).*

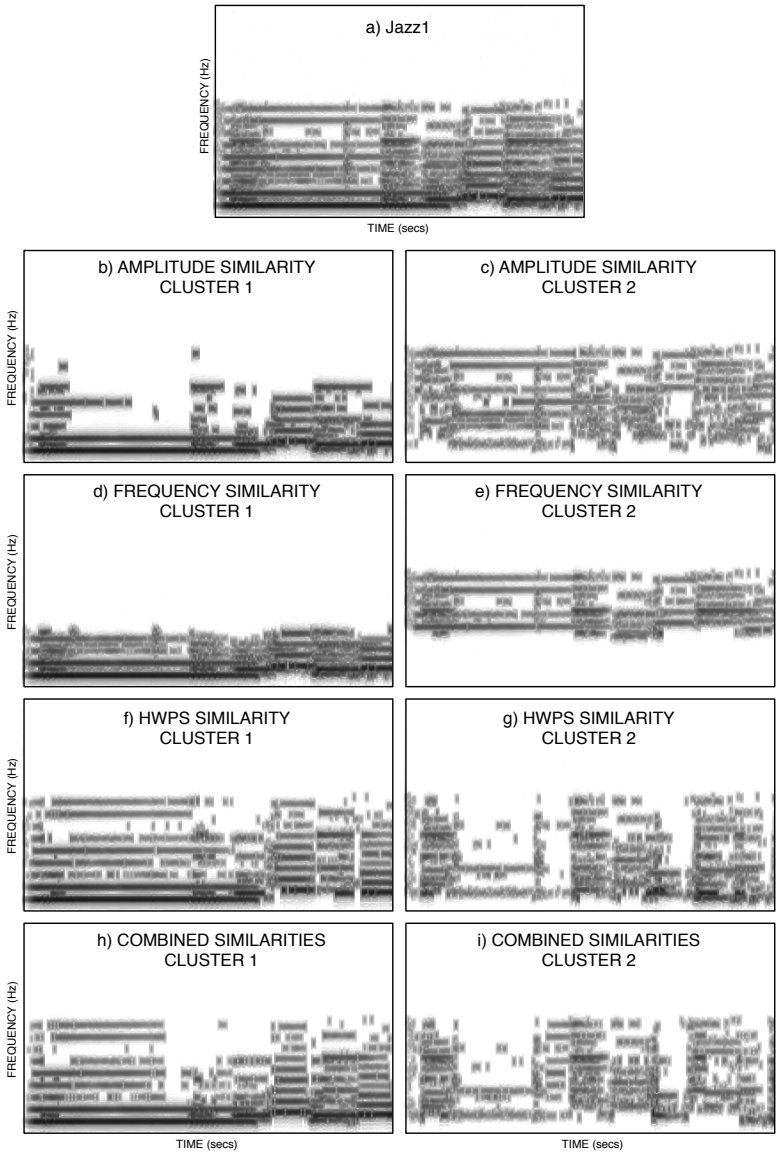


Figure 19: *Bipartite segregation results of the signal "Jazz1" using different similarity cues. Panel (a) shows the time-frequency plot of the sinusoidal representation of the mixed signal, where darker shades of gray represent higher amplitude peak values. The first three rows in the figure (plots (b) to (g)) represent the pairwise clusters obtained by using in each case a single similarity cue, while the last row (plots (h), (i)) show the clusters obtained by combining all the previous similarity cues (as discussed in Section 3.7.3).*

In this case, the components in the mixture end up grouped into low and high frequency groups, depending on their frequencies.

Figure 19 also shows similar experiments, this time for a real-world signal (specifically, a 2 second excerpt from the Jazz example used in the previous examples). Similarly to Figure 18, panel *a* presents the time-frequency plot of the sinusoidal representation of the mixture signal where the spectral peaks resulting from the sinusoidal analysis are depicted. As before, darker shades of gray represent higher amplitude values. Plots *b* and *c* clearly show that the spectral peaks with the highest amplitudes ended up grouped together in one cluster while the peaks with lower amplitudes were clustered into another. In a similar fashion, when using frequency similarity, low frequency spectral components are grouped together into cluster 1 (see plot *d*) while the peaks with higher frequencies end up grouped into the second cluster (see plot *e*).

However, all these results show that, by themselves, the amplitude and the frequency similarity functions are not able to correctly segregate the harmonic "sources" in the signals. Nevertheless, their role is important for grouping peaks from non-harmonic sound sources (i.e. unpitched sounds), or inharmonic or noisy frequency components in harmonic sounds. The following section will propose a novel harmonicity cue that, as will be shown, is able to successfully segregate harmonic components in a mixture signal.

3.7.2 Harmonically Wrapped Peak Similarity (HWPS)

A wide variety of sounds produced by humans are harmonic, from singing voice and speech vowels, to musical sounds. As stated by Alain de Cheveigné: "*Harmonicity is the most powerful among ASA cues. It is also the cue most often exploited in computational ASA systems and voice-separation systems*" [Wang and Brown, 2006]. It is therefore a critical cue for separating harmonic sounds which are particularly important for musical signals.

Accordingly, some of the source separation and multiple fundamental frequency estimation algorithms iteratively estimate the dominant fundamental frequency f_0, and then remove the spectral components that are most likely to belong to the source attached to the corresponding f_0 (e.g. [Klapuri, 2004]). Few studies have focused on the identification of harmonic relations between peaks without any prior fundamental frequency estimation.

By contrast, in this work the focus is on the definition of a similarity function between time-frequency components, of the same frame or of different frames, that considers the harmonicity cue without fully relying on the prior knowledge of the f_0's (although such prior information can be easily embedded and improve the grouping capability of the proposed similarity function – see discussion later in this section). The challenge is therefore

to define a similarity measure between two frequency components (i.e. peaks) that is high for harmonically related peaks and low for peaks that are not harmonically related.

Existing Harmonicity Cues

Most existing approaches use the mathematical properties of the harmonically related frequencies to build a harmonicity similarity measure for a single frame [Virtanen and Klapuri, 2000, Srinivasan and Kankanhalli, 2003, Rosier and Grenier, 2004, Martins and Ferreira, 2002]. For example, Virtanen and Klapuri [Virtanen and Klapuri, 2000] consider whether the ratio of the frequencies of the components is a ratio of small positive integers, while Martins and Ferreira [Martins and Ferreira, 2002] select peaks whose frequencies are integer multiples of a candidate f_0 peak to form harmonic clusters in each frame.

Srinivasan and Kankanhallli consider a harmonicity map that can be precomputed to estimate the harmonic similarity between two spectral bins. Considering two bin indexes i and j, the map is computed as follows:

$$\text{hmap}(i,j) = 1 \text{ if } \text{mod}(i,j) = 0 \text{ or } \text{mod}(j,i) = 0. \tag{20}$$

This map is next smoothed to allow an increasing level of inharmonicity using a Gaussian function, normalized so that the sum of its elements is unity. The standard deviation of the Gaussian function is set to be 10% of its center frequency, see [Srinivasan and Kankanhalli, 2003] for further details. The similarity between peaks p_l and p_m is then:

$$W_s(p_l, p_m) = \text{shmap}(M_l, M_m) \tag{21}$$

where shmap is the smoothed and normalized version of hmap, and M_l corresponds to the bin index of peak p_l. The frames indexes are omitted for clarity sake.

According to Virtanen and Klapuri [Virtanen and Klapuri, 2000], if two peaks p_l and p_m are harmonically related, the ratio of their frequencies f_l and f_m is a ratio of two small positive integers a and b (which correspond to the harmonic rank of each peak, respectively). By assuming that the fundamental frequency cannot be below the frequency resolution of the sinusoidal modeling front-end (i.e. $f_{\min} = 50$ Hz), it is possible to obtain an upper limit for a and b, respectively $a < \left\lfloor \frac{f_l}{f_{\min}} \right\rfloor$ and $b < \left\lfloor \frac{f_m}{f_{\min}} \right\rfloor$. A harmonic distance measure can be defined:

$$W_v(p_l, p_m) = 1 - \min_{a,b} \left| \log \left(\frac{f_l / f_m}{a/b} \right) \right|, \tag{22}$$

by considering all the ratios for possible a and b and choosing the closest to the ratio of the frequencies.

There are several issues concerning these approaches, both from the technical and perceptual points of view. First, this type of measures can not be safely considered for peaks belonging to different frames, which is a strong handicap when trying to simultaneously optimize partial tracking and source formation (thus the importance of the use of texture windows as proposed in Section 3.5). The reason of this restriction is that the fundamental frequency of the source can change across frames.

Secondly, these mathematical conditions are not sufficient to determine whether two peaks are part of a harmonic source. From a perceptual point of view, the fact that two components have harmonic frequencies is not directly linked to the fact that an audible pitch is perceived. And inversely, the fact that there is an audible pitch does not imply that all of the frequencies of the spectral components of the pitched source will be in perfect harmonic relation. Thus, two peaks should be close on the "harmonic" axis if these peaks belong to a perceptible compound of harmonically-related peaks in the spectrum and not simply because their frequencies happen to have a harmonic relation (probably caused by noise or spurious spectral components from the analysis front-end). This fact perhaps explains why some sound separation algorithms first attempt to identify the pitch of the sounds within the mixture by considering the spectral information globally, and then assign frequency components to each estimated pitch (e.g. [Parsons, 1976, Weintraub, 1985, Every, 2006, Li and Wang, 2006, Li and Wang, 2007]).

In contrast, a novel similarity measure termed *Harmonically Wrapped Peak Similarity* (HWPS) is proposed in the next section and works reasonably well without estimating the underlying pitch. HWPS tries to take a global view when defining harmonic relations between time-frequency components in the signal. A comparison of the HWPS cue with some of the state of the art harmonicity measures discussed above will be presented in Chapter 4.

The HWPS Algorithm

The *Harmonically Wrapped Peak Similarity* (HWPS) is a novel criterion for computing similarity between sinusoidal peaks that are potentially harmonically related. The main goal of the HWPS measure is to take advantage of the flexibility of a harmonically-related similarity between peaks which not only considers each peak in isolation, but also takes into consideration the entire spectral information associated with the remaining peaks. This measure can be used both for peaks within the same frame and among peaks of

different frames (although with slightly different properties, as will be discussed later).

The basic mechanism behind the HWPS measure is to assign each peak a spectral pattern. The pattern captures information about the spectrum in relation to the specific peak. The degree of matching between two spectral patterns is used as a similarity measure between the two peaks thus utilizing more spectral information than just frequency of the two peaks. Additionally, HWPS takes into account the relative amplitudes of all the peaks involved in the computation, implicitly assigning more importance to the stronger peaks and reducing the impact of weak peaks usually resulting from noise or spurious components in the signal. As the spectral pattern of a peak might shift when changing frames and contains peaks belonging to multiple harmonic sources, a harmonically wrapped frequency space is used to align the two spectral patterns corresponding to the peaks. The goal is that the similarity between peaks belonging to the "same" harmonic series is higher than the similarity of peaks belonging to different harmonic series.

In the following, a formal description about the three steps involved in the HWPS algorithm is introduced, after which a more motivational discussion is presented.

STEP 1 – Shifted Spectral Pattern

The HWPS approach relies on a description of the spectral content using estimates of the frequency and amplitude of local maxima of the power spectrum, i.e. the peaks. Therefore, a given spectral pattern F_l^k is assigned to each peak p_l^k (recall that l is the peak index, and k is the frame index), defined as the set of frequencies (in Hz):

$$F_l^k = \{f_i^k, \forall i \in [1, L^k]\} \tag{23}$$

where L^k is the number of peaks in frame k.[6] A shifted version of this spectral pattern $\tilde{F}_l^k$ can be defined as follows:

$$\tilde{F}_l^k = \{\tilde{f}_i^k | \tilde{f}_i^k = f_i^k - f_l^k, \forall i \in [1, L^k]\}. \tag{24}$$

This shifted spectral pattern is essentially a shift of the set of peak frequencies such that the frequency of the peak corresponding to the pattern maps to 0 (when i is equal to l). One can easily see that two peaks of different frames modeling the same partial will have roughly similar spectral patterns under the assumption that the spectral parameters evolve

[6] Although the F_l^k definition in Equation 23 does not depend on the peak index l (i.e. according to the current spectral pattern definition, peaks in a same frame will all have a similar spectral pattern; $F_l^k = F_m^k, \forall l, m \in [1, L^k]$), it was nevertheless chosen, for the sake of generality, to keep the l index in the expression.

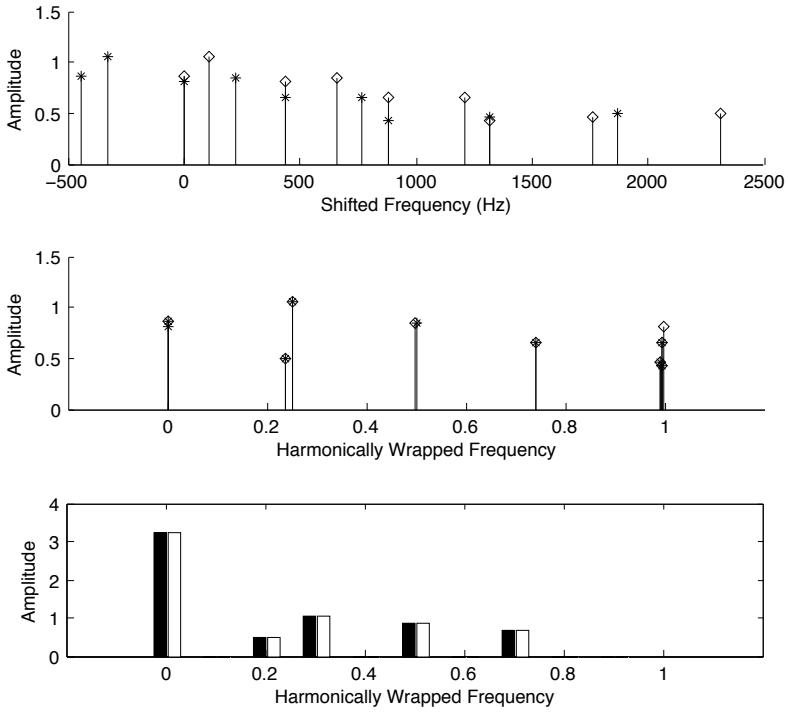


Figure 20: *HWPS calculation for peaks A_0 $\Diamond$ and A_1 $*$, from Figure 12. From top to bottom: Shifted Spectral Pattern, Harmonically-Wrapped Frequency and Histogram of Harmonically-Wrapped Frequency. Notice the high correlation between the two histograms at the bottom of the Figure.*

slowly with time. This spectral pattern forms a peak-specific view of the spectral content which is used to calculate a pitch invariant representation using a wrapped frequency space as described in the following step. The top graphs of Figures 20 and 21 show overlaid peak-specific spectral patterns for two pairs of peaks from the harmonic mixture depicted in Figure 12 and defined as in Table 1.

STEP 2 – Wrapped Frequency Space

To estimate whether two peaks p_l^k and p_m^{k+n} belong to the same harmonic source, the proposal is to measure the correlation between the two shifted spectral patterns corresponding to the peaks. To achieve this, it would be helpful to find a way to transform the peak-specific shifted spectral patterns in such a way that when the peaks under consideration belong to the same harmonic series the correlation is higher than when they belong to different harmonic sources. In order to achieve this the following operations are performed: the energy distribution of a harmonic source along the frequency axis can be seen as a cyclic unfolding with periodicity equal to the fundamental frequency of the

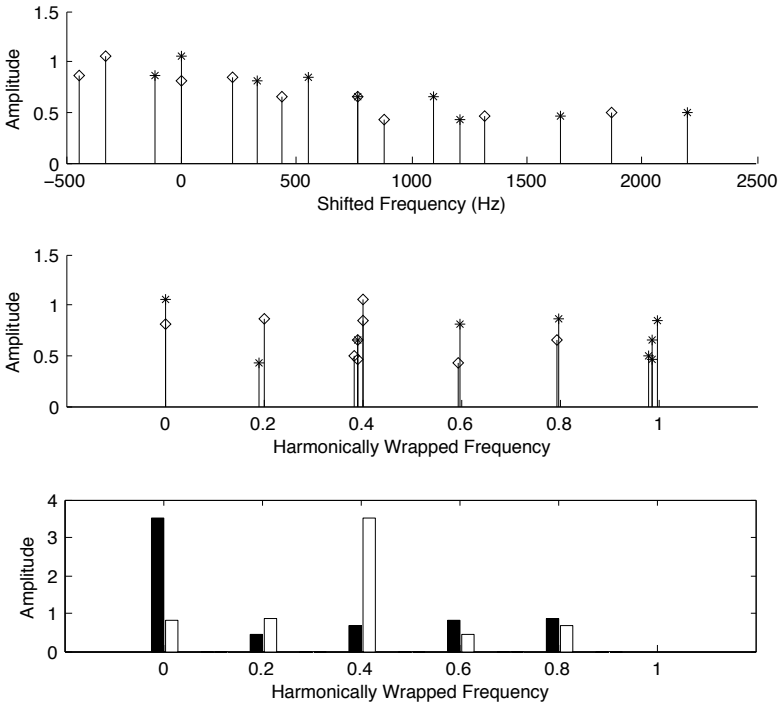


Figure 21: *HWPS calculation for peaks A_1 $\Diamond$ and B_1 $*$, from Figure 12. From top to bottom: Shifted Spectral Pattern, Harmonically-Wrapped Frequency and Histogram of Harmonically-Wrapped Frequency. Notice the lack of correlation between the two histograms at the bottom of the Figure.*

"source". To concentrate these energies as much as possible before correlating them, it is proposed to wrap the frequencies of each spectral pattern as follows:

$$\hat{f}_i^k = \mathrm{mod}\left(\frac{\tilde{f}_i^k}{h}, 1\right) \tag{25}$$

where h is the wrapping frequency function and mod is the real modulo function.[7]

This wrapping operation would be perfect with the prior knowledge of the fundamental frequency. With this knowledge it would be possible to parametrize the wrapping operation h as:

$$h = \min(f_{0l}^k, f_{0m}^{k+n}) \tag{26}$$

where f_{0l}^k is the fundamental frequency of the source of the peak p_l^k. Without such prior,

[7]The real modulo function is inhere defined as a modulo operation according to the following conditions: if taking $D = q \times d + r$, where D is the dividend, d is the divisor, $q \in \mathbb{N}$ is the quotient, and the returned remainder $r \in \mathbb{R}$ is in the range $0 < r < d$. E.g. mod(5.3, 2) = 1.3 or mod(5.3, 1) = 0.3.

a conservative approach h' is considered instead, although it will tend to over estimate the fundamental frequency (as will be discussed later in this text):

$$h' = \min(f_l^k, f_m^{k+n}). \tag{27}$$

Notice that the value of the wrapping frequency function h is the same for both patterns corresponding to the peaks under consideration. Therefore the resulting shifted and wrapped frequency pattern will be more similar if the peaks belong to the same harmonic "source". The resulting shifted and wrapped patterns are pitch invariant and can be seen in the middle plot of Figures 20 and 21.

Different approaches could have been taken for the definition of the fundamental frequency estimation function h'. One possibility would be to select the highest amplitude peak in the union of the two spectral patterns under consideration as the f_0 estimate (i.e. $h' = \{f_i | i = argmax_i(A_i)\}, \forall i \in [1, \#A]$, where $A = A_l^k \cup A_m^{k+n}$, $\#A$ is its number of elements and A_l^k is the set of amplitudes corresponding to the spectral pattern F_l^k). The motivation for this approach is the fact that the highest amplitude partial in musical signals often corresponds to the fundamental frequency of the most prominent harmonic 'source" active in that frame, although this assumption will not always hold.

A more robust approach, though more computationally expensive, would be to calculate all the frequency differences between all peaks in each spectral pattern and compute a histogram. The peaks in these histograms would be good candidates for the fundamental frequencies in each frame (in order to avoid octave ambiguities, a second histogram with the differences between all the candidate f_0 values could be again computed, where the highest peaks would be selected as the final f_0 candidates). The HWPS could then be iteratively calculated using each f_0 candidate in this short list, and select the one with the best value as the final choice. In fact, this technique could prove an interesting way to robustly estimate the number of harmonic "sources" in each frame, including their pitches, but experimental evaluations are still required to validate these approaches.

STEP 3 – Discrete Cosine Similarity

The last step is now to correlate the two shifted and harmonically wrapped spectral patterns ($\hat{F}_l^k$ and $\hat{F}_m^{k+n}$) to obtain the HWPS measure between the two corresponding peaks. The proposal is to discretize each shifted and harmonically wrapped spectral pattern into an amplitude weighted histogram, H_l^k, corresponding to each spectral pattern $\hat{F}_l^k$. The contribution of each peak to the histogram is equal to its amplitude and the range between 0 and 1 of the Harmonically-Wrapped Frequency is divided into 20 equal-size bins

(preliminary tests conducted during experimental evaluation of the system, described in Chapter 4, have shown that the use of 20 bin histograms resulted in the best segregation results for the specific applications at hand).

In addition, the harmonically wrapped spectral patterns are also folded into a a pitch-invariant profile. For example, in Figure 20, the energy of the spectral pattern in wrapped frequency 1 (all integer multiples of the wrapping frequency) is mapped to histogram bin 0.

The HWPS similarity between the peaks p_l^k and p_m^{k+n} is then defined based on the cosine distance between the two corresponding discretized histograms as follows:

$$W_h(p_l^k, p_m^{k+n}) = \text{HWPS}(p_l^k, p_m^{k+n}) = e^{-\left(1-\frac{c(H_l^k, H_m^{k+n})}{\sqrt{c(H_l^k, H_l^k) \times c(H_m^{k+n}, H_m^{k+n})}}\right)^2} \quad (28)$$

where c is the dot product between two vectors (which correspond to the histograms in this particular case) and is defined as:

$$c(H_a^b, H_c^d) = \sum_i H_a^b(i) \times H_c^d(i). \quad (29)$$

The use of the cosine distance (which is bounded to the interval $[0, 1]$) allows to put the emphasis on the alignment of the two histograms while reducing the impact of any existing scale differences (i.e. if the two histograms have a similar shape, their distance will be small, regardless of the absolute occurrence values at each bin). A Gaussian function is then once again used to convert the cosine distance into a similarity value, where the neighborhood width of the harmonicity similarity cue can be controlled by means of its σ_h parameter (a $\sigma_h = 1$ is implicitly used in the current system implementation, as will be discussed in Section 3.7.2).

Similarly to the cases of amplitude and frequency similarity functions, peaks with a low HWPS distance will have a high HWPS value, while peaks far apart in the harmonicity distance space will end up with a low HWPS value. One may also notice that due to the wrapping operation of Equation 25, the size of the histograms can be relatively small (e.g. 20 bins), thus being computationally inexpensive.

An Intuitive View on HWPS

The three steps that compose the HWPS algorithm presented in the previous section can be more intuitively understood if looking at the following example scenarios.

As a first example, imagine that in a frame k there are six peaks from two harmonic

series (i.e. harmonic "sources") A and B, so that $\{A_0, A_1, A_2\} \in A$ and $\{B_0, B_1, B_2\} \in B$ (see Figure 22). Assume for now and for the sake of clarity that their fundamental frequencies are somehow known and have values f_{0_A}, f_{0_B}, respectively. In practice the true fundamental frequencies of the harmonic series in each frame are not known, and their values will have to be estimated (as discussed in the previous section while describing the second step in the HWPS algorithm). A later discussion in this section will show that although the use of f_0 estimates may decrease the HWPS accuracy due to estimation errors, its impact is limited and still allows to compute the harmonic similarity between peaks with an acceptable accuracy.

Suppose now that the objective is to compute the HPWS between two peaks in frame k, and that the two peaks selected are harmonically related, say A_1 and A_0. This implies that the peaks will have to necessarily belong to a same harmonic series, in this case A. In such a scenario, HWPS(A_1, A_0) is expected to result high valued, as will be shown in the following. Under these conditions, the two peaks would have frequencies $f^k_{A_1} = a \times f_{0_A}$ and $f^k_{A_0} = b \times f_{0_A}$, where a and b are integer numbers, which for this particular example assume the values 2 and 1, respectively. Additionally, because the two peaks come from the same frame k, they will have the same spectral pattern, i.e. $F^k_{A_1} = F^k_{A_0}$ (see the top panel in Figure 22).

For visualization's sake, assume that the spectral patterns $F^k_{A_1}, F^k_{A_0}$ of each peak are now represented along helixes (named $\bar{F}^k_{A_1}$ and $\bar{F}^k_{A_0}$ in Figure 22), and that their "spiraling" periodicity is set accordingly to the idealized wrapping operation defined in Equation 26 $h = f_{0_A}$ (i.e. the fundamental frequency of the harmonic series A). Since the two peaks are from the same frame, their helix-shaped spectral patterns $\bar{F}^k_{A_1}$ and $\bar{F}^k_{A_0}$ are also equal.

As previously explained, the first step in the HWPS algorithm corresponds to performing a shifting operation in each peak's spectral pattern, where all the frequency values in $F^k_{A_1}$ are subtracted by the frequency of peak A_1 (i.e. $f^k_{A_1} = 2f_{0_A}$) and all the frequencies in $F^k_{A_0}$ are subtracted by the frequency of peak A_0 (i.e. $f^k_{A_0} = f_{0_A}$). This results in two wrapped and shifted spectral patterns $\tilde{F}^k_{A_1}$ and $\tilde{F}^k_{A_0}$, which are no longer equal (see the second and fourth helixes depicted in Figure 22). Intuitively, this shifting operation can be understood as "spiraling" down all the peaks in the helixes $\bar{F}^k_{A_1}$ and $\bar{F}^k_{A_0}$ by the same amounts, $f^k_{A_1}, f^k_{A_0}$, respectively (see the shifting operations depicted in Figure 22). But because in this example both $f^k_{A_1}$ and $f^k_{A_0}$ are integer multiples of the "spiraling" periodicity of the helixes (i.e. $h = f_{0_A}$), the result is that all peaks will be moved down in each helix a different but integer number of turns (i.e. "twists").

In fact, the key aspect in this shifting operation is not the absolute shifting applied to

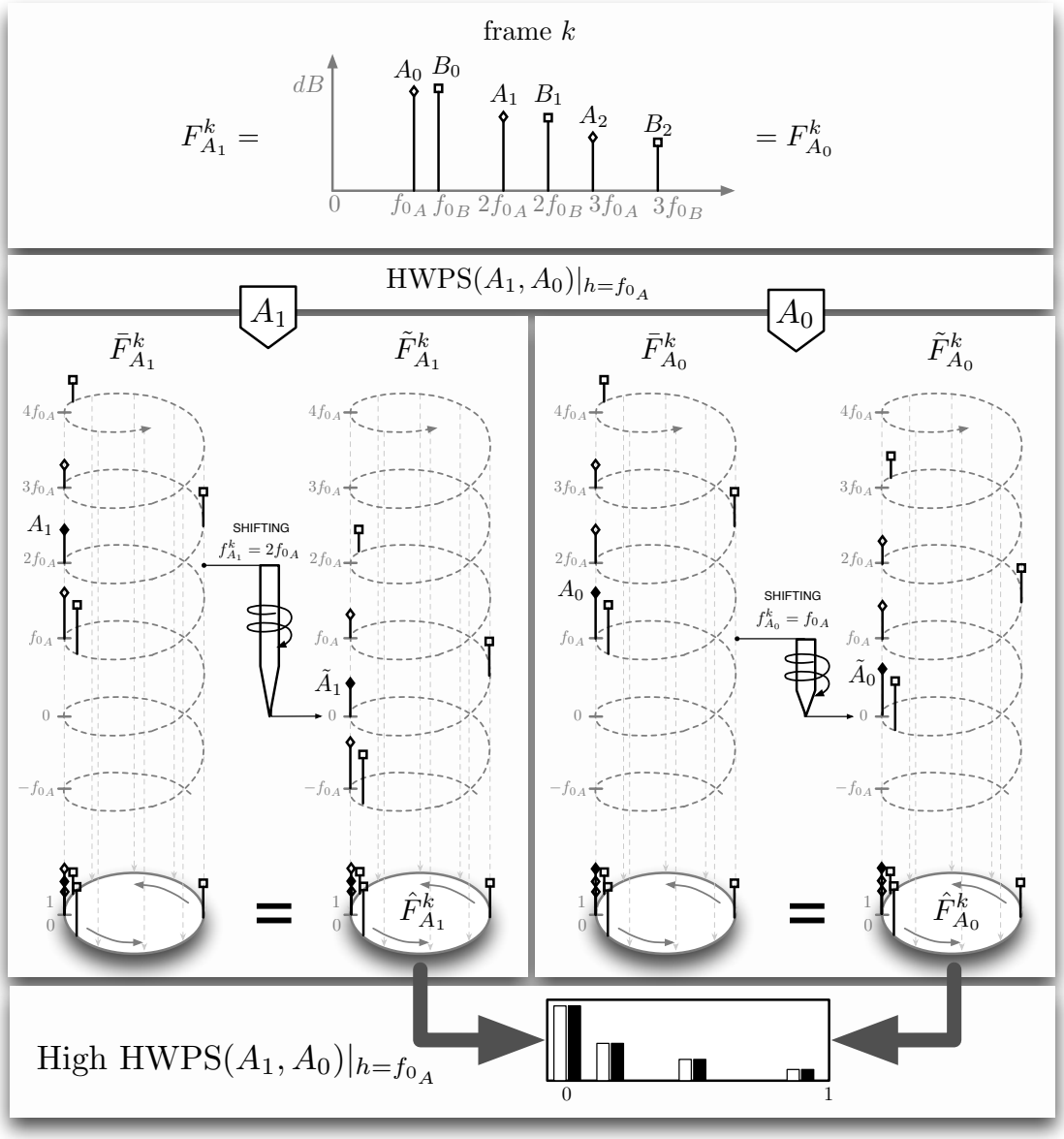


Figure 22: *Example of the HWPS computation for two harmonically related peaks, A_1 and A_0, from a same frame k, and assuming an ideal wrapping function h. The top panel shows the illustrative spectrum representation used in this example and the helixes represent the shifting and wrapping steps performed by the HWPS algorithm for each peak (see text for details). The bottom panel shows the resulting combined histograms, where the white bars refer to the histogram $\hat{F}^k_{A_1}$ and the black bars refer to $\hat{F}^k_{A_0}$. Given the similarity between the two histograms, the HWPS value will result high, as expected for two harmonically related peaks.*

each spectral pattern, but instead whether the shifting is an integer multiple of the helix periodicity h. A shift corresponding to an integer number of turns will mean that the relative position of each peak around the helix (i.e. discarding their "height" in the helix) will still be the same. This is similar to the chroma and octave notions in music scales, where a note when moved up or down by an integer number of octaves does not change its chroma.

If the helixes corresponding to $\tilde{F}^k_{A_1}$ and $\tilde{F}^k_{A_0}$ are then projected down vertically (i.e. flattened), the end result will be two wrapped spectral patterns, $\hat{F}^k_{A_1}$, $\hat{F}^k_{A_0}$ (see the circles at the bottom of each helix in Figure 22), where all the frequency values are wrapped to the range $[0, 1[$ (this corresponds to the second step of the HWPS algorithm). This wrapped representation will result unaltered as long as the shifting applied to each helix is a multiple integer of the helix periodicity h, as is the case of the peaks A_1, A_0.

Consequently, if the wrapped representations corresponding to each peak under evaluation, $\hat{F}^k_{A_1}$, $\hat{F}^k_{A_0}$, are used to compute discretized histograms (where the magnitudes of all peaks are stacked into the respective bins – this is where the magnitudes of each peak come into play in the HWPS), it becomes possible to compute a distance between the two histograms (e.g. the cosine distance, as defined in Equation 28 and used in the third step of the HWPS algorithm). Accordingly, for peaks that belong to a same harmonic series, their histograms will be very similar, resulting in an anticipated high harmonicity value (i.e. an HWPS close to 1.0 – see bottom panel of Figure 22).

However, the same will not happen if the two peaks under HWPS evaluation do not belong to a same harmonic "source". As can be seen in Figure 23, in this case the HWPS is being computed for the peaks A_1 and B_0, which clearly do not belong to the same harmonic source. Because the two peaks still come from the same frame k, and the ideal wrapping factor h is still equal to f_{0_A}, the two peaks share the same spectral patterns (i.e. $F^k_{A_1} = F^k_{B_0}$) and their helix-shaped representations are still similar (i.e. $\bar{F}^k_{A_1} = \bar{F}^k_{B_0}$). However, the shifting operation will no longer result into an integer number of turns down the helix for the case of B_0, causing a loss of alignment between the peaks in $\hat{F}^k_{A_1}$ and $\hat{F}^k_{B_0}$. As desired, this will result in quite different histogram representations for each peak and by consequence into a substantially smaller HWPS value for peaks that do not have a harmonic relation (see bottom panel of Figure 23).

In summary, for peaks in a same frame that have a harmonic relation, the shifting operation in the HWPS algorithm will not affect the final shifted and wrapped spectral pattern, $\hat{F}^k_l$, resulting in a high HWPS value. On the other hand, peaks in a same frame whose frequencies are not harmonically related will have substantially different shifted and

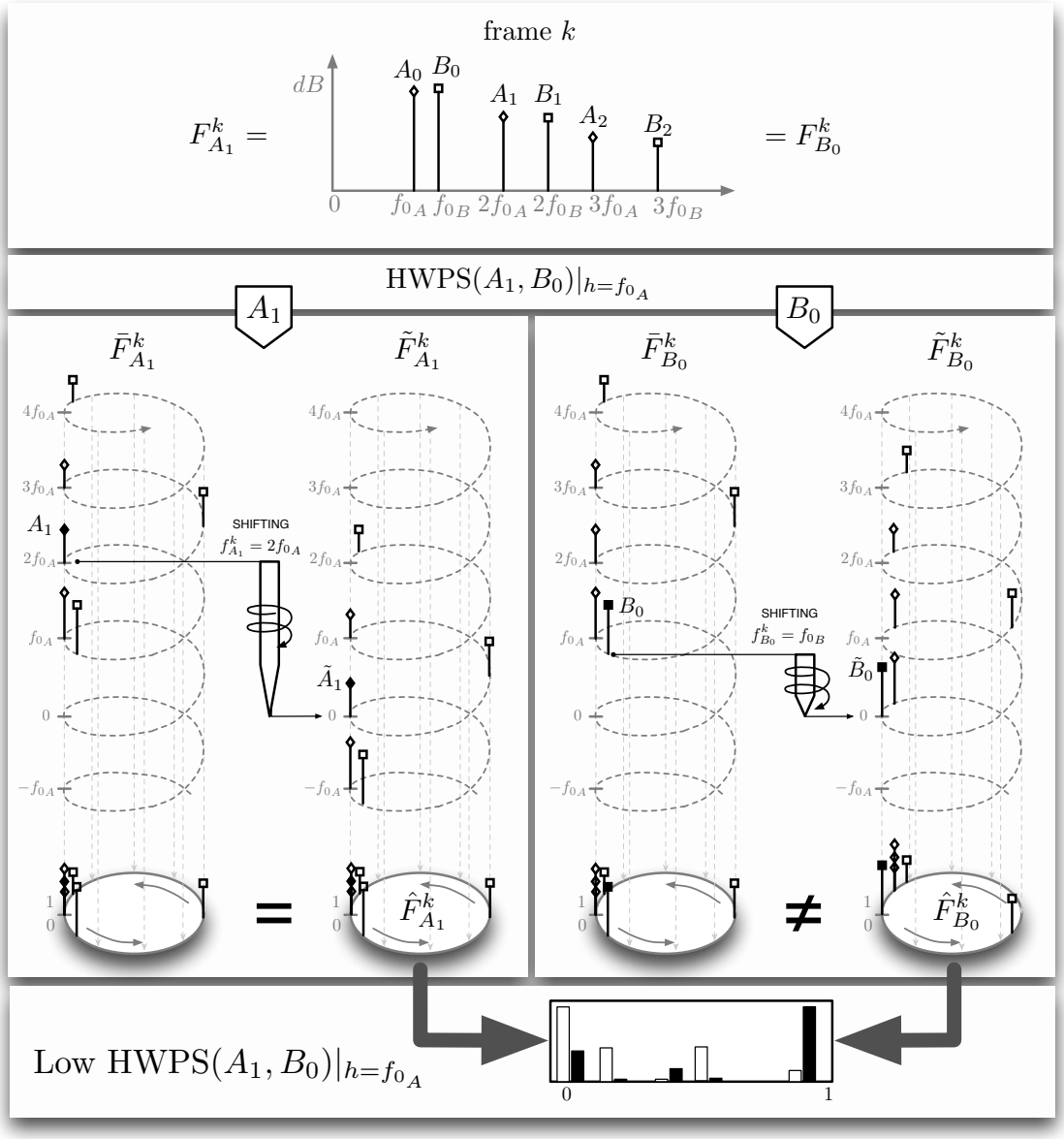


Figure 23: *Example of the HWPS computation for two non-harmonically related peaks, A_1 and B_0, from a same frame k, and assuming an ideal wrapping function h. The top panel shows the illustrative spectrum representation used in this example and the helixes represent the shifting and wrapping steps performed by the HWPS algorithm for each peak (see text for details). The bottom panel shows the resulting combined histograms, where the white bars refer to the histogram $\hat{F}^k_{A_1}$ and the black bars refer to $\hat{F}^k_{B_0}$. Given the difference between the two histograms, the HWPS value will result low, as expected for two non-harmonically related peaks.*

wrapped spectral patterns, resulting in a low HWPS value.

Until now it has been assumed that the two peaks under HWPS evaluation belong to the same frame. However, when taking peaks from two distinct frames there is some probability that the two corresponding spectral patterns are substantially different. This will happen more often for frames that are far apart in time, (e.g. due to different sources or notes occurring at each time instant) since it is assumed that, because spectral parameters evolve slowly with time, frames close in time will have similar spectral patterns.

Figure 24 depicts an illustrative scenario, where the peaks in two frames, k and $k+n$ (which are arbitrarily separated in time by n frames) are not exactly the same (see top panel of the figure). In the first frame k, two harmonic series A and B, similar to the ones used in the two previous examples, coexist. In a later frame, $k+n$, the harmonic "source" A is still active, but B is no longer present. Instead, a new harmonic series C, with a different fundamental frequency f_{0_C} is now mixed with A.

In this example the objective is to compute the HWPS between two peaks, A_1^k and A_0^{k+n}, from a same harmonic series but now from different frames. Although being from different instants in time, their HWPS should nevertheless be high valued since they still have a harmonic relation.

Under these conditions, it is easy to see that their spectral patterns, $F_{A_1}^k$ and $F_{A_0}^{k+n}$, are no longer the same, and consequently their helix representations $\bar{F}_{A_1}^k$ and $\bar{F}_{A_0}^{k+n}$ are also different (see Figure 24). Notice however that the idealized wrapping function h still results in the same f_{0_A} value for the peaks selected for this example (recall Equation 26).

Looking at the shifted spectral patterns of each peak, $\tilde{F}_{A_1}^k, \tilde{F}_{A_0}^{k+n}$, it becomes clear that all peaks from the harmonic series A are aligned in similar positions in the helixes of each peak. The difference when compared to the example given in Figure 22 (where the HWPS was also computed for peaks A_1 and A_0, though they were then from a same frame) is that now the remaining peaks are not the same in each spectral pattern (as they were in Figure 22). The end result of this difference is that the histograms will exhibit a high similarity for bin 0 (which corresponds to the aligned and stacked peaks from "source" A), but the remaining bins will be substantially different since they result from different sources at each frame (i.e. B and C). However, because the discretized histogram stacks the individual amplitudes of the peaks closely aligned on the wrapped representations, $\tilde{F}_{A_1}^k, \tilde{F}_{A_0}^{k+n}$, the values of the bins corresponding to the peaks from B and C (which are not aligned and therefore not stacked together into a same bin) can be neglected when compared to the bin stacking the peaks from "source" A (i.e. bin 0). As a result, the HWPS between harmonically related peaks, even if in different frames, is still high valued,

even if slightly lower than when considering similar peaks form the same frame.

In fact, the influence of different spectral components at each frame (i.e. the "interference") has a direct impact on the HWPS computation. One way to minimize this "interference" could be to modify the current HWPS implementation by just taking the bin 0 of the histograms when computing their distances. Looking at the histograms at the bottom of Figures 22, 23 and 24 it becomes clear that when the two peaks under HWPS evaluation have a harmonic relation (e.g. HWPS(A_1, A_0), HWPS(A_1^k, A_0^{k+n})) the bin 0 of their histograms will have similarly high values, while they present quite different values otherwise (e.g. HWPS(A_1, B_0)). As mentioned previously, the bins other than the 0 bin in the histograms refer to the peaks coming from the other "interfering sources", and as a result they do not directly contribute to the goal of obtaining a precise harmonicity measure between two peaks. Given so, measuring the distance between the bins 0 of the histograms could hypothetically provide a more robust harmonicity measure for peaks in frames far apart in time. Additionally, such a distance could be implemented using an Euclidean distance, which is more computationally efficient than the cosine distance used in the current implementation of the HWPS, simplifying at the same time its combination with the other Euclidean-based similarity cues (as will be discussed in Section 3.7.3). However, this idea still requires experimental evaluation, and as a result, the system proposed in this thesis and all the experiments presented in Chapter 4 are based on the computation of the cosine distance between the entire discretized histograms.

Finally, and regarding the impact of the use of estimated fundamental frequencies instead of the true ones in the HWPS computation, consider now the case of the mixture of two pitched sound sources, A and B, each consisting of four harmonics with fundamental frequencies of 440 Hz and 550 Hz respectively, as presented in Table 1 and depicted in Figure 12. For these experiments, random frequency deviations of a maximum of 5 Hz are added to test the resilience of the algorithm to frequency estimation errors. Once again, if considering two peaks of the same source A_0 and A_1, the discrete version of the harmonically-wrapped sets of peaks are highly correlated, as can be seen in the histograms at the bottom of Figure 20. On the other hand, if two peaks of different sources, A_1 and B_0, are considered, the correlation between the two discretized histograms is low (see Figure 21).

However, if instead of using the true fundamental f_0, any harmonic of it is used as the wrapping frequency (as proposed for the conservative wrapping function h', defined in Equation 27), the correlation between two histograms of harmonically related peaks will still work, though to a lesser extent. Figure 25 (left) shows a HWPS similarity

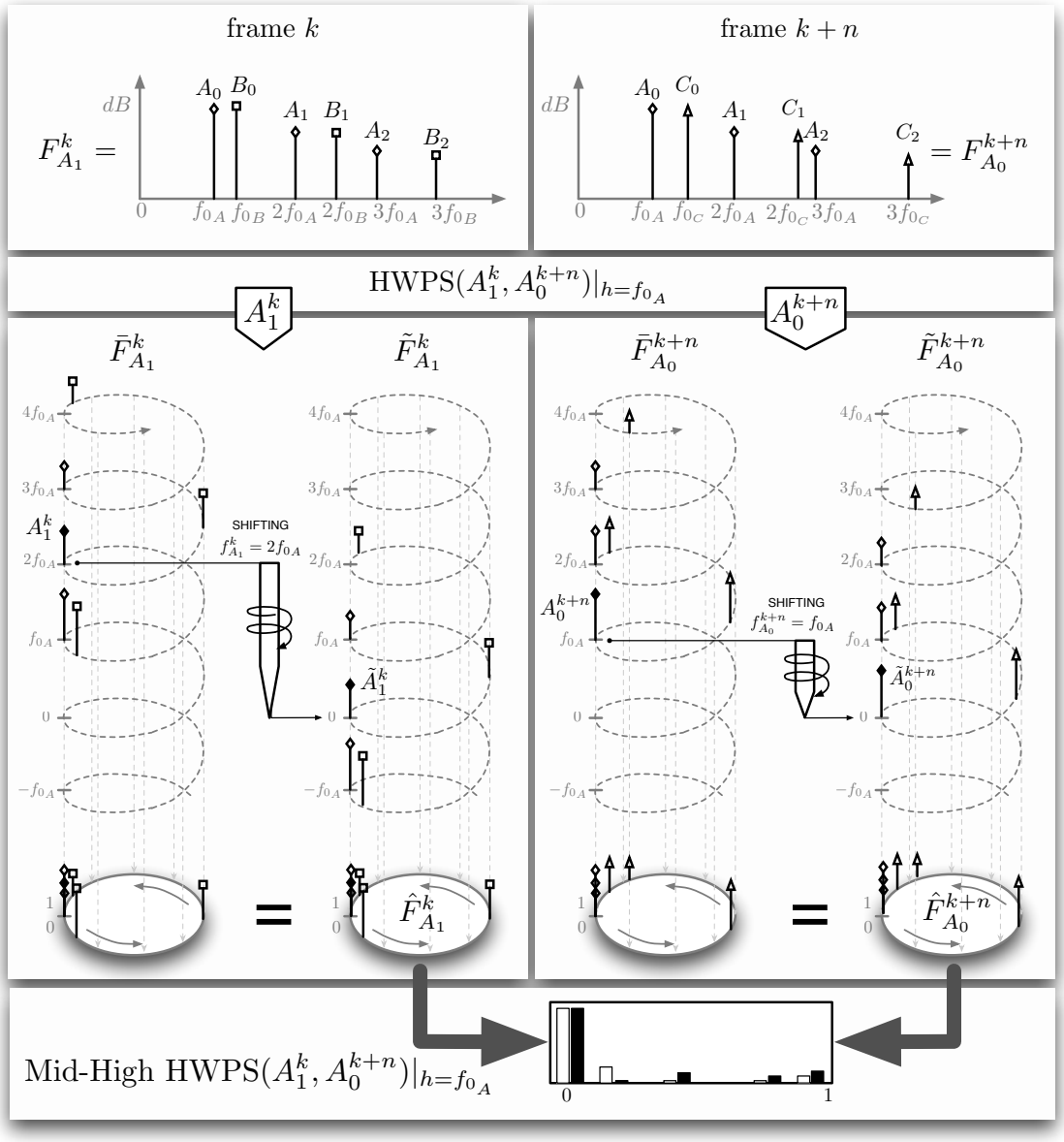


Figure 24: *Example of the HWPS computation for two harmonically related peaks, A^k_1 and A^{k+n}_0, from two different frames k and $k+n$, and assuming an ideal wrapping function h. The top panel shows the illustrative spectrum representations for each frame as used in this example. The helixes represent the shifting and wrapping steps performed by the HWPS algorithm for each peak (see text for details). The bottom panel shows the resulting combined histograms, where the white bars refer to the histogram $\hat{F}^k_{A_1}$ and the black bars refer to $\hat{F}^{k+n}_{A_0}$. The similarity between the two histograms is basically given by the high and similar values in bin 0, which correspond to the aligned peaks from the harmonic series A. Given the low values of the remaining bins, their impact into the HWPS result will be negligible. As a result, the HWPS will still result high valued, although smaller than the value obtained for similar peaks when in the same frame (see Figure 22).*

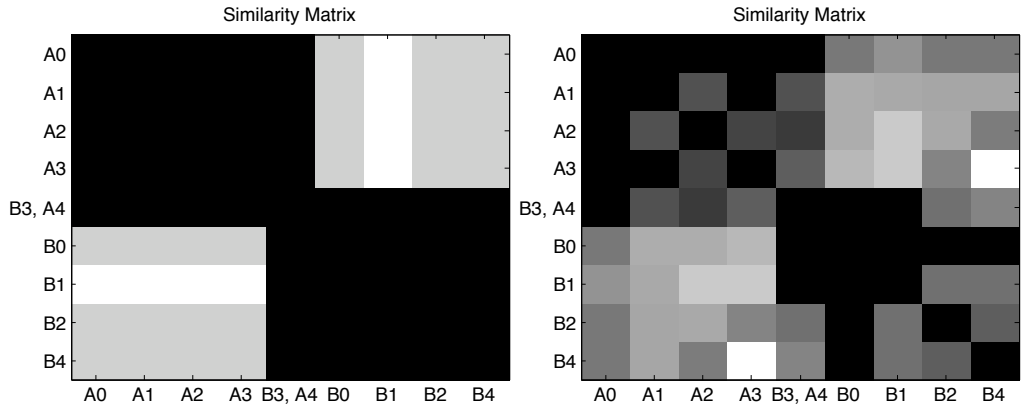


Figure 25: *Harmonically-Wrapped Peak Similarity (HWPS) matrix for two harmonic sources using the correct f_0 estimates (left), and using the conservative estimate of wrapping frequency (likely a harmonic of the "true" f_0) (right). High similarity values are mapped to black and low similarity values to white.*

matrix computed among the peaks of two overlapping harmonic sounds within a frame (as shown in Figure 12) with perfect knowledge of the fundamental frequency for each peak respectively. As can be seen clearly from the figure, the similarity is high for pairs of peaks belonging to the same source and low for pairs belonging to different sources. Figure 25 (right) shows the HWPS similarity matrix computed among the peaks of the same two overlapping harmonic sounds within a frame using the conservative approach to estimate the wrapping frequency (i.e. h', where basically the lower peak is considered as the "wrapping" frequency). As can be seen from the figure, although the similarity matrix on the right is not as clearly defined as the one on the left, it still clearly shows higher values for pairs of peaks belonging to the same sound source. Other approaches for estimating the "pitches" present in each frame would allow better candidates to the fundamental frequencies, potentially improving the current HWPS accuracy. However, such approaches still need experimental evaluation.

Using HWPS for Sound Segregation

Comparably to the examples given for the sole use of the amplitude and frequency similarities, W_a and W_f, in sound segregation (see Section 3.7.1), this section will present some illustrative scenarios where the results from clustering of sound components based on just using the HWPS cue will be discussed.

Plots d, h, l and p of Figure 15 depict the HWPS distances (i.e. the cosine distance, as defined in the exponent of Equation 28) for the four sound examples used in the previous

sections (see Section 3.7.1 for detailed descriptions on these signals).

For the case of the *Tone A* example, the HWPS distance histogram has two peaks. The first at bin 0 represents, as expected, the harmonically related peaks in the signal, for which HWPS correctly returned a null distance. In fact, in this particular example, all peaks are known to be harmonically related (recall Table 1) and therefore they all should be represented in bin 0. However, the histogram exhibits a second peak around 0.1. This peak results from the use of a conservative fundamental frequency estimation h' instead of the true f_0 value given by the ideal function h (as defined in Section 3.7.2 and Equations 26 and 27). As discussed in the previous section, the current system implementation computes the HWPS using the conservative f_0 estimation h', which for high order harmonics provides poor estimates, hurting to some extent the final HWPS precision. Similar observations can be drawn for the *Tones A+B* example (see plot *h*), where three peaks appear in the HWPS distance histogram instead of the expected two.

The HWPS distance histograms for the two real-world examples (plots *l, p*) present a similarly shaped distribution, and contrary to the cases of amplitude and frequency distances, most peaks do not exhibit a harmonic relation. This is not surprising since in a real-world music signal, although there is a highly predominant harmonic content in the signals, there are still a lot of spectral components produced from unpitched sounds (e.g. drums, percussion) or from the transient parts of pitched sounds (i.e. the attacks) or from other noisy events in the signal.

Plot *f* in Figures 16 and 17 shows the W_h similarity function histogram for the two example sound files, after applying the Gaussian function, as defined in Equation 28. Similarly to the cases of amplitude and frequency similarity functions, peaks with a low HWPS distance will have a high HWPS value, while peaks far apart in the harmonicity distance space will end up with a low HWPS value.

Because the HWPS computation is based on the cosine distance, which is bounded to the interval $[0, 1]$, there is no need to perform any type of normalization to the distance values. Furthermore, and as previously mentioned, the HWPS calculation uses an implicit neighborhood value $\sigma_h = 1.0$ (represented as the dashed gray lines in plots *e* of Figures 16 and 17), which was found satisfactory during the experimental evaluations presented in Chapter 4. Nevertheless, similarly to the amplitude and frequency cases, it could be more adequate and perceptually motivated to set σ_h based on prior knowledge about JNDs for harmonicity [Gerald Kidd et al., 2003]. Given the range of values output by the cosine distance, another option would be to set σ_h to some value around 0.4, which would smoothly divide the range of distances between similar and dissimilar values (see

Figure 11). In any case, comprehensive experimental evaluation of such approaches would be necessary to be conducted in order to validate their adequacy.

Plots *f,g* in Figures 18 and 19 show the segregation results when only using the HWPS similarity function W_h. In the case of the *Tones A+B* example, it is clear that the HWPS enabled the segregation algorithm to correctly separate the two harmonic sets, A and B (note that the shared harmonic $\{A_4, B_3\}$ was grouped with the components from harmonic set B).

For the real-world example *Jazz1*, it is also clear that the HWPS allowed the grouping algorithm to successfully group the most predominant harmonic content in the mixture into a single cluster (see plot f), separating it from the harmonic background and more noisy components in the sound mixture (see panel g)[8].

Section 4.2.1 will present additional experimental evaluations as well as a comparative study including some of the most well known approaches for the computation of harmonicity similarity. Additionally, Appendix D describes the use of the HWPS as the basis of a computationally efficient scheme for the dominant harmonic source separation, further demonstrating the ability of this novel similarity cue to correctly identify the harmonic content in complex sound mixtures.

3.7.3 Combining Similarity Functions

The use of a single similarity cue for sound segregation has been presented in Sections 3.7.1 and 3.7.2, and the results have shown that each cue allows to obtain a different representation of the complex audio mixture. As a result, each cue by itself only provides a limited ability to represent the different sound events in a complex mixture. Therefore, the combination of different similarity cues could allow to make the best use of their isolated grouping abilities towards a more meaningful segregation of a sound mixture.

Following the work of Shi and Malik [Shi and Malik, 2000], who proposed to compute the overall similarity function as the product of the individual similarity cues used for image segmentation, the current system combines the amplitude, frequency and HWPS grouping cues presented in the previous sections into a combined similarity function W as follows:

$$W(p_l, p_m) = W_{afh}(p_l, p_m) = W_a(p_l, p_m) \times W_f(p_l, p_m) \times W_h(p_l, p_m). \quad (30)$$

[8] Audio clips of the signals plotted in Figures 18 and 19 are available at `http://www.fe.up.pt/~lgustavo`

Plots *g* in Figures 16 and 17 show the histogram of the values resulting from the combined similarity functions for the two sound examples, *Tones A+B* and *Jazz1*, respectively. In both cases, it becomes clear that the HWPS cue contributes to the reduction of the number of similar peaks (notice the peak around bins 0.7 and 0.5 in plots *g* in Figures 16 and 17, respectively), which would be more likely to be considered as similar if only looking at their frequency and amplitude similarities.

Plots *h, i* in Figures 18 and 19 show the bipartite clusters for the two example signals *Tones A+B* and *Jazz1* as used in previous illustrative scenarios. For the case of the simpler example *Tones A+B*, the clustering results of the combined use of the three grouping cues is equal to the one obtained when just using the HWPS cue (compare plots *f,g* and *h,i* in Figure 18). This means that the harmonicity cue was, for specific example, sufficiently strong to overcome the amplitude and frequency cues.

On the other hand, in the case of the more complex real-world signal *Jazz1*, the clusters obtained from the use of the combined similarity function (see plots *h, i* in Figure 19) seem to take a strong influence from the HWPS cue (see plots *f,g*), but also from the amplitude similarity cues (see plots *b,c*). Still, comparing the results of the combined cues in plot *h*, with the ones obtained using the HWPS cue alone (plot *f*), there seems to exist some degradation, probably resulting from the impact of the frequency cue, which influences clustering into two non-overlaping frequency bands (see plots *d,e*). Although these segregation differences are clearly noticeable to the eye in the spectrogram representation, they are hard to identify when hearing the actual audio signals[9], which once again shows that the evaluation of segregation results is a hard and ill-posed problem.

Actually, this raises some questions about the way the similarity functions are currently being combined into the overall similarity W_{afh}. Because the combination is performed by taking the product of the corresponding values of each similarity cue, this means that if two peaks are very dissimilar regarding any of the similarity cues (i.e. their similarity value is zero or close to zero) their overall similarity will also be very low, regardless of the values of all the remaining cues. This makes the specification of the neighbourhood width parameters σ in the similarity Gaussian functions critical, since it will control how fast each cue will get values close to zero, and consequently outclassing the impact of all the remaining cues.

Given the difficulty to specify a meaningful value for each similarity's σ (recall the discussion in Sections 3.6.2, 3.7.1 and 3.7.2), and the fact that the HWPS values are

[9]Audio clips of the signals plotted in Figures 18 and 19 are available at `http://www.fe.up.pt/~lgustavo`

intrinsically bounded to the range $[0, 1]$, the current system implementation attempts to simplify this problem by first normalizing all distance values to the interval $[0, 1]$ and then using a fixed and empirically derived value for each neighbourhood width (i.e. $\sigma_a = \sigma_f = 1.2$, $\sigma_h = 1.0$, which allowed to achieve satisfactory results in the experiments described in Chapter 4). These fixed σ values can also be seen as weighting factor for each similarity cue, but, as previously discussed, the use of prior knowledge could motivate a more appropriate specification of these values.

Furthermore, although this fixed logical "AND" combination of similarity cues may be appropriate for some cases (e.g. if trying to segregate the strongest harmonic components in a complex signal, two peaks would only be considered similar if their harmonicity "AND" amplitude proximities were high) it may be too restrictive for expressing more perceptually motivated combination schemes. For instance, if trying to segregate peaks that are close in amplitude "AND" frequency, "OR" harmonically related, but (i.e."AND") that are placed in the same azimuth in a stereo signal, the corresponding logical expression of the similarity cues combination could end up being something like $W_{afh} = [(W_f \wedge W_a) \vee W_h] \wedge W_s$ (where W_s is some stereo azimuth proximity cue). The logical "AND" and "OR" operators could be easily implemented by means of product and sum operators, or minimum or maximum expressions, respectively.

However, all these hypothesis require experimental validation, and are subject for future work.

3.8 Sound Event Resynthesis

The last blocks of the framework proposed in this thesis (see Figure 6) are the ones responsible for selecting and resynthesizing the segregated clusters as independent audio signals.

Although it is commonly accepted that the human auditory system does not resynthesize the sounds it represents internally, the ability to achieve perceptual invertibility may be considered conceptually equivalent to a representation that captures all the relevant information [Ellis and Rosenthal, 1995]. Indeed, the resynthesis of the segregated sounds is important when attempting to inspect and evaluate the segregation results, both by human listening or by some objective evaluation metrics (see the discussion in Section 2.4.5 and the experiments in Chapter 4). Furthermore, tractable inversion schemes are mandatory for some CASA applications, such as advanced hearing prostheses or auditory scene reconstruction.

The following sections will focus on the approach taken for the selection and subsequent resynthesis of the sound events segregated from an audio mixture.

3.8.1 Cluster Selection

The work proposed in this thesis is a generic framework for sound event segregation in complex music signals, and therefore aims at identifying and segregating the main musical sound events in a polyphonic mixture, mostly inspired by the way a naive listener would do (recall the discussion in Section 2.3.2). However, even if able to effectively identify and resynthesize all the perceptually relevant sound events occurring at each time instant in a music signal (i.e. corresponding to some texture window), the problem of how to sequentially group them over time remains (e.g. how could the system only resynthesize the sound events belonging to a saxophone playing the main melody or to the main singing voice).

While the current framework implementation already includes some sequential grouping principles at the texture window level (achieved mainly by frequency and amplitude proximity cues, as described in Section 3.7.1), it does not yet fully incorporate sequential grouping among sound clusters detected in different texture windows. In fact, the same frequency and amplitude grouping cues, added to timbre and smoothness cues among others, could be used in the future to provide such a sequential integration following the principles proposed by Bregman [Bregman, 1990] and discussed in Section 2.3.1.

Although not yet fully implementing sequential grouping, and given the relevance of the predominant melodic source in popular western music, the current implementation of the framework is already able to attempt to segregate the dominant melodic source among all other musical events playing together in a polyphonic music signal (e.g. try to segregate the singing voice from the accompaniment instruments, such as bass, piano, drums and others). The approach used is general and straightforward and it does not rely on any prior knowledge of the predominant source (e.g. singing voice, specific music instruments).

The approach implemented in the current system starts by computing five clusters for each texture window, from which the two clusters with the highest "density" (as defined below) are selected as the ones corresponding to the predominant source signal. The peaks corresponding to the selected clusters are then used to resynthesize the segregated predominant signal using a bank of sinusoidal oscillators (as described in the following section).

Specifically, among the several clusters C_i identified by the grouping algorithm in each

texture window (see Section 3.6.5), the idea is then to select the clusters that most likely represent the predominant source signal. A cluster of peaks corresponding to a predominant harmonic source should be "dense" in the feature space in which the similarities are computed. The reason is that peaks belonging to a prominent harmonic "source" have more precise parameter estimates and therefore comply better to the implicit model expressed by the various similarity functions. The peaks of a prominent source will therefore tend to be more similar (mostly in terms of harmonicity) to peaks belonging to the same source than to other sources. Thus, the intra-cluster similarities should be high for this particular cluster. Accordingly, let a cluster of peaks P_c of cardinality $\#P_c$ be defined as the set of peaks whose cluster or grouping label is c:

$$P_c = \{p_m^k | \text{label}(p_m^k) = c\}. \tag{31}$$

It is then possible to define a density criterion as:

$$d(P_c) = \frac{1}{(\#P_c)^2} \sum_{p_l^k \in P_c} \sum_{p_m^j \in P_c} W(p_l^k, p_m^j) \tag{32}$$

where k, j are the frame indices within the texture window and l, m are respectively the peak indices within the frames k and j. The function W refers to the overall similarity weight between peaks, as defined in Equation 30 and discussed in Section 3.7.3).

The numbers of clusters to segment and to subsequently select for resynthesis were found during preliminary experiments. Although giving reasonable results (see Chapter 4), they are in no way an optimal choice for all types of musical signals. However, given the high number of degrees of freedom in the proposed system, it was for now necessary to fix some of its parameters using the described values. This allowed to focus at this stage on the evaluation of other equally important aspects of the framework, as will be shown by the experiments presented in Chapter 4.

3.8.2 Additive Synthesis

Once the clusters resulting from the segregation process described in the previous sections of this chapter are identified, it is now possible to resynthesize them as independent sound signals. By picking the highest amplitude peaks of the spectrum (see Section 3.4), it is usually possible to achieve fair resynthesis quality using a small number of sinusoids per frame with a significant savings in computation time.

As a result, and given that clusters consist of a maximum of L^k spectral peaks

$p_l^k = \{f_l^k, a_l^k, \phi_l^k\}$ at each frame (in the current system L^k is set to 20), it is now simple to synthesize the corresponding audio signals using a bank of sinusoidal oscillators implementing Equation 1 and using a 50% overlap-add scheme.

More details about the actual software implementation of the resynthesis of the segregated signals may be found in Chapter 5.

3.9 Summary

This chapter presented and discussed the framework for sound segregation in music signals proposed in this thesis. The different building blocks comprising the system were described and the choices used for their current implementation were justified, allowing to provide a base ground for future developments and improvements to the system. Some preliminary and illustrative examples of the way the algorithms work were provided, but a more objective and comprehensive evaluation and validation of the performance of the system for the segregation of real-world signals will be conducted in the following chapter.

Chapter 4

Experimental Validation and Applications

"If it sounds good and feels good, then it is good!"

Duke Ellington

4.1 Introduction

Typical audio content analysis systems for music signals represent statistically the entire polyphonic sound mixture [Pachet and Cazaly, 2000, Tzanetakis, 2002]. There is some evidence that this approach has reached a "glass ceiling" [Aucouturier and Pachet, 2004] in terms of analysis and retrieval performance. One obvious direction for further progress is to attempt to individually characterize the different sound events or sources comprising the polyphonic mixture.

The overall goal of the sound segregation framework proposed in the previous chapter is to extract structural information from a complex sound scene consisting of simultaneously occurring sound events, mainly produced by musical sources (although in principle nothing prevents environmental sources from also being considered). Still, the main objective of this work is not to achieve an actual separation of all the physical sources in a sound mixture. Instead, the aim is to provide a perceptually motivated topological description of the sound scene similar to the way a naive listener would perceive it (as discussed in Section 2.3.2).

However, and as previously discussed in Section 3.8.1, the current framework implementation, although already including some sequential grouping principles at the texture window level, does not yet fully incorporate sequential grouping among sound clusters

detected in different texture windows. In addition, and as discussed in Section 3.6.4, the current system does not yet implement an automatic estimation of the number of sound events at each time instant (or texture window) of a sound mixture. While these constraints prevent at this stage to achieve a complete description of a sound scene, they do not hinder the use and evaluation of the proposed framework in application specific scenarios (e.g. in MIR tasks).

In fact, one of the challenges faced when developing such a sound analysis framework is the need to objectively evaluate the resulting structured representation. As discussed in Section 2.4.5, one plausible way to validate the proposed framework is to demonstrate that this approach can achieve a fair degree of segregation of the sound events corresponding to, for example, the main melody in a polyphonic music signal. Due to practical reasons, such an evaluation is performed using an objective measurement approach where the original isolated sources (i.e. the "physical" sounds) are used as references. This raises a pertinent question of whether such an evaluation is in fact assessing the ability of the system to segregate "perceptual" sounds. The final assumption is that such an evaluation approach, although being amenable to further improvements by conducting additional listening tests, is nevertheless an adequate compromise solution for a first measurement of the framework ability to perform sound segregation. Still, a supplementary way to substantiate the relevance of such a structured representation is to show that, when compared to typical audio content analysis and MIR systems, the use of segregated signals in MIR tasks leads to improved (or comparable) final results.

Accordingly, this chapter will present a set of experiments and evaluation scenarios that validate the suitability of the framework proposed in this work for the segregation of sound events in music signals. It will start with a preliminary evaluation of the perceptual grouping cues proposed in Section 3.7, and proceed with more application-specific scenarios, where different aspects of a music analysis system are evaluated.

Unless stated otherwise, all experiments whose results are presented and discussed in this chapter use the following parameters (see Chapter 3 for more details): 46 ms analysis windows using a hop size of 11 ms, 20 spectral peaks are selected from the highest amplitude peaks in each analysis frame, and texture windows are set to a fixed length of about 150 ms.

One of the first decisions to make is which type of evaluation criteria to use for each evaluation task. Following the discussion in Section 2.4.5, various approaches and different objective evaluation metrics (see Appendix B) will be used in each specific evaluation scenario. Different datasets will also be used for each particular experiment, depending

on the features under evaluation. Thus, each section will present a detailed description of the evaluation corpus and information about the experimental setup used.

The interested reader may find some audio examples related to most of the experiments presented in this chapter in `http://www.fe.up.pt/~lgustavo`, where it is possible to appreciate the relative perceptual relevance of the obtained segregation results.

4.2 Preliminary Evaluation of Perceptual Grouping Cues

The grouping cues presented in Section 3.7 play a determinant role on the performance of the sound segregation framework proposed in this thesis. As a result, it is of upmost importance to start by objectively validating their influence on the segregation ability of the system. In particular, the novel HWPS cue will be shown to have an important impact on the system capacity to identify harmonic sources in complex audio signals.

4.2.1 Evaluation of the HWPS Cue

In this section, the properties of the HWPS cue, introduced in Section 3.7.2, will be studied and compared to existing state-of-the art harmonicity cues using a generic evaluation methodology. The experiments show the improvement in segregation performance achieved by using the HWPS similarity cue compared to two existing cues proposed in Srinivasan [Srinivasan and Kankanhalli, 2003] and Virtanen [Virtanen and Klapuri, 2000], as described in Section 3.7.2.

To evaluate the capabilities of the presented harmonic cues, two synthetic sets of peaks with harmonically related frequencies and exponentially decaying amplitude envelope were considered. The first set of peaks has a fundamental frequency of 440 Hz, whereas the f_0 of the second set is iteratively changed to values from 10 to 5000 Hz, using a 10 Hz step (both sets of peaks follow harmonic distributions similar to the ones presented in Table 1).

It is then possible to define a Fisher criterion $\mathcal{F}$ (loosely based on the Fisher discriminant commonly used in statistical analysis [Duda et al., 2000, pp.117]) as the sum of the inter-class scatter divided by the sum of the intra-class scatter. Since the Fisher criterion is not scale invariant, it may not be the best choice to compare the performance of distinct cues. Nevertheless, it is still an interesting way of evaluating the performance of a metric with respect to different scenarios.

Given so, a Density criterion $\mathcal{D}$, computed as the number of peaks that have the closest neighboring peak in the feature space belonging to the same set, is also defined. This criterion is scale invariant and closer to the one considered by clustering algorithms.

	W_s	W_v	W_h	$W_h(f_0)$
$\mathcal{F}$	1.44 (0.31)	1.00 (0.01)	1.22 (0.05)	2.27 (0.37)
$\mathcal{D}$	0.50 (0.01)	0.55 (0.11)	0.80 (0.12)	0.94 (0.16)

Table 2: *Results for the separation of two harmonics sets of peaks. Mean and standard deviation values of the Fisher and Density criteria are computed for the Srinivasan (W_s), Virtanen (W_v), HWPS (W_h), and HWPS with prior knowledge of the two f_0's ($W_h(f_0)$).*

The partitioning of a set of elements X is represented using an indicator function, as follows:

$$\begin{array}{rccc} E\colon & X & \rightarrow & \mathbb{N} \\ & x & \mapsto & i \end{array}$$

where i is the partition index x belongs to. The closest neighbour of a peak $a \in X$, $V(a)$, can be represented as:

$$V(a) = b | b \in X \setminus \{a\} \wedge d(a,b) = min_{c \in X \setminus \{a\}} d(a,c). \tag{33}$$

The Density criterion can then be defined as:

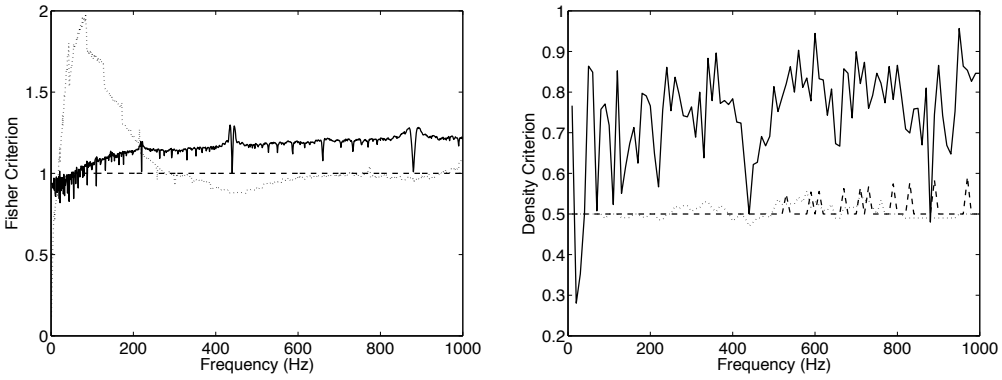
$$\mathcal{D}(X) = \frac{1}{(\#\tilde{X})^2} \tag{34}$$

where $\tilde{X}$ is defined as:

$$\tilde{X} = \{c | c \in X \wedge E(c) = E(V(c))\}. \tag{35}$$

This Density criterion will result higher when peaks belonging to a same harmonic set end up close together in the similarity space (and hence, there is a higher chance of being the nearest neighbours of each other), as desired. Table 2 presents the average performance of the evaluated harmonic cues in the $[100, 5000]$ Hz range, using the Fisher and Density criteria. The last column shows the performance of the HWPS with prior knowledge of the f_0's of the two sets of peaks.

Figure 26(a) shows the evolution of the Fisher criterion with respect to the f_0 of the second set for the three harmonic cues. The Srinivasan cue shows the expected behavior, with minimal performance when the two sources have close f_0's (around 440 Hz). Another local minima is found around 880 Hz and the performance globally increases with the second f_0. Since the Virtanen and HPWS cues consider more precise frequency estimates, a finer behavior can be noticed around frequencies multiple of the first peak f_0. Differently



(a) (b)

Figure 26: *Fisher (a) and Density (b) criteria versus the f_0 of the second source for the Srinivasan cue (dotted line), the Virtanen cue (dashed line) and the HPWS cue (solid line). The f_0 of the first source is set to 440 Hz.*

from the cue proposed by Virtanen, the HWPS performance increases with the frequency difference between the two f_0's, as desired. As shown in Figure 26(b), the HWPS performs well as far as the Density criterion is concerned, except at frequency locations multiple of 440 Hz, the f_0 of the reference set.

4.2.2 Combined Use of Grouping Cues

For the evaluation of the combination of the different grouping cues an experimental setup inspired by the "old+new" heuristic described by Bregman [Bregman, 1990, pp.261] is used. The dataset consists of synthetically-created mixtures of isolated instrument sounds, voice, harmonic sweeps and noise. Each clip is approximately 1 second long and is created by mixing two sound sources in the following way: for the first part of the sound only the "old" sound source is played followed by the addition of the "new" sound source (i.e. "old+new") in the second part of the sample.

Ncut clustering is then performed over the entire duration of the clip. The clusters that contain peaks in the initial "old"-only part are selected as the ones forming the separated source. The remaining peaks are considered to be part of the "new" sound source. The Signal-to-Distortion ratio (SDR – see Appendix B) is used to measure the distortion/interference caused by this "new" sound source to the separation algorithm. Similar experiments were presented in [Lagrange and Tzanetakis, 2007].

Table 3 compares different mixtures of isolated sounds separated using only frequency

Table 3: *SDR values (in dB) for experiments using different "old+new" mixtures. The following conventions are used : X is saxophone, N is noise, V is violin, S is harmonic sweep, and C is voice. A, F, H correspond to using amplitude, frequency and HWPS similarities, respectively.*

	XN	XS	VN	VS	CN	CS
A+F	12.87	9.33	10.11	7.67	2.94	1.52
A+F+H	13.05	9.13	11.54	7.69	3.01	2.09

Table 4: *SDR values (in dB) using "texture windows" for experiments with different "old+new" mixtures. The same experimental setup and conventions described in Table 3 are used.*

	XN	XS	VN	VS	CN	CS
A+F	9.79	3.09	3.29	6.50	3.01	3.01
A+F+H	7.33	5.03	4.73	5.35	3.08	3.07

and amplitude similarities, and also separated with the additional use of the HWPS similarity. As can be seen from the table, in almost all cases the use of the HWPS improves the SDR measure of separation performance.

A second set of experiments where the "old+new" mixtures are separated directly using the approach described in section 3.8.1 to select the dominant sound source were conducted. Unlike the previous experiments, the spectral clustering is performed separately for each "texture window" and the highest density cluster is selected as the separated voice. This is a more realistic scenario as no knowledge of the individual sound sources is utilized. The results are presented in Table 4 and, as expected, the SDR values are lower than the ones in Table 3, but once again the use of the HWPS improves separation performance in most cases.

4.3 Predominant Melodic Source Segregation

The predominant melodic source is an important component of musical signals. It is usually related to the singing voice or, in the case of instrumental music, to the musical instrument playing the main melodic line. Given the relevance of the singing voice in popular western music, and because it was easier to find audio datasets where the isolated singing voice track was available to be used as the segregation reference, in this section the predominant melodic source will be assumed to be the singer's voice. In fact, the voice and melodic characteristics of singers are some of the primary features of music listeners relate to, and its separation and characterization has a large number of applications in MIR.

Most existing query-by-humming systems [Ghias et al., 1995, Dannenberg et al., 2004] can only retrieve songs from a database containing music in symbolic format. By performing pitch extraction on the extracted voice signals, it is possible to perform query-by-humming in databases of audio signals. Another potential application is singer identification that is independent of the instrumentation or the "album" effect [Kim et al., 2006]. Other possible applications include automatic accompaniment, music transcription, and lyrics alignment.

There has been limited work on singing voice separation from monaural recordings. Many existing systems require predominant pitch detection in order to perform separation [Li and Wang, 2006, Li and Wang, 2007] or rely on prior source models [Ozerov et al., 2005]. Other approaches are based on statistical methods such as Independent Component Analysis (ICA) [Abdallah, 2002] and Non-Negative Matrix Factorization (NMF) [Vembu and Baumann, 2005]. The non-stationarity of the singing voice and music signals as well as their heavy computational requirements are some of the challenges of applying statistical methods to this problem.

In contrast, the CASA framework proposed in this thesis (see Chapter 3), being based on basic perceptually-inspired grouping cues [Bregman, 1990], allows to directly perform the segregation of sound events (including the prominent melodic source) in polyphonic music signals without first estimating the predominant pitch.

Accordingly, the following sections will present the results obtained for the task of extracting the predominant source (namely the singing voice) from "real-world" polyphonic music. A description of the audio dataset used for the evaluation will be firstly presented, followed by the discussion about experimental setup and the obtained results. The use of the dynamically adjusted texture windows, based on the automatic onset detection as described in Section 3.5 will also be evaluated using this same experimental scenario.

4.3.1 Corpus Description and Experimental Setup

This experiment will evaluate the performance of the segregation of the predominant sound source in "real-world" polyphonic music signals using different similarity measures. A dataset of 10 polyphonic music signals (with lengths in the range 3-5 minutes), each including their original vocal and music accompaniment tracks before mixing as well as the final mix, is used for this evaluation. Although relatively small, this dataset is diverse and covers different styles of singing and music background, containing the following types of music: rock, celtic and hiphop.

Given the analysis front-end used in this work (see Section 3.4), a sinusoidal representation of the original voice-only track using 20 sinusoidal peaks per analysis frame is also used as the reference signal for computing the SDR. This assures that the SDR comparison is measured between similar representations of the original and separated signals, and therefore is more meaningful. This representation of the reference signal is perceptually very similar to the original voice-only signal and captures the most important information about the singing voice, such as the identity of the singer, pitch, vibrato.

In this experiment fixed length texture windows are used (recall the discussion about time segmentation in Section 3.5) and the dominant sound source is separated and selected directly using the approach described in Section 3.8.1. Furthermore, no post-processing of the extracted signals was performed in order to provide a better insight about the algorithm and its limitations. For example, a dominant cluster is always selected independently of the presence of a main melody line (in this specific case, the singing voice).

4.3.2 Experimental Results

Several polyphonic music separation experiments were conducted using different combinations of similarity measures, whose results are summarized in Table 5. The first column in the table shows the performance of the proposed system using only the amplitude and frequency similarities (defined in Section 3.7.1). The other columns show the performance of using the three different harmonicity similarities (see Section 3.7.2) in addition to amplitude and frequency. All the configurations utilize the same parameters for the Ncut algorithm (see Section 3.6.3) and the only thing that changes is the definition of the similarity function.

As can be seen, in most cases the HWPS similarity provides better results than the Virtanen similarity (HV) [Virtanen and Klapuri, 2000] and the Srinivasan similarity (HS) [Srinivasan and Kankanhalli, 2003] and behave similarly otherwise, confirming the preliminary evaluation results of Section 4.2. Finally the last two columns show the importance for the HWPS computation of the precise frequency estimation (described in Section 3.4) when compared to rough frequency estimation directly from FFT bins. A similar drop in performance between rough and precise estimation was also observed for HV and HS but not included in the table.

Once again, these results illustrate the improvement in separation performance when using the HWPS, which confirms the importance of the harmonicity cue for the segregation of pitched components from complex audio mixtures.

Table 5: *SDR measurements (in dB) of predominant sound source separation in polyphonic music using different similarity measures. The first column (AF) shows the results of using only the amplitude and frequency similarity cues, while the remaining ones present the performance of adding the use of the harmonicity cues proposed by Srinivasan (HS), Virtanen (HV) and the rough (rHWPS) and precise (HWPS) frequency estimation HWPS cues, respectively.*

Track Title	AF	HS	HV	rHWPS	HWPS
bentOutOfShape	2.66	1.19	1.17	5.65	8.03
intoTheUnknown	0.65	3.24	0.81	3.29	4.05
isThisIt	1.94	2.71	2.07	2.18	2.65
landingGear	1.29	5.29	0.81	4.40	6.37
schizosonic	0.57	3.11	0.59	2.86	3.96
smashed	0.22	1.15	0.29	1.17	1.54
chavalierBran	4.25	7.21	1.6	4.02	6.83
laFee	7.7	6.62	2.48	4.88	6.93
lePub	0.23	0.48	0.23	0.38	0.47
rockOn	0.96	1.78	0.79	1.60	1.74

4.3.3 On the Use of Dynamic Texture Windows

Following the discussion about the time segmentation of music audio signals in Section 3.5, this section presents a comparative evaluation on the impact of the use of fixed or dynamically adjusted texture windows for the segregation of the predominant melodic source from monaural polyphonic music signals. As detailed in Section 3.5.2, the dynamic adjustment of the length of the texture windows is based on the use of an onset detector.

For this experiment, a dataset of 10 polyphonic music signals almost identical to the one used in the previous evaluation was used. Similarly to the previous setup, a sinusoidal representation of the original voice-only track using 20 sinusoidal peaks per analysis frame is also used as the reference signal for computing the SDR_{ogf}. The fixed texture windows have a length of 10 frames ($\approx$ 150 ms, given the use of a window size of about 46 ms and a hop size of 11 ms), and for the case of the dynamic texture windows, their sizes will be in the range 50 ms - 300 ms.

Table 6 presents the comparative results for the predominant melodic source segregation for the two types of the texture windows. There is on average a small improvement of 0.15 dB in the SDR_{ogf} measure when using texture windows dynamically adjusted by the automatically detected onsets in the signal. Nevertheless, the statistical relevance of such a small difference may not be as significant as expected (recall the discussion in Section 3.5).

This small improvement may in fact result from the cluster selection approach used

Table 6: *SDR_{ogf} measurements (in dB) of predominant melodic source segregation from monaural polyphonic audio recordings using fixed and dynamic adjustment of the length of texture windows.*

Track Title	Fixed Text.Windows	Dynamic Text.Windows	ΔdB
bentOutOfShape	5.46	5.50	0.05
intoTheUnknown	2.45	2.46	0.01
isThisIt	1.90	1.84	-0.06
landingGear	4.21	4.13	-0.07
schizosonic	2.09	2.15	0.06
smashed	0.78	0.81	0.04
chavalierBran	4.24	6.01	1.77
lePub	3.02	2.78	-0.23
rockOn	0.90	0.91	0.02
latinoRemix	2.05	2.07	0.06
tudoBem	2.14	2.19	0.06
Mean (Stdev)	*2.66 (1.45)*	*2.81 (1.71)*	*0.15 (0.54)*

for the current implementation (recall the discussion in Section 3.8.1). In this method, the peaks in each texture window are over-segmented into five clusters, from which the two most "dense" ones are selected for resynthesis. Because the dynamic adjustment of the texture windows to inter-onset segments will desirably increase the probability of getting windows containing just a single sound event (e.g. a single note), this over-segmentation approach may, in such a case, end up breaking a single event into incoherent sub-structures, hurting the final segregation. Differently, when using fixed windows the probability of having multiple (complete or partial) events per texture window is higher and consequently the over-segmentation will better fit the different events under analysis.

One drawback of using dynamically adjusted texture windows is related to the computational costs involved in the computation of the Ncut for longer texture windows. As discussed in Section 3.6, longer texture windows will in most cases account to a higher number of peaks, resulting in larger similarity matrices. Because the specific NCut implementation in this work is based on SVD computation, its complexity is in the order of $O(n^3)$. Consequently its computational cost increases steeply with the increase of the number of peaks n to cluster, having as a consequence an increased average computation time when using dynamically adjusted texture windows. In fact, this problem is more related to the use of the Ncut than to the dynamic texture window approach, and some solutions and alternatives have been proposed to improve the computational cost of the clustering procedure (e.g. [Dhillon et al., 2007]). As a result, and given the small improvements provided by the use of dynamically adjusted texture windows, mainly when compared to

the resulting computational costs, all the evaluations and experiments presented in the following sections will use fixed length texture windows.

Nevertheless, these slightly better segregation results show that the use of the detected onsets to automatically segment the audio signal into texture windows may be a promising strategy. As a result, further experimentation and evaluation as well as the implementation of more computationally efficient clustering methodologies should be explored in the future in an attempt to get the most out of this approach.

4.4 Main Melody Pitch Estimation

The main melodic line is a critical piece of information for describing music and is very influential in the identity of a musical piece. A common approach to automatic melody extraction is to attempt multiple pitch extraction on the polyphonic mixture and select the predominant pitch candidate as the pitch of the singing voice (e.g. [Paiva, 2006]). The detected pitch can then be used to inform source separation algorithms. In the method evaluated in this section the singing voice is first separated and the melodic pitch is subsequently extracted directly from the separated audio.

It is important however to note that the main goal of the framework proposed in this thesis is not just predominant pitch estimation but also separation and resynthesis of the separated voice preserving additional information such as timbral characteristics and singer identity. As a result, this evaluation should be taken as an additional way to validate the proposed method.

Entire works have been dedicated to this topic (e.g. [Ryynanen and Klapuri, 2006, Li and Wang, 2006, Klapuri et al., 2006, Paiva et al., 2006, Poliner et al., 2007]), and the interested reader should refer to them for a comprehensive discussion on the definition of the problem, its challenges and proposed solutions.

4.4.1 Corpus Description and Experimental Setup

For each song in the dataset of 10 songs already used for the experiments presented in Section 4.3 (for which, recall, the original voice-only tracks are available) the pitch contours were calculated for three configurations: the original clean vocal signal, the polyphonic recording with both music and vocals (VM), and the vocal signal separated by the algorithm proposed in this work ($VSep$).

Two pitch extraction algorithms were utilized: a time domain autocorrelation monophonic pitch extraction algorithm implemented in Praat [Boersma and Weenink, 2006],

and a recent multipitch estimation algorithm developed by Klapuri [Klapuri et al., 2006]. Both approaches were configured to estimate fundamental frequencies in the range $[40, 2200]$ Hz, using a hop size of 11ms and an analysis window with a length of about 46ms.

The pitch contours estimated using Praat from the polyphonic recordings with both music and vocals will be referred in the text, figures, and tables as VM_{praat}, while the ones extracted using Klapuri's algorithm will be referred as VM_{klap}. Similarly, for the separated vocal signals $VSep_{praat}$ and $VSep_{klap}$ will be used. For ground truth Praat is used to extract a reference pitch contour from the original voice-only track of each song. This ground truth was confirmed as sufficiently correct and therefore suitable for this evaluation by listening to the generated contours.

For the purpose of this specific evaluation only the singing segments of each song are considered. They are identified as the segments in the ground truth pitch contours that present non-zero frequency values. For each pitch contour the normalized pitch error is computed at each frame as follows:

$$NE[k] = \left| \log_2 \frac{f[k]}{f_{\text{ref}}[k]} \right| \tag{36}$$

where $f_{ref}[k]$ corresponds to the frequency values of the ground truth pitch contour, measured in Hertz, and k is the frame index. $f[k]$ is either related to the frequency value of the pitch contour extracted from the mixed signal (i.e. VM_{praat} or VM_{klap}), or to the pitch contour extracted using the vocal track separated by the proposed algorithm (i.e. $VSep_{praat}$ and $VSep_{klap}$). The normalized error NE is zero when the pitch estimation is correct, and an integer number for octave errors (i.e. when $f[k] = 2^n \times f_{ref}[k], n \in \mathbb{N}$). Since only the singing segments of the signals are being considered, both $f[k]$ and $f_{ref}[k]$ will never be zero.

Given that this evaluation is related to (musical) pitch estimation, a chroma-based error measure NE_{chr} derived from NE was also defined. In this measure the errors are folded into a single octave as follows:

$$NE_{chr}[k] = \begin{cases} 0 & \text{if} \quad NE[k] = 0 \\ 1 & \text{if} \quad NE[k] \neq 0 \ \wedge \ \text{mod}(NE[k], 1) = 0 \\ \text{mod}(NE[k], 1) & \text{otherwise} \end{cases} \tag{37}$$

where mod() is the real modulo function[1]. This allows bringing to evidence the chromatic

[1]The real modulo function is inhere defined as a modulo operation according to the following conditions:

Table 7: *Normalized Pitch Errors (NE and NE_{chr}) and Gross Errors (GE and $GE-8^{ve}$) across corpus, for different evaluation scenarios.*

	NE	NE_{chr}	$GE(\%)$	$GE-8^{ve}(\%)$
VM_{praat}	8.62 ($\pm$11.86)	0.51 ($\pm$0.35)	82.44	66.00
VM_{klap}	0.55 ($\pm$0.83)	0.26 ($\pm$0.33)	55.70	48.68
$VSep_{praat}$	3.89 ($\pm$8.35)	0.35 ($\pm$0.36)	64.45	55.23
$VSep_{klap}$	1.34 ($\pm$1.09)	0.56 ($\pm$0.37)	79.65	62.04

distribution of the errors, wrapping all pitch inaccuracies into the interval $[0, 1]$, where 0 corresponds to no pitch estimation error and 1 accumulates all the octave-related errors.

The following section will also present the results of similar experiments conducted using the corpus created for the MIREX automatic melody extraction evaluation exchange[2]. It consists of 23 clips of various styles including instrumental and MIDI tracks for which case the dominant melodic voice is estimated. In this case there was no access to the original melody-only tracks, but ground truth pitch contours were provided for the evaluation. As a result, this gives an opportunity to test the system's segregation performance for signals where there is no singing voice, but instead a predominant musical instrument usually playing the main melody in a song.

4.4.2 Experimental Results

Table 7 shows the normalized pitch errors across the dataset of 10 songs for the different evaluation scenarios. It also presents the gross error (GE) for each case, defined as the percentage of all errors larger than half a semitone (i.e. for $NE > \frac{1}{24}$). This tolerance allows accepting estimated frequencies inside a one semitone interval centered around the true pitch as correct estimates. Also presented is the gross error excluding all octave errors ($GE - 8^{ve}$). Octave ambiguities can be accepted as having smaller impact than other errors for many musical applications.

From the results obtained from the VM_{praat} evaluation, a GE in excess of 82% confirms the expected inability of monophonic pitch detectors such as the Praat algorithm to accurately estimate the most prominent pitch on polyphonic music recordings. However, if using the same exact pitch estimation technique on the voice signal separated by the proposed system (i.e. $VSep_{praat}$), the results demonstrate a clear improvement, reducing the gross error rate GE to about 64%. Although the proposed scheme does not compare favourably to the state-of-the-art multipitch algorithm by Klapuri (GE of 55.7%),

if taking $D = q \times d + r$, where D is the dividend, d is the divisor, $q \in \mathbb{N}$ is the quotient, and the returned remainder $r \in \mathbb{R}$ is in the range $0 < r < d$. E.g. mod(5.3, 2) = 1.3 or mod(5.3, 1) = 0.3.

[2] http://www.music-ir.org/mirex2005/index.php/Main_Page

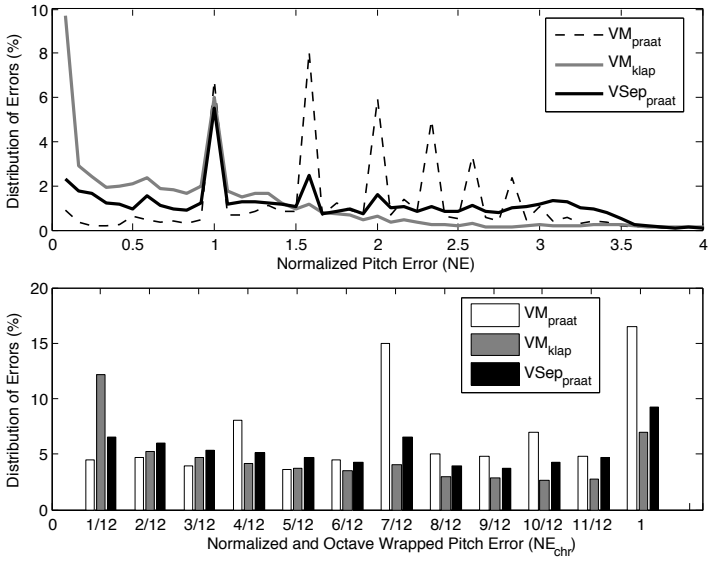


Figure 27: *Distribution of errors as percentages of total pitch estimates across corpus. The top plot presents a segment of the normalized error distribution, NE (no significant error values exist outside the plotted ranges), while the bottom plot depicts the corresponding octave wrapped error distribution, NE_{chr}. The pitch errors from using Praat on the separated voice signals are represented in black colour, while its use on mixed signals is represented by the slashed line and white bars. The multiple pitch approach of Klapuri is represented in gray.*

it shows the ability of the proposed method to simplify the acoustic scene by focusing on the dominant harmonic source.

Intriguing to note are the results obtained when using the algorithm by Klapuri in the separated signal (i.e. $VSep_{klap}$). Given the good results obtained when applying this algorithm to the mixed signals (i.e. VM_{klap}), it would be expectable to get even better results when applying the same method to the separated signals. Resynthesis artifacts and phase distortions in the separated signals may be in the origin of these poor results, but additional research and study of Klapuri's algorithm will have to be conducted in order to fully understand these results.

It is also interesting to look at the distribution of errors. Figure 27 shows the distribution of the normalized and octave wrapped errors as percentages over the total number of estimated pitch frames (the upper plot presents the significant section of the NE distribution while the lower plot shows the NE_{chr} distribution). All three evaluations presented in the plots show a similar tendency to output one-octave ambiguities (i.e. about 6% for $NE = 1$). VM_{praat} presents several additional high-valued error peaks caused by incorrect

Table 8: *Normalized Pitch Errors (NE and NE_{chr}) and Gross Errors (GE and $GE - 8^{ve}$) for MIREX Dataset, using different evaluation scenarios.*

	NE	NE_{chr}	$GE(\%)$	$GE - 8^{ve}(\%)$
VM_{praat}	3.29 ($\pm$6.62)	0.48 ($\pm$0.37)	76.02	55.87
VM_{klap}	0.34 ($\pm$0.75)	0.15 ($\pm$0.28)	34.27	29.77
$VSep_{praat}$	1.34 ($\pm$4.13)	0.36 ($\pm$0.40)	54.12	34.97
$VSep_{klap}$	0.48 ($\pm$0.58)	0.33 ($\pm$0.40)	53.07	36.67

pitches estimated due to the presence of multiple overlapping notes from the musical background. These errors are significantly reduced in the case of the pitch estimation on the separated signal using the proposed method. When compared to VM_{klap} most of the pitch estimation errors from $VSep_{praat}$ result from octave and perfect-fifth (i.e. $NE_{chr} = 7/12$) ambiguities.

Similar experiments were also conducted using the corpus created for the MIREX automatic melody extraction evaluation exchange. The MIREX examples include some synthesized MIDI pieces, which are simpler to separate as they do not include reverberation and other artifacts found in realworld signals. Most of the examples also have a more pronounced vocal line or dominant melody than the corpus used for the previous experiments, and therefore most of the results were better. Table 8 shows the normalized pitch errors and gross errors for the MIREX corpus. The distribution of the normalized errors are depicted in Figure 28.

4.5 Voicing Detection in Polyphonic Music Signals

Given the relevance of the singing voice in popular western music, a system able to reliably identify those portions of a music audio file containing vocals would be useful to obtain a "signature" of the piece or to serve as pre-processing stage to automatic recognition of lyrics. Several authors have been working in this specific problem, and the interested reader can find detailed discussions and approaches to this problem in [Berenzweig and Ellis, 2001, Nwe et al., 2004, Rocamora and Herrera, 2007]. In particular, Rocamora and Herrera recently explored a set of descriptors proposed in the literature to perform this task and compared the performance of a statistical classifier using each one of them, concluding that Mel-Frequency Cepstral Coefficients (MFCC) are the most appropriate [Rocamora and Herrera, 2007]. Accordingly, the goal of the experiments described in this section is to determine whether the proposed sound segregation algorithm can be used to improve voicing detection accuracy in monaural polyphonic recordings.

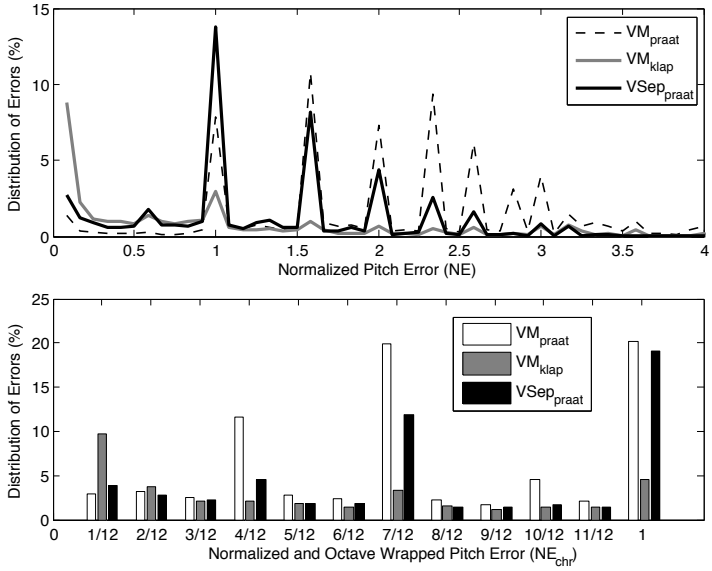


Figure 28: *Distribution of errors over the MIREX audio melody extraction dataset. The top plot presents a segment of the normalized error distribution, NE (no significant error values exist outside the plotted range), while the bottom plot depicts the corresponding octave wrapped error distribution, NE_{chr}. The pitch errors from using Praat on the separated voice signals are represented in black colour, while its use on mixed signals is represented by the slashed line and white bars. The multiple pitch approach of Klapuri is represented in gray.*

4.5.1 Corpus Description and Experimental Setup

The dataset of the 10 polyphonic music pieces for which the original separate vocal track are available (described in Section 4.3.1) was once again used for the experiments. For this evaluation, voiced/unvoiced decisions extracted using Praat [Boersma and Weenink, 2006] from the original vocal track were used as the ground truth. A supervised learning approach was then used to train voiced/unvoiced classifiers for three configurations:

- VM_{MFCC} refers to using MFCC [Davis and Mermelstein, 1980] calculated over the mixed voice and music signal;
- $VSep_{MFCC}$ refers to MFCC calculated over the automatically separated voice signal;
- $VSep_{CPR}$ refers to using the Cluster Peak Ratio (CPR), a feature that can be directly calculated on each extracted clusters of peaks. It is defined as:

$$CPR = \frac{\max(A^k)}{\text{mean}(A^k)} \tag{38}$$

where A^k are the extracted peak amplitudes for frame k. Voiced frames tend to have more pronounced peaks than unvoiced frames and therefore tend to have higher CPR

Table 9: *Voicing Detection Percentage Accuracy using different classifiers and evaluation scenarios, where NB refers to a Naive Bayes classifier, SVM to a support vector machine and ZeroR is the "random" baseline classifier.*

	ZeroR	*NB*	*SVM*
VM_{MFCC}	55	69	69
$VSep_{MFCC}$	55	77	86
$VSep_{CPR}$	55	73	74

values.

4.5.2 Experimental Results

The experiments were performed using the Weka machine learning framework [Witten and Frank, 2005], where *NB* refers to a Naive Bayes classifier and *SVM* to a support vector machine trained using the sequential minimal optimization (SMO) . The *ZeroR* classifier classifies everything as voiced and was used as a baseline. The goal was to evaluate the relative improvement in classification performance when using the separated voice signal rather than building an optimal voicing detector. No smoothing of the predictions was performed.

Table 9 shows the classification accuracy (i.e the percentage of frames correctly classified using these 3 configurations). All the results were computed using 10-fold cross-validation over features extracted from the entire corpus. In 10-fold cross-validation the feature matrix is shuffled and partitioned into 10 "folds". The classification accuracy is calculated by using 9 of the folds for training and 1 fold for testing and the process is repeated 10 times so that all partitions become the testing set once. The classification results are averaged. 10-fold cross-validation is used to provide a more balanced estimate of classification accuracy that is not as sensitive to a particular choice of training and testing sets [Witten and Frank, 2005].

The results show that using the automatically separated voice results in significant improvements in voicing detection accuracy. Additionally, the use of the simple and direct CPR feature still outperforms a more complex classifier trained on the mixed data. Since CPR is computed directly on the mid-level peak representation (see Section 3.4), this result shows that such a compact sound event representation can be directly used for MIR tasks, without the need of resynthesizing the separated signals.

4.6 Timbre Identification in Polyphonic Music Signals

The increasing quantity of music titles available in digital format added to the huge amount of personal music storage capacity available today has resulted in a growing demand for more efficient and automatic means of indexing, searching and retrieving music content. The computer identification of the instruments playing in a music signal can assist the automatic labeling and retrieval of music.

Several studies have been made on the recognition of musical instruments on isolated notes or in melodies played by a single instrument. A comprehensive review of those techniques can be found in [Herrera et al., 2003]. However, the recognition of musical instruments in multi-instrumental, polyphonic music is much more complex and presents additional challenges. The main challenge stands from the fact that tones from performing instruments can overlap in time and frequency. Therefore, most of the isolated note recognition techniques that have been proposed in the literature are inappropriate for polyphonic music signals. Some of the proposed techniques for the instrument recognition on polyphonic signals consider the entire audio mixture, avoiding any prior source separation [Essid et al., 2005, Livshin and Rodet, 2004]. Other approaches are based on the separation of the playing sources, requiring the prior knowledge or estimation of the pitches of the different notes [Kashino and Murase, 1999, Kostek, 2004]. However, robustly extracting the fundamental frequencies in such multiple pitch scenarios is difficult.

In this section, the segregation framework proposed in this thesis is used and evaluated for the task of timbre classification of polyphonic, multi-instrumental music signals. This work and the corresponding results have also been published in [Martins et al., 2007]. A work on source separation based on the use of the same timbre models, originally proposed in [Burred et al., 2006], can be found in [Burred and Sikora, 2007].

4.6.1 System Overview

Figure 29 presents a block-diagram of the complete timbre identification system. After grouping the spectral peaks into separated sound events (which will ideally have a perceptual correspondence to the notes played by each instrument in the sound mixture – see Chapter 3 for a detailed description of the sound source formation block) each identified cluster is matched to a collection of six timbre models, namely piano, oboe, clarinet, trumpet, violin and alto sax. These models are a compact description of the spectral envelope and its evolution in time, and were previously trained using isolated note audio recordings. The design of the models, as well as their application to isolated note classification, were

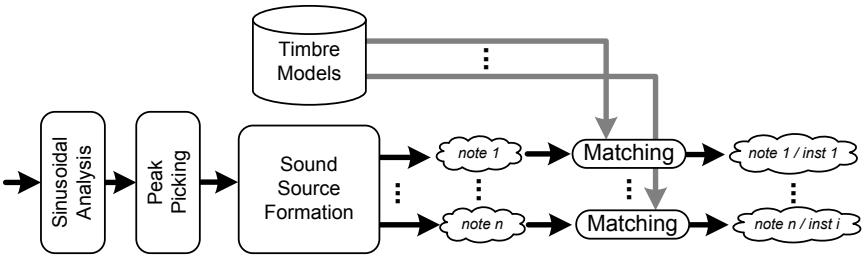


Figure 29: *Block diagram of the timbre recognition system.*

originally proposed in [Burred et al., 2006].

As in the previous experiments, in this evaluation the system was configured to extract a maximum of 20 sinusoids per frame which are 46 ms long, using a hop size of 11 ms. However, texture windows were now set to the duration of the audio signals (i.e. the duration of the notes being played).

However, the number of clusters to be segregated and selected for resynthesis (recall the discussion in Section 3.8.1) was in this case set manually to correspond to the known number of notes playing in each audio signal (see the description of the datasets used for this evaluation presented in Section 4.6.3 for more details on the type of sound signals under analysis).

Although this approach requires prior knowledge about the number of notes playing at each instant (which were in this case available as ground truth), and may raise the question whether each note would be in fact perceived as a single sound event by a naive listener (recall the discussion about "physical" versus "perceptual" sources in Section 2.3.2), one objective of this experiment was to verify if the segregated events had some perceptual correspondence to known timbres and their corresponding models. While listening tests would have been the ideal way to evaluate the perceptual relevance of the segregated sounds, due to practical constraints a more pragmatic evaluation was conducted, as presented in the following sections. Nevertheless, the interested reader may find some audio examples in `http://www.fe.up.pt/~lgustavo`, where it is possible to appreciate the relative perceptual relevance of the segregation results.

As an example of the output of the sound source formation block, Figure 30 depicts the result of the sound segregation for a single-channel audio signal with mixture of two notes (E4 and B4[3], same onset, played by a piano and an oboe, respectively). Each

[3]Throughout this section the convention A4 = 440Hz will be used.

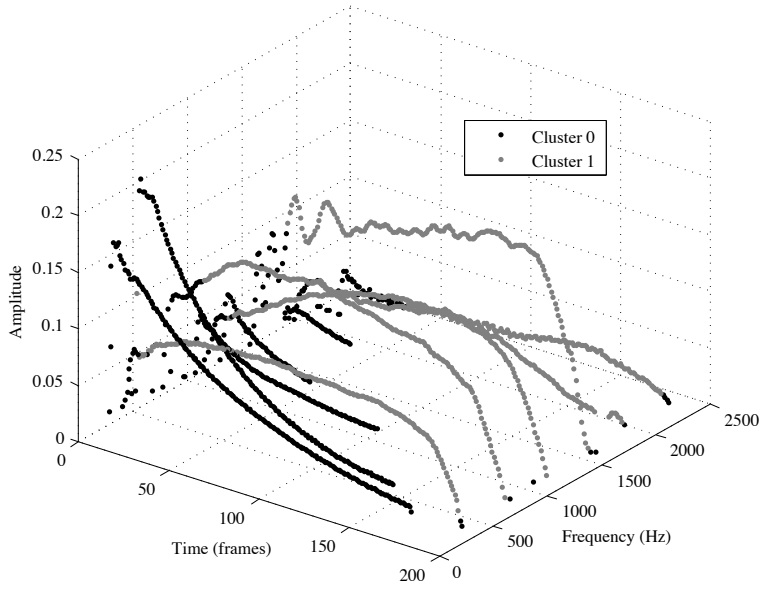


Figure 30: *Resulting sound source formation clusters for two notes played by a piano and an oboe (E4 and B4, respectively).*

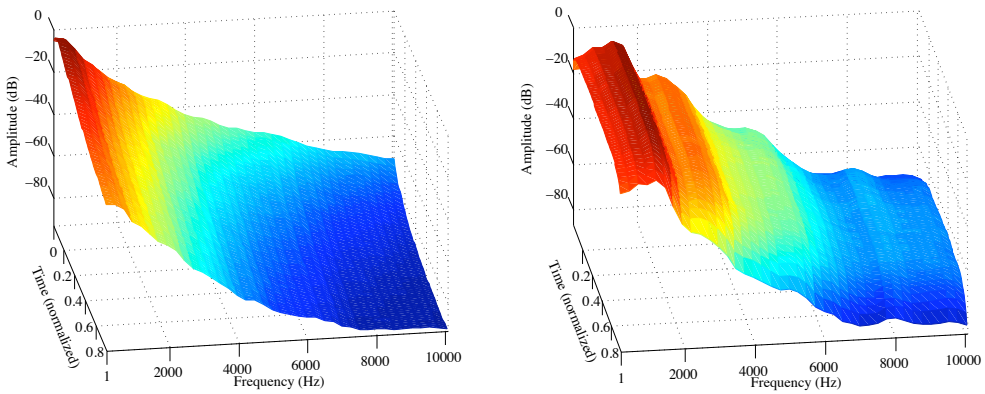
dot corresponds to a peak in the time-frequency space (see Section 3.4) and the different coloring reflects the cluster to which it belongs (i.e. its source). Each cluster is then matched to a set of pre-trained timbre models, described in the following section.

4.6.2 Timbre Models and Instrument Recognition

Once each single-note cluster of sinusoidal parameters has been extracted, it is classified into an instrument from a predefined set of six: piano (**p**), oboe (**o**), clarinet (**c**), trumpet (**t**), violin (**v**) and alto sax (**s**). The method, originally described in [Burred et al., 2006], models each instrument as a set of time-frequency templates, one for each instrument. The template describes the typical evolution in time of the spectral envelope of a note. The spectral envelope is an appropriate representation to generate features to analyze sounds described by sinusoidal modeling, since it matches the salient peaks of the spectrum, i.e., the amplitudes a_l^k of the partials (recall the notation defined in Section 3.4).

Training the Timbre Models

The training process consists of arranging the training dataset as a time-frequency matrix $\mathbf{X}(g, k)$ of size $G \times K$, where g is the frequency bin index and k is the frame index, and performing spectral basis decomposition upon it using Principal Component Analysis



(a) Piano (b) Oboe

Figure 31: *Examples of prototype envelopes for a range of one octave.*

(PCA). This yields a factorization of the form $\mathbf{X} = \mathbf{BC}$, where the columns of the $G \times G$ matrix $\mathbf{B}$ are a set of spectral basis sorted in decreasing order of contribution to the total variance, and $\mathbf{C}$ is the $G \times K$ matrix of projected coefficients. By keeping a reduced set of $R < G$ basis, a reduction of the data needed for a reasonable approximation is obtained and, more importantly for the purpose of this application, a representation is also obtained that is based only on the most essential spectral shapes.

Having as goal a pitch-independent classification, the time-frequency templates should be representative for a wide range of notes. In the training process, notes from several pitches must be considered to give rise to a single model. The training samples are subjected to sinusoidal modeling, and arranged in the data matrix $\mathbf{X}$ by linearly interpolating the amplitude values to a regular frequency grid defined at the locations of the G bins. This is important for appropriately describing formants, which are mostly independent of the fundamental frequency.

The projected coefficients of each instrument in the R-dimensional PCA space are summarized as a prototype curve by interpolating the trajectories corresponding to the individual training samples at common time points and point-wise averaging them. When projecting back into the time-frequency domain by a truncated inverse PCA, each P^i-point prototype curve will correspond to a $G \times P^i$ prototype envelope $\mathbf{M}^i(g, k)$ for instrument i. The same number of time frames $P = P^i$ are considered for all instrument models. Figure 31 shows the obtained prototype envelopes for the fourth octave of a piano and of an oboe.

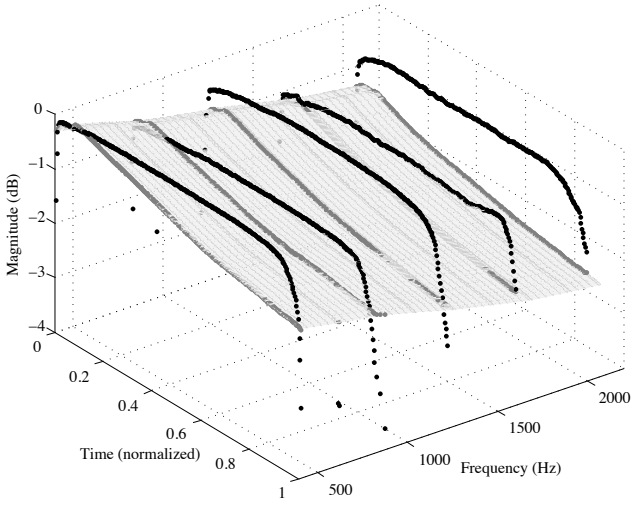


Figure 32: *Weak matching of an alto sax cluster and a portion of the piano prototype envelope.*

As argued in [Burred et al., 2006], depending on the application, it can be more convenient to perform further processing on the reduced-dimensional PCA space or back in the time-frequency domain. When classifying individual notes, a distance measure between unknown trajectories and the prototype curves in PCA space has proven successful. In the current source separation application, the clusters to be matched to the models can contain regions of unresolved overlapping partials or outliers, which can introduce important interpolation errors when adapted to the G-bin frequency grid needed for projection onto the bases. This makes working in the time-frequency domain more convenient in the present case.

Timbre Matching

Each one of the clusters obtained by the sound source separation step is matched against each one of the prototype envelopes. Let a particular cluster of K frames be represented as an ordered set of amplitude and frequency vectors $\mathbf{A} = (\mathbf{a}^1, \ldots, \mathbf{a}^K)$, $\mathbf{F} = (\mathbf{f}^1, \ldots, \mathbf{f}^K)$ of possibly differing lengths $L^1, \ldots, L^K$.

It becomes now necessary to evaluate the prototype envelope of model i at the frequency support of the input cluster j. This operation is denoted by $\tilde{\mathbf{M}}(i,j) = \mathbf{M}_i(\mathbf{F}(j))$. To that end, the time scales of both input and model are first normalized. Then, the model frames closest to each one of the input frames in the normalized time scale are selected. Finally, each new amplitude value $\tilde{m}_l^k(i,j)$ is linearly interpolated from the neighboring amplitude values of the selected model frame.

It is then possible to define the distance between a cluster j and an interpolated prototype envelope i as

$$d(\mathbf{A}(j), \tilde{\mathbf{M}}(i,j)) = \frac{1}{K(j)} \sum_{k=1}^{K(j)} \sqrt{\sum_{l=1}^{L^k(j)} (a_l^k(j) - \tilde{m}_l^k(i,j))^2} \tag{39}$$

i.e., the average of the Euclidean distances between frames of the input clusters and interpolated prototype envelope at the normalized time scale.

The model $\tilde{\mathbf{M}}(i,j)$ minimizing this distance is chosen as the predicted instrument for classification. Figure 32 shows an attempt to match a cluster extracted from an alto sax note and the corresponding section of the piano prototype envelope. As it is clearly visible, this weak match results in a high distance value.

Other alternative metrics could have been used (e.g. since the timbre models are in fact parametrically defined as probability distributions [Burred et al., 2006], the likelihood of a cluster corresponding to a specific template would be an interesting approach to implementing the matching procedure), but preliminary tests and the simplicity and computationally efficiency of the Euclidean distance ended up determining its choice for this specific evaluation.

4.6.3 Corpus Description and Experimental Setup

The current implementation of the sound source formation framework does still not fully take into consideration timing information and continuity issues, such as note onsets and durations (see the discussion on Section 3.8.1). Given so, the evaluation procedure will be limited to the separation and classification of concurrent notes sharing the same onset and played from different instruments.

The evaluation dataset was artificially created mixing audio samples of isolated notes of piano, oboe, clarinet, trumpet, violin and alto sax, all from the RWC Music Database [Goto et al., 2003]. The training dataset used to derive the timbre models for each instrument (as described in Section 4.6.2) is composed of audio samples of isolated notes, also from the RWC Music Database. However, in order to get meaningful timbre recognition results, independent instances of each instrument for the evaluation dataset and for the training dataset were used. Ground-truth data was also created for each mixture and includes information about the notes played and the corresponding instrument. Given that the timbre models used in this work have shown good results for a range of about two octaves [Burred et al., 2006], the notes used for evaluation were constrained to the

True instruments						*classified*
p	**o**	**c**	**t**	**v**	**s**	*as*
100	0	0	0	0	0	**p**
0	**100**	8	8	0	0	**o**
0	0	**67**	0	33	0	**c**
0	0	0	**92**	0	8	**t**
0	0	0	0	**58**	8	**v**
0	0	25	0	8	**83**	**s**

Table 10: *Confusion matrix for single–note instrument identification. Six different instruments from the RWC database were considered: piano (***p***), oboe (***o***), clarinet (***c***), trumpet (***t***), violin (***v***), alto sax (***s***).*

range C4 to B4. Furthermore, for simplicity's sake, only notes with a fixed intensity were considered for this evaluation .

4.6.4 Experimental Results

Several experiments were conducted for evaluating the system ability to identify sound events produced by specific instruments in a sound mixture. These evaluations range from a simple single-note instrument identification problem (as a way to define a baseline evaluation of the sinusoidal model and subsequent matching algorithm), go through the evaluation of the ability to detect the presence of a specific timbre in a sound mixture, to the final test where the system tries to perform event segregation and subsequent timbre recognition in signals with a varying number of notes playing simultaneously.

Timbre identification for single note signals

The first evaluation tested the performance of the timbre matching block (as described in Section 4.6.2) for the case of isolated notes coming from each of the six instruments modeled. This provides a base-ground with which it will be possible to compare the ability of the framework to classify notes separated from mixtures. For the case of isolated notes, the sound source separation block reduces its action to just performing sinusoidal analysis, since there are no other sources to be separated. This basically only results in the loss of the non-harmonic residual, which although not irrelevant to timbre identification, has been demonstrated to have a small impact in the classification performance [Livshin and Rodet, 2006].

Table 10 presents the confusion matrix for the instrument classification for a dataset of 72 isolated notes, ranging from C4 to B4, from each one of the six considered instruments. The system presents an overall classification accuracy of 83.3%, being violin and clarinet the instruments causing the biggest difficulties. This result is not as strong as the one

	2-note			3-note			4-note			total		
	RCL	**PRC**	**F1**	**RCL**	**PRC**	**F1**	**RCL**	**PRC**	**F1**	**RCL**	**PRC**	**F1**
p	83	100	91	22	100	36	0	0	0	23	100	38
o	100	75	86	100	46	63	67	40	50	86	50	63
c	33	100	50	33	100	50	40	86	55	36	93	52
t	89	100	94	58	100	74	58	64	61	67	85	75
v	67	67	67	83	45	59	83	36	50	80	43	56
s	100	43	60	67	60	63	60	75	67	67	62	64
total	**75**	**79**	**77**	**56**	**64**	**59**	**46**	**56**	**50**	**56**	**64**	**60**

Table 11: *Recall and precision values for instrument presence detection in multiple-note mixtures.*

published in [Burred et al., 2006], based on the same timbre models, where a 94.86% classification rate was presented. This difference may result from the use of independent datasets for training and testing in this work, instead of cross-validation over the entire dataset.

Instrument presence detection in mixtures of notes

In the next experiment the objective was to evaluate the ability of the system to separate and classify the notes from audio signals with up to 4 simultaneously sounding instruments. A combination of 54 different instruments and mixtures of 2-, 3- and 4-notes was created (i.e. 18 audio files for each case).

The first and simplest evaluation performed was to test the system ability to detect the presence of an instrument in a mixture of up to 4 notes. In this case it was just a matter of matching each one of the six timbre models with all the separated clusters and counting the *true* and *false positives* for each instrument.

A *true positive (TP)* is here defined as the number of separated clusters correctly matched to an instrument playing in the original mixture (such information is available in the dataset ground-truth). A *false positive (FP)* can be defined as the number of clusters classified as an instrument not present in the original audio mixture. Given these two values, it is then possible to define three performance measures for each instrument – *Recall (RCL)*, *Precision (PRC)* and *F-Measure (F1)*:

$$RCL = \frac{TP}{COUNT}, \quad PRC = \frac{TP}{TP + FP}, \quad F1 = \frac{2 \times RCL \times PRC}{RCL + PRC}, \tag{40}$$

where $COUNT$ is the total number of instances of an instrument over the entire dataset (i.e. the total number of notes it plays). As shown in Table 11, the system was able to correctly detect 56% of the occurrences of instruments in mixtures of up to 4 notes, with

a precision of 64%. Piano appears as the most difficult timbre to identify, specifically for the case of 4-note mixtures, where from the existing 15 notes playing in the dataset, none was correctly detected as coming from that instrument. As anticipated, the system performance degrades with the increase of the number of concurrent notes. Nevertheless, it was still possible to retrieve 46% of the present instruments in 4-note mixtures, with a precision of 56%.

Note separation and timbre identification in mixtures of notes

Although informative, the previous evaluation has a caveat – it does not allow to precisely verify if a separated and classified cluster does in fact correspond to a note played with the same instrument in the original audio mixture. In order to fully assess the separation and classification performance of the framework, an attempt to make a correspondence between each separated cluster and the notes played in the mix (available in the ground-truth) was conducted.

A possible way to obtain such a correspondence is by estimating the pitch of each one of the detected clusters, using the following simple technique. For each cluster the histogram of peak frequencies was calculated. Since the audio recordings of the instruments used in this evaluation are from notes with steady pitch over time (i.e. no vibrato, glissandos or other articulations), the peaks on the histogram provide a good indication of the frequencies of the strongest partials. Having the set of the strongest partial frequencies, another histogram of the differences among all partials was computed, and the highest mode as the best f_0 candidate for that cluster was selected.

Given these pitch correspondences, it is now possible to check the significance of each separated cluster as a good note candidate, as initially hypothesized. For the entire dataset, which includes a total of 162 notes from all the 2-, 3- and 4-note audio mixtures, the system was able to correctly establish a pitch correspondence for 55% of the cases (67%, 57% and 49% for the 2-, 3- and 4-note mixtures, respectively). These results can not however be taken as an accurate evaluation of the sound source separation performance, as they are influenced by the accuracy of the used pitch estimation technique.

The results in Table 12 show the correct classification rate for all modeled instruments and multiple-note scenarios, excluding the clusters whose correspondence was not possible to establish. This allows decoupling the source separation/pitch estimation performance from the timbre identification accuracy. Table 12 shows a correct identification rate of 47% of the separated notes overall, diminishing sharply its accuracy with the increase of concurrent notes in the signal. This shows the difficulties posed by the overlap of spectral

	Instrument Detection Rate			
	2-note	**3-note**	**4-note**	**overall**
p	67	67	0	55
o	100	86	60	81
c	33	29	19	26
t	75	33	22	43
v	67	100	50	75
s	75	36	42	44
total	65	50	33	**47**

Table 12: *Instrument classification performance for 2-, 3- and 4-note mixtures.*

True instruments						*classified*
p	**o**	**c**	**t**	**v**	**s**	*as*
55	0	0	4	0	0	**p**
27	**81**	21	30	8	15	**o**
9	0	**26**	0	8	3	**c**
0	6	5	**43**	0	32	**t**
0	6	28	4	**75**	6	**v**
9	6	21	17	8	**44**	**s**

Table 13: *Instrument classification confusion matrix for 2-, 3- and 4-note mixtures*

components from different notes/instruments into a single detected cluster. Table 13 presents the overall confusion matrix for the instrument classification.

4.7 Summary

This chapter presented a set experiments where the sound segregation framework proposed in Chapter 3 was evaluated.

The first experiments attempted to directly demonstrate the influence of the proposed perceptual grouping cues in the segregation performance of the system, both individually and in combination. In particular, the novel HWPS cue has been shown to compare favourably to two state-of-the-art harmonicity cues, and that its combination with other grouping cues, such as frequency and amplitude similarity, improved the overall separation performance.

In addition, experimental results for several MIR tasks using "real-world" polyphonic music signals showed that the use of segregated signals allows to achieve final results that compare or outperform typical and state-of-the-art audio content analysis systems, which traditionally represent statistically the entire polyphonic sound mixture.

Although not all of these experiments directly evaluate the separation performance of the proposed system using low-level measures, they allowed to indirectly assess the improvements achieved in specific application tasks.

Chapter 5

Software Implementation

"Everything should be made as simple as possible but not simpler."

Albert Einstein

5.1 Introduction

Attempting to do research on the topic of audio analysis and processing poses challenging demands on the development of software modules and tools so that any proposed hypothesis and algorithms can be objectively implemented and evaluated. In fact, the use of working software prototypes in research is essential to the validation and improvement of the implemented algorithms, ultimately allowing to create a *"self-improving feedback loop"* [Tzanetakis, 2002, p.10]. Inevitably, during the development of this thesis a great deal of work has been invested into software development, mainly focused on the sound segregation framework proposed and discussed in the previous chapters.

The following sections will discuss the requirements, choices and the major contributions towards the development of a software framework for the computational analysis of sound signals. However, a thorough and profound discussion about Software Engineering specificities is out of the scope of this thesis, and the reader will be redirected to the specialized literature whenever appropriate.

5.2 Design Requirements

The design and implementation of a multimedia signal processing software framework is challenging especially when efficiency or real-time performance is at stake. Given the huge

amounts of multimedia data available today (where audio and music content is included), real-world multimedia applications should be able to process content in a timely manner, otherwise their usefulness is compromised. In addition, if the framework is to remain useful with the passing of time, an effort should be put into designing it so that it makes the best use of the increasingly available computational power, allowing it to scale as well as possible with increasing amounts of data to process.

Furthermore, modularity and code reusability are usually highly desired attributes in a flexible software framework. This allows developers to perform rapid-prototyping of complex algorithmic processes or even develop fully fledged applications without the need to repeat from scratch the cycle of coding, testing, debugging and validating all the building blocks that make up such a project, being able instead to focus on the development of new and highly specialized software modules.

Another important feature is portability, which means that it should be possible to use the same software code in different conditions (e.g. as a console service running in a server, or as a GUI application providing easy interaction with the end user), architectures (e.g. x86, Power-PC, SPARC) and platforms (e.g. UNIX/Linux, MacOSX, Microsoft Windows). However, this requires the use of well established coding standards (e.g. ANSI C, portable C++, JAVA) and discourages the use of architecture and platform specific features and optimisations (e.g. assembly coding, platform specific libraries or technologies).

This issue is particularly difficult to tackle for the case of multimedia drivers (e.g. audio and MIDI drivers), whose technologies, implementation and APIs are traditionally distinct and incompatible between different operating systems and platforms (e.g. DirectShow in Microsoft Windows, CoreAudio in MacOSX, ALSA in Linux). This requires specific software implementations for each particular case, which can put an unacceptable burden into the development cycle. One possible solution is to make good use of external libraries (commercially or freely available) that already implement a common API for abstracting the end-user from all the platform-specific code calls (e.g. RTAUDIO and RTMIDI [Scavone and Cook, 2005]).

In fact, multi-platform external libraries are also available for other code modules, such as processing algorithms (e.g. FFTW[1]), input and output handling of file formats (e.g. libsndfile[2]), graphic user interfaces (e.g. Trolltech Qt4[3], GTK+[4]), numerical routines (e.g.

[1] http://www.fftw.org/
[2] http://www.mega-nerd.com/libsndfile/
[3] http://trolltech.com/products/qt
[4] http://www.gtk.org/

BLAS[5]), among others. Consequently, code reusability can additionally be achieved by integrating with already existing libraries for specific tasks, but some tradeoff should be exerted if the software framework is to remain with as limited as possible dependencies in order to avoid portability issues.

Interoperability with existing software packages or applications (e.g. MATLAB[6], WEKA[7], Python[8], Trolltech Qt4) is another valuable feature. This is different from the use of external libraries, as presented above. In this case there is no actual integration of external code into the framework (i.e. by means of some sort of static or dynamic linking with external libraries). Instead, interoperability simply implies the usage of the features provided by those packages or applications by means of some existing data communication interface (e.g. input/output of a common file format or some way of run-time interprocess communication using a vendor provided API) [Tzanetakis et al., 2008]. This has an added virtue of also making the software framework a potentially useful package those applications can interface and build upon.

A final but significant feature of a software framework is its code complexity. Avoiding highly complex or over-engineered software architectures (an aspect easily overlooked when dealing with feature-rich and sophisticated projects) may prevent an excessive burden on code understandability, creation and usability. This takes great importance when the framework aims at bringing non-experts in software engineering to contribute with the creation of valuable and specialized software modules, but who would otherwise be turned down by a steep learning curve or high complexity and programming overheads. However, this is always a compromise situation, where increasing flexibility and efficiency usually bring along added complexity.

5.3 Implementation Strategies

Different strategies may be adopted when facing the task of implementing software modules that are just a small part of a larger system.

[5] http://www.netlib.org/blas/
[6] http://www.mathworks.com/products/matlab/
[7] http://www.cs.waikato.ac.nz/ml/weka/
[8] http://www.python.org/

One possible approach would be to use commercially available platforms (e.g. MATLAB, Simulink[9], LabView[10], MAX/MSP[11]). Traditionally, these software packages already provide some ready-to-use building blocks and routines for commonly known and used tasks (which may range from basic mathematical or numerical routines to advanced signal processing algorithms), and allow in most of the cases coding new user-defined ones. However, and at the light of the discussion in the previous section, it may not always be easy to find a commercial software package that perfectly fits all the requirements posed by a specific research task. And since the majority of the available commercial software packages is released as closed source, it may become difficult to tailor the platform to the specific needs of the project at hands. Finally, the cost of purchasing and maintaining a commercial software license is often high, turning such solutions unsustainable for a low-budget or unfunded projects.

Another option would be to write all the software from scratch and find some way to integrate the different software modules into a working system. If good Software Engineering practices are followed, this would usually require the definition of a system architecture that should take into consideration the design requirements of the project. It would demand taking the time and effort to implement, test and evaluate the system and all its processing modules. Although potentially complex and challenging (both in time and in expertise), this approach has obvious advantages: it allows defining and fine-tunning the software architecture taking into consideration any specific requirements, staying in complete control of the way the software is designed and subsequently implemented. The learning experience gained from undertaking such an endeavor would also be a valuable added bonus.

In fact, some of these individual or team efforts have been released to the community as Free[12] and Open Source[13] Software (FOSS) projects. Successful examples include projects like the Synthesis ToolKit (STK)[14] [Cook and Scavone, 1999], MARSYAS[15] [Tzanetakis, 2008], CLAM[16] [Amatriain, 2007], Aubio[17], PureData (Pd)[18],

[9] `http://www.mathworks.com/products/simulink/`
[10] `http://www.ni.com/labview`
[11] `http://www.cycling74.com/products/maxmsp`
[12] As defined by the *Free Software Foundation* (`http://www.fsf.org/`)
[13] As defined by the *Open Source Initiative* (`http://www.opensource.org/`)
[14] `http://ccrma.stanford.edu/software/stk/`
[15] `http://marsyas.sourceforge.net`
[16] `http://clam.iua.upf.edu/`
[17] `http://aubio.org/`
[18] `http://crca.ucsd.edu/~msp/software.html`

"The RESPITE CASA Toolkit Project" (CTK)[19], among others[20].

These projects are an interesting opportunity for carrying on existing work and contributing back with any original or relevant achievements (be it software, algorithms or improved results). This usually ends up originating a positive feedback loop of contributions of software modules, tools, collections of routines and even complete frameworks around a field of study. An entire community may end up using the software and putting it to the test, eventually reporting any found deficiencies or limitations, posting new feature requests, or even becoming an active collaborator of the project.

Specially relevant when used for research, FOSS allows using code as a means of communication, where publications can not possibly describe all the nuances and details of how an algorithm is implemented. At the same time, replication of experiments is essential for progress in research especially in new emerging areas such as MIR and CASA. For complex systems and algorithms it is almost impossible to know if a reimplementation is correct and therefore the ability to run the original code is crucial.

Finally, FOSS solutions have a lower cost when compared to proprietary or comercial software, a particularly important point given that traditionally researchers have limited financial resources.

5.4 Developing with MARSYAS

At the light of the previous discussions, and taking into account the past experience from doing research using both commercial software packages (i.e. MATLAB) [Martins, 2002, Martins and Ferreira, 2002] and open source software frameworks (i.e. MARSYAS-0.1 [Tzanetakis and Cook, 2000]) [Kotti et al., 2006a, Kotti et al., 2006b], a decision was made to look for a suitable FOSS framework.

Among the available options, MARSYAS-0.2[21], a recent version of the original MARSYAS-0.1 framework, stood out as a strong candidate [Tzanetakis, 2008, Tzanetakis et al., 2008]. As presented in Appendix C, MARSYAS-0.2 meets most of the discussed requirements for a research software framework and the previous acquaintance with the original version implied a smoother learning curve and a good prospect of easily reusing most of the code previously developed. All things considered, MARSYAS[22] was

[19] `http://www.dcs.shef.ac.uk/spandh/projects/respite/ctk/`

[20] For an extensive list of the most used software tools in the area of MIR and CASA (both FOSS and commercial), visit: `http://www.music-ir.org/evaluation/tools.html`

[21] `http://marsyas.sourceforge.net`

[22] For simplicity sake, from now on the more compact MARSYAS designation will be used when referring to the specific MARSYAS-0.2 version of the framework.

the software framework of choice.

Given the free and open source nature of MARSYAS, this decision opened the opportunity for the involvement in the development of the framework. This resulted in a close collaboration with the MARSYAS development team (which at the time was basically composed by MARSYAS original creator, George Tzanetakis, but that has since then grew into an actual group of people working around several aspects of the software framework[23]). Their openness and availability for discussing and integrating new ideas into the framework gave space for most of the software development contributions that will be described in the following sections.

5.4.1 Contributions to the MARSYAS Framework

MARSYAS provides a general, extensible and flexible framework that enables the easy and efficient combination of a variety of existing building blocks as dataflow components which ultimately allow to implement efficient audio analysis and synthesis tools. Appendix C presents the base architecture of MARSYAS and some of its key features (e.g. Implicit Patching and Composition architecture). It also provides a detailed description of the MARSYAS basic processing blocks, composite modules, dynamic access and linking to controls and modules as well as the interoperability with other software platforms [Tzanetakis et al., 2008].

As a result, MARSYAS provided a solid software base upon the sound segregation framework proposed in this thesis was implemented. However, given the challenging and specific requirements posed by the proposed sound segregation framework, its implementation as a MARSYAS processing network asked for the substantial development of new processing (e.g. an HWPS module) and composite modules as well as some changes, optimizations and some profound and extensive refactoring of the core MARSYAS code. The following list summarizes some of the main contributions to the MARSYAS software framework conducted in the scope of this thesis.

Refactoring and optimization of the `MarSystem` update mechanism. The complexity of the MARSYAS networks required for the implementation of the sound source segregation system proposed in this thesis (see Figures 33 to 38) posed new challenges to the way the MARSYAS data flow configuration mechanism worked. Changes in the data flow configuration (e.g. different number of peaks at each texture window) had now to be handled dynamically and efficiently at runtime, requiring the optimization of the update routines and a redefinition of the `MarSystem` API.

[23] For more information about the MARSYAS core development team, visit `http://marsyas.sf.net/`

Refactoring and optimization of MARSYAS controls. Controls in MARSYAS (i.e. the objects `MarControl`, `MarControlValue`, `MarControlPtr`) are now implemented using C++ *templates*, increasing code generality and flexibility while keeping read and write accesses to controls fully encapsulated.

Refactoring and optimization of the control linking mechanism. The linking of controls is a powerful mechanism which allows to specify parameters of processing modules as well as to pass information among modules in the processing network. However, the way they were originally implemented made it hard to achieve a flexible management and reconfiguration of intricate linking configurations such as the ones required by the sound segregation system proposed in this work. Links between controls are now directed links, allowing to easily reconfigure complex linking schemes between child and Composite `MarSystem`'s. Furthermore, a profound change in the way the linked control values are kept in sync was implemented: linked controls now point to a same shared memory object, facilitating the synchronization between writing and reading operations and optimizing memory usage (important when linking controls which store big real valued matrices, such as similarity matrices).

New Composite `MarSystem`'s. Several new *composite* `MarSystem`'s have been implemented. Two of the most important ones (see Figure 36) are the `FanOutIn` (which is the key element in the flexible way similarity cues can now be combined) and the `SimilarityMatrix` composite (which was implemented as a generic similarity matrix computation module, already proven usable out of the scope of this work – e.g. [Tzanetakis et al., 2007]).

New helper classes. The use of complex data structures encoded in MARSYAS `realvec`'s required the implementation and use of helper classes, such as the `peakView`. This class provides a simple and concise API for handling peak data structures (see Figure 34/**B**) encoded in a `realvec`.

The complexity and time requirements of this endeavor demanded the involvement and commitment of the entire MARSYAS core development team at different stages of the development. At the end, this effort allowed the implementation of the sound segregation framework proposed in Chapter 3 and to subsequently conduct the evaluation experiments presented in Chapter 4.

Furthermore, because it is designed as a flexible and efficient platform, MARSYAS allows the current sound segregation framework to be easily expanded to include future ideas and algorithms, such as the ones anticipated during the course of this work, and compiled in Section 6.2.

The following sections will present and describe the main aspects of the implementation of the sound segregation framework proposed in this thesis as a MARSYAS processing network. It is assumed that the reader is sufficiently familiarized with the MARSYAS platform. Detailed information about the MARSYAS specificities can be found in the documentation[24] and in Appendix C.

5.5 Implementing the Sound Segregation Framework in MARSYAS

Figure 33 shows the overall MARSYAS network that implements the system proposed in this thesis (see Chapter 3). Each dark box in the figure represents a basic `MarSystem` while the white boxes represent *composite* `MarSystem`'s. Optional `MarSystem`'s are represented as light gray boxes, and their existence is not mandatory for the base purpose of the system, being used for added functionality. For the sake of clarity, Figures 35 and 36 present detailed views of some of the more complex composite `MarSystems`, using numbers inside black circles to cross-reference the different views of a same composite along the different figures. In all figures, a *link* between two controls is represented by dashed directed lines (whose orientation represents the original linking direction).

As described in Appendix C, MARSYAS uses two-dimensional real valued matrices (known as `realvec`'s) for expressing data flowing in a network. However, the data at each point in the network can be very different, depending on the particular `MarSystem` producing it (e.g. it can be used to represent time domain audio samples at the output of an audio file source, or a complex spectrum data at the output of an FFT block). As a result, Figure 34 presents schematic views on how data is expressed using `realvec`'s at some relevant points in the network. Capital letters inside dark and bracketed squares are used as cross-references between all the figures.

5.5.1 Implementing Texture Windows

The first block in the main `Series/mainNet` composite `MarSystem` (see Figure 33), named `Accumulator/textWinNet`, is a composite module where audio frames are read from an audio file (by the `Series/oriNet`), their spectrum computed (in `Parallel/par`, and as discussed in the following sections) and the resulting outputs accumulated into texture windows. This is implemented using an `Accumulator` composite `MarSystem`.

[24] Available at the MARSYAS website (`http://marsyas.sf.net`) or in the code distribution.

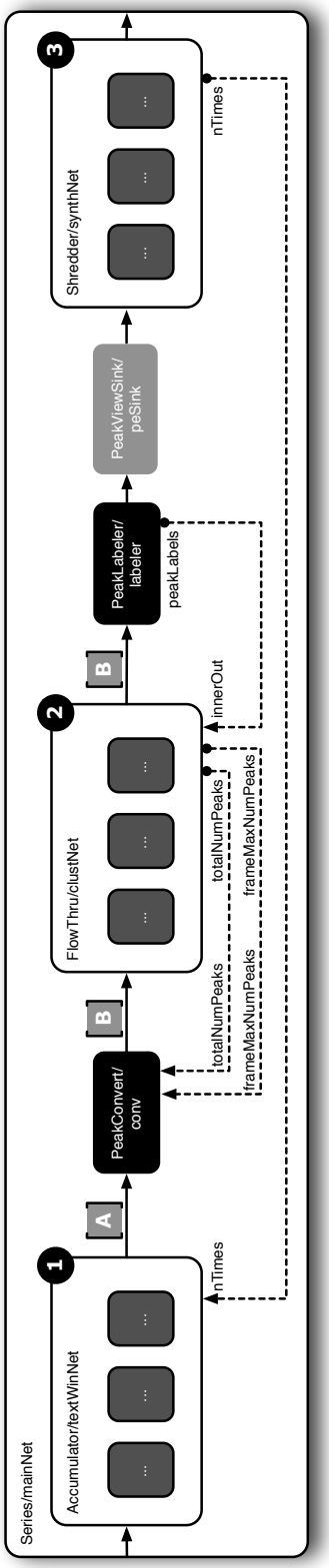


Figure 33: *Overview of the MARSYAS network used for the sound segregation framework proposed in this thesis. Detailed views of the blocks signaled with numbered dark circles and the data structures signaled with letters inside gray squares are presented in Figures 33, 35, 36, 37 and 38.*

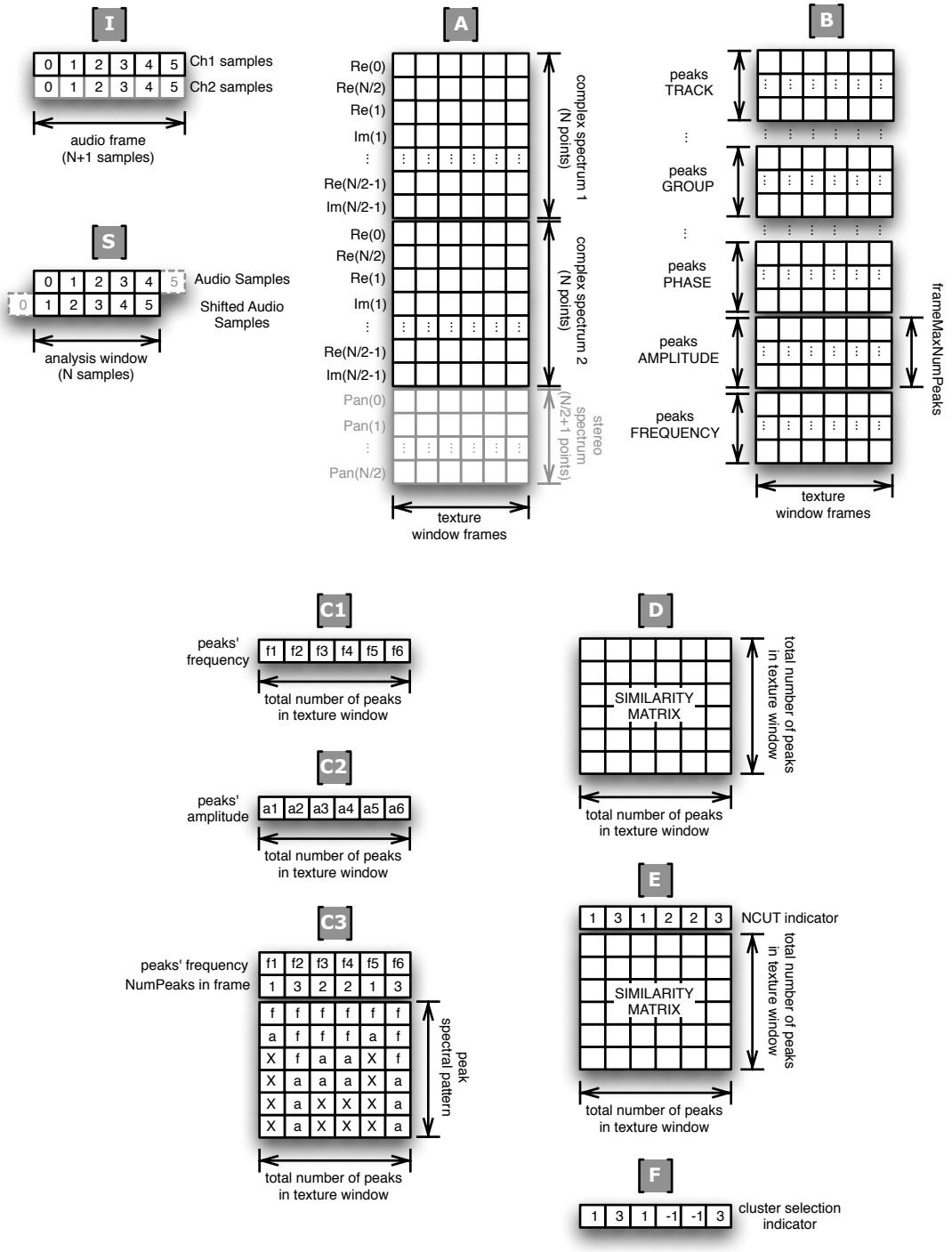


Figure 34: *Graphic representation of the data structures used at different points of the data flow of the MARSYAS networks depicted in Figures 33, 35, 36, 37 and 38.*

Whenever *ticked*, an `Accumulator` iteratively *ticks* all its children modules, accumulating their output into an internal memory buffer. When the number of internal *ticks* reaches a specified value, or when the accumulator is explicitly *flushed*, the internal *ticking* is stopped and the accumulated results are sent to its output for further processing by the following module (in this case the `PeakConvert/conv` module, described later).

Accordingly, and following the discussion in Section 3.5, the length (in frames) of each texture window can be set either as a fixed value, by setting the control `nTimes` of the `Accumulator` to some integer value (e.g. 10 frames), or dynamically, based on the onsets detected in the signal. In such a case, an optional onset detector implemented in `FlowThru/onsetdetector` (see the composite network labeled as **1a** in Figure 35) will automatically toggle the `Accumulator` `flush` control whenever an onset is detected (notice the link between the `onsetDetected` control in `PeakerOnsetPeaker/peaker` and the `flush` control of the `Accumulator/textWinNet`).

The size (in samples) of each audio frame is specified in the `ShiftInput/si` block (see Figure 35), by setting its `windowSize` control to the desired number of samples. The hop size is defined by setting the number of samples to be read at each *tick* by the `SoundFileSource/src` inside `Series/oriNet`. Given the way the MARSYAS architecture propagates data flow configurations along a network (see Appendix C and the MARSYAS documentation[25]), the correct way to specify this number is by setting the `inSamples` control of the top most composite, i.e. `Series/mainNet` (see Figure 33).

An optional audio source is included in `FanOutIn/mixer` (see the `Series/mixSeries` composite in Figure 35), but is only used for SNR evaluations and experiments not important for the sound segregation process. It is nevertheless depicted in the figure for the sake of the correspondence with the C++ MARSYAS code.

5.5.2 Implementing Spatial Cues from Stereo Audio Signals

Although this thesis does not discuss the use of spatial proximity cues for sound segregation, the proposed sound segregation framework has been presented as being flexible enough to allow the implementation of spatial localization cues.

In fact, Figure 35 depicts some of the modules already implemented in the current MARSYAS system (see the `Series/stereoSpkNet` composite), which are enabled when taking a stereo audio file. These modules implement an azimuth discrimination technique known as *ADRess*, proposed in [Cooney et al., 2006] (another similar stereo technique proposed in [Avendano, 2003] is also implemented in MARSYAS and can be easily used

[25] `http://marsyas.sf.net`

in a similar way).

The output of this composite is a spectrum representation where a pan or azimuth coefficient is computed for each spectral bin (see the lower section of the data structure with the label **A** in Figure 34) [Tzanetakis et al., 2007]. The idea is that bins in the spectrum with similar azimuth values are probably originating from a same source, becoming a good proximity cue for sound segregation. Further evaluations and experiments still need to be conducted in order to demonstrate the relevance of spatial cues in the overall segregation performance of the proposed system.

5.5.3 Implementing Sinusoidal Modeling

The framework proposed in this thesis is based on the use of a sinusoidal model of the audio signals under analysis, where precise estimates of the frequency, amplitude and phase of the most prominent peaks in the spectrum are computed (see Section 3.4). The specific technique used to obtain these precise estimates of each peak requires the calculation of the spectrum of two audio analysis windows both with a size of N samples but shifted by a single sample.

A way to achieve this in MARSYAS is to set the output frame size in `ShiftInput/si` (see Figure 35) to $N + 1$ samples (i.e. by setting its `windowSize` control to the corresponding value – see the **I** data structure in Figure 34). Afterwards, a `Shifter` module can be used and configured to perform a shift of one sample, resulting in two shifted versions of the initial audio frame, as depicted by the data structure **S** of Figure 34.

After applying a Hamming windowing function (performed by the `Windowing/win` block) to the two shifted analysis windows, they can now be used to compute two complex spectra (performed by `Spectrum/spk1` and `Spectrum/spk2` in `Parallel/par`, as shown in Figure 35, respectively).

The two resulting spectra (stacked on top of each other and accumulated over the specified texture window as depicted in the data structure **A** of Figure 34) can now be sent to the following processing module, `PeakConvert/conv` (see Figures 33 and 35). This module will use the provided spectral information to estimate the precise frequency, amplitude and phase of a user-specified number of the highest amplitude peaks from each frame in the texture window (e.g. 20 peaks per frame as used in the current system implementation, specified by setting the `frameMaxNumPeaks` control of the `PeakConvert` module).

Additionally, and in the case of a stereo signal, the optional azimuth spectrum information (see the lower section of the data structure with the label **A** in Figure 34) could

also be used by `PeakConvert/conv` to add panning information to each detected peak.

Encoding Peak Information into a MARSYAS `realvec`

The `PeakConvert` module outputs several features for each detected peak in each frame (up to the user-defined `frameMaxNumPeaks` value), including their precise frequency, amplitude, phase, panning (in case of a stereo signal). The way `PeakConvert` encodes such information into an output `realvec` is depicted in the data structure **B** in Figure 34, where each column represents a frame, and the peak's features are organized in groups of rows (where each peak maintains its relative position inside each feature group of rows).

Several other features are also provided for each peak in this structure, such as the group (i.e. the cluster) each peak belongs to (initially set to zero but later filled by the Ncut modules when clustering peaks in a texture window – see Section 5.5.5), their panning and volume and support for future features (e.g. the track each peak belongs to, allowing to encode trajectories resulting from some type of McAulay-Quatieri processing [McAulay and Quatieri, 1986] into this representation).

In order to avoid potential memory reallocations and network reconfigurations at each processing *tick* (which would drastically hurt the computational efficiency of the network) the information is encoded in the data structures using a fixed memory footprint. For instance, for the case of the **B** data structure, its memory allocation is set to accommodate the specified maximum number of peaks per frame (i.e. `frameMaxNumPeaks`). When the number of detected peaks in a frame is lower than the specified maximum number of peaks per frame, the remaining peak slots in the data structure are encoded as peaks with a zero frequency value, meaning that all the remaining features are irrelevant and should be ignored.

In order to simplify the access and interaction with this somewhat complex data structure, a `peakView` viewer class was implemented in MARSYAS (look for the `peakView.h/.cpp` C++ files in the MARSYAS source code). It provides utility and helper methods that simplify the encoding of the information about the peaks in the frames of a texture window into a generic `realvec` data structure, as depicted in Figure 34/**B**. The interested reader should refer to the source code for more detailed information.

5.5.4 Implementing Similarity Cues

The sinusoidal model information provided by the module `PeakConvert` (described in the previous section) can now be used to compute different similarity metrics, which will

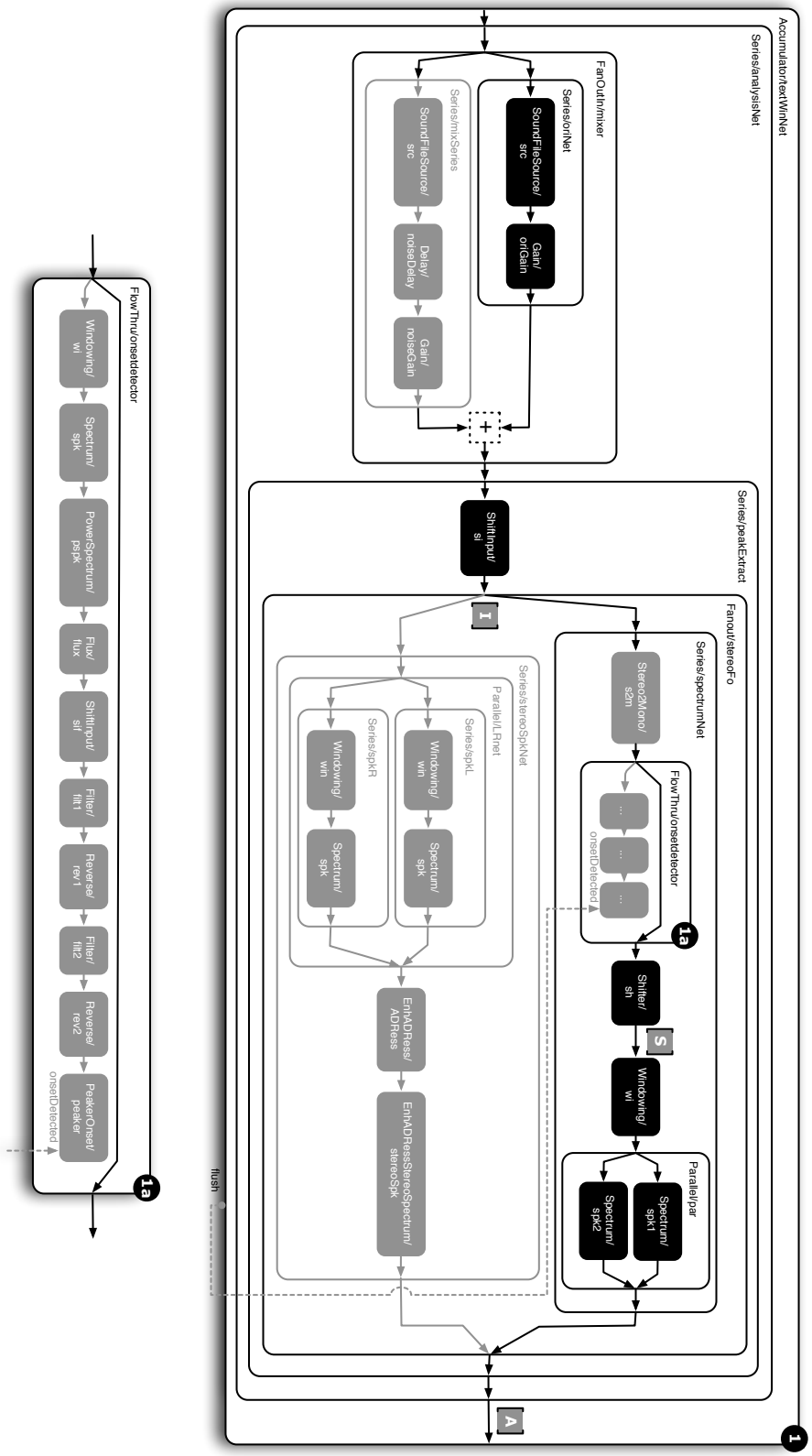


Figure 35: *Detailed view of the MARSYAS network used for the computation of the texture windows and the corresponding spectral representations used as the base for the sinusoidal modeling of the audio signals. A detailed view of the optional MARSYAS network which implements the onset detector used for the dynamic adjustment of the texture windows is depicted at the bottom of the figure.*

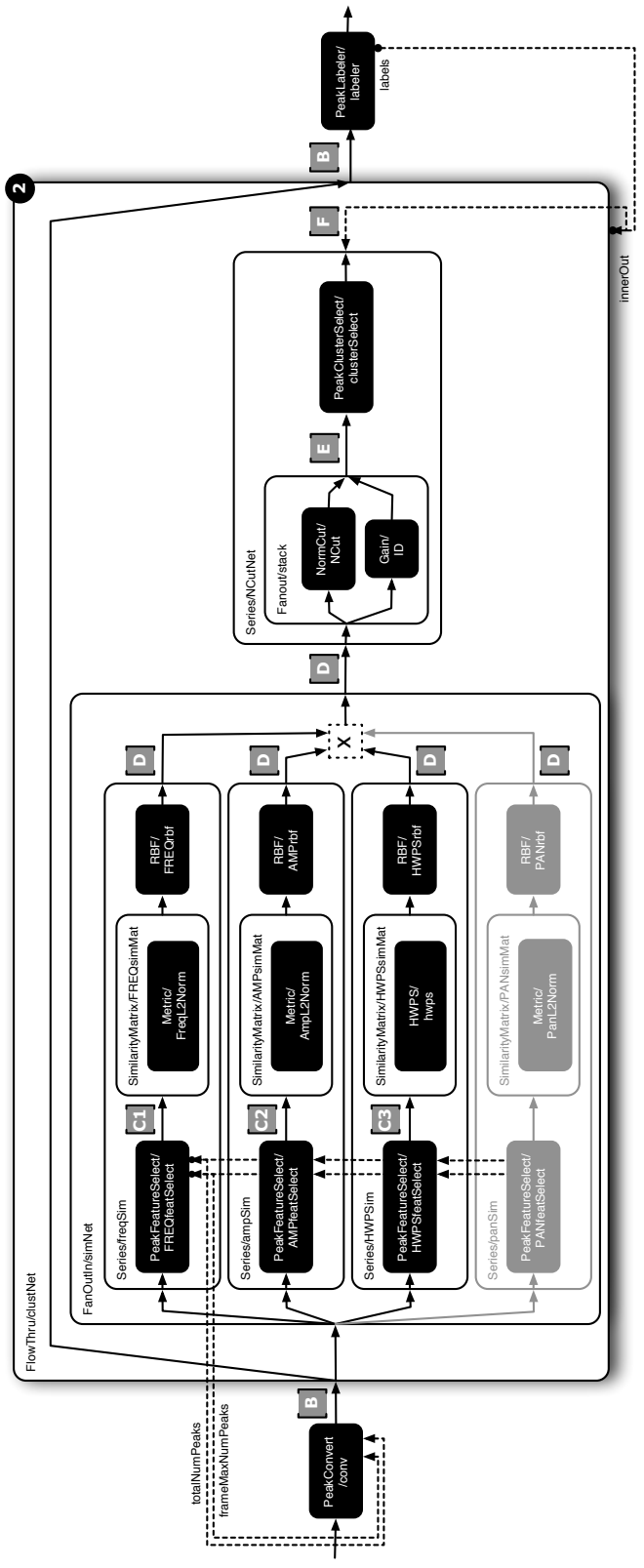


Figure 36: *Detailed view of the MARSYAS network used for the computation of the different similarity cues and the subsequent grouping process. This composite and hierarchical architecture is flexible enough to support the future inclusion of new similarity cues and the expression of more complex combination schemes of similarity cues (as exemplified in Figure 37).*

ultimately result in similarity matrices that encode the pairwise proximity between all peaks in each texture window (as discussed in Section 3.6.2).

This similarity computation is performed in the `FlowThru/ClustNet` composite (see Figures 33 as well as the more detailed block diagram in Figure 36). Following the discussion in Section 3.7, the current system implements frequency, amplitude and HWPS (i.e. harmonicity) similarities, as shown in the `FanOutIn/simNet` composite depicted in Figure 36.

Although not discussed in this thesis, a spatial proximity cue (only enabled when analyzing stereo signals) is also implemented and depicted in the figure (in the module `Series/panSim`). It is presented here as an example of the flexibility of the proposed framework to easily incorporate additional grouping cues into the sound segregation process. This allows to anticipate the straightforward addition of new similarity cues to the system, such as common onsets/offsets (e.g. using peak trajectories [McAulay and Quatieri, 1986]), timbre models, among others.

In the current system, each similarity computation is performed in their own branch of the `FanOutIn/simNet` composite, all implemented in a similar manner. From the peak information at the input, the required features for the computation of the frequency, amplitude and HWPS distances are first selected by the corresponding `PeakFeatureSelect` modules, outputting the data structures **C1**, **C2** and **C3** depicted in Figure 34, respectively. **C1** and **C2** are basically vectors containing all the frequency and amplitudes of each peak in the texture window. Differently, **C3** contains additional information given that the HWPS computation requires not only the frequency of each peak in the texture window but also the corresponding spectral pattern (as described in Section 3.7.2). The spectral pattern for each peak is encoded in **C3** as the number of peaks in each peaks' frame as well as their frequencies and amplitudes.

The `PeakFeatureSelect` modules need to know beforehand the total number of peaks in each texture window (which may vary over time), as well as the specified maximum number of peaks per frame. The latter was already specified in the `PeakConvert/conv` (see the previous section) and therefore a simple link between the controls `frameMaxNumPeaks` in `PeakConvert/conv` and each one of the `PeakFeatureSelect` modules allows to easily and automatically configure the similarity computation network (see Figure 36). Regarding the total number of peaks at each texture window, such information is only available while actually processing the

input audio data, and is expected to vary over time. The `PeakConvert/conv` module should therefore inform the `PeakFeatureSelect` modules about the actual number of peaks at each texture window (i.e. at each *tick*). Once again, the use of links between the `frameMaxNumPeaks` controls of `PeakConvert/conv` and the different `PeakFeatureSelect` modules enable to efficiently pass this information down the network. Given the MARSYAS flexible and dynamic dataflow architecture, the network can then automatically and efficiently reconfigure itself at each *tick* accordingly.

Having all the required features for each peak in the texture window currently under analysis, the `SimilartyMatrix` block at each similarity branch is then responsible for the pairwise computation of some distance metric between all peaks in the texture window, generating at its output the corresponding distance matrix. In fact, and in order to allow greater flexibility, the `SimilartyMatrix` module was designed as a composite `MarSystem`. This allows it to use a `Metric` or a `HWPS` module (or any other distance algorithm module) as a child, where the distance computation is actually performed. Different metrics are provided by the `Metric` module (e.g. Euclidean, Mahalanobis, cosine), while the `HWPS` provides a novel and more specialized harmonicity distance computation, as described in Section 3.7.2.

The `SimilarityMatrix` module also provides the ability to compute normalization of the data at its input, either based on the mean and standard deviation values (estimated from the input data or manually set by means of controls), or based on its maximum and minimum values. Diagonal of full covariance matrices can also be computed, and these values are automatically passed using controls to the child `Metric` (or any other) module, allowing, for e.g. to compute both a simple or standardized Euclidean distance. This mechanism provides the flexibility to perform the normalization of the amplitude and frequency values of the peaks to the range $[0, 1]$, as well as to manually set the neighbouring widths σ used for each similarity function (see the detailed discussion about this subject in Section 3.7).

Following the computation of the pairwise distances between all peaks, the `RBF` modules at each branch are used to apply a Gaussian function to the values (i.e. the `RBF` modules implement a Gaussian radial basis function of the form $y = e^{-x^2\beta}$, where β is set to 1.0 – other functions, such as multi quadratic or thin plate spline [Carlin, 1992] are also implemented and can be selected using the `RBFtype` control). As defined in Section 3.6.2, this results in the final similarity function (see Equation 8) and the output of each branch will be a square similarity matrix corresponding to the pairwise amplitude, frequency, HWPS and azimuth (for stereo signals) peak similarity values, respectively (see

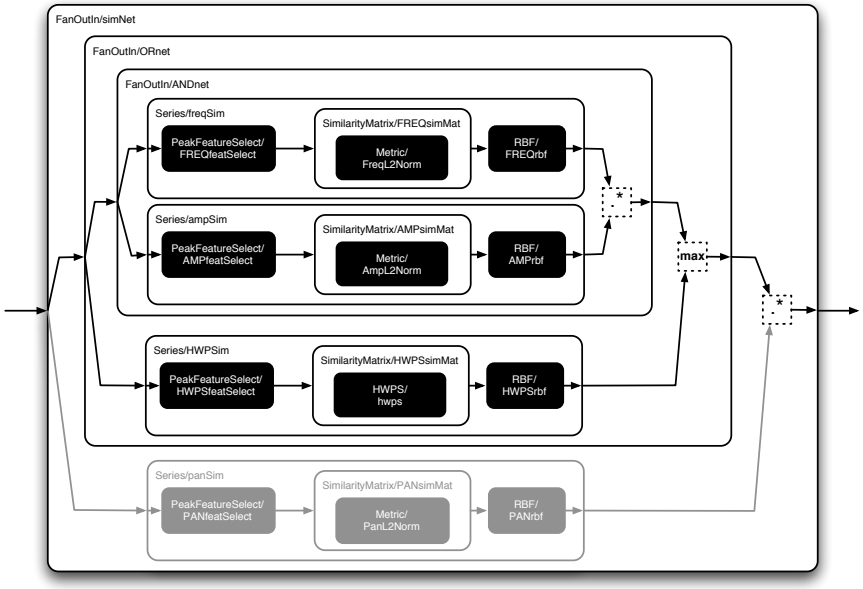


Figure 37: *The hierarchical composition and Implicit Patching architecture of MARSYAS allows to efficiently express complex combination schemes of similarity cues by using different instances of* `FanOutIn` *composite* `MarSystem`*'s. The figure shows an illustrative composition expressing* $W_{afh} = max[(W_f .* W_a), W_h] .* W_s$ *(where the* $.*$ *operator represents the element-by-element matrix multiplication).*

Figure 34/**D**).

The `FanOutIn/simNet` composite module will now combine the individual similarity matrices into an overall matrix, which will be used by the following NCut modules for peak grouping. As discussed in Section 3.7.3, this combination of the individual matrices is currently done by means of a multiplication operation, but the `FanOutIn` composite already provides other means of combination, such as the sum of the individual values of each matrix or the choice of the maximum or minimum values at each matrix position.

Furthermore, the *hierarchical composition* and *Implicit Patching* architecture of MARSYAS allows to easily and efficiently define more complex similarity combination schemes by hierarchically combining several instances of `FanOutIn` modules. Because each `FanOutIn` instance allows to use a different combination operation, it becomes straightforward to efficiently express more perceptually motivated similarity combination schemes (see Figure 37 and recall the discussion in Section 3.7.3).

5.5.5 Implementing the Grouping Algorithm

The grouping algorithm described in Section 3.6.5 is implemented in the module `NormCut/NCut` (see Figure 36). It takes the overall similarity matrix for each texture window (see previous section) and outputs an indicator vector where a cluster label (i.e. a number in the range $[1, k]$, where k is the specified number of clusters – see Section 3.6.3) is assigned to each peak, which represent the grouped peaks.

Given the way clusters are selected in the current system (see Section 3.8.1), the peaks indicator vector output by the `NormCut/NCut` module is stacked on top of the input similarity matrix (see Figure 34/**E**) before being sent to the following cluster selection module, `PeakClusterSelect/clusterSelect`. This module uses the indicator vector output by the normalized cut module as well as the similarity values in order to select the "denser" clusters, in a number which can be manually specified by means of its `numClustersToKeep` control.

The `PeakClusterSelect/clusterSelect` outputs a modified indicator vector, where the selected clusters will preserve their numeric label, while the remaining will be set to -1 (see Figure 34/**F**).

The selected clusters indicator vector is then used by the subsequent `PeakLabeler/labeler` module (see Figure 36), which is the module responsible to fill the corresponding cluster information into the `GROUP` field of the original peak feature data structure (see Figure 34/**B**).

This is efficiently achieved by passing the final indicator vector using the `innerOut` control of the `FlowThru/clustNet` composite, which is linked to the `labels` control of `PeakLabeler/labeler`, while the data structure containing the original peak information is passed in the data flow (see Figure 36).

Following this procedure, the final clustered peaks information can optionally be saved to a text file using the `PeakViewSink/peSink` module (see Figure 33) and then sent to the resynthesis modules, described next.

5.5.6 Implementing the Sound Event Resynthesis

Once the clusters resulting from the grouping process are identified and selected (see previous section), it is now possible to resynthesize them as independent sound signals. The resynthesis of the clustered and selected peaks is performed at the `Shredder/synthNet` composite (see Figure 33 and a more detailed view in Figure 36).

The `Shredder` composite `MarSystem` takes texture windows and "shreds" them

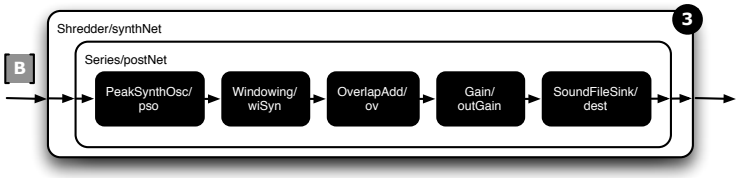


Figure 38: *Sound event resynthesis MARSYAS network, based on an additive synthesis approach and implemented as a bank of sinusoidal oscillators using an overlap-add scheme.*

into individual frames (i.e. separates the data structure in Figure 34/**B** into individual column vectors). This requires that the `Shredder` module to be in sync with the `Accummulator/textWinNet` module, where the texture windows are created. This is achieved by simply linking their `nTimes` controls, as depicted in the figures.

For each frame, and as discussed in Section 3.8.2, it is now possible to synthesize the audio signals corresponding to each cluster of peaks (i.e. to each one of the sound "sources" segregated from the mixture signal). The synthesis is done using a bank of sinusoidal oscillators implemented in `PeakSynthOsc/pso` (see Equation 1) and using a 50% overlap-add scheme carried away by the `Windowing/wSyn` and `OverlapAdd/ov` modules.

It is now possible to resynthesize each one of the segregated groups of peaks into independent audio files, but the current system is configured by default to just output a single audio stream where the "denser" clusters selected by the preceding `PeakClusterSelect` block are mixed together (see previous section).

At the end, the resulting segregated audio data is finally written to an audio file by the `SoundFileSink/dest` block (using e.g., the WAV format).

5.5.7 The `peakClustering` Tool

The sound segregation system whose implementation as a MARSYAS network was described in the previous sections is available as a command line application known as `peakClustering`. Its source code can be found in the MARSYAS distribution in the `src/apps/peakClustering` path.

After building MARSYAS (instructions for building MARSYAS in several platforms are provided in detail in the documentation), it is possible to experiment with the current system implementation by running `peakClustering`, found at `bin/release`, as follows:

```
> peakClustering myAudioFile.wav
```

After completing execution, the program returns two files: an audio file (in this example, named `myAudioFileSep.wav`) which is the result of the resynthesis process explained in Section 5.5.6 and a text file (named for this example, `myAudioFile.peak`) which lists all the peaks in the signal and their corresponding features (i.e. time frame, frequency, amplitude, phase, group – see Figure 34/**B**) in a table format.

By running `peakClustering` without any options, the default parameters are used (e.g. maximum number of peaks per frame is set to 20; number of clusters k is set to 5, from which the 2 "denser" clusters are selected at each texture window for resynthesis; texture windows are set to have a fixed length and equal to 10 frames). Several other options exist and they can be consulted by calling `peakClustering` with the `-h` option.

The `peakClustering` although being quite efficient, still does not run in realtime (mainly depending on the specified texture window length and the maximum number of peaks extracted per frame). Preliminary code profiling indicate that most of the computational cost is taken by the Normalized Cut calculations. Future code optimizations and different algorithms (e.g. [Dhillon et al., 2007]) may allow achieving realtime sound segregation of real world complex music signals.

5.6 Summary

Following the proposal of a sound segregation framework proposed in Chapter 3, this chapter discussed some of the requirements, choices and the major contributions towards the development of an open source software platform (based on the MARSYAS project) for the computational analysis of sound signals.

Some implementation details about the main building blocks of the sound segregation framework proposed in this work were described, where the efficiency, flexibility and code reusability aspects taken into consideration during the software development, were highlighted.

Chapter 6

Conclusions

"About the time we think we can make ends meet, somebody moves the ends."

Herbert Hoover

6.1 Results and Contributions

This dissertation proposed a flexible and extensible CASA framework for modeling perceptual grouping in music listening. The goal of the proposed framework is to partition a monaural acoustical mixture into a perceptually motivated topological description of the sound scene (similar to the way a naive listener would perceive it) instead of attempting to accurately separate the mixture into its original and physical sources.

The presented framework is data-driven and takes the view that perception primarily depends on the use of low-level sensory information, and therefore requires no training or prior knowledge about the audio signals under analysis. It is however flexible enough to allow the use of prior knowledge and high-level information in the segregation procedure. Designed to be causal and efficient (close to real-time on a fast computer), the resulting system can be used to segregate sound events in a complex polyphonic mixture of "real-world" music signals, paving the way to applications such as the predominant melody line detection, separation of the singing voice, instrument segregation and identification, voicing detection, among others.

The proposed system is based on a sinusoidal modeling analysis front-end, from which spectral components are segregated into sound events using perceptually inspired grouping cues. Grouping cues based on amplitude, frequency and harmonicity are incorporated in an unified optimization framework, and a novel harmonicity similarity measure, termed

"Harmonically-Wrapped Peak Similarity" (HWPS), was also proposed. The segregation process is then based on spectral clustering methods (in particular in the Normalized Cut criterion) for graph partitioning, a technique originally used to model perceptual grouping tasks in the computer vision field.

Experimental validation of the perceptual grouping cues have shown that the novel HWPS cue presented in this dissertation compare favourably to other state-of-the-art harmonicity cues, and that its combination with other grouping cues, such as frequency and amplitude similarity, improved the overall separation performance. In addition, experimental results for several MIR applications were presented. The use of segregated signals in these tasks allowed to achieve final results that compare or outperform typical and state-of-the-art audio content analysis systems, which traditionally represent statistically the entire polyphonic sound mixture.

In this dissertation a specific implementation of the sound event segregation framework is also presented, where several assumptions had to be considered due to practical constraints. Nevertheless, the proposed framework was designed to be modular, efficient and flexible enough to be able to utilize different analysis front-ends and to incorporate new perceptually-inspired grouping criteria in a straightforward manner.

A software implementation of the system described in this thesis was also made available as free and open source software. Together with the belief that this work showed the potential of spectral clustering methods for sound event segregation, it is expected that the software implementation may stimulate further research in this area as it can have significant impact in many MIR applications such as singer identification, music transcription and lyrics alignment.

6.2 Future Work

After a great deal of investment in the area of algorithm development, which has given rise to the framework proposed in Chapter 3 and to the results presented in Chapter 4, paving the way to several papers submitted for publication with success, there are nevertheless several lines of future work that are now possible to anticipate.

The development and evaluation of additional grouping cues are interesting possibilities for future work. The proposed framework is able to easily accommodate future similarity functions such as common amplitude and frequency modulation, frequency or time masking [Lagrange et al., 2006], spatial proximity, onsets, timbre models, or other grouping principles, allowing to model an increasing number of perceptual mechanisms

involved in human hearing.

Regarding spatial proximity cues, an interesting possibility is the addition of common panning cues for stereo signals as proposed in [Jourjine et al., 2000, Avendano, 2003]. The idea is that bins in the spectrum with similar azimuths are probably originating from a same source, becoming a good proximity cue for sound segregation.

The preliminary results presented in Section 3.5 have shown that the use of an onset detector to automatically segment the audio signal into texture windows may be a promising strategy. Indeed, further experimentation and evaluation as well as the implementation of more computationally efficient clustering methodologies should be explored in the future in an attempt to get the most out of this approach. Furthermore, onsets can also be used to estimate beat and tempo, and therefore allow the incorporation of top-down anticipations and expectations in the segregation process.

Although not necessary for the operation of the algorithm, prior knowledge such as sound source models or score representations could easily be incorporated into the similarity calculation. For example the likelihood that two peaks belong to the same sound source model [Vincent, 2006] could be used as an additional similarity cue.

Following the discussion in Section 3.7.2, different approaches should also be considered for the definition of the fundamental frequency estimation function in the HWPS calculation. Improved methods for estimating the "pitches" present in each frame would allow better candidates to the fundamental frequencies, potentially improving the current HWPS accuracy. Nevertheless, all such approaches will require careful experimental evaluations. Additionally, and given the segregation performance of the HWPS cue for harmonic events in complex music mixtures [Lagrange et al., 2008b], further work could be conducted towards the use of Harmonically Enhanced Spectral Representations for chroma-based music analysis [Gomez, 2006, Ellis, 2007].

The definition of the similarity functions, discussed in Section 3.7, could make good use of prior information to help setting up the neighborhood width for each similarity cue. The knowledge about the just noticeable differences (JND) in amplitude and frequency perception may be a good criteria for defining the similarity widths, where distance values below the corresponding JND would be considered fully similar, while all the other distances would gradually becomes more dissimilar.

The combination of the different grouping cues, as discussed in Section 3.7.3, should also receive additional attention. Although the current use of a logical "AND" combination of similarity cues may be appropriate for some cases, it may be too restrictive for

expressing more perceptually motivated combination schemes. More expressive combination operators, such as the logical "OR", the maximum and minimum of the different similarities, could result in better segregation results.

The current version of the framework presented in this work still requires the prior specification of the number of sound events or sources at each texture window. This is a key limitation of the current implementation, but the framework is flexible enough to include new approaches to an automatic estimation of the number of clusters to segregate at each time instant. Section 3.6.4 presented some pertinent options, but additional implementation and evaluation efforts must be conducted in order to validate their application to the task of sound segregation in music signals.

In addition, in the current implementation, clustering and cluster selection are performed independently for each "texture" window. In the future the goal is to explore cluster continuity constraints (for example neighbouring clusters in time corresponding to the same source should have similar overall characteristics) as well as more sophisticated methods of cluster selection. While the current framework implementation already includes some sequential grouping principles at the texture window level (achieved mainly by frequency and amplitude proximity cues, as described in Section 3.7.1), it does not yet fully incorporate sequential grouping among sound clusters detected in different texture windows. In fact, the same frequency and amplitude grouping cues, added to timbre and smoothness cues among others, could be used in the future to provide such a sequential integration following the principles proposed by Bregman [Bregman, 1990] and discussed in Section 2.3.1. Aspects such as temporal masking phenomena in perception or prior knowledge on timbre models could prove helpful to achieve sequential continuity of the segregated sound events (e.g. see [Lagrange et al., 2006, Burred and Sikora, 2007]).

Alternative analysis front-ends such as perceptually-informed filterbanks or sinusoids+transient representations could be a way to address some of the sinusoidal modeling limitations. Even though some grouping of components corresponding to transients and consonants can be achieved by amplitude and frequency similarity cues, the sinusoidal representation is not particularly suited for non-pitched sounds such as transient sounds. In what regards frequency estimation methods of sinusoidal components, alternative approaches could be considered, such as parabolic interpolation [Abe and Smith, 2004] or subspace methods [Badeau et al., 2006]. In particular, a technique based on the Odd-DFT filter bank that allows the accurate estimation of the frequency, phase and magnitude of stationary sinusoids proposed by [Ferreira, 2001] has been used with some promising results in a multi-pitch detection system presented

in [Martins, 2002, Martins and Ferreira, 2002].

The resynthesis also suffers from artifacts that result from the limitations of the sinusoidal representation. An interesting alternative would be to retain the sinusoidal modeling front-end for grouping but use the entire STFT spectrum for the resynthesis of the segregated sound events. As studied in [Lagrange et al., 2007b], such a resynthesis stage is more flexible and reduces artifacts.

Another option worth exploring is the use peak tracking methods (e.g. [McAulay and Quatieri, 1986]) as a pre- or post-processing stage. After computing the sinusoidal representation, and prior to clustering, peak tracking could be used to identify spectral tracks between spectral peaks, which could be subsequently used to derive grouping cues (e.g. peaks that belong to a same track probably result from a same sound event; peaks that belong to tracks whose onset or offset are close in time have a higher chance to belong to a same sound event). On the other hand, partial tracking could be applied to each segregated cluster of peaks, where peaks not belonging to a track would be discarded, reducing the existence of spurious spectral content at the resynthesis stage (though this would also have a negative impact on the resynthesis of the non-harmonic content of each segregated sound event).

The extraction of descriptors (e.g. pitch, timbre features, timing information) directly from the mid-level representation (which results from the analysis front-end) of each segregated cluster without the need of resynthesis is also an interesting line of future work. The Cluster Peak Ratio (CPR) feature proposed in Section 4.5, a feature extracted directly from the sinusoidal mid-level representation, already presented some promising results when trying to perform voicing detection in polyphonic music signals.

Appendix A

Graph Laplacians for Spectral Clustering

Spectral clustering is mainly based on the properties of *graph Laplacian* matrices (the naming comes from the fact that graph Laplacians formally look like a continuous Laplace operator – refer to [von Luxburg, 2007, pp.28] for more insight). There exists a whole area of study dedicated to those matrices, called *spectral graph theory* (e.g. see [Chung, 1997]). Basically, the spectral theory provides conditions under which an operator or matrix can be represented as a diagonal matrix in some basis.

Accordingly, this section will summarize the most important properties of graph Laplacians and try to point out their importance in the way spectral clustering works. The interested reader may find more detailed discussion about this topic in [von Luxburg, 2007].

In the following discussion it will be always assumed that G (i.e. the similarity graph) is an undirected, weighted graph with weight matrix W, where $w_{ij} = w_{ji} \geq 0$. It will not necessary be assumed that, when taking the eigenvectors of a matrix, they will be normalized. This means that, for instance, the constant one vector $\mathbb{I}$ and a multiple $a\mathbb{I}$ for some $a \neq 0$ will be considered as the same eigenvectors. Eigenvalues will always be ordered increasingly, respecting multiplicities. By "the first k eigenvectors" it is meant to refer to the eigenvectors corresponding to the k smallest eigenvalues.

Two main types of graph Laplacians have been traditionally proposed: the *unnormalized* and the *normalized* graph Laplacians (which correspond to *unnormalized* and *normalized* spectral clustering, respectively). The latter can yet be divided in two subtypes: *symmetric* or *random walk* (see [von Luxburg, 2007, pp.3]).

This raises the question of which of the three graph Laplacian matrices should be used for spectral clustering. In order to make a decision, it is usually advised to look at the degree distribution of the similarity graph (see Equation 10). If the graph is very regular and most vertices have approximately the same degree, then all the three types of

Laplacians will result very similar to each other, and will work equally well for clustering. However, if the degrees in the graph are very broadly distributed, then the Laplacians differ considerably.

The use of normalized rather than unnormalized spectral clustering is usually advocated, and in particular the recommendation goes to the *random walk* variant. This advice is justified by several arguments, as clearly discussed in [von Luxburg, 2007, pp.24] and summarized next.

In general, clustering has two different objectives: find a partition such that points in different clusters are dissimilar to each other (i.e. minimize the between-cluster similarity) and find a partition such that points in the same cluster are similar to each other (i.e.maximize the within-cluster similarities). It is shown that normalized spectral clustering implements both clustering objectives, while unnormalized spectral clustering only implements the first one.

Furthermore, and on a different argument based on the statistical analysis of both normalized and unnormalized versions of spectral clustering, it was demonstrated the superiority of the normalized spectral clustering in what regards the consistency of results [von Luxburg, 2007, pp.24]. More specifically, if taking a statistical setting where it is assumed that the data points $x_1, \ldots, x_T$ have been sampled i.i.d. according to some probability distribution P on some underlying data space $\mathcal{X}$, the question of consistency is whether the clustering results of spectral clustering converge to a useful partition of the underlying space $\mathcal{X}$ while drawing more and more data points from $\mathcal{X}$. Once again, it can be proved that, while the above question is true for both variants of normalized spectral clustering, the same can not be said when using unnormalized clustering. For the latter case, it was shown that convergence may completely fail, or may end up converging to trivial solutions, which construct clusters consisting of one single point of the data space [von Luxburg, 2007].

Finally, and after ruling out the unnormalized spectral clustering approach, the question about which one of the two variants of the normalized clustering remains. It was also shown that all the above arguments favour the *random walk* variant rather than the *symmetric* one [von Luxburg, 2007, pp.27]. The main reason results from the way both matrices are defined, which for the *symmetric* case ends up leading to some undesired artifacts. Given that there are no computational differences between the two variants, the final advise goes to the use of the *random walk normalized spectral clustering* approach, the one used in this thesis.

In the following the properties of the graph Laplacian will be discussed, starting with

the *unnormalized* graph Laplacian, and subsequently deriving the *normalized random walk* variant.

A.1 The Unnormalized Graph Laplacian

The *unnormalized graph Laplacian matrix* L is defined as

$$L = D - W \tag{41}$$

where D is the degree matrix (see Equation 11) and W is the affinity matrix corresponding to the similarity graph (see Equation 9). This matrix L has several interesting properties (see [Mohar, 1991]) and one of the most relevant ones for spectral clustering is that if taking any indicator vector $f \in \mathbb{R}^T$ (where T is the number of data points to cluster and that $f = (f_1, \ldots, f_T)'$ indicates the cluster "label" assigned to each data point – see Section 3.6) then the following applies (see [von Luxburg, 2007] for the proof):

$$f'Lf = \frac{1}{2} \sum_{i,j=1}^{T} w_{ij}(f_i - f_j)^2. \tag{42}$$

The challenge of spectral clustering is then to minimize $f'Lf$. Thus, what this expression basically states is that data points close in the similarity space (i.e. with high w_{ij} values) will be penalized if they are clustered into different clusters (i.e. $f_i \neq f_j$). In case they are not similar (i.e. w_{ij} is negligible and close to zero), it becomes irrelevant in what cluster they end up (this fact gives some justification on why the unnormalized spectral clustering fails on the second objective of clustering, as discussed before).

Additionally, L is symmetric (which directly follows from the symmetry of D and W) and positive semi-definite (from Equation 42 is trivial to see that $f'Lf \geq 0$), which by definition means that L will have T non-negative, real valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_T$. It is also easy to prove that the eigenvector corresponding to $\lambda_1 = 0$ will be a constant one vector $\mathbb{I}$.

Also interesting to note is that the unnormalized graph Laplacian does not depend on the diagonal elements of the affinity matrix W. Each affinity matrix which coincides with W on all off-diagonal positions leads to the same unnormalized graph Laplacian L. This means that the self-similarities in a graph do not change the corresponding graph Laplacian.

From these properties an important observation results: the multiplicity k of the eigenvalue 0 of L equals the number of connected components $A_1, \ldots, A_k$ in the graph.

The eigenspace of eigenvalue 0 is spanned by the indicator vectors $\mathbb{I}_{A_1}, \ldots, \mathbb{I}_{A_k}$ of those components. This can be easily proven as follows. Starting with the case $k = 1$ (i.e. the graph is connected), assume that f is an eigenvector with eigenvalue 0. This implies that

$$0 = f'Lf = \sum_{i,j=1}^{T} w_{ij}(f_i - f_j)^2. \tag{43}$$

In other words, given that the weights w_{ij} are non-negative, this sum can only vanish if all terms $w_{ij}(f_i - f_j)^2$ vanish. Thus, if two vertices v_i and v_j are connected (i.e. $w_{ij} > 0$), then f_i needs to equal f_j. This implies that f needs to be constant for all vertices which can be connected by a path in the graph. Since all vertices of a connected component in an undirected graph can be connected by a path, f needs to be constant on the whole connected component. In a graph consisting of only one connected component the eigenvector corresponding to the eigenvalue 0 will have to be the constant one vector $\mathbb{I}$, which obviously is the indicator vector of the connected component.

If now considering the case of k *connected components*, assume that, without loss of generality and for the sake of visualization, the vertices are ordered according to the connected components they belong to. In this case, the affinity matrix W has a block diagonal form, and the same is true for the matrix L:

$$L = \begin{bmatrix} L_1 & & & \\ & L_2 & & \\ & & \ddots & \\ & & & L_k \end{bmatrix}. \tag{44}$$

It is interesting to note that each one of the blocks L_i in L is a proper graph Laplacian on its own. They correspond to the subgraph of the i-th connected component. As it is the case for all block diagonal matrices, it is known that the spectrum of L is given by the union of the spectra of L_i, and therefore the corresponding eigenvectors of L are nothing more than the eigenvectors of L_i, filled with 0 at the positions of the other blocks. Because each L_i is a graph Laplacian of a connected graph, then every L_i will have its eigenvalue 0 with multiplicity 1, and the corresponding eigenvector will obviously be the constant one vector on the i-th connected component. Thus, the matrix L will have as many eigenvalues 0 as there are connected components, and the corresponding eigenvectors will be nothing more than the indicator vectors of the connected components.

A.2 The Normalized Graph Laplacian

Taking the definition of the unnormalized graph Laplacian presented above, it is now easy to define the *normalized random walk graph Laplacian* L_{rw} as

$$L_{rw} := D^{-1}L = I - D^{-1}W. \tag{45}$$

Note that $D^{-1}W$ is a *random walk* transition matrix (hence the name of this graph Laplacian). A random walk on a graph is a stochastic process which randomly jumps from vertex to vertex, where the transition probability of jumping in one step from vertex v_i to vertex v_j is proportional to the edge weight w_{ij} and is given by $p_{ij} := w_{ij}/d_i$. Under this formulation, spectral clustering can also be interpreted as the attempt to find a partition of the graph such that the random walk stays long within the same cluster and seldom jumps between clusters (see [von Luxburg, 2007, pp.14]).

Similarly to the unnormalized graph Laplacian, L_{rw} is also positive semi-definite and has T non-negative real-valued eigenvalues $0 = \lambda_1 \leq \cdots \leq \lambda_T$. Its eigenvalue 0 will also have the constant one vector $\mathbb{I}$ as eigenvector. In general, λ will be an eigenvalue of L_{rw} with eigenvector u if and only if λ and u solve the generalized eigenproblem $Lu = \lambda Du$. As is the case for the unnormalized graph Laplacian, the multiplicity of the eigenvalue 0 of L_{rw} is related to the number of connected components: its eigenspace is spanned by the indicator vectors $\mathbb{I}_{A_i}$ of those components.

Furthermore, by the standard Rayleigh-Ritz theorem [Golub and Loan, 1989], the second smallest eigenvector of this generalized eigensystem is the real valued solution to the bipartite Normalized Cut minimization [Shi and Malik, 2000, von Luxburg, 2007]. Similarly, it is possible to show that the eigenvector with the third smallest eigenvalue is the real valued solution that optimally subpartitions the first two parts. In fact, this line of argument can be extended to show that one can subdivide the existing graphs, each time using the eigenvector with the next smallest eigenvalue.

Thus, once the eigenvectors are computed, it becomes possible to partition the graph into k pieces using the k smallest eigenvectors. In the ideal case, the eigenvectors should only take on two discrete values and their values would indicate exactly how to partition the graph. However, and because of the applied relaxation (see Section 3.6.3), the computed eigenvectors can now take on continuous values and it is therefore necessary to transform them into a discrete form.

However, in practice, because the approximation error from the real valued solution

to the discrete valued solution accumulates with every eigenvector taken and all eigenvectors have to satisfy a global mutual orthogonality constraint, solutions based on higher eigenvectors become unreliable.

Several approaches to achieve this discretization have been proposed and the interested reader can find detailed information about them in [Shi and Malik, 2000, von Luxburg, 2007].

Appendix B

Evaluation Metrics

B.1 Signal-to-Distortion Ratio (SDR)

One of the most common measures used to evaluate CASA systems is the signal-to-distortion ratio (SDR) [Klapuri and Davy, 2006], which summarizes the separation quality as the ratio of the energies of the reference signal and the error between the separated and reference signal. Although the SDR is only an approximate measure of the perceptual quality of the separated signals, its simplicity of calculation will justify its use in some of the experiments presented in this chapter. It is defined in decibels as:

$$\mathrm{SDR[dB]} = 10 \log_{10} \frac{\sum_t s(t)^2}{\sum_t [\hat{s}(t) - s(t)]^2} \tag{46}$$

where $s(t)$ is the reference signal with the original separated source and $\hat{s}(t)$ is the extracted source (both represented as row vectors).

B.2 Signal-to-Distortion Ratio with Optimized Gain Factor (SDR_{ogf})

When computing the SDR between the original and the separated signals, it is important to assure that both signals are normalized to a similar gain. Otherwise the SDR will get biased by the gain differences between the two signals and loose its ability to measure accurately the distortion level between the two signals. Because it is not always easy (or even possible) to control the gain of the resynthesis process used to obtain the separated signals, a more convenient solution is to find a way to compensate the gain diferences in the SDR calculation itself. This is the objective of the SDR with optimized gain factor (SDR_{ogf}), which can be calculated as follows (in dB):

$$\mathrm{SDR}_{ogf}[\mathrm{dB}] = 10\log_{10}\frac{1}{1-r^2}, \tag{47}$$

where r is the normalized correlation coefficient:

$$r = \frac{\sum_t [s(t)\hat{s}(t)']}{\sqrt{\sum_t s(t)^2 \sum_t \hat{s}(t)^2}}, \tag{48}$$

where $s(t)$ is the reference signal with the original separated source and $\hat{s}(t)$ is the extracted source (both represented as row vectors). In other words, the SDR_{ogf} calculation corresponds to using an optimized gain factor Sf for the separated signal $\hat{s}(t)$, as follows:

$$Sf = \frac{\sum_t [s(t)\hat{s}(t)']}{\sum_t \hat{s}(t)^2}. \tag{49}$$

B.3 Segmental Signal-to-Distortion Ratio (SSDR)

A simple objective measure which is perceptually more plausible than the SDR is the segmental signal-to-distortion ratio (SSDR), which is calculated as the average of frame-wise SDR's [Mermelstein, 1979]. Unlike the normal SDR, this measure takes into account the fact that errors in low-intensity segments are usually more easily perceived. The segmental SDR has often been used to measure the subjective quality of speech, and it can be defined as follows. For each segment k, a signal-to-distortion ratio $\mathrm{SS}(k)$ is calculated as:

$$\mathrm{SS}(k) = 1 + \frac{\sum_t s(t)^2}{\delta + \sum_t [\hat{s}(t) - s(t)]^2}. \tag{50}$$

The term δ in the denominator prevents divides by zero. The additive unity term discounts segments with SNR's less than unity. The final average segmental SDR in dB is calculated as:

$$\mathrm{SSDR}[\mathrm{dB}] = 10\log_{10}\left(10^{\sum_k \frac{\log_{10}\mathrm{SS}(k)}{N}} - 1\right) \tag{51}$$

where N is the total number of segments considered. The exponent term is the geometric mean of $\mathrm{SS}(k)$. The subtraction of the unity term tends to compensate for the unity term in $\mathrm{SS}(k)$.

Appendix C

MARSYAS: An Open Source Audio Processing Framework

MARSYAS (**M**usic **A**nalysis **R**etrieval and **SY**nthesis for **A**udio **S**ignals) is a free and open source software (FOSS) framework with specific emphasis on building music information retrieval (MIR) and sound analysis and processing systems. Originally created in 1998 by George Tzanetakis [Tzanetakis and Cook, 2000, Tzanetakis, 2002], it has been under steady development and is used for a variety of projects, both in academia and industry. The guiding principle behind the design of MARSYAS has always been to provide a flexible, efficient and extensible framework without sacrificing computational efficiency.

MARSYAS provides a general, extensible and flexible framework that enables experimentation with algorithms and provides the fast performance necessary for developing efficient and real-time and audio analysis and synthesis tools. A variety of existing building blocks that form the basis of many published algorithms are provided as dataflow components that can be combined to form more complicated algorithms (black-box functionality). In addition, it is straightforward to extend the framework with new building blocks (white-box functionality). These blocks can be combined into data flow networks that can be modified and controlled dynamically while they process data in soft real-time[1].

[1] A system is said to be real-time if the total correctness of an operation depends not only upon its logical correctness, but also upon the time in which it is performed. The classical conception is that in a hard or immediate real-time system, the completion of an operation after its deadline is considered useless, ultimately leading to a critical failure of the complete system. A soft real-time system on the other hand will tolerate such lateness, and may respond with decreased service quality (e.g. dropping frames while playing an audio stream).

C.1 Framework Architecture

MARSYAS-0.2 [Tzanetakis, 2008, Tzanetakis et al., 2008] evolved from MARSYAS-0.1 which focused mostly on audio analysis [Tzanetakis and Cook, 2000]. One of the main additions to the new version was support for audio synthesis, which got inspiration from the design of the Synthesis ToolKit (STK) [Cook and Scavone, 1999]. Supporting audio synthesis is an welcome new feature, since the ability to hear the results of audio analysis algorithms can be very useful for validating and understanding algorithms.

Other influences in MARSYAS[2] include the powerful but more complex architecture of CLAM [Amatriain, 2007], the patching model and strong timing of ChucK[3] [Wang and Cook, 2004], and ideas from the Aura project [Dannenberg and Brandt, 1996].

C.1.1 Dataflow Programming

Dataflow programming is based on the idea of expressing computation as a network of processing nodes/components connected by a number arcs/communication channels. Dataflow programming has a long history. The original (and still valid) motivation for research into dataflow was to take advantage of parallelism. Motivated by criticisms of the classical von Neumann hardware architecture such as [Ackerman, 1982], dataflow architectures for hardware were proposed as an alternative in the 1970s and 1980s. During the same period a number of textual dataflow languages such as Lucid [Ashcroft and Wadge, 1977] were proposed. Despite expectations that dataflow architectures and languages would take over from von Neumann concepts this didn't happen. However during the 1990s there was a new direction of growth in the field of dataflow visual programming languages that were domain specific. In such visual languages programming is done by connecting processing objects with "wire" to create "patches". Successful examples include LabView, Simulink and in the field of Computer Music MAX/MSP [Zicarelli, 2002] and Pure Data[4] [Puckette, 1997].

In fact, expressing audio processing systems as dataflow networks has several advantages [Tzanetakis, 2008]. The programmer can provide a declarative specification of what needs to be computed without having to worry about the low level implementation details of how it is computed. The resulting code can be very efficient and have a small memory

[2]For simplicity sake, from now on the more compact MARSYAS designation will be used when referring to the specific MARSYAS-0.2 version of the framework.

[3]`http://chuck.cs.princeton.edu/`

[4]`http://crca.ucsd.edu/~msp/software.html`

footprint as data just "flows" through the network without having complicated dependencies. In addition, dataflow approaches are particularly suited for visual programming. One of the initial motivations for dataflow ideas was the exploitation of parallel hardware and therefore dataflow systems are particularly good for parallel and distributed computation. A comprehensive overview of audio processing frameworks from a Software Engineering perspective can be found in [Amatriain, 2007].

Another recent trend has been to view dataflow computation as a software engineering methodology for building systems using existing programming languages [Manulescu, 1997].

As a result, the idea of dataflow programming has been determinant in the design of MARSYAS-0.2, where complex network of processing objects can be assembled to form systems that can handle audio and data flows with expressiveness and efficiency.

C.1.2 Implicit Patching and Composition

The basic idea behind *Implicit Patching* [Bray and Tzanetakis, 2005] is to use object composition [Gamma et al., 1995] rather than explicitly specifying connections between input and output ports in order to construct the dataflow network. It evolved from the integration of different ideas that were developed independently in previous versions of MARSYAS [Tzanetakis, 2008].

Combined with Implicit Patching, the expressive power of composition is increased and a large variety of complex dataflow networks can be expressed only using object composition, and therefore no Explicit Patching.

Another side benefit of Implicit Patching is that it enforces the creation of trees and therefore avoids problems with cycles in the dataflow graph.

C.1.3 Basic Processing Units and Networks

As previously described, systems in Marsyas are expressed as interconnected dataflow networks of processing modules. Each processing module performs a specific task that always consists of a matrix transformation.

Audio and other types of data are represented by matrices with some semantics associated with them: rows represent *observations* (over time) and columns represent *samples* in time. For instance, a stereo audio signal might be processed in chunks of 2 observations (i.e. two channels) and 512 samples in time. This clean data structure, although quite specific, suits well audio processing applications. All modules process one input matrix (known as a *slice*) and store the result on another matrix so it can be shared with the next

processing module. Hence, each module accepts only one input and produces one output. Processing is performed on defined chunks of data and is executed whenever the `tick()` function of the module is called.

All processing blocks in MARSYAS are called *MarSystems* and provide the basic components out of which more complicated networks can be constructed. Essentially any audio processing algorithm can be expressed as a large *composite MarSystem* (discussed below) which is assembled by appropriately connected basic *MarSystems*.

Some representative examples of *MarSystems* include sound file reading and writing (e.g. wav, au, mp3 and ogg audio file formats), real-time audio input and output (i.e. from and to a soundcard), signal processing algorithms (e.g. filters, STFT, DWT), feature extraction (e.g. MFCC, centroid, rolloff, flux) and machine learning modules (e.g. KNN, GMM, SVM, PCA, SVM).

To assemble multimedia processing systems, modules are *implicitly connected* using hierarchical *composition*. Special *"Composite"* modules such as *Series*, *Fanout*, *Parallel* are used for this purpose. The basic idea behind *Implicit Patching* [Bray and Tzanetakis, 2005] is to use object composition rather than explicitly specifying connections between input and output ports in order to construct the dataflow network. For instance, modules added to a *Series* composite will be connected in series, following the order they were added – the first module's output is shared with the second module's input and so on. Moreover, the `tick()` method is called sequentially following the same order. Figure 39 shows an example of how composite and non-composite modules can be used. This paradigm differs from typical processing tools based on explicit patching such as CLAM [Amatriain, 2007], MAX/MSP [Zicarelli, 2002] or PD [Puckette, 1997].

As another example, consider a small network consisting of a *Series* module which contains three other modules: a source, a processing module and a sink. When `tick()` is called in the *Series*, implicitly the `tick()` method is called in all three modules as shown in Figure 40.

Dataflow in MARSYAS is synchronous which means that at every *"tick"* a specific slice of data is propagated across the entire dataflow network. This eliminates the need for queues between processing nodes and enables the use of shared buffers which improves performance. This is similar to the way UNIX pipes are implemented but with audio specific semantics.

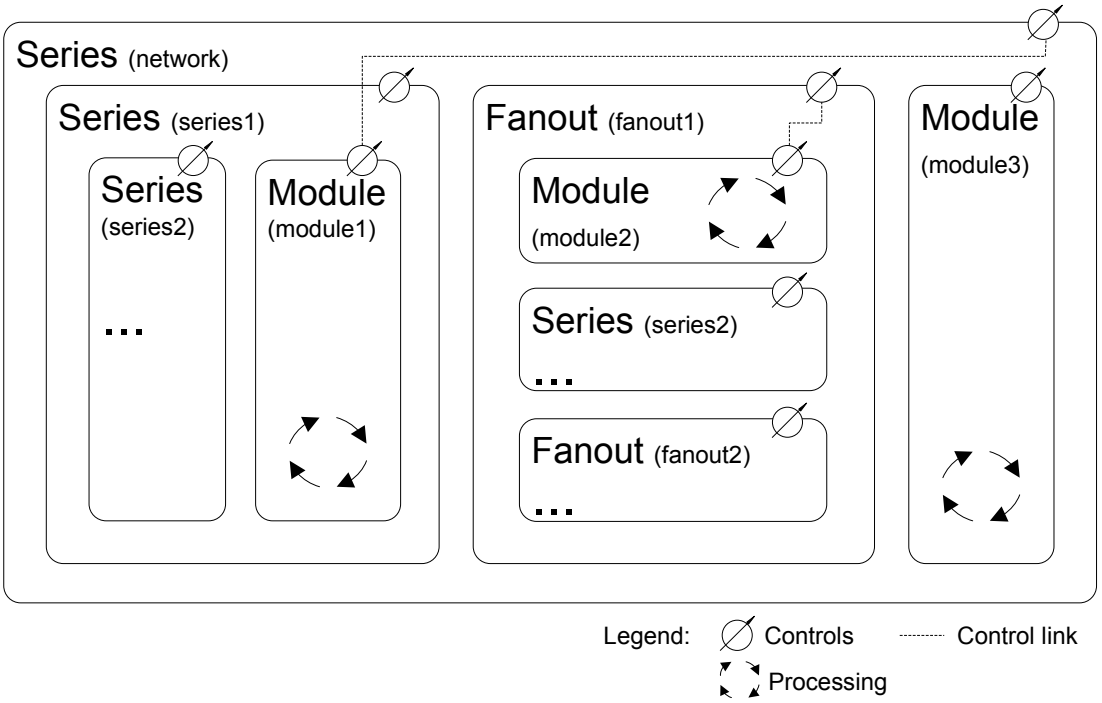


Figure 39: *Building blocks in MARSYAS 0.2.*

C.1.4 Dynamic Access to Modules and Controls

In MARSYAS, each module in a processing network can be accessed by querying the system with a path-like string. In addition to being able to process data, *MarSystems* need additional information that can change their functionality while they are running (i.e. processing data). For example a sound file reader *MarSystem* needs the name of the sound file to be opened, while a volume block (known as *Gain MarSystem* in MARSYAS) can be adjusted while data is flowing through in real-time. This is achieved by a separate message passing mechanism, where there is a clear distinction between dataflow, which is synchronous, and control flow, which is asynchronous.

Taking the example shown in Figure 39, if we wanted to reach the processing module named `module1`, the query path would be `/Series/network/Series/series1/Module/module1`. The first "/" indicates the outermost module and the rest of the path is always composed by the concatenation of `Type/Name` strings. This naming scheme was inspired from the way messages are exchanged in Open Sound Control (OSC)[5] [Wright et al., 2003]. It is possible to have access to some of the internal parameters of the modules using *controls*. Each module

[5] `http://opensoundcontrol.org/`

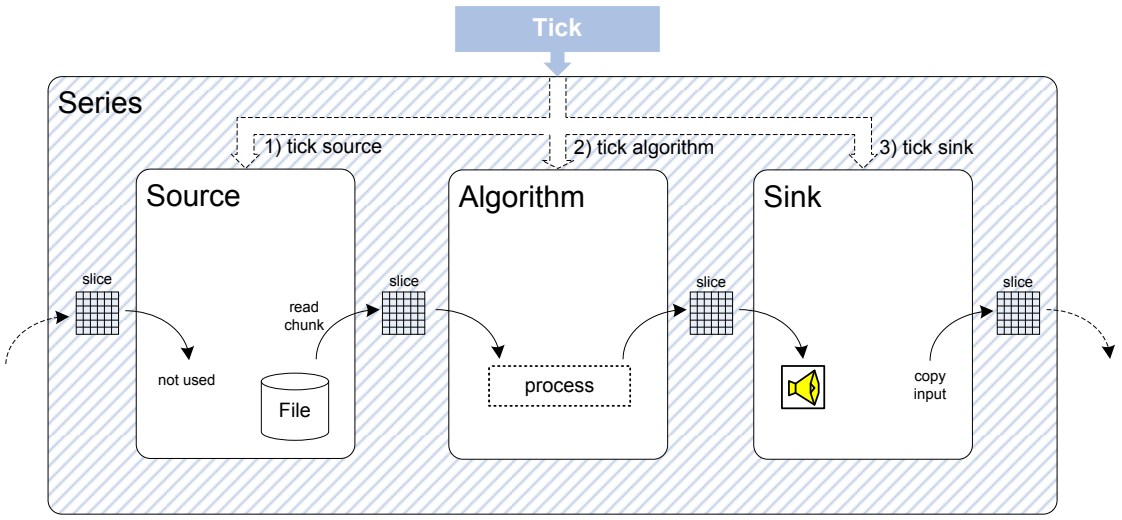


Figure 40: *Processing slices of data in MARSYAS 0.2.*

exports a list of controls which may be of different types (e.g. integers, floats, strings, vectors, or arbitrary user-defined types). They can be accessed for reading or writing by specifying the path to their parent module plus the `Type/Name` corresponding to the control.

One of the most useful characteristics of MarSystems is that they can be instantiated at run-time. Because they are hierarchically composable that means that any complicated audio computation expressed as a dataflow network can be instantiated at run-time. For example multiple instances of any complicated network can be created as easily as the basic primitive MarSystems. This is accomplished by using a combination of the Prototype and Composite design patterns [Gamma et al., 1995].

C.1.5 Dynamic Linking of Controls

Controls can be linked as shown in Figure 39, so that changes to the value of one control are automatically propagated to all the others. There are plenty of interesting uses for this feature: parameter values that must be passed to more than one module in a system; feedback loops where results from modules ahead in the processing network are sent back to the first modules in the chain; shortcuts for other links, etc. Links can be defined (both at compile-time and at run-time) for controls with the same value type, either belonging to a same module, or to any other module in the network.

Links can also be used to create *proxy controls* – in order to create a shortcut to a control from a module deep inside other composite modules, it is possible to link it to a

proxy control in the outmost module, created on demand for this task. This way multiple and easy to understand views for the control of the same algorithm can be created.

C.2 Interoperability

MARSYAS has provided almost since its creation some interoperability layers with other software packages and projects (e.g. WEKA). More recently, several additional interoperability interfaces have been implemented and included in the framework [Tzanetakis et al., 2008]. The following section will describe the implementation and use of the interfaces to software platforms such as MATLAB, Qt4, MAX/MSP, OSC, Python, Ruby and Java.

C.2.1 MATLAB

MATLAB is a powerful and widely used tool in several areas of research and development, with a large community of users and available routines for math and multimedia analysis and processing algorithms. Additionally, MATLAB provides easy to use and advanced plotting facilities, a major asset for researchers developing algorithms for audio, image and video processing. Until recently, developers always had to make a hard choice regarding their development language: either opt for the flexibility and ease of use of MATLAB or decide in favor of efficiency and performance as provided by an OOP language like C++. In its latest versions, MATLAB includes the ability to exchange data in run-time with applications developed in Fortran, C or C++, through and API named MATLAB Engine[6]. Marsyas implements a singleton wrapper class for the MATLAB Engine API, enabling Marsyas developers to easily and conveniently send and receive data (i.e. integers, doubles, vectors and matrices) to/from MATLAB in run-time. It is also possible to execute commands in MATLAB from calls in the C++ code as if they have been called in the MATLAB command line. This enables the execution of MATLAB scripts and the access to all MATLAB functions and toolboxes from within Marsyas C++ code. The MATLAB wrapper class in Marsyas provides three basic methods:

- `MATLAB_PUT(Marsyas_var, MATLAB_var_name);`
- `MATLAB_GET(Marsyas_var, MATLAB_var_name);`
- `MATLAB_EVAL(MATLAB_cmd);`

[6] `http://www.mathworks.com`

By means of function overloading, these three methods allow exchanging different types of variables from Marsyas/C++. They can be called from anywhere in the Marsyas C++ code without any need of changes in the Marsyas interfaces, making it simple to send data structures to MATLAB for convenient inspection and analysis, calculations and plotting, and then get them back in Marsyas for additional processing. These features are only available when MATLAB is installed in the system and Marsyas is built with MATLAB Engine support. Any MATLAB Engine calls in Marsyas code are automatically ignored otherwise. A code snippet in C++ is presented next, illustrating how a vector of real values can be processed and exchanged with MATLAB, using the Marsyas MATLAB Engine wrapper class:

```
// create a std::vector of real numbers
std::vector<double> vector_real(4);
vector_real[0] = 1.123456789;
vector_real[1] = 2.123456789;
vector_real[2] = 3.123456789;
vector_real[3] = 4.123456789;

// send a std::vector<double> to MATLAB
MATLAB_PUT(vector_real, "vector_real");

// do some dummy math in MATLAB
MATLAB_EVAL("mu = mean(vector_real);");
MATLAB_EVAL("sigma = std(vector_real);");
MATLAB_EVAL("vector_real = vector_real/max(vector_real);");

// get values from MATLAB
double m, s;
MATLAB_GET(m, "mu");
MATLAB_GET(s, "sigma");
MATLAB_GET(vector_real, "vector_real");
```

C.2.2 Trolltech Qt4

Trolltech's Qt4 is a comprehensive development toolkit that includes features, capabilities and tools that enable the development of cross-platform C++ applications. Such features include multi-platform APIs and classes for the development of Graphic User Interfaces (GUIs), signalling, and multi-threaded execution. In its 4th version, Qt is available as a dual-license software toolkit for all the supported platforms (i.e. Linux, MacOSX and Windows). For open source applications such as Marsyas one of the licences is open-source GPL.

Marsyas, although not bound specifically to Qt, uses this toolkit as its preferred solution for the development of GUIs. Its use is however totally optional, allowing the

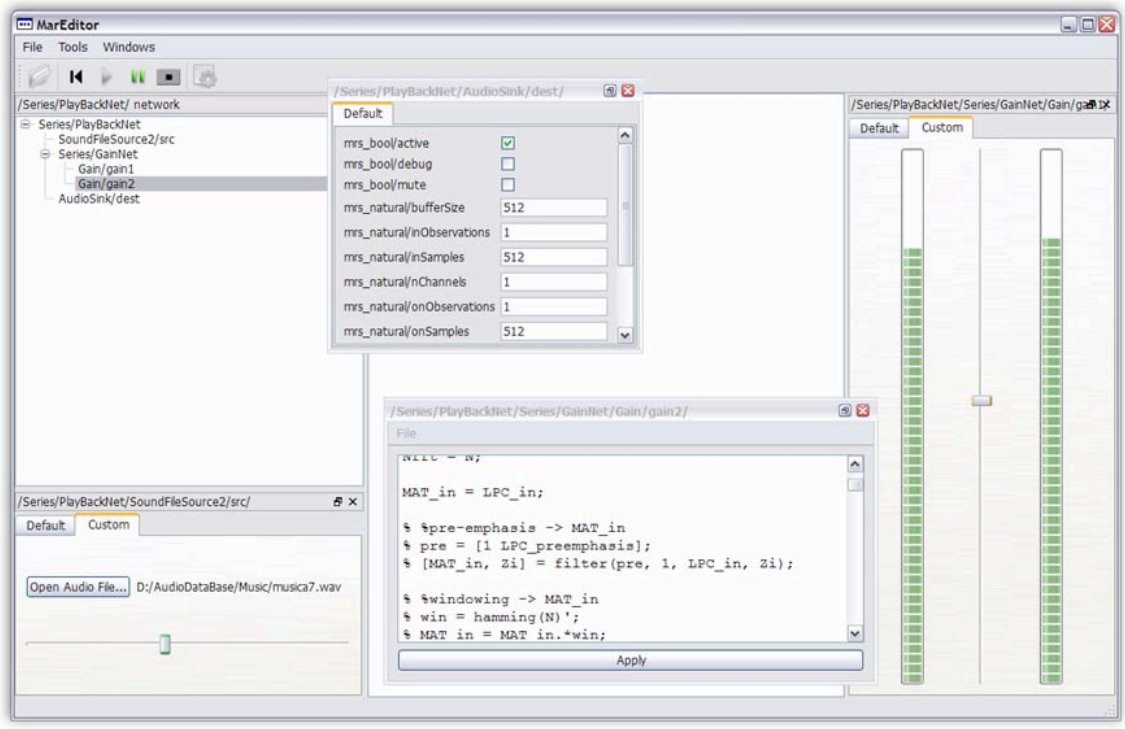


Figure 41: *Common and specialized GUIs for Marsyas modules.*

developer to choose any other library, or even including no GUI support at all (missing in this case all of the GUI extra features already implemented in some Marsyas classes and applications). There are currently two approaches available for GUI development using Qt4 in Marsyas.

The first way is using a *delegation design pattern*, where the core C++ Marsyas classes in charge of the actual processing are wrapped by an entity that takes care of all the message passing between the GUIs and the processing network. Additionally, using Qt's multithread features, this wrapper makes sure that GUIs and the processing code are executed in independent threads. This allows the implementation of responsive GUIs and the best use of the last generation multi-core processors. This approach is most suited for the development of customized GUI front-ends for Marsyas based applications. It allows interacting with the processing network (by means of reads/writes to its module's controls) in an intuitive and real-time manner.

The second way is to conditionally make all Marsyas modules inherit from Qt's base class *QObject*. This automatically embeds Qt's most advanced features (such as *signals* and *slots*) into most Marsyas core classes. This avoids the use of middle-layers for message exchange between GUIs and the processing modules. Additionally to improving efficiency,

this approach facilitates the implementation of more advanced functionalities for GUI and multi-threaded processing. As a drawback, this implies a tighter compile-time bind between Marsyas and Qt, which can make the desired independence between the two frameworks more difficult to assure and maintain. Given the additional features inherited from Qt, this approach allows the common implementation of GUIs for all Marsyas modules, such as widgets for viewing/modifying the list of controls from any Marsyas module, or even the creation of specialized GUIs for data plotting or parameter modification (see Figure 41.

C.2.3 Open Sound Control

Open Sound Control (OSC) [Wright et al., 2003] is a protocol for communication among computers, sound synthesizers, and other multimedia devices that is optimized for modern networking technology. There many implementations of OSC and most computer music environments (such as Max/MSP, PD, Chuck, CSound) have the ability to send and receive open sound control messages.

The control mechanism in Marsyas was inspired from OSC so the mapping of controls to OSC messages is very straightforward. The path notation is used to specify the full name of the control and the value of the message is directly mapped to the value of the control.

The mapping of OSC messages to Marsyas controls is part of the Qt4/Marsyas integration code. *OscMapper* is the interface between OSC, Marsyas and Qt4. It acts as both an OSC server and client and allows particular OSC hosts and clients to be associated with particular MarSystems. The communication is abstracted as signals and slots following the way Qt4 structures communication between interface components. The user interface programmer only needs to specify the information about where the OSC messages will be coming from and all the rest is taken care directly by the mapping layer.

For example, this way it is straightforward to use PureData to send OSC messages to modify the parameters of a phasevocoder running in Marsyas. The data flowing through a Marsyas network is also accessible through controls so audio information can also be exchanged.

C.2.4 Marsyas runtime as a Max/MSP external

Max/MSP[7] allows the creation of so called "external" processing units which can be written in C/C++ following a specific API. These externals can then be used as building blocks in the visual programmming environment.

Efforts have been put into the implementation of a general external that can be used to load any audio processing system expressed in Marsyas. The main challenge was to completely decouple the audio buffer rate of Max/MSP from the audio buffer size used by the Marsyas runtime. This is achieved through a dynamic rate adjusting sound source and sound sink for the input and output to the Marsyas part of the patch.

For example if the audio buffer size of the Max/MSP patch is 64 samples and Marsyas requires buffers of 256 samples then four buffers of 64 samples will be accumulated before sent to Marsyas for processing. Similarly at the Marsyas output the 256 samples will be broken into 64 sample buffers to be sent back to Max/MSP. Arbitrary sizes are supported and there is no requirement that one buffer should be smaller than the other. This is achieved by using circular buffers with dynamically adjustable length. Controls can be read and written through control outlests and inlets of the external.

C.2.5 SWIG Bindings

There are only a few commands that need to be supported in order to interface the Marsyas runtime. They basically consist of commands for assembling the dataflow network through hierarchical composition (create, addMarSystem) and commands for linking, updating and setting controls (linkctrl, updctrl, setctrl). SWIG[8] is a software development tool that connects programs written in C and C++ with a variety of high-level programming languages. It includes support for both scripting and non-scripting languages and can be used to create high-level interpreted programming environments. We have used it to provide Marsyas bindings for several programming languages (currently Lua, Ruby, Python, Java).

The following piece of code shows a simple soundfile player written in Ruby using the bindings:

```
msm = Marsyas::MarSystemManager.new
file = msm.create "SoundFileSource","file"
sink = msm.create "AudioSink","sink"
net = msm.create "Series","net"
```

[7] http://www.cycling74.com
[8] http://www.swig.org

```
net.addMarSystem file
net.addMarSystem sink

fn = net.getctrl "SoundFileSource/file/mrs_string/filename"
ne = net.getctrl "SoundFileSource/file/mrs_bool/notEmpty"
filename.setctrl_string "test.wav"

while ne.to_bool
      net.tick
end
```

A more complex examples shows how feature extraction can be written in Python:

```
import marsyas_python
# Create top-level patch
mng = marsyas_python.MarSystemManager()
fnet = mng.create("Series", "featureNetwork");

# functional short cuts to speed up typing
create = mng.create
add = fnet.addMarSystem
link = fnet.linkControl
upd = fnet.updControl
get  = fnet.getControl
msr = marsyas.MarControlPtr.from_string

# Add the MarSystems
add(create("SoundFileSource", "src"))
add(create("TimbreFeatures", "featExtractor"))
add(create("TextureStats", "tStats"))
add(create("Annotator", "annotator"))
add(create("WekaSink", "wsink"))

# link the controls to coordinate things
link("mrs_string/filename",
     "SoundFileSource/src/mrs_string/filename")
link("mrs_bool/notEmpty",
     "SoundFileSource/src/mrs_bool/notEmpty")
link("WekaSink/wsink/mrs_string/currentlyPlaying",
     "SoundFileSource/src/mrs_string/currentlyPlaying")
link("Annotator/annotator/mrs_natural/label",
     "SoundFileSource/src/mrs_natural/currentLabel")
link("SoundFileSource/src/mrs_natural/nLabels",
     "WekaSink/wsink/mrs_natural/nLabels")

upd("mrs_string/filename", mstr("bextract_single.mf"))
upd("WekaSink/wsink/mrs_string/labelNames",
    get("SoundFileSource/src/mrs_string/labelNames"));

while (get("mrs_bool/notEmpty")):
```

```
    fnet.tick()
```

As an example of using the bindings with a compiled language, it is shown next how the same network as the Ruby example can be created and used to play sound in Java. This approach utilizes the JNI (Java Native Interface) and therefore would not be portable across different operating systems. However it allows Java programs to utilize the Marsyas functionality which has much higher run-time performance than a full Java port would have.

```
import edu.uvic.marsyas.*;

class Test {
    static {
        System.loadLibrary("marsyas");
   }
 public static void main (String [] args){
     MarSystemManager msm = new MarSystemManager();

     MarSystem file = msm.create("SoundFileSource","file");
     MarSystem gain = msm.create("Gain", "gain");
     MarSystem sink = msm.create("AudioSink","sink";)
     MarSystem net = msm.create("Series","net");
     net.addMarSystem(file);
     net.addMarSystem(gain);
     net.addMarSystem(sink);

     MarControlPtr filename =
      net.getControl("SoundFileSource/file/mrs_string/filename");
     MarControlPtr notempty =
      net.getControl("SoundFileSource/file/mrs_bool/notEmpty");

     filename.setValue_string("test.wav");
     while (notempty.to_bool()) net.tick();
}
```

C.3 MARSYAS Source Code

MARSYAS source code is available at Sourceforge[9], an internet hosting service for open source projects. It is written in portable C++ (as much as possible) and it compiles under Linux, MacOSX and Microsoft Windows. Subversion (SVN) is used for version control and the latest unstable source code can be obtained from the webpage.

[9] http://marsyas.sourceforge.net

Appendix D

Using HWPS for Efficient Dominant Harmonic Source Segregation

The leading voice or instrument is an important feature of musical pieces and can often be considered as the dominant harmonic source. This appendix describes a new scheme for the purpose of efficient dominant harmonic source separation.

In comparison to the evaluation conducted for the predominant melodic source separation approach presented in Section 4.3, a simpler but more computationally efficient scheme for the dominant harmonic source separation will be described and evaluated in this section . Although both evaluations focus a similar problem, the approach presented in this section is far from being a complete and generic framework for sound segregation, where much more demanding requirements exist. Nevertheless, its efficiency makes it well suited for being integrated in feature extraction algorithms with real-time constraints or as a good choice for a MIR pre-processing stage for very large datasets, allowing to focus on the dominant harmonic content in the audio signals.

The proposed technique is based around the HWPS cue (described in Section 3.7.2) and the resulting representation is termed *Harmonically Enhanced Spectral Representation* (HESR). Consequently, this evaluation also serves the purpose of demonstrating the ability of the HWPS cue to identify the harmonic content in complex audio mixtures and future work could potentially show the suitability of this approach as good pre-processing stage for chroma-based analysis of music signals (see for e.g. [Gomez, 2006, Ellis, 2007, Ellis et al., 2008, Serrà and Gómez, 2007, Serrà and Gómez, 2008]).

Given so, this new separation scheme will be compared to the generic CASA framework described in Chapter 3 (termed in the following sections as the Ncut approach). Therefore, its evaluation results presented previously in this chapter will be considered as the reference

for the experiments described in the following sections. The HESR approach and the results described in this section have also been presented in [Lagrange et al., 2008b].

D.1 Proposed Algorithm

The novel HWPS cue proposed in Section 3.7.2 estimates the degree of harmonic relationship between two frequency components and more precisely the likelihood that those two components belong to a dominant harmonic source. Focusing on the task of determining the dominant harmonic source in an audio signal using the HWPS cue only, it becomes possible to substantially reduce the the computational effort by considering the following algorithm: for each audio frame the precise frequencies and magnitudes of spectral peaks are estimated (see Section 3.4). An harmonicity factor, assigned to each selected peak p_l, can then be computed as follows:

$$h_l = \sum_{i \neq l} a_i W_h(p_l, p_i) \tag{52}$$

where $W_h(p_l, p_i)$ is the HWPS between the peak p_l under analysis and all the remaining peaks p_i in the same frame , being a_i their corresponding amplitudes. This factor indicates the likelihood that the considered peak belongs to a dominant harmonic source, and the peaks with the highest factor value can be amplified or selected for creating a harmonically enhanced spectral representation (HESR).

In fact, there are several alternatives to compute the HESR in practice. The one chosen for this specific task and used in the evaluations presented in the following section selects, for each frame, the 10 peaks with the highest harmonicity factor h_l among the 20 peaks output by the sinusoidal analysis front-end used to represent the input audio signals (see Section 3.4). Only the selected peaks are used in the resynthesis, which becomes equivalent to applying a harmonicity-based mask to the input spectral peaks. Other approaches could have been used such as scaling the amplitudes of the peaks depending on their h_l factor (equivalent to a " soft masking" procedure). In the current implementation, no smoothing or inter-frame information is used as an attempt to improve the harmonic selection over time frames. This allows to, for now, focus the evaluation on the raw ability of the HWPS cue to identify the harmonic content in complex music signals.

The next section will evaluate the performance of the HESR against the Ncut approach using the different tasks considered in the previous sections, namely singing voice separation, dominant pitch estimation, and voice detection.

	Ncut	HESR
Separation Performance (SSDR in dB)	4.25	1.07
Pitch Estimation Accuracy (Praat) (%)	46	64
Voice Detection Accuracy (MFCC/SMO) (%)	86	83
Computational Cost (× real-time)	1.91	0.23

Table 14: *Performance comparison between the Ncut and HESR approaches. Computational cost is expressed in terms of real-time and separation performance in terms of SSDR (in dB).*

D.2 Corpus Description and Experimental Setup

The same 10 song dataset and the 23 music clips from the MIREX audio melody extraction dataset used for the evaluation of the Ncut approach (both described in Section 4.3.1) were used for the evaluations of the HESR. This allows a direct comparison with the Ncut results presented in the previous sections.

D.3 Experimental Results

For the separation experiment, the 10 song dataset was used similarly to the way explained in Section 4.3.1. Table 14 presents the comparative results between the Ncut and the HESR approaches for different tasks. The first row in the table presents the mean value of the SSDR achieved when using all the signals in the dataset. As it is possible to see, the separation performance drops by approximately 3 dB when using the HESR method. This may be due to the fact that the HESR algorithm only considers the HWPS cue to select the frequency components with the highest harmonicity index, ignoring all other grouping cues (e.g. frequency and amplitude proximity) used by the Ncut approach.

For the main pitch estimation experiment, the music tracks from the MIREX dataset were used. The pitch contour from the dominant melodic voice was estimated for each audio signal using the Praat pitch estimation [Boersma and Weenink, 2006] on the resynthesized signals coming out of both HESR and Ncut methods. In this experiment the HESR achieved better results, which seems to confirms the hypothesis that the proposed harmonicity criterion is able to select a significant number of components from the dominant harmonic source.

A voicing detection evaluation was also conducted, where the objective was to identify whether a given time frame contains a "melody" pitch or not. The goal of this experiment was to determine whether the HESR algorithm can be used to achieve a good voicing detection accuracy in monaural polyphonic recordings. The dataset of the 10 polyphonic music

pieces was used for this experiment. As before, the voicing regions were manually labeled from the original vocal track and were used as the ground truth. A supervised learning approach was then used to train voicing/no-voicing classifiers for two configurations, as presented in table 14: Ncut refers to using Mel-Frequency Cepstral Coefficients (MFCC) calculated over the Ncut automatically separated voice signal, and HESR refers to MFCC calculated over the HESR mixed voice and music signal. The experiments were once again conducted using the Weka machine learning framework [Witten and Frank, 2005], where a support vector machine was trained using the sequential minimal optimization (SMO).

As shown, the HESR MFCC accuracy compares well to the slightly superior value achieved when using the Ncut and MFCC approach. Not presented in the table, but of comparative interest, is the accuracy of the MFCC feature using the same classifier but when applied directly to the original mixed signal (i.e. without any Ncut separation or HESR processing). For this case a 69% accuracy is obtained, a quite inferior value in comparison to the two values discussed above.

For all the above experiments a 2.2 GHz Intel machine was used for running the HESR and Ncut algorithms, both implemented in C++ using the Marsyas framework [1] (see Chapter 5 for more details on the software implementation conducted in the scope of this thesis). The last row of Table 14 shows a relative measure of the computation time taken by running the HESR and Ncut algorithms on the all the songs in the 10 song dataset. The HESR approach is about 8 times faster than the Ncut algorithm, making it a more practical option, mainly when used as a front-end of pre-processing step in MIR tasks. Nevertheless, the much more complex and generic Ncut algorithm is just roughly 2 times real-time, and this figure will eventually be improved and allow real-time operation by mainly introducing optimizations in the normalized cut implementation.

D.4 Discussion

All the presented experiments show that the proposed HESR approach, even though not capable of achieving comparable results for the source separation task, is able to attain convincing results as a front-end for MIR tasks such as pitch estimation and voicing detection at a low computational cost. Additionally, and because the HESR is based on the sole use of the HWPS, the results confirm the relevance of this grouping cue to the segregation of harmonic content from complex audio mixtures. Thus, HESR could potentially be used as a efficient front-end for chroma-based analysis of music signals.

[1] `http://marsyas.sourceforge.net`

List of References

[Abdallah, 2002] Abdallah, S. A. (2002). *Towards Music Perception by Redundancy Reduction and Unsupervised Learning in Probabilistic Models*. Phd thesis, Department of Electronic Engineering, King's College, London.

[Abe and Smith, 2004] Abe, M. and Smith, J. O. (2004). Design Criteria for Simple Sinusoidal Parameter Estimation based on Quadratic Interpolation of FFT Magnitude Peaks. In *Proc. 117th Convention of the Audio Engineering Society*, San Francisco, USA. Preprint 6256.

[Ackerman, 1982] Ackerman, W. (1982). Data flow languages. *Computer*, 15(2):15–25.

[Amatriain, 2007] Amatriain, X. (2007). CLAM, a framework for audio and music application development. *IEEE Software*, 24(1):82–85.

[Ashcroft and Wadge, 1977] Ashcroft, E. A. and Wadge, W. (1977). Lucid, a nonprocedural language with iteration. *Communications of the ACM*, pages 519–526.

[Aucouturier and Pachet, 2004] Aucouturier, J.-J. and Pachet, F. (2004). Improving timbre similarity: How high's the sky? *Journal of Negative Results in Speech and Audio Sciences*, 1(1).

[Avendano, 2003] Avendano, C. (2003). Frequency-domain source identification and manipulation in stereo mixes for enhancement, suppression, and re-panning applications. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, USA.

[Bach and Jordan, 2004] Bach, F. and Jordan, M. I. (2004). Blind one-microphone speech separation: A spectral learning approach. In *Proc. Neural Information Processing Systems (NIPS)*, Vancouver, Canada.

[Bach and Jordan, 2006] Bach, F. R. and Jordan, M. I. (2006). Learning spectral clustering, with application to speech separation. *Journal of Machine Learning Research*, 7:1963–2001.

[Badeau et al., 2006] Badeau, R., David, B., and Richard, G. (2006). High resolution spectral analysis of mixtures of complex exponentials modulated by polynomials. *IEEE Transactions on Signal Processing*, 54(4):1341–1350.

[Balakrishnan, 1997] Balakrishnan, V. K. (1997). *Graph Theory*. McGraw-Hill, 1st edition.

[Beerends, 1998] Beerends, J. G. (1998). Audio quality determination based on perceptual measurement techniques. In Kahrs, M. and Brandenburg, K., editors, *Applications of Digital Signal Processing to Audio and Acoustics*, pages 1–38. Kluwer Academic Publishers.

[Beerends et al., 2002] Beerends, J. G., Hekstra, A. P., Rix, A. W., and Hollier, M. P. (2002). Perceptual evaluation of speech quality (PESQ), the new ITU standard for end-to- end speech quality assessment, part II — psychoacoustic model. *Journal of the Audio Engineering Society*, 50(10).

[Beerends and Stemerdink, 1992] Beerends, J. G. and Stemerdink, J. A. (1992). A perceptual audio quality measure based on a psychoacoustical sound representation. *Journal of the Audio Engineering Society*, 40(12).

[Bello et al., 2005] Bello, J., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., and Sandler, M. (2005). A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035–1047.

[Bello, 2003] Bello, J. P. (2003). *Towards the Automated Analysis of Simple Polyphonic Music: A Knowledge-based Approach*. Phd thesis, Queen Mary, University of London.

[Beranek, 1990] Beranek, L. L. (1990). *Acoustics*. American Institute of Physics.

[Berenzweig and Ellis, 2001] Berenzweig, A. L. and Ellis, D. P. W. (2001). Locating singing voice segments within music signals. In *Proc. IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 119–122, New Paltz, New York.

[Bey and McAdams, 2002] Bey, C. and McAdams, S. (2002). Schema-based processing in auditory scene analysis. *Perception and Psychophysics*, 64:844–854.

[Bigand, 1993] Bigand, E. (1993). *Contributions of Music to research on human auditory cognition*, chapter 8, pages 231–273. Thinking in Sound. Oxford University Press.

[Bigand and Poulin-Charronnat, 2006] Bigand, E. and Poulin-Charronnat, B. (2006). Are we "experienced listenes"? a review of the musical capacities that do not depend on formal musical training. *Cognition*, 100(1):100–130.

[Boersma and Weenink, 2006] Boersma, P. and Weenink, D. (2006). Praat: doing phonetics by computer (version 4.5.06). Retrieved December 13, 2006, from http://www.praat.org/.

[Bray and Tzanetakis, 2005] Bray, S. and Tzanetakis, G. (2005). Implicit patching for dataflow-based audio analysis and synthesis. In *In Proceedings of International Music Conference (ICMC)*.

[Bregman, 1990] Bregman, A. (1990). *Auditory Scene Analysis – The Perceptual Organization of Sound*. MIT Press.

[Bregman, 1993] Bregman, A. S. (1993). *Auditory Scene Analysis: hearing in complex environments*, chapter 2, pages 10–34. Thinking in Sound. Oxford University Press.

[Bregman and Ahad, 1996] Bregman, A. S. and Ahad, P. A. (1996). Demonstrations of auditory scene analysis: The perceptual organization of sound. Audio CD with booklet, Department of Psychology, McGill University.

[Brown, 1994] Brown, G. J. (1994). *Computational Auditory Scene Analysis: A Representational Approach*. Phd thesis, University of Sheffield.

[Brown and Cooke, 1994] Brown, G. J. and Cooke, M. (1994). Computational auditory scene analysis. *Computer Speech and Language*, 8(4):297–336.

[Burred et al., 2006] Burred, J. J., Röbel, A., and Rodet, X. (2006). An accurate timbre model for musical instruments and its application to classification. In *Proc. Workshop on Learning the Semantics of Audio Signals*, Athens, Greece.

[Burred and Sikora, 2007] Burred, J. J. and Sikora, T. (2007). Monaural source separation from musical mixtures based on time-frequency timbre models. In *International Conference on Music Information Retrieval (ISMIR 2007)*, Vienna, Austria.

[Cai et al., 2005] Cai, R., Lu, L., and Hanjalic, A. (2005). Unsupervised content discovery in composite audio. In *Proc. ACM Multimedia*.

[Cano et al., 2006] Cano, P., Gómez, E., Gouyon, F., Herrera, P., Koppenberger, M., Ong, B., Serra, X., Streich, S., and Wack, N. (2006). ISMIR 2004 Audio Description Contest. MTG Technical Report MTG-TR-2006-02 (under the Creative Commons Attribution-NonCommercial-NoDerivs 2.5 licence), Music Technology Group (MTG), Pompeu Fabra University, Barcelona, Spain.

[Carlin, 1992] Carlin, M. (1992). Radial basis function networks and nonlinear data modelling. In *Proc. Neural Networks and their Applications (Neuro-Nimes'92)*, pages 62363–3.

[Channappayya et al., 2006] Channappayya, S. S., Bovik, A. C., and Heath, R. W. (2006). Design of a linear image estimator optimized for the structural similarity index and its application to image denoising. In *Proc. IEEE International Conference on Image Processing*, Atlanta, Georgia.

[Cherry, 1953] Cherry, E. C. (1953). Some experiments on the recognition of speech, with one and with two ears. *Journal of Acoustical Society of America*, 25(5):975–979.

[Chowning, 1973] Chowning, J. (1973). The synthesis of complex audio spectra by means of frequency modulation. *Journal of the Audio Engineering Society*, 21(7).

[Chung, 1997] Chung, F. (1997). *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. Conference Board of the Mathematical Sciences, Washington.

[Cook and Scavone, 1999] Cook, P. and Scavone, G. (1999). The synthesis toolkit (STK) version 2.1. In *Proc. International Computer Music Conference (ICMC)*, Beijing, China.

[Cooke, 1991] Cooke, M. P. (1991). *Modelling Auditory Processing and Organization*. Phd thesis, University of Sheffield.

[Cooney et al., 2006] Cooney, R., Cahill, N., and Lawlor, R. (2006). An enhanced implementation of the ADRess (Azimuth Discrimination and Resynthesis) Music Source Separation Algorithm. In *Proc. 121st Convention of the Audio Engineering Society*, San Francisco, USA.

[Dannenberg and Brandt, 1996] Dannenberg, R. and Brandt, E. (1996). A flexible real-time software synthesis system. In *Proc. International Computer Music Conference (ICMC)*, pages 270–273.

[Dannenberg et al., 2004] Dannenberg, R. B., Birmingham, W. P., Tzanetakis, G., Meek, C., Hu, N., and Pardo, B. (2004). The MUSART testbed for query-by-humming evaluation. *Computer Music Journal*, 28(2):34–48.

[Darwin, 1997] Darwin, C. J. (1997). Auditory grouping. *Trends in Cognitive Science*, 1:327–333.

[Davis and Mermelstein, 1980] Davis, S. and Mermelstein, P. (1980). Experiments in syllable-based recognition of continuous speech. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 28:357–366.

[Deutsch, 1975] Deutsch, D. (1975). Musical illusions. *Scientifc American*, 233:92–104.

[Deutsch, 1999] Deutsch, D., editor (1999). *The Psychology of Music*. Academic Press, San Diego, 2nd edition.

[Deutsch and Deutsch, 1993] Deutsch, J. A. and Deutsch, D. (1993). *Attention: Some theoretical considerations*, volume The Psychology of Attention of *International Library of Critical Writings in Psychology*. Edward Elgar Publishing, Aldershot.

[Dhillon et al., 2007] Dhillon, I., Guan, Y., , and Kulis, B. (2007). Weighted graph cuts

without eigenvectors: A multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, (11):1944–1957.

[Dixon, 2006] Dixon, S. (2006). Onset detection revisited. In *Proc. International Conference on Digital Audio Effects (DAFx)*, Montreal, Canada.

[Donath and Hoffman, 1973] Donath, W. E. and Hoffman, A. J. (1973). Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17:420–425.

[Downie, 2003] Downie, J. (2003). Music information retrieval. *Annual Review of Information Science and Technology*, 37:295–340.

[Dubnov and Appel, 2004] Dubnov, S. and Appel, T. (2004). Audio segmentation by singular value clustering. In *Proc. International Computer Music Conference (ICMC)*.

[Duda et al., 2000] Duda, R. O., Hart, P. E., and Stork, D. G. (2000). *Pattern Classification*. Wiley-Interscience, 2nd edition.

[Ellis, 2007] Ellis, D. (2007). Classifying music audio with timbral and chroma features. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, pages 339–340, Vienna, Austria.

[Ellis et al., 2008] Ellis, D., Cotton, C., and Mandel, M. (2008). Cross-correlation of beat-synchronous representations for music similarity. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 57–60, Las Vegas, NV, USA.

[Ellis and Lee, 2004] Ellis, D. and Lee, K. (2004). Minimal-impact audio-based personal archives. In *Proc. ACM Workshop on Continuous Archival and Retrieval of Personal Experience (CARPE)*, New York, USA.

[Ellis, 1996] Ellis, D. P. W. (1996). *Prediction-driven computational auditory scene analysis*. Phd thesis, Massachusetts Institute of Technology (MIT).

[Ellis, 2006] Ellis, D. P. W. (2006). *Model-Based Scene Analysis*, chapter 4, pages 115–143. Computational Auditory Scene Analysis: Principles, Algorithms and Applications. Wiley-IEEE Press.

[Ellis and Rosenthal, 1995] Ellis, D. P. W. and Rosenthal, D. F. (1995). Mid-level representations for computational auditory scene analysis. In *Proc. International Joint Conference on Artificial Intelligence*, Montreal, Quebec.

[Esquef et al., 2002] Esquef, P. A. A., Välimäki, V., and Karjalainen, M. (2002). Restoration and enhancement of solo guitar recordings based on sound source modeling. *Journal of the Audio Engineering Society*, 50(5).

[Essid et al., 2005] Essid, S., Richard, G., and David, B. (2005). Instrument recognition in polyphonic music. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Philadelphia, USA.

[Every, 2006] Every, M. R. (2006). *Separation of Musical Sources and Structure from Single-Channel Polyphonic Recordings*. Phd thesis, University of York, Department of Electronics.

[Ferreira and Sinha, 2005] Ferreira, A. and Sinha, D. (2005). Accurate spectral replacement. In *Proc. 118th Convention of the Audio Engineering Society*.

[Ferreira, 2001] Ferreira, A. J. S. (2001). Accurate estimation in the ODFT domain of the frequency, phase and magnitude of stationary sinusoids. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, New York.

[Fletcher, 1940] Fletcher, H. (1940). Auditory patterns. *Review of Modern Physics*, 12:47–65.

[Gamma et al., 1995] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns: Elements of reusable object-oriented software*. Addison Wesley.

[Gerald Kidd et al., 2003] Gerald Kidd, J., Mason, C. R., Brughera, A., and Chiu, C.-Y. P. (2003). Discriminating harmonicity. *The Journal of the Acoustical Society of America*, 114(2):967–977.

[Ghias et al., 1995] Ghias, A., Logan, J., Chamberlin, D., and Smith, B. (1995). Query by Humming: Musical Information Retrieval in an Audio Database. *ACM Multimedia*, pages 213–236.

[Gibson, 1979] Gibson, J. (1979). *The Ecological Approach to Visual Perception*. Houghton Mifflin, Boston.

[Godsmark and Brown, 1999] Godsmark, D. and Brown, G. J. (1999). A blackboard architecture for computational auditory scene analysis. *Speech Communication*, 27(3-4):351–366.

[Godsmark, 2000] Godsmark, D. J. (2000). *A Computational Model of the Perceptual Organization of Polyphonic Music*. Phd thesis, University of Sheffield.

[Golub and Loan, 1989] Golub, G. H. and Loan, C. F. V. (1989). *Matrix Computations*. John Hopkins Press.

[Gomez, 2006] Gomez, E. (2006). Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing*, 18(3):294–304.

[Goto, 2006] Goto, M. (2006). *Analysis of Musical Audio Signals*, chapter 8, pages 251–295. Computational Auditory Scene Analysis: Principles, Algorithms and Applications. Wiley-IEEE Press.

[Goto et al., 2003] Goto, M., Hashiguchi, H., Nishimura, T., and Oka, R. (2003). RWC music database: Music genre database and musical instrument sound database. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, pages 229–230.

[Gouyon, 2005] Gouyon, F. (2005). *A computational approach to rhythm description — Audio features for the computation of rhythm periodicity functions and their use in tempo induction and music content processing*. Phd thesis, Universitat Pompeu Fabra, Barcelona, Spain.

[Gouyon et al., 2008] Gouyon, F., Herrera, P., Gómez, E., Cano, P., Bonada, J., Loscos, À., Amatriain, X., and Serra, X. (2008). *Content processing of music audio signals*. Sound to sense, sense to sound: A state-of-the-art. Logos Verlag. (in press).

[Handel, 1989] Handel, S. (1989). *Listening – An Introduction to the Perception of Auditory Events*. MIT Press.

[Hartmaan, 1988] Hartmaan, W. M. (1988). *Pitch perception and the segregation and integration of auditory entities*, pages 623–645. Auditory Function: Neurobiological Bases of Hearing. Wiley, New York.

[Hartmann, 1998] Hartmann, W. M. (1998). *Signals, Sound, and Sensation*. AIP Press - Springer.

[Herrera et al., 2003] Herrera, P., Peeters, P., and Dubnov, S. G. (2003). Automatic classification of musical instrument sounds. *Journal of New Music Research*, 32(1):3–22.

[Humphreys and Bruce, 1989] Humphreys, G. W. and Bruce, V. (1989). *Visual Cognition: Computational, Experimental and Neuropsychological Perspectives*. Psychology Press.

[Huron, 1991] Huron, D. (1991). Auditory scene analysis: The perceptual organization of sound by albert s. bregman. *Psychology of Music*, 19(1):77–82.

[Jones, 1990] Jones, M. R. (1990). Learning and the development of expectancies: an interactionist approach. *Psychomusicology*, 9:193–228.

[Jourjine et al., 2000] Jourjine, A., Richard, S., and Yilmaz, O. (2000). Blind Separation of Disjoint Orthogonal Signals: Demixing N Sources From 2 Mixtures. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

[Kashino, 2006] Kashino, K. (2006). *Auditory Scene Analysis in Music Signals*, chapter 10, pages 299–325. Computational Auditory Scene Analysis: Principles, Algorithms and Applications. Wiley-IEEE Press.

[Kashino and Murase, 1999] Kashino, K. and Murase, H. (1999). A sound source identification system for ensemble music based on template adaptation and music stream extraction. *Speech Communication*, (27):337–349.

[Kayser et al., 2005] Kayser, C., Petkov, C., Lippert, M., and Logothetis, N. (2005). Mechanisms for allocating auditory attention: An auditory saliency map. *Current Biology*, 15(21):1943–1947.

[Kim et al., 2006] Kim, Y., Williamson, D., and S.Pilli (2006). Towards quantifying the album effect in artist identification. In *Proc. International Conference on Music Information Retrieval (ISMIR)*.

[Klapuri, 2006] Klapuri, A. (2006). Multiple fundamental frequency estimation by summing harmonic amplitudes. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, Victoria, BC, Canada.

[Klapuri, 2008] Klapuri, A. (2008). Multipitch analysis of polyphonic music and speech signals using and auditory model. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2). Special Issue on MIR.

[Klapuri and Davy, 2006] Klapuri, A. and Davy, M., editors (2006). *Signal Processing Methods for Music Transcription*. Springer-Verlag.

[Klapuri, 2004] Klapuri, A. P. (2004). Automatic music transcription as we know it today. *Journal of New Music Research*, 33(3):269–282.

[Klapuri et al., 2006] Klapuri, A. P., Eronen, A. J., and Astola, J. T. (2006). Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech and Language Processing*, 14(1):342–355.

[Kostek, 2004] Kostek, B. (2004). Musical instrument classification and duet analysis employing music information retrieval techniques. *Proceedings of the IEEE*, 92(4):712–729.

[Kotti et al., 2006a] Kotti, M., Benetos, E., Kotropoulos, C., and Martins, L. G. (2006a). Speaker change detection using BIC: A comparison on two datasets. In *Proc. International Symposium on Communications, Control and Signal Processing*, Marrakech, Morocco.

[Kotti et al., 2006b] Kotti, M., Martins, L. G., Benetos, E., Cardoso, J. S., and Kotropoulos, C. (2006b). Automatic speaker segmentation using multiple features and distance measures: A comparison of three approaches. In *Proc. IEEE International Conference on Multimedia and Expo (ICME)*, Toronto, Canada.

[Krumhansl, 1992] Krumhansl, C. L. (1992). *Grouping processes in infants' music perception*. Grouping in Music. Royal Swedish Academy of Music, Stockholm, Sweden.

[Lagrange and Marchand, 2007] Lagrange, M. and Marchand, S. (2007). Estimating the instantaneous frequency of sinusoidal components using phase-based methods. *Journal of the Audio Engineering Society*, 55(5):385–399.

[Lagrange et al., 2007a] Lagrange, M., Marchand, S., and Rault, J. (2007a). Enhancing the tracking of partials for the sinusoidal modeling of polyphonic sounds. *IEEE Transactions on Acoustics, Speech and Signal Processing*.

[Lagrange et al., 2008a] Lagrange, M., Martins, L. G., Murdoch, J., and Tzanetakis, G. (2008a). Normalized cuts for predominant melodic source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2). Special Issue on MIR.

[Lagrange et al., 2007b] Lagrange, M., Martins, L. G., and Tzanetakis, G. (2007b). Semi-automatic mono to stereo up-mixing using sound source formation. In *Proc. 112th Convention of the Audio Engineering Society*, Vienna, Austria.

[Lagrange et al., 2008b] Lagrange, M., Martins, L. G., and Tzanetakis, G. (2008b). A computationally efficient scheme for dominant harmonic source separation. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Las Vegas, Nevada, USA.

[Lagrange et al., 2006] Lagrange, M., Murdoch, J., and Tzanetakis, G. (2006). Temporal constraints for sound source formation using the normalized cut. In *Proc. Neural Information Processing Systems Workshop (NIPS)*, Whistler, BC, Canada.

[Lagrange and Tzanetakis, 2007] Lagrange, M. and Tzanetakis, G. (2007). Sound source tracking and formation using normalized cuts. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Honolulu, USA.

[Levine, 1998] Levine, S. N. (1998). *Audio Representations for Data Compression and Compressed Domain Processing*. Phd thesis, Stanford University, USA.

[Levinson, 1997] Levinson, J. (1997). *Music in the moment*. Cornell University, Ithaca, NY.

[Li and Wang, 2006] Li, Y. and Wang, D. (2006). Singing voice separation from monaural recordings. In *Proc. International Conference on Music Information Retrieval (ISMIR)*.

[Li and Wang, 2007] Li, Y. and Wang, D. (2007). Separation of singing voice from music accompaniement for monaural recordings. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1475–1487.

[Lim, 1983] Lim, J. S., editor (1983). *Speech Enhancement*. Prentice Hall, Englewood Cliffs, NJ.

[Livshin and Rodet, 2004] Livshin, A. and Rodet, X. (2004). Musical instrument identification in continuous recordings. In *Proc. International Conference on Digital Audio Effects (DAFx)*, Naples, Italy.

[Livshin and Rodet, 2006] Livshin, A. and Rodet, X. (2006). The importance of the non-harmonic residual for automatic musical instrument recognition of pitched instruments. In *Proc. 120th Convention of the Audio Engineering Society*, Paris.

[Manulescu, 1997] Manulescu, D. A. (1997). A dataflow pattern language. In *Proc. Pattern languages of Programming*, Monticello, Illinois.

[Marchand and Lagrange, 2006] Marchand, S. and Lagrange, M. (2006). On the equivalence of phase-based methods for the estimation of instantaneous frequency. In *Proc. European Conference on Signal Processing (EUSIPCO)*.

[Marolt, 2004] Marolt, M. (2004). A connectionist approach to automatic transcription of polyphonic piano music. *IEEE Transactions on Multimedia*, 6(3):439–449.

[Marr, 1982] Marr, D. (1982). *Vision*. W. H. Freeman, San Francisco, CA.

[Martins, 2002] Martins, L. G. (2002). PCM to MIDI Transposition. MSc thesis, Faculdade de Engenharia da Universidade do Porto (FEUP).

[Martins et al., 2007] Martins, L. G., Burred, J. J., Tzanetakis, G., and Lagrange, M. (2007). Polyphonic instrument recognition using spectral clustering. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria.

[Martins and Ferreira, 2002] Martins, L. G. and Ferreira, A. J. S. (2002). PCM to MIDI Transposition. In *Proc. 116th Convention of the Audio Engineering Society*, Munich, Germany.

[McAdams, 1984] McAdams, S. (1984). *Spectral fusion, spectral parsing and the formation of auditory images*. Phd thesis, CCRMA Stanford University.

[McAdams and Bigand, 1993] McAdams, S. and Bigand, E., editors (1993). *Thinking in Sound*. Oxford University Press.

[McAulay and Quatieri, 1986] McAulay, R. and Quatieri, T. (1986). Speech analysis/synthesis based on a sinusoidal representation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(4):744–754.

[Meila and Shi, 2001] Meila, M. and Shi, J. (2001). A random walks view of spectral segmentation. In *Proc. International Wokshop on Artificial Inteligence and Statistics (AISTATS)*.

[Mellinger, 1991] Mellinger, D. (1991). *Event Formation and Separation in Musical Sounds*. Phd thesis, Stanford University.

[Mermelstein, 1979] Mermelstein, P. (1979). Evaluation of a segmental snr measure as an indicator of the quality of adpcm coded speech. *Journal of the Acoustical Society of America*, 66(6).

[Mohar, 1991] Mohar, B. (1991). *The Laplacian spectrum of graphs*, volume 2 of *Graph teory, combinatorics, and applications*, pages 871–898. Wiley, New York.

[Moore, 1989] Moore, B. C. J. (1989). *An Introduction to the Psychology of Hearing*. Academic Press.

[Moore, 1995] Moore, B. C. J. (1995). *Hearing (Handbook of Perception and Cognition)*. Academic Press, second edition.

[Nakatani et al., 1995] Nakatani, T., Okuno, H. G., and Kawabata, T. (1995). Residue-driven architecture for computational scene analysis. In *Proc. International Joint Conference on Artificial Inteligence (IJCAI)*, pages 165–172, Montreal, Quebec, Canada.

[Ng et al., 2002] Ng, A., Jordan, M., and Weiss, Y. (2002). *On spectral clustering: analysis and an algorithm*, pages 849–856. Advances in Neural Information Processing Systems 14. MIT Press.

[Nwe et al., 2004] Nwe, T. L., Shenoy, A., and Wang, Y. (2004). Singing voice detection in popular music. In *Proc. ACM Multimedia Conference*, New York, NY, USA.

[Oppenheim and Schafer, 1975] Oppenheim, A. V. and Schafer, R. W. (1975). *Digital Signal Processing*. Prentice-Hall.

[Orio, 2006] Orio, N. (2006). Music retrieval: A tutorial and review. *Foundations and Trends in Information Retrieval*, 1(1):1–90.

[Ozerov et al., 2005] Ozerov, A., Philippe, P., Gribonval, R., and Bimbot, F. (2005). One microphone singing voice separation using source-adapted models. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, USA.

[Pachet and Cazaly, 2000] Pachet, F. and Cazaly, D. (2000). A classification of musical genre. In *Proc. RIAO Content-Based Multimedia Information Access Conference*.

[Paiva, 2006] Paiva, R. (2006). *Melody Detection in Polyphonic Audio*. Phd thesis, Universidade de Coimbra, Coimbra, Portugal.

[Paiva et al., 2006] Paiva, R. P., Mendes, T., and Cardoso, A. (2006). Melody detection in polyphonic musical signals: Exploiting perceptual rules, note salience, and melodic smoothness. *Computer Music Journal*, 30(4):80–98.

[Palmer, 1999] Palmer, S. E. (1999). *Vision Science*. MIT Press, Cambridge, Massachusetts.

[Parsons, 1976] Parsons, T. W. (1976). Separations of speech from interfering speech by means of harmonic selection. *Journal of the Acoustical Society of America*, 60:911–918.

[Patterson and Moore, 1986] Patterson, R. and Moore, B. (1986). *Auditory filters and excitation patterns as representations of frequency resolution*, pages 123–177. Frequency Selectivity in Hearing. Academic Press, London.

[Pereira and Ebrahimi, 2002] Pereira, F. and Ebrahimi, T. (2002). *The MPEG-4 Book*. Prentice Hall PTR. ISBN: 0-130-61621-4.

[Pinker, 1984] Pinker, S. (1984). Visual cognition: an introduction. *Cognition*, 18(1-3):1–63.

[Poliner et al., 2007] Poliner, G., Ellis, D., Ehmann, A., Gomez, E., Streich, S., and Ong, B. (2007). Melody transcription from music audio: Approaches and evaluation. *IEEE Transactions on Audio, Speech and Language Processing*, 15(4).

[Puckette, 1997] Puckette, M. (1997). Pure data. In *In Proceedings of International Music Conference (ICMC)*, pages 269–272.

[Puckette and Brown, 1998] Puckette, M. S. and Brown, J. C. (1998). Accuracy of frequency estimates using the phase vocoder. *IEEE Transactions on Audio and Speech Processing*, 6(2).

[Rabiner and Juang, 1993] Rabiner, L. and Juang, B.-H. (1993). *Fundamentals of speech recognition*. Prentice-Hall, Inc.

[Rocamora and Herrera, 2007] Rocamora, M. and Herrera, P. (2007). Comparing audio descriptors for singing voice detection in music audio files. In *Proc. 11th Brazilian Symposium on Computer Music*, São Paulo, Brazil.

[Rosenthal and Okuno, 1998] Rosenthal, D. F. and Okuno, H. G., editors (1998). *Computational auditory scene analysis*. Lawrence Erlbaum Associates, Inc.

[Rosier and Grenier, 2004] Rosier, J. and Grenier, Y. (2004). Unsupervised classification techniques for multipitch estimation. In *Proc. 116th Convention of the Audio Engineering Society*.

[Roweis, 2000] Roweis, S. T. (2000). One microphone source separation. In *Proc. Neural Information Processing Systems (NIPS)*, pages 793–799.

[Ryynanen and Klapuri, 2006] Ryynanen, M. and Klapuri, A. (2006). Transcription of the singing melody in polyphonic music. In *Proc. International Conference on Music Information Retrieval (ISMIR)*.

[Scavone and Cook, 2005] Scavone, G. and Cook, P. (2005). RTMIDI, RTAUDIO, AND A SYNTHESIS TOOLKIT (STK) UPDATE. In *Proc. International Computer Music Conference (ICMC)*, Barcelona, Spain.

[Scheirer, 1998] Scheirer, E. D. (1998). The MPEG-4 Structured Audio standard. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 6, pages 3801–3804, Seattle, Washington, USA.

[Scheirer, 2000] Scheirer, E. D. (2000). *Music-Listening Systems*. Phd thesis, Massachusetts Institute of Technology (MIT).

[Serrà and Gómez, 2007] Serrà, J. and Gómez, E. (2007). A cover song identification system based on sequences of tonal descriptors. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria.

[Serrà and Gómez, 2008] Serrà, J. and Gómez, E. (2008). Audio cover song identification based on tonal sequence alignment. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Las Vegas, NV, USA.

[Serra, 1989] Serra, X. (1989). *A System for Sound Analysis/Transformation/Synthesis based on a Deterministic plus Stochastic Decomposition*. Phd thesis, Stanford University.

[Serra and Smith, 1990] Serra, X. and Smith, J. (1990). Spectral modeling synthesis - a sound analysis synthesis system based on a deterministic plus stochastic decomposition. *Computer Music Journal*, 14(4):12–24.

[Serra et al., 2007] Serra, X., Widmer, G., and Leman, M. (2007). *A Roadmap for Sound and Music Computing*.

[Sethares and Staley, 1999] Sethares, W. A. and Staley, T. W. (1999). Periodicity transforms. *IEEE Transactions on Signal Processing*, 47(11):2953–2964.

[Shepard, 1981] Shepard, R. N. (1981). *Psycophysical complementarty*, pages 279–341. Perceptual Organization. Erlbaum, Hillsdale, NJ.

[Shi and Malik, 2000] Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

[Shneiderman and Aris, 2006] Shneiderman, B. and Aris, A. (2006). Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5).

[Slaney, 1998] Slaney, M. (1998). *A Critique of Pure Audition*, chapter 3. Computational Auditory Scene Analysis. Lawrence Erlbaum Associates, Inc.

[Slaney and Lyon, 1993] Slaney, M. and Lyon, R. F. (1993). *On the importance of time – a temporal representation of sound*. Visual representation of Speech Signals. John Wiley & Sons.

[Smith, 1992] Smith, J. O. (1992). Physical modeling using digital waveguides. *Computer Music Journal,*, 16(Winter):74–91.

[Srinivasan, 2004] Srinivasan, S. H. (2004). Auditory blobs. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 4, pages 313–316, Montreal, Canada.

[Srinivasan and Kankanhalli, 2003] Srinivasan, S. H. and Kankanhalli, M. S. (2003). Harmonicity and dynamics based audio separation. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 5, pages v–640 – v–643, Hong Kong, China.

[Stevens, 1957] Stevens, S. S. (1957). On the psychophysical law. *Psychology Revue*, 64(3):153–181.

[Summerfield et al., 1990] Summerfield, Q., Lea, A., and Marshall, D. (1990). Modelling auditory scene analysis: strategies for source segregation using autocorrelograms. In *Proc. Institute of Acoustics*, volume 12, pages 507–514.

[Terhardt, 1987] Terhardt, E. (1987). *Gestalt principles and music perception*, pages 157–166. Auditory processing of complex sounds. Erlbaum, Hillsdale, NJ.

[Tolonen, 2000] Tolonen, T. (2000). *Object-Based Sound Source Modeling*. Phd thesis, Helsinki University of Technology.

[Torkkola, 2000] Torkkola, K. (2000). *Blind separation of delayed and convolved sources*, pages 321–375. Unsupervised Adaptive Filtering – Volume 1: Blind Source Separation. John Wiley & Sons.

[Trehub and Trainor, 1993] Trehub, S. E. and Trainor, L. J. (1993). *Listening strategies in infancy: the roots of music and language development*, chapter 9, pages 278–317. Thinking in Sound. Oxford University Press.

[Treisman, 1964] Treisman, A. (1964). Selective attention in man. *British Medical Bulletin*, 20:12–16.

[Tzanetakis, 2002] Tzanetakis, G. (2002). *Manipulation, Analysis and Retrieval Systems for Audio Signals*. Phd thesis, Princeton University.

[Tzanetakis, 2008] Tzanetakis, G. (2008). *Marsyas: a case study in implementing Music Information Retrieval Systems*, pages 31–49. Intelligent Music Information Systems: Tools and Methodologies. Information Science Reference. ISBN 978-1-59904-663-1.

[Tzanetakis et al., 2008] Tzanetakis, G., Castillo, C., Jones, R., Martins, L. G., Teixeira, L. F., and Lagrange, M. (2008). Interoperability and the marsyas 0.2 runtime. In *Proc. International Computer Music Conference (ICMC)*, Belfast, Northern Ireland.

[Tzanetakis and Cook, 2000] Tzanetakis, G. and Cook, P. (2000). Marsyas: a framework for audio analysis. *Organized Sound, Cambridge University Press*, 4(3).

[Tzanetakis and Cook, 2002] Tzanetakis, G. and Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 10:293–302.

[Tzanetakis et al., 2007] Tzanetakis, G., Jones, R., and McNally, K. (2007). Stereo panning features for classifying recording production style. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria.

[van Rijsbergen, 1979] van Rijsbergen, K. (1979). *Information Retrieval*. Butterworths, London, 2nd edition.

[Vembu and Baumann, 2005] Vembu, S. and Baumann, S. (2005). Separation of vocals from polyphonic audio recordings. In *Proc. International Conference on Music Information Retrieval (ISMIR)*, London, UK.

[Verma and Meng, 1998] Verma, T. and Meng, T. (1998). An analysis/synthesis tool for transient signals that allows a flexible sines+transients+noise model for audio. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Seattle, WA, USA.

[Vincent, 2006] Vincent, E. (2006). Musical source separation using time-frequency source priors. *IEEE Transactions on Audio, Speech and Language Processing*, 14(1):91–98.

[Vincent et al., 2006a] Vincent, E., Gribonval, R., and Fevotte, C. (2006a). Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1462 – 1469.

[Vincent et al., 2006b] Vincent, E., Jafari, M. G., and Plumbley, M. D. (2006b). Preliminary guidelines for subjective evalutation of audio source separation algorithms. In Nandi, A. K. and Zhu, X., editors, *Proc. ICA Research Network International Workshop*, pages 93–96, Liverpool, UK.

[Vincent and Plumbley, 2005] Vincent, E. and Plumbley, M. D. (2005). A prototype system for object coding of musical audio. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, USA.

[Virtanen, 2006] Virtanen, T. (2006). *Sound Source Separation in Monaural Music Signals*. Phd thesis, Tampere University of Technology.

[Virtanen and Klapuri, 2000] Virtanen, T. and Klapuri, A. (2000). Separation of harmonic sound sources using sinusoidal modeling. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Istanbul, Turkey.

[von Luxburg, 2007] von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416.

[von Noorden, 1975] von Noorden, L. P. A. S. (1975). *Temporal Coherence in the Perception of Tone Sequences*. Phd thesis, Eindhoven University of Technology.

[Wang, 1994] Wang, A. L.-C. (1994). *Instantaneous and Frequency-Warped Signal Processing Techniques for Auditory Source Separation*. Phd thesis, Stanford University, USA.

[Wang and Brown, 2006] Wang, D. and Brown, G. J., editors (2006). *Computational Auditory Scene Analysis: Principles, Algorithms and Applications*. Wiley-IEEE Press.

[Wang and Cook, 2004] Wang, G. and Cook, P. (2004). ChucK: A Programming Language for On-the-fly, Real-time Audio Synthesis and Multimedia. In *Proc. ACM Multimedia*, New York, USA.

[Warren and Warren, 1970] Warren, R. M. and Warren, R. P. (1970). Auditory illusions and confusions (auditory illusions, investigating phonemic restorations, verbal transformations and perceptual organization). *Scientifc American*, 223:30–36.

[Weintraub, 1985] Weintraub, M. (1985). *A Theory and Computational Model of Auditory Monaural Sound Separation*. Phd thesis, Stanford University.

[Witten and Frank, 2005] Witten, I. and Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. 2nd Edition, Morgan Kaufmann.

[Wright et al., 2003] Wright, M., Freed, A., and Momeni, A. (2003). OpenSound Control: State of the Art 2003. In *Proc. International Conference on New Interfaces for Musical Expression (NIME)*, Montreal, Canada.

[Wrigley and Brown, 2004] Wrigley, S. N. and Brown, G. J. (2004). A computational model of auditory selective attention. *IEEE Transactions on Neural Networks*, 15(5):1151–1163.

[Yost, 1997] Yost, W. A. (1997). *The cocktail party problem: Forty years later*, pages 329–347. Binaural and Spatial Hearing in Real and Virtual Environments. Lawrence Erlbaum, Mahwah, NJ.

[Yost, 2001] Yost, W. A. (2001). *Fundamentals of Hearing: An Introduction*. Rinehart and Wirston, New York, 4th edition edition.

[Yost et al., 2007] Yost, W. A., Popper, A. N., and Fay, R. R., editors (2007). *Auditory Perception of Sound Sources*. Springer, 1st edition.

[Zelnik-Manor and Perona, 2004] Zelnik-Manor, L. and Perona, P. (2004). Self-tuning spectral clustering. In *Proc. Neural Information Processing Systems (NIPS)*.

[Zicarelli, 2002] Zicarelli, D. (2002). How i learned to love a program that does nothing. *Computer Music Journal*, 26(4):44–51.

[Zölzer, 2002] Zölzer, U., editor (2002). *DAFX Digital Audio Effects*. Wiley.

[Zwicker and Fastl, 1990] Zwicker, E. and Fastl, H. (1990). *Psychoacoustics: Facts and Models*. Springer-Verlag Berlin.