VOICE-CONTROLLED INTERFACE FOR DIGITAL MUSICAL INSTRUMENTS

STEFANO FASCIANI

B. SC. ENG. UNIVERSITY OF ROME TOR VERGATA 2003

M. SC. ENG. UNIVERSITY OF ROME TOR VERGATA 2006

A THESIS SUBMITTED FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

NUS GRADUATE SCHOOL FOR

INTEGRATIVE SCIENCES AND ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2014

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis. This thesis has also not been submitted for any degree in any university previously.

Stefano Fasciani

31st July 2014

Abstract

In recent decades, the sonic capabilities of digital musical instruments have significantly increased and today musicians confront a very high dimensional control space for interacting with these complex devices. The exploitation of the musical potential represents a significant challenge, as can be appreciated by surveying the proliferation of novel interfaces and techniques to map the performer gesture to instrument controls. Body gesture and in particular hand interaction have intrinsic limits on the number of instrument parameters that can be controlled simultaneously. This thesis proposes an approach in which gestures derived from the vocal timbre are used to extend common musical interfaces, providing an additional control layer for any sound synthesis or processing device.

The use of human voice as input modality for musical purposes has been proposed in the past, but it is yet under-exploited. Previous research has succeeded only in certain scenarios or with specific instruments. Existing methods for mapping voice to instrument are too specialized to be broadly utilized, while generative mapping techniques present limitations in adopting voice as the gesture, and may require complicated training procedures. This research investigates the musical control potential in the human voice and explores novel mapping approaches, within the real-time, low latency and reliability constraints of generic musical interfaces. The high variabilities across digital musical instrument characteristics, user vocal timbres and interaction preferences are open challenges addressed here.

The main contribution of this thesis is a generic and adaptive method to map the voice to digital musical instruments, based on several real-time signal processing and off-line machine learning algorithms for producing and using maps between heterogeneous spaces, with the aim of maximizing the interface expressivity and minimizing the user intervention in the system setup. This comprises a technique to extract robust, continuous and multidimensional gestural data from specific performer's voice, optimizing the computation towards noise minimization and gesture accentuation; a gestural controller based on a novel application and training of self- organizing map technique; a framework to model analytically the relationship between the variation of instrument parameters and the perceptual sonic changes in any deterministic sound synthesis or processing device; a strategy to minimize the dimensionality of instruments control space retrieving discontinuity-free parameters from a reduced sound map; and a dual-layer generative mapping strategy to transform gestures in the vocal space into trajectories in the instrument sonic space, which

confers linear perceptual response and topological coherence, while maximizing the breadth of explorable sonic space over a set of instrumental parameters.

This dissertation also presents a study involving expert musicians in training, exploring and performing with an open-source proof-of-concept prototype of the voice-controlled interface for digital musical instruments. The study provides an additional validation of the proposed method and it includes participants' vision about application, improvement and benefit from the system in different contexts.

The software system embodying the techniques described in this thesis provides a novel end-to-end solution to implement ad hoc vocal interfaces, engendering new paradigms in musical performances that better exploit creativity and virtuosity. Furthermore the modular design of the system and the self-contained contributions are beneficial and relevant in the broader contexts of sound and music computing as well as human-computer interaction.

Acknowledgements

First and foremost, I would like to offer my sincere thanks to my supervisor Lonce Wyse for giving me the opportunity to work with him and for the invaluable mentoring, guidance and inspiration provided through my PhD journey. A very special thanks goes to Kay O'Halloran for encouraging me in pursuing my graduate studies and for her precious advice over the years. My gratitude also goes to my advisory committee members Roger Zimmermann and Mandar Chitre for their support and feedback.

I wish to thank the folks in the NUS Arts and Creativity Lab: Norikazu Mitani, Sudarshan Balachandran and Hiroki Nishino; my former colleagues at the NUS Multimodal Analysis Lab: Alexey Podlasov, Bradley Smith, Sabine Tan and Marissa E; my former colleagues at Atmel Roma DSP Design Centre: Piergiovanni Bazzana, Benedetto Altieri, Elena Pastorelli, Maurizio Cosimi, Antonio Costa, Antonio Cerruto, Andrea Michelotti, Andrea Ricciardi and Pierstanislao Paolucci, Alberto Dell'Olio, Andrea Gentili, Marisa Iannarelli; my former advisors at the University of Rome Tor Vergata Microelectronics & DSPVLSI Lab: Marco Re and Giancarlo Cardarilli; that in one way or another provided me with their assistance, support, or mentoring over the years, contributing to the development of essential skills for this research.

I am very grateful to all musicians that participated in the user studies for evaluating and testing my system and I also would like to thank the friendship and open-source tools from the people I've met at NIME and ICMC conferences.

I gratefully acknowledge the funding received towards my PhD from the NUS NGS Scholarship program and the support received from executives and administration staff of the NUS Graduate School for Integrative Sciences and Engineering and of the Interactive and Digital Media Institute.

Finally, I would like to thank my family, my friends in Rome and the countless buddies in Singapore from all over the world for their persistent moral support.

Contents

List of Figures

List of Tables

List of Abbreviations

ADC	Analog-to-Digital Converter
AM	Amplitude Modulation
ANN	Artificial Neural Network
ASR	Automatic Speech Recognition
CPU	Central Processing Unit
DAW	Digital Audio Workstation
dB	Decibel
DFT	Discrete Fourier Transform
DMI	Digital Musical Instrument
DSP	Digital Signal Processing
DVI	Direct Voice Input
ERB	Equivalent Rectangular Bandwidth
FIR	Finite Impulse Response
FM	Frequency Modulation
GC	Gestural Controller
GUI	Graphical User Interface
HCI	Human-Computer Interaction
HMM	Hidden Markov Model
ICA	Independent Component Analysis
IDW	Inverse Distance Weighting
IIR	Infinite Impulse Response
LDA	Linear Discriminant Analysis
LFO	Low Frequency Oscillation
LLE	Locally Linear Embedding
LPC	Linear Predictive Coding
MDS	Multi-Dimensional Scaling
MFCC	Mel Frequency Cepstrum Coefficients
MIDI	Musical Instrument Digital Interface
MIR	Music Information Retrieval
ML	Machine Learning
MSE	Mean Squared Error
NLDR	Non-Linear Dimensionality Reduction
NIME	New Interfaces for Musical Expression
OSC	Open Sound Control

PCA	Principal Component Analysis
PLP	Perceptual Linear Predictive
RMD	Relative Mean Difference
SDK	Software Development Kit
SNR	Signal-to-Noise Ratio
SOG	Self-Organizing Gestures
SOM	Self-Organizing Maps
STFT	Short Time Fourier Transform
VCI4DMI	Voice-Controlled Interface for Digital Musical Instruments
VST	Virtual Studio Technology

Chapter 1

Introduction

> "For keyboard players who need an extra hand. If you're like most keyboard performers, you've got your hands full – but you still want more sound. The new ARP Sequencer adds rich new textures to your music while it frees both hands for playing keyboards."
>
> – ARP Instruments Inc. (ARP Instruments, 1976)

> "I wish I had more hands so I could do more."
>
> – Hans-Peter Lindstørm (Lindstrøm, 2007)

1.1 Motivation

Since their introduction, electronic or Digital Musical Instruments (DMI) presented an expansive potential that had empowered musicians with novel sonic and performing resources, beyond all the limits of the traditional acoustic instruments. One of the key features of DMI is the physical independence of the sound generation component from the interface, regardless of the specific synthesis technique and the user input modality. The player mechanical energy does not excite directly any instrument's resonant body, but it is converted into electric control signals for powered analog or digital circuitry, which generates an electric wave transduced to sound by loudspeakers.

Improving the sonic capabilities of synthesis or processing algorithms was the prevailing research focus for decades, while interfacing aspects had a minor emphasis. Simple buttons, rotary dial, faders or piano-like keyboards were used extensively, but these had showed their limitation in providing expressive control over modern DMI (Levitin, McAdams, and Adams, 2002), especially in timbre and texture manipulation which may expose hundreds of parameters to the user. In the twenty-first century, providing effective control strategies to transfer the rich and complex sonic potential to performers became a central challenge. It motivated a wide spectrum of works on *New Interfaces for Musical Expression* (NIME), initially

considered a branch of Human-Computer Interaction (HCI) (Poupyrev et al., 2001). In a few years musical interfaces design gained a significant importance and a growing scientific community had successfully brought in concepts from human-machine interaction, sensors and actuators engineering, computer vision, digital signal processing, machine learning, artificial intelligence and robotics. A variety of novel interfaces and control strategies have recently emerged, supported by the introduction of several hardware and software development tools. These have drastically reduced prototyping time and cost, have hidden the engineering burden of real-time and low latency design required by musical interface, coming to the point where even just a little programming expertise is sufficient to craft customized musical instruments.

Notwithstanding the sensor technology employed, a DMI interface requires some sort of a physical motor interaction of the performer, which can be captured with or without contact (Bongers, 2000). In the vast majority of the musical controllers available today, the interaction exploits performer's hands in different ways (tactile, haptic, gestural). This overlap in the input modality across interfaces determines a physical limit that restricts the number of synthesis or processing parameters controlled simultaneously by a single performer. Providing musicians with multiple interfaces does not usually expand the bandwidth of musical intentions flowing toward instruments. A single interface can easily demand all of a performer's interaction capabilities and prevent the parallel use of another control device. In live electronics and in electronic/computer music live performances automation strategies, where a machine acts as co-player, are often used to cope with this limitation, especially when the ratio "performers over instruments" is particularly low. These constraints minimize the possibility of improvisation and to a certain extent distort the live nature of the performance itself. This issue is rarely addressed since considerations about musical interfaces integrations and cooperation are not common in the conceptual design phase. It pinions and levels off the virtuosity of certain performers, leaving a sense of frustration often expressed in the desire of having extra hands to do more. Augmented instrument may represent an exception since they aim to saturate any *spare bandwidth* (Cook, 2001) of a performer engaged with a single and specific instrument, placing on it extra physical sensors for additional musical control.

The voice can be considered as spare bandwidth in the vast majority of scenarios in which a performer is engaged with one or more musical interface that control computer generated music. The voice is our most proficient and prolific means of communications, it can be considered also as a musical instrument or as a complex

source of controllable sound (Benade, 1990), and the variety of timbres that the vocal tract can generate is very broad compared with most acoustic musical instruments. The underlying idea in this thesis is the adoption of the performer's voice as the gestural input for a generic DMI interface, and we investigate how and what can be mapped between voice and the instrument. We believe that this approach can provide an additional layer of musical control, and that it can widen the control limitation of common physical interfaces. In order to find applications and provide benefits in the largest spectrum of contexts we propose a method which has no constraints in terms of voice and instrument characteristics. This carries several challenges that we address with a flexible mapping strategy, a self-configured interface, and adoption or modification of generic Machine Learning (ML) techniques.

1.2 Aims and main contributions

The aim of this thesis is to introduce a generic and adaptive method to extract and map gestural characteristics of the human voice to an arbitrary number of DMI parameters, meeting the requirements and constraints of musical interfaces such as real-time, minimal latency and reliability. The mapping should ideally not depend on any prior decision or knowledge about the voice or about the instrument, nor from any explicit user specifications, but it should be algorithmically generated starting from an automatic study of the DMI parameters-to-sound characteristic, and from an unsupervised analysis of the performer vocal attributes. From these aims we developed the *Voice-Controlled Interface for Digital Musical Instruments* (VCI4DMI), a system for the implementation of ad hoc vocal interfaces that minimizes the user involvement in the system setup, but, at the same time, maximizes the breadth of explorable sonic space over a set of instrument real-valued parameters. It constitutes an extension to generic interface to effectively perform hands-free musical control tasks. The main contributions of this dissertation, which are essential parts of the system, are:

- A procedure that learns how to extract robust and continuous gestural data from specific voice examples, representative of the control intention, optimizing the computation towards noise minimization and spatial gesture accentuation (Chapter 3).
- A multi dimensional gestural controller based on the output lattice of a self-organizing map trained with a modified algorithm that prevents topology distortions (Chapter 3).

- A framework to model analytically the relationship between an arbitrary number of instrument parameters and the perceptual sonic changes in any deterministic sound synthesis or processing device (Chapter 4).
- A method to minimize the dimensionality of the control space of any DMI retrieving discontinuity-free parameters from a reduced sound map (Chapter 4 and Chapter 5).
- A dual-layer generative strategy to map across heterogeneous spaces, which is used to transform gestures in the vocal space into trajectories in the instrument sonic space, providing linear perceptual response and topological coherence (Chapter 5).
- An open-source proof-of-concept functional prototype of the VCI4DMI that exposes to the end user intermediate settings and mapping options for experimenting different system configuration (Chapter 6).

1.3 Associated publications

Parts of this dissertation have been published in international conferences proceedings and journal articles as listed here.

- Fasciani, S. and Wyse, L. 2012a. "A voice interface for sound generators: adaptive and automatic mapping of gestures to sound". In *Proceedings of the 12th international conference on New Interfaces for Musical Expression*. Ann Arbor, US. – The paper presents the adaptive and generative approach to the vocal control of instrument parameters and a basic functional implementation.
- Fasciani, S. 2012. "Voice features for control: a vocalist dependent method for noise measurement and independent signals computation". In *Proceedings of the 15th international conference on Digital Audio Effects*. York, UK. – This work describes an early adaptive method to compute musical control signals from performer's voice, together with the experimental study on different vocal features.
- Fasciani, S. and Wyse, L. 2012b. "Adapting general purpose interfaces to synthesis engines using unsupervised dimensionality reduction techniques and inverse mapping from features to parameters". In *Proceedings of the 2012 International Computer Music Conference*. Ljubljana, Slovenia. – The paper presents the technique to provide an adapted dimensionality reduction

of the control space of sound generators through the analysis of the perceptual sonic response of an instrument.

- Fasciani, S. and Wyse, L. 2013a. "A self-organizing gesture map for a voice-controlled instrument interface". In *Proceedings of the 13th international conference on New Interfaces for Musical Expression*. Daejeon, Korea. – This work describes the gestural controller based on the modified self-organizing maps, its application for vocal control purposes and a fully working prototype of the system.
- Fasciani, S. and Wyse, L. 2013b. "One at a time by voice: performing with the voice--controlled interface for digital musical instruments". In *Proceedings of the NTU/ADM symposium on Sound and Interactivity 2013*, Singapore, and extended for *eContact! 16.2*. – This article provides an overview of the whole system, including the mapping generation, the runtime operations, the user perspective, and it includes details about the setup of a performance solely based on the developed prototype.

Live solo performances exclusively based on the VCI4DMI were featured in the NTU/ADM Symposium on Sound and Interactivity 2013 concert, in the 2014 Margaret Guthman Musical Instrument Competition[1], where the VCI4DMI was selected as a semi-finalist, and in the fall 2014 Lindblad Electronic Music Meet 2014. Moreover the VCI4DMI was featured in systemic improvised duets and quartets at the 2014 Culture Night at the Academy of Music and Drama of Gothenburg.

1.4 Thesis outline

The structure of the thesis is organized in eight chapters. In Chapter 1 we present context and motivation of this work, detailing aims and contributions of this dissertation. Chapter 2 is dedicated to the scientific background and to the survey on existing systems for musical control driven by voice, including those that can be extended in that direction. From a critical review about their limitation, and from the perspective of the thesis aims we draw a set of design principles for our system. These are refined and further elaborated in Chapter 3 where we first identify the challenges introduced by the thesis objectives, and then we present our vocal gestural controller based on machine learning, that extract intermediate signals from the voice, representative of the performer's control intention. Chapter 4 introduces a taxonomy

[1] http://guthman.gatech.edu

of digital musical instrument, and a framework, which covers all the cases, to automatically obtain a sonic map relative to variation of synthesis or processing parameters. This chapter also defines a technique to reduce the dimensionality of the instrument's control exploiting the computed sonic space. Chapter 5 addresses the issue of how to map across the two heterogeneous spaces, proposing a generative dual-layer mapping strategy that maximizes the overlap, minimizes the loss in explorable instrument sonic space, and avoids discontinuities in parameters retrieval. Next we dedicate Chapter 6 to the proof-of-concept prototype implementation, functionalities and user perspective. It also explains how it can be used for mapping fine tuning, for complex live performances as well as for further research on mapping strategies and instruments interfaces. In Chapter 7 we detail the methodology, experimental setup and results of the user evaluation. Finally in Chapter 8 we conclude by summarizing the contributions of this work, discussing their potential impact, and proposing future work directions. In Figure 1.1 we illustrate the overall thesis outline showing the interrelation between the different chapters.

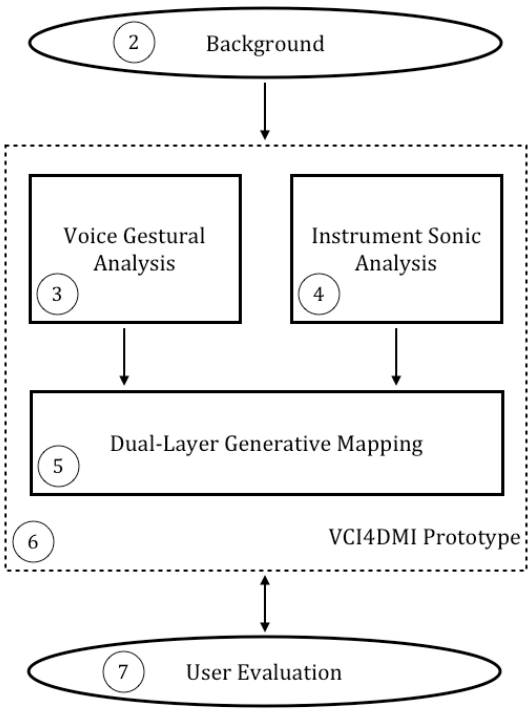

Figure 1.1: Illustration of the thesis structure by chapter content.

Chapter 2

Background

This chapter introduces the scientific background and the main research area in which this thesis is developed. At first we focus on the control strategies for DMI, presenting a history, characteristics, trends, and limitations of instrument interfaces, with particular attention to live performances. This is followed by a survey of previous work in which we analyze the different strategies for vocal control in the context of musical instrument interface. The review includes certain systems that are not specifically designed for musical task or to accept vocal input, but their extension can clearly provide interaction between voice and DMI. Finally, from the critical analysis of the related works, and in the direction of the thesis aim, we conclude the chapter establishing our research strategy and drawing a set of design principles and requirements for our system. Two other background sections on human voice with a perspective on machines interaction and on sound synthesis and processing characteristics of DMIs will be presented in details in Chapter 3 and Chapter 4 respectively.

2.1 Musical interfaces and controllers

Acoustic instruments have slowly evolved over millennia to the canonical forms we know today. In contrast electric, electronic and digital musical instruments have been around for just over a hundred years, in which, supported by the rapid advances in related technologies, they continue progressing at a growing pace. Instrument design has hence turned from craftsmanship to an exact science, encompassing sound generation aesthetic as well as control aspects. For modern instruments this involves a variety of disciplines such as math, physics, engineering, computer science, psychology and music composition (Bernardini et al., 2007). The abstraction of musical interfaces started to exist *per se* only with the appearance of electronic instruments. In DMIs sound production is totally supported by electric energy and controlled by electric signals. The sound is generated as a sequence of numbers first, converted into an electric wave that is amplified and finally transduced into a mechanical longitudinal wave by loudspeakers, from which we perceive the sound.

There is no direct transfer of mechanical energy from the player to drive and excite a resonant body, as there is in acoustic instruments. The player interacts with an interface that generates controls signals only. The introduction of DMIs voided or subverted concepts valid across centuries for traditional instruments and the following key characteristics were central in their development and evolution:

- The sound generation mechanism, the sound reproduction unit, and musical interface are independent components of a DMI. These parts can be physically decoupled and designed individually. For traditional instruments these are inseparable within the same physical body.
- A DMI can generate any sound, exploiting a variety of sound synthesis techniques (J. O. Smith, 1991). From the accurate emulation of natural sound and acoustic musical instruments, to the generation of novel sounds not existing in nature. Since any waveform can be synthetized there are no theoretical bounds to the sonic potential of a DMI. For a given acoustic instrument, the timbre is fixed and natural.

2.1.1 Brief history

In the early days of electronic instruments, musical controllers and sound synthesis components were still designed and packaged together. The interfaces recalled acoustic instruments with the piano-like keyboard dominating the scene. The *Theremin*, in Figure 2.1, introduced by Lev Sergeyevich Termen in 1920, presented the first touch-less and gesture-free interface. It was a ground breaking introduction but remained a singularity for decades. At the same time, keyboards started to be equipped with pedals, switches, and continuous controllers for the real-time timbre manipulation, that constitutes a novel performance paradigm possible only with electronic instruments. A great boost to design and development of musical controllers was given by the standardization of a communication protocol between synthesis engine and interface. This had already happened three times in the short history of DMIs, and each time it provided greater control potential in line with the current technologies. In the 1960's the manufacturers of voltage-controlled analog synthesizers adopted the logarithmic 1-volt-per-octave pitch control as the standard in the industry, which was introduced by Robert Moog in his synthesizers from 1964 (Pinch and Trocco, 2002). The Musical Instrument Digital Interface (MIDI) protocol specifications were published in 1983 by the homonymous consortium, providing a digital serial communication method still widely used today. Finally in 1997 Matt Wright and Adrian Freed presented the Open Sound Control (OSC) protocol (Wright

and Freed, 1997), that overcomes several MIDI shortcomings such as limited numeric resolution and inflexible symbolic addressing. Moreover OSC provides native support for network-based transport mechanisms, allowing the use of existing infrastructure such as local networks or the Internet to exchange control messages between instrument modules. These introductions drastically eased the decoupling of the controller from the synthesizer, and thus promoted the modularity of DMI design. Sound modules and stand-alone generic interfaces started to appear, devices from different manufacturers could be interfaced, while users started to personalize their instruments control strategy and interconnections. Moreover multiple instruments could be controlled from a single interface, making stage setups more portable. However until the late 1980's manufacturers still integrated synthesis and control in the same physical device, while some time later standalone modules started to be more common. Since the piano-like keyboard was often integrated in the DMI, this remained the most common musical controller. It provided velocity and pressure sensitive keys and, more recently, it integrates generic faders, button, knobs, and pads. Its popularity affected the way synthesis engines bundle and expose expressivity parameters. At that time musical controllers not inspired by acoustic instruments such as the keyboard or wind controllers (Wiffen, 1988), were limited to prototypes or market niches.

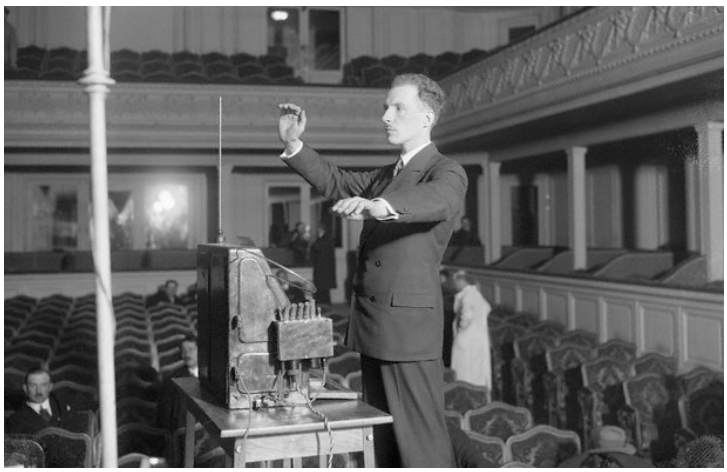

Figure 2.1: Lev Termen demonstrating the Theremin in December 1927.

Concurrently to the introduction and improvements of the musical communication standards, computer based sound generation became mature and real-time capable, due to the exponential increases in clock speed and memory of general-purpose personal computers. Specific languages or software to design audio synthesis and processing started to appear, drastically lowering the coding burden for users,

such as musicians, who may have little or no exposure to programming languages. In 1957 Max Mathews wrote MUSIC, the first program for generating digital audio waveforms by direct synthesis in digital computer (Mathews and Guttman, 1959). It gave rise to MUSIC-N series that in 1985 Barry Vercoe reimplemented as *Csound*. In 1986 Miller Puckette developed at IRCAM a non-graphical program to control the 4X synthesizer (Favreau et al., 1986) (Puckette, 1986). In 1988 he developed the graphical version called *The Patcher* (Puckette, 1988), later called *Max* after Max Mathew, and commercialized by Cycling'74. It evolved into Max/MSP, a modular software package, including Digital Signal Processing (DSP) functionalities, for dataflow programming by graphical interconnection of routines that exist in form of shared libraries. In 1996 Miller Puckette presented *Pure Data* (Puckette, 1996), a software system similar to Max/MSP in scope and design, but released as open-source. Another "strongly-timed" audio programming language, used for synthesis, performance, composition, and very popular among live coding artists was introduced in 2003 by Ge Wang and Perry Cook under the name of *ChucK*. The Virtual Studio Technology (VST) and the related and Software Development Kit (SDK), introduced by Steinberg in 1996, drastically boosted the development of third party commercial or free software synthesizer and audio effects. The VST is a standard cross platform software interface for the integration of virtual DMI as "plug-in" into audio editors, hard disk recording systems and Digital Audio Workstations (DAW). All these introductions turned computers into musical instruments (Mathews, 1963) that could finally be used to perform live, and not only to compose by offline coding. However they were generally still lacking in sophisticated musical interfaces. This fact stimulated the development of controllers that make synthetic sound production less disjunct from the body, providing visual representation and physical manipulation of real-time sound synthesis and processing. Moreover the figure of the performer started to arise and garner stature equal to the composer (Keislar, 2009). Today the musical interface remains often the only dedicated hardware component in computer-based instruments, while the large availability of powerful personal computers embodied the other components in software form, and they became part of any studio or performance setup, promoting formations such as the *Princeton Laptop Orchestra* (Trueman et al., 2006).

The development of a variety of sensor technology and recognition algorithms enabled the detection and tracking of physical expression of a performer, enabling the implementation of advanced and idiosyncratic musical controllers. Moreover open microcontroller hardware platforms facilitated the interconnection of transducers to computers (Wilson et al., 2003), permitting the fast prototyping of the hardware side

of musical interfaces as well. From *The Hands* (Waisvisz, 1985), the first experimental musical gestural interface based on the conversion of analog sensor data into MIDI control signals, to the more recent *Reactable* (Jordà et al., 2005), the first complete tabletop tangible user interface for musical application, hundreds of musical controllers have been presented due to the opening of vast design possibilities (Miranda and Wanderley, 2006).

Current trends include the implementation of musical interfaces on portable devices such as smartphones and tablets, which offer a complete platform including a large number of sensors, powerful multicore processors, advanced operating systems, and seamless communications. These are relatively low cost and widespread to further increase the accessibility to musical controllers, as demonstrated by, for example, the *Stanford Mobile Phone Orchestra* (Wang, Essl, and Penttinen, 2008).

Today, challenges in interface design are presented by the complexity and high dimensionality of data coming from the sensors. These devices are often equipped with a large network of heterogeneous transducers, aiming for enhancements in control efficiency and engagement (Tanaka, 2000). The data coming from the sensors cannot be easily and directly linked to synthesis parameters. A stage of processing to extract the musical intentions of the performer and the strategies to relate these streams of control data to instrument input has become an essential factor of any interface. Personalization and re-configurability features of DMI interfaces using ML techniques are gaining popularity, following the pioneering work of Wessel (1991) (Lee and Wessel, 1992) with Artificial Neural Networks (ANN). This has turned out in a design oriented towards a greater adaptability to the player (Paradiso and O'Modhrain, 2003). Recent musical controllers are no longer hardware-only devices, but they integrate a crucial algorithmic component, usually implemented in software, that users are starting to recognize as having primary value.

2.1.2 Main characteristics

The design of musical interfaces, despite their diversified nature (Paradiso, 2002), is today considered a specialized branch of HCI (Orio, Schnell, and Wanderley, 2001) where the simultaneous and continuous control of multiple parameters, the instantaneous response, and the necessity of user practice are key aspects (A. Hunt and Kirk, 2000). Across Rasmussen's (1986) human information processing models, the skill-based behavior is the most compliant with the musical interfaces interaction (Cariou, 1992), consisting of the continuous response to a continuous signal in real-time (Malloch et al., 2006). The interface of a DMI has a central role because often it

is the only physically accessible part of the sound generation chain and it represents the link between the mechanical and the electrical domain. It is the key element that enables the realization of the performer's musical intention, expressed through gestures, into sound. Conceptualization and design of musical interfaces is challenging because it involves the transformation of sonic generation and control information across heterogeneous domains. The design complexity grows with the musical potential and control features of the synthesis and processing algorithms. The interaction with any musical controller can be reduced and generalized to the block diagram in Figure 2.2, which shows actors, grouping, domains, and flow that realizes the performer's musical idea.

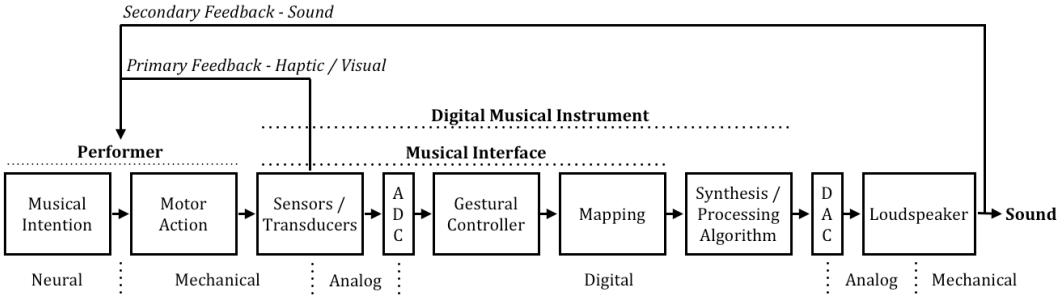

Figure 2.2: Block diagram representing the data flow, the key processes, the domains, and the grouping in the interaction between a performer and a DMI.

For the performer, the musical intention is expressed with a motor action called *gesture*, which can vary drastically depending on the physical characteristics of the interface itself (Cadoz and Wanderley, 2000). The ability to perform with a musical instrument is in general not acquired quickly, because the interface usually favors the expressive potential at the expenses of simplicity. However through extensive practicing musicians can achieve a level of "control intimacy" that minimizes the cognitive complexity necessary to produce the desired sound (F. R. Moore, 1988), to the point where players consider instruments as an extension of their bodies (Fels, 2000). High intimacy bestows smoothness to the performance and permits virtuosity.

The acquisition of the gestural data relies on sensors or transducers, which convert the mechanical energy of the gesture into an electric signal. Depending on the sensor characteristics and their application the *gestural acquisition* system can be *direct*, *indirect* or *physiological* (Wanderley and Depalle, 2004). In the first case each sensor signal is representative of a single basic feature of the gesture. In indirect acquisition the gesture is captured through an audio (Puckette and Lippe, 1994) or video signal, while electromyography (Tanaka and Knapp, 2002) or

electroencephalography are examples of physiological acquisition. Indirect and physiological acquisitions, besides allowing wearable or touch-less musical interfaces, present a more complex and high-dimensional gestural data stream. Therefore after digitizing the sensor signals, the *gestural controller* (Rovan et al., 1997) runs tracking, recognition or detection algorithms to eliminate noise, nonlinearities, redundancies, and correlation in order to isolate the desired musical control information. For simple cases with direct acquisition the gestural controller is often not included because operations such as segmentation, scaling, and limiting are sufficient to process the gestural data. Instead, the gestural controller eases the design of complex interfaces by separating the operations into two separate layers, and it produces a set of intermediate abstract control parameters as output.

The following stage, often considered the core of the musical interface, performs the *mapping* between control parameters and sound synthesis parameters (A. Hunt, Wanderley, and Kirk, 2000). Depending on the number of input and output signals, the mapping can be one-to-one, divergent (one-to-many) or convergent (many-to-one) (Wanderley, 2001). A variety of techniques have been used to establish the mapping, from direct linear relationship to more sophisticated geometrical methods that become challenging when continuity and differentiability are required and the parameters dimensionality is high (Van Nort, Wanderley, and Depalle, 2004). The existing mapping strategies can be divided in two classes: *explicit* and *generative* (Andy Hunt and Wanderley, 2003). In the first one the designer defines *a priori* the relationship between control and synthesis parameters, while in the second the relationship is the outcome of a training procedure using ML or other adaptive techniques. However, in both cases, the mapping algorithm itself should satisfy the real-time and low latency requirements of musical interfaces. The mapping chain, when paired with a synthesis algorithm, can also mutate a continuous gestural input to a discrete musical output and vice versa. Therefore the nature of the sensors of an interface does not restrict the kind of musical interaction that can be achieved (Kvifte and Jensenius, 2006). Depending on gestural data acquisition, gestural controller, and mapping Wanderley and Depalle (2004) classify the existing controllers into four categories: *instrument-like*, where the design of the input device tends to duplicate features of existing acoustic instruments; *instrument-inspired controllers*, same as before but conceived for a different use; *extended instruments*, which are existing instrument augmented with the addition of extra sensors; and *alternate controllers*, that do not resemble any traditional musical controller and neither restrict the performer motion in any way (Mulder, 2000).

Similarly to their acoustic counterparts, DMIs provide feedback that performers use for grounding the proprioception of the playing act and sonic interaction. For both instrument categories, the output sound constitutes the secondary feedback while the physical characteristic of the device provides a visual passive primary feedback such as the position of a fader or the status of a key. In acoustic instruments the active mechanical transfer of energy to the sound generating mechanism implicitly provides an active feedback, usually vibrotactile or haptic. This characteristic, often missing, can be emulated on DMI interfaces using mechanical transducers, relating it coherently to the gestural input. The active primary feedback can also be augmented or replaced with graphical or additional sonic feedback. However its design can be strongly limited by the choices on the gestural data acquisition system. An absolute lack of primary feedback, as in the *Theremin*, can make skill mastering and performing extremely burdensome.

2.1.3 Artistic impact

The novelties in the DMI characteristics have drastically affected the perspective on the instrument for designers, performers and listeners. In the early days the aim was to imitate acoustic instruments in their sound, interface and functionality. It was only when artists, composers and musicians started to go beyond the traditional music paradigms that new sounds and new ways of playing became "musically accepted", drastically stimulating the development of DMIs. Luigi Russolo, who introduced his *Intonarumori* in 1913, is considered the pioneer in using noise material in musical composition, influenced by sounds of the industrial revolution. He inspired the work of Pierre Schaeffer and *Musique Concrète* artists that in the 1940's started to use a wide sound palette in their pieces, which includes sounds from the real world. Cage in his compositions and writing foresees the future of electronic music and instruments, predicting

> "I believe that the use of noise to make music will continue and increase until we reach a music produced through the use of electrical instruments which will make available for musical purposes any and all sounds that can be heard. Photoelectric, film and mechanical mediums for the synthetic production of music will be explored." "Wherever we are, what we hear is mostly noise. When we ignore it, it disturbs us. When we listen to it, we find it fascinating." "The present methods of writing music […] will be

> inadequate for the composer, who will be faced with the entire field of sound."
>
> – John Cage (1937).

Therefore technological and socio-cultural changes had brought music to the point where every sound is admissible, theoretically realizable, and equally accessible (Wyse, 2003), and this represents the reason behind the continuous evolutionary process of sound related technologies. There are no limitations that confine the designers to particular timbral characteristics, sound generation mechanism, physical dimension, and performance style to control the instrument.

Through experience and senses humans can identify mechanical aspects of the event that generates sound. We search and guess material, shape, cause and location by auditory cues (Emmerson, 1998). In the same way we can predict the sound of a mechanical event by visual cue. This principle applies also to musical instruments. When a musician is about to play we expect to hear a distinct timbre, coming from a specific direction, generated by a clear relationship between gesture and sound. The advent of DMIs has radically changed the perspective of listeners, breaking all the certainties. DMIs have "acousmatically dislocated" sound from player action in space, in mechanical causality, and in time (Emmerson, 1994). Visual cues as well as experience may not help to predict timbre, gestural control, source location and spatial characteristics, because musical interfaces often differ from familiar physical forms of acoustic instruments. This lack of correlation induces the audience, usually passive but in some cases with an active role (Blaine and Fels, 2003), to build a relationship between aural and visual information during the performance itself. If mapping between gesture and sound changes, then the model of their relationship must be rebuilt. The DMI interface is fundamental in this process because it is often the only visible music-related object on stage. More than a drawback, musicians exploit this characteristic of DMI as a novel artistic potential by performing with unique alternate or extended controllers. The sonic palette, the performer interaction and the instrument itself became integral parts of the artwork. Anything can be used to control any aspect of any sound, since musical interfaces can be also mapped to control sound alteration algorithms, also called audio effects. Filters that modify audio signals properties had been used extensively, but with static parameters. Today the use and control of these algorithms in live performance is common and commensurate to sound synthesis (Wanderley and Depalle, 2001), and it provides further real-time manipulation of timbral and spatial aspects of the sound.

2.1.4 Limitations and trends

Learning to play with a new musical interface does not follow the same path of traditional acoustic instruments. It is not possible to acquire the complex representation of the relationship between motor action and sonic response of the instrument when the same controller can be used with different mappings and different synthesis algorithms. Moreover these evolve so quickly that musicians rarely have enough time to develop virtuosity (Paradiso and O'Modhrain, 2003). Making a NIME is usually easier than playing it well (Lyons and Fels, 2012), and the number of virtuosi or professional musicians using these as their main instrument is exceptionally small while many computer musicians still compose and perform using mouse and keyboard or, at the most, a generic fader box (Jordà, 2004a). The reasons behind this are generally the commercial unavailability or the high cost, plus issues related to efficiency and learnability. Some successful interface designs favor virtuosity and nuances, requiring extensive skills mastering training, others present a low entry fee, capturing the interest of beginners, but mostly both fail in promoting continuous exploration, discovery and creative use (Machover, 2002). Wessel and Wright (2001) argue that both characteristics should belong to a NIME, the ease of use in the early stage should not be at the expense of the potential to develop musical expressivity. They believe that musicians will be more prone to skill development and personalization, which ease generation of musically attractive experiences, only if the instrument presents potential for control intimacy. This is in accordance with design principle of tools that support creativity (Resnick et al., 2005), but in contrast with the general goal of human factors and ergonomics, generally inclined to facilitate the use of devices. Easy interfaces require less effort to perform and this often results in a loss of expressive power (Vertegaal and Ungvary, 1995). Rather than being just easy, a musical instrument should be highly compatible with the performer, so that the "naturalness" of the interface leads to its "transparency" (Norman, 1988). This depends on the consistency and adaptability of an interface to the user preferences (Shneiderman and Plaisant, 2010). Therefore there is an emerging trend in instrument design that is going towards a greater flexibility to accommodate the individual performing style and input modality (Paradiso and O'Modhrain, 2003). Nevertheless Cook warns that when the instrument learns directly from the player the training and performance modes must be well separated since learning to play an ever-changing instrument is rather difficult. Moreover interfaces filled with user options and programmable features provide an infinite

landscape for experimentation and creativity, but these rarely get used in real performances or artworks (Cook, 2009).

Decoupling interface and sound generation in DMIs represents a challenge and an opportunity. Today the design of controller and musical algorithm is often completely agnostic. The designer of a controller can hardly predict the final user's choices on synthesis mapping and vice versa. Both halves make use of the communication standards and expose a set of generic parameters, sometimes low-level and without direct perceptual meaning. This implies the user's central role in defining and implementing the mapping that usually follows explicit and fixed rules. When the devices present sophisticated characteristics this method fails to represent the complex and indeterministic relationship between a performer and computer based musical instruments. The mapping algorithms should also react dynamically to the performer input, changing arbitrary relationships between controls (Chadabe, 2002). This is the principal drawback that separation between controllers and sound can bring to DMI design (Jordà, 2004b), and even advanced mapping techniques show limits in coping with it. Currently this issue is addressed with the renaissance of mutual design of synthesis and interface (Cook, 2004), or the generative adaptation of the interface mapping to the sonic characteristics of the specific sound synthesis algorithm (Arfib et al., 2002).

Improving the DMI expressivity through additional interface control potential had been among the main goals of musical controller designers and researchers for decades. Augmented instrument-like or instrument-inspired interfaces are evident examples of this tendency. The most popular and widespread piano-like keyboard, for instance, has been recently proposed in many variations with capacitive multi touch keys (McPherson, 2012), integrated analog sequencer (Snyder and McPherson, 2012), pedals, knee levelers, bowing, breath controllers (Wierenga, 2012), and optical sensors capturing the key motion (McPherson, 2013). These works show a pattern in exploiting performer's spare bandwidth to provide extra continuous control that is lacking in many common interfaces. The urge for continuous controllers surged when algorithms, running on faster processors, started to support runtime modification of real-valued synthesis and processing parameters without generating glitches in the output sound. This gave rise to novel composing and playing paradigms, where morphing the sound timbre, texture or spatial features across a contiguous and infinite sonic space prevail over traditional melodic and harmonic elements (Rowe, 1995). This trend is particularly evident in electronic music that is often denigrated for its simplicity, but must be credited for its obsessive exploration and application of new musical technologies and for exposing these to the masses (Collins, 2009). Live

performances of electronic music, thanks to the power of modern general-purpose computers, often involves the use of a DAW that hosts and synthesizes the equivalent of a large ensemble in real-time. The hundreds of events and parameters contrast with the small number of players involved in the performance, often one only. Therefore the performer often raises the abstraction level of the interaction by programming sequencers or triggering pre-recorded material, alternating his role between conductor and single instrument player. In this way the performer may control richer but less flexible sonic objects, and the required musical interaction skills can be dramatically different. Moreover the real-time aspect of the control is also lowered because the performer can asynchronously schedule events in a pipe that is executed by a machine. However the automatic sequencing of predefined or computer generated musical events degrades the live nature of the performance itself. A limit is often the overlapping hand-based input modality of the interfaces, although foot controllers are a common and effective partial workaround. Therefore, current research directions to address these problems are looking at exploiting spare bandwidth of players, and at the runtime re-configurability of controllers.

In this section we have discussed some of the limitations, trends and challenges related in modern DMI interfaces, focusing the attention only on those that contribute to determining the aim of this thesis.

2.2 Related works

Several musical controllers driven by human voice have been proposed starting from the late 1960's. The *Voder*, named as an acronym of "voice operating demonstrator" invented by Homer Dudley at Bell Labs in 1928, patented in 1939, and later renamed *Vocoder*, started to be used from the 1960's to generate synthetic sound driven by voice. It was originally a complex machine consisting of manually operated oscillators, noise generators and a filter bank that skilled operators were using to produce recognizable speech. It found musical application when the sound of a synthesizer was used as the input of the filter bank (Tompkins, 2011). Morphing of the voice signal for musical application has today evolved to more extensive systems allowing skilled beatboxers (Stowell, 2008) to layer and morph their voice extensively, sometimes even obtaining virtual polyphony from a single voice (Foreman, 2013). These systems perform only a modification of the voice applying processing algorithms within the audio signal domain, and thus do not provide any explicit control opportunity. In this section we focus only on methods that capture the

performer voice signal, usually with a microphone, break down frames of the order of milliseconds into low-level features, and generate arbitrary control signals or sound interaction strategies, similarly to the performance-drive approach (Rowe, 1993). We present a survey on recent works with the intent of identifying the limitations and drawbacks in state-of-the-art. In reviewing the literature we discuss various aspects such as the gestural acquisition, the gestural controller, system setup, eventual training procedure, and musical control. The survey is divided in two parts that detail related works presenting explicit and generative mapping strategies respectively. We also include some systems not specifically designed for generating musical output or for accepting vocal input such as generic HCI vocal interfaces and ML mapping tools, but we discuss their extension for providing interaction between voice and DMIs.

2.2.1 Explicit mapping

In this category of mapping strategies the designer defines a clear and fixed relationship between performer actions represented by the gestural data and the instrumental control parameters. Generic pitch-to-MIDI converters have been used for decades to map pitch from voice to instrument. The Roland SPV-355 was introduced in the late 1970's presenting basic functionalities. The more recent VP-70 and software plugins such as *Widisoft WIDI* and the *DigitalEar* offer more advanced capabilities such as of polyphonic pitch tracking, pitch bending and attack detection. The energy is tracked and mapped to MIDI velocity. These devices present an underlying mapping strategy that establishes an identity between voice and instrument loudness and pitch. Since the human voice is intrinsically monophonic, there are obvious limitations in exploiting instrument polyphony.

The energy and pitch of human voice can be also captured non-acoustically through electroglottography, which is a noninvasive measure of the laryngeal behavior based on the variations of the electrical impedance across the throat (Lecluse, Brocaar, and Verschurre, 1975). This physiological gestural acquisition is adopted as an alternative to the sound input in the *SynchroVoice MIDIVox* that presents a mapping strategy identical to pitch-to- MIDI converters. Similarly, the *Larynxophone* (Loscos, Cano, and Bonada, 2005) concatenative cross-synthesis engine, is directly driven by pitch and energy computed from the voice. The pitch at onset time is used to query the trumpet samples database, while other spectral features such as the excitation gain, slope, and depth are mapped respectively to

velocity, modulation, and aftertouch. After the onset these values are used to continuously modulate pitch and aftertouch.

The *Singing Tree* (Oliver, Yu, and Metois, 1997), part of the MIT *Brain Opera* installation, was the first work to present a more extended idea of mapping, in which voice nuances are used to interact with an ensemble of instruments. From the singing voice, 10 different dynamic parameters are extracted and mapped to an ensemble of MIDI instruments using different sound sources to resynthesize the character of the singing voice. The detected pitch is used to control the progress of a musical sequence towards its goal. The other features such as loudness, formants, cepstra, and their deviations, are mapped to multiple parameters using dynamic set assignment probability and random number generation.

In the *Wahwactor* (Loscos and Aussenac, 2005), the central frequency of the resonant filter of the wah-wah effect is controlled by the guitarist's voice, which varies across the phonemes /u/ and /a/. The most interesting aspect of this work is the preliminary study to identify which vocal features are the most robust and reliable for the musical control task. The authors conclude that the low- band spectral weighted area yields the smoothest and most stable response, improving the noisy performances given by all other considered features.

Janer (2005a) presents two different plucked bass synthesis techniques controlled by a singing voice. The author argues that the selection of voice features and mapping depends on the instrument type as well as on the synthesis technique, and therefore the generic mapping model encompasses two layers. The first is related to the instrument interface, and the second to the controllable synthesis parameters. For the physical model, the string excitation is triggered by the voice energy envelope onset detection, and the pitch defines the length of the string algorithm delay line, initially filled with the attack samples of sung note. In the spectral morphing synthesis algorithm the plucked bass sound is generated by the concatenation of spectral frames from a database, storing information about spectrum, harmonic peaks, pitch, dynamic and attack type. Finally pitch and harmonic peaks computed from the voice are used to find the closest element in the database, while the "attack unvoiceness"is used to select between fingered and sharp slap attacks. Janer extends and further generalizes this work in the *Singing- Driven Interfaces for Sound Synthesizers* (Janer, 2008) proposing a system based on the imitation of the sound of the instrument by the user's voice, performing a temporal segmentation of the voice based on syllables, and mapping voice pitch and loudness to the corresponding instrument features. The system generates a real-time score and one continuous value parameter derived from the first two formants, which can be used for timbre modulation. The author applied

this method to control acoustic instruments emulators, arguing that for this aim a single parameter would be sufficient because the vocal apparatus has a wider range of timbre variation than acoustic instruments, which present nearly fixed timbres. On the contrary for non-natural or acoustic sounds synthesis this approach shows limitations since the timbre variation range is generally wider, and it requires multiple parameters for full control.

Image-based gestural data acquisition is used in the *Mouthesizer* (Lyons, Haehnel, and Tetsutani, 2003) and in *Tongue 'n' Groove* (Vogt et al., 2002). The first uses a head-worn camera to track the mouth height, mouth width and mouth aspect ratio. These are mapped respectively to the wah-wah filter frequency, amplifier distortion, and formant filters morphing in a guitar effect chain application. The second acquires the image of the tongue from a two-dimensional medical ultrasound scanner. The tongue motion is tracked by contour or optical flow extraction, providing a set of coefficients mapped directly to the synthesis algorithm further processed to control pitch, note triggering and filtering of basic synthesizers. These two acquisition systems ignore the source component of the vocal apparatus, focusing the attention on a partial estimation of the filter part, which depends, as we will see later, on both the shape of the mouth opening and position of the tongue. These approaches present the advantage of not being affected by external noise or breathing pauses that interfere and interrupt instrument control in other systems.

In *Software Tool for Vocal Control of Musical Elements* (Deacon, 2014) the voice drives different aspects of the system depending on the temporal duration of the command. Repetitions of short commands toggle the status of different sections of the system. Medium-length commands are analyzed by a simple word recognition system then mapped to selection of the active sample bank in a sampler or synthesizer. In longer commands, the pitch and the duration are recognized and used to trigger different samples, while for the synthesizer the pitch of the voice is directly mapped to the instrument note. Despite the fact that the mapping is explicitly tuned to the designer use preferences and limits of the implementation, this work shows two interesting aspects. The author proposes the use of the voice as an additional and multifunctional control layer for performers already fully engaged with other instruments. Moreover the work demonstrates that vocal control can be extended integrating words or speech recognition, to control non real-time and non-musical parameters of a DMI such as switch across presets or control target.

2.2.1.1 Generic HCI interfaces

The use of human voice to interact with machines is an emerging trend in several HCI fields. This is often limited to speech recognition techniques, which falls short of central DMI interface requirements such as low latency and multi parametric continuous control. However some HCI vocal interfaces establish the interaction at non-verbal level (Igarashi and Hughes, 2001), enabling real-valued control and fast system response, demonstrating musical control task compliancy. In these systems the different vowel sounds are detected to control an emulation of a drawing tablet (Harada, Wobbrock, and Landay, 2007), a joystick (Bilmes et al., 2005), or to command a robot arm (House, Malkin, and Bilmes, 2009). The eight vowel classifier outputs are mapped to the movement direction, while energy, pitch and vowel quality provide three additional independent levels of freedom, mapped onto other continuous value parameters such as the speed of movement. Despite the basic mapping that emulates common machine controllers, these works show a value for the motor impaired individuals and for hands-busy environments. They also report encouraging user-study results showing acceptable learning rate and proficiency (Harada et al., 2009).

2.2.2 Generative mapping

Mapping strategies classified as generative consist of the adoption of supervised or unsupervised ML algorithms to establish the relationship between performer's actions and instrument controlled parameters. The adoption of a learning algorithm eases the mapping definition, and provides adaptive and reconfigurable characteristics to the instrument interface.

The *Billaboop* (Hazan, 2005) is a real-time system that translates beatboxing onomatopoeic vocal sounds into synthetic or sampled drum sounds. This work presents a hybrid mapping approach. Onset detection explicitly triggers sounds, while other descriptors are mapped to effects or synthesis parameters. A set of spectral vocal features including the centroid, the high frequency content, and energy of three bands computed over the onset frame are used to query a decision tree that returns the label of the sound to be triggered. The decision tree classifier is trained using the early descriptors of a large set of vocal hits from a specific performer, manually labeled and grouped into three macro categories. The wide range of sound used in beatboxing is not formally defined and may vary drastically across performers. In this context, a supervised classifier offers an effective solution, providing the capability to

adapt to vocal style and voice characteristics of the performer. A similar method applied on larger temporal grain with rhythmic pattern detection is used to "query-by-beatboxing" a database containing loops and songs (Kapur, Benning, and Tzanetakis, 2004), implicitly providing a less real-time interacting environment.

Orio (1997) uses the inner shape of the oral cavity as the gesture and this approach does not require the user to utter vocal sounds. Here the oral cavity represents a resonator for an external sound source, similarly to the Jew's harp case. Two physical waveguides are used to feed the oral cavity with white noise and capture the filtered signal. Deconvolution of input and output signals provides the parameters of the linear filter that approximates the performer intention through variation of the inner shape of the oral cavity. The system undergoes a training phase in which the performer assumes all the postures used for control, and the relative gestural data made of 12 Linear Predictive Coding (LPC) coefficients is used to find the Principal Component Analysis (PCA) projection matrix. In the real-time use of the interface, two or three principal components of a new incoming LPC vector are used to control continuous valued parameters of an arbitrary instrument.

In *Auracle* (Ramakrishnan, Freeman, and Varnik, 2004), an interactive network based collaborative musical instrument, root-mean square, zero crossing rate, fundamental frequency, frequencies and bandwidths of the first two formants are fed to a higher level analytical stage in which frames are segmented and classified into gestures. The low-level features envelopes are classified using a PCA for dimensionality reduction followed by an ANN. The control data obtained is transmitted by multiple users over the network, and merged by a single server to control a single sound synthesis system. The mapping to the specific synthesis algorithm is still implemented explicitly by the user on the server side.

Similar approaches are taken in the *scrambled?HaCkZ!* (König, 2006) and in (Janer and De Boer, 2008), where the voice data is used to retrieve and then sequence sounds slices from a previously analyzed database. In the first case a classifier selects the most similar sound frames running a classifier of a large set of features computed on the voice live input. The sound chunks in the database are extrapolated offline from music videos, in sizes of fractions notes, and played back with the original associated clip. In the second case, vocal timbre features are projected onto their principal components and then mapped statistically onto the timbre principal components of the database of an audio mosaicing system. The mapping aims to optimize the overlapping of the voice timbre space to the sound database features by inserting an intermediate and case dependent transformation stage.

Stowell (2010) presents a mapping strategy that overlaps two timbre spaces, one derived from the analysis of a voice gesture and another related to the sonic variation of a synthesis engine subjected to different input parameters. The author analyzes the voice and the sound of the instrument computing identical high dimensional features, which are projected onto their principal components, and then two different methods to implement the "remapping" are proposed. In the first one, a piecewise warping transformation is adopted to distort and rearrange both spaces spanned by the principal components in an equal manner. The coordinates of new incoming voice vectors are then used to pick a point in the sound synthesis space. The second one is based on a prior reorganization of the spaces made by an auto-associative regression tree (Stowell and Plumbley, 2010), which recursively divides the two spaces and finds cross associations for real-time mapping. Finally the synthesizer is driven with the parameters associated with the target timbre. In this kind of approach, the relationship between synthesis parameters and sound may not be one-to-one, and it may cause discontinuities in the output of the instrument.

2.2.2.1 Machine learning mapping tools

The implementation of a generative mapping strategy is usually complex and requires significant programming skills, usually beyond the competence of average DMI users. Several research works have provided software tools to musicians for exploring and performing with personalized mappings generated by algorithms rather than manually defined. Using ML techniques is often a necessity rather than a choice, because the explicit definition of mappings that involves high dimensional gestural data and a high number of synthesis parameters is impracticable rather than difficult. To date a variety of ML techniques for classification, clustering and regression has been applied to DMI interfaces, in both the gestural controller or the mapping component (Caramiaux and Tanaka, 2013). In this section we review those generative methods supporting mapping of continuous gesture-to-instrument parameters and without constraints on the gestural data input, implicitly supporting the mapping of voice-to-instrument given a proper stage of gestural data acquisition or a gestural controller.

The IRCAM *Gesture Follower* (Bevilacqua et al., 2010, 2011) is a generic system allowing real-time gesture following and recognition. Live audio input comes across the various supported input modalities providing Mel Frequency Cepstrum Coefficients (MFCC) analysis, commonly used in many voice applications. The Gesture Follower is based on a left-to-right Hidden Markov Model (HMM), which

continuously outputs a generic real-valued parameter indicating the temporal correspondence between the observed and the reference gesture. This can be used for a basic DMI interaction or to dynamically change the map between the performer's gesture and the instrument of another system. However the adoption of the HMM, independent from the dimensionality of the gestural data, results in a mono-dimensional output and introduces causality constraints between gesture and the system output. This work has been extended using Segmental HMMs, a generative method for shape modeling, which allows continuous signals to be segmented and indexed at the same time (Caramiaux, Wanderley, and Bevilacqua, 2012), and providing hierarchical multilevel time structure (Francoise, Caramiaux, and Bevilacqua, 2012). These improve the mapping potential establishing different relationships at long and short temporal frames, as well as providing additional output from musical control consisting of the direction, scale, angle, and speed of the current gesture against those stored in the HMM.

The *Wekinator* (Fiebrink, 2011) offers musicians and performers a diversified range of generative methods for mapping between gestural data and instrument. It is a meta-instrument to train and modify standard ML algorithms for interactive mapping purposes. Basic audio feature extraction is provided, while more advanced features computed externally can also be fed to the input via OSC. Except for the ANN, the available ML algorithms are supervised and provide only discrete output. With the discrete classifiers it is possible to map vocal features to discrete values of instruments, trigger events, or switch between parameters presets. The application of the ANN presents interesting possibilities for the control of multiple time-continuous and real-valued instrument parameters, but it relies on the quantity and consistency of the training data, in input-output pair form, provided by the user. The required quantity of training data can be considerable if the underlying model is highly nonlinear. To ease this task, the *Wekinator* provides a novel "play along" modality to generate the training data live (Fiebrink, Cook, and Trueman, 2009). A potential limitation of this approach is the lack of feedback for the user about common NN shortcomings such as unlearnable models, over and under fitting, requiring the system use to verify whether the ANN learnt the user control intent. Manual tuning is not available, so changes in the system response require modification of the training data and system re-training.

The *SARC EyesWeb Catalog* (Gillian, Knapp, and O'Modhrain, 2011) packages the ML algorithms and functionalities comparable to the *Wekinator* but provides a Graphical User Interface (GUI) for dataflow programming that supports the combination of different ML algorithm, feature extraction, and training data

manipulation. It supports the prediction of the real-time gesture recognition and tracking performances by showing results of testing and validating of the system after training.

A package of Max/MSP externals implementing a collection of ML algorithms supporting online training is described and provided by B. D. Smith and Garnett (2012). This includes spatial encoding techniques and ANN based methods such as self-organizing maps, a multi-layer perceptron, and the Adaptive Resonance Theory (Grossberg, 1987), a self-supervised technique in which the ANN is trained unsupervisedly by an automatic supervisor (B. D. Smith and Garnett, 2011).

2.2.3 Drawbacks and open challenges

The number of musicians and performers making regular use of any voice-control techniques is still very limited, in stark contrast to the wide use of human voice in the musical context. This suggests that the majority of the work described here still presents technical or conceptual design issues, even if some, in particular Fiebrink, or Janer and Stowell, introduced outstanding contributions in this context. The works based on explicit mapping propose closed and stand-alone interfaces with minimal or no tuning options, which conflict with the growing needs of performers for personalization, configurability, versatility and integration with other control devices. Moreover they generally fail to cope with different vocal tract characteristics of different users. Some of these works present naïve mappings that simply relate characteristics of the vocal apparatus such as pitch and intensity, to the homologous of a synthesis algorithm. Singing to trigger notes of an instrument, in addition to presenting high latency and error proneness compared to a basic keyboard, are furthermore restricted to a smaller monophonic note range. Moreover most of these works are still limited in instrumental expressivity in terms of sonic manipulation.

The works based on a generative mapping definitely offer greater control potential and flexibility, but the user is still required to provide training data, and sometimes to choose some ML preferences. For supervised learning methods such as ANNs, the preparation of the training data can be even more tedious since they require large sets of input-output pairs. These must be consistent and representative of an underlying learnable model. Moreover, it is often unclear to the user whether the mapping idea expressed in training data was successfully captured from the ML algorithm, and real-time manual fine tuning options are difficult or impossible. All these issues represent an entry barrier in time and knowledge too high for most users, calling for a simplification of the setup process, at least for a basic first usage.

With the exception of the works of Loscos and Stowell, the selection of the signal processing analytical techniques applied to the voice for the computation of low-level feature vectors, are not justified or supported by prior studies on real data. We believe that this aspect must be further explored because a larger but flexible low-level feature computation, especially if adapted to user specific data, may drastically improve some aspects of the interface. At the same time, the curse of dimensionality (Bellman, 1972), which undermines the learning effectiveness of many ML algorithms, must be addressed with a dimensional reduction stage in the gestural controller, that dispatches more compact and representative data at the input of the mapping block.

Those works that take the characteristic of the output sound into account in the mapping generation can certainly provide a sort of co-design or adaptation between controller and DMI, which is one of the issues discussed in Section 2.1.4. The method to analyze the sound of the instrument must be able to capture any sound nuances, and thus different from the one used for the voice since the two sources are respectively indeterminate and fixed. The analysis of the DMI response represents the most promising strategy to provide adaptation to specific synthesis or processing characteristics for complex mapping scenarios. This approach requires further development, integration, and diversification to be universally compliant.

2.3 Strategy

From the analysis of the limitations and open challenges discussed in Sections 2.1.4 and 2.2.3, and in the light of the motivation and aim described in Section 1.1 and 1.2, we formulate a set of design principles and requirements for the VCI4DMI system that determines the research topics of this thesis. We favor a modular strategy for approaching self-contained entities that can be easily integrated to contribute towards the main goal of this dissertation. In particular we separate the investigation of voice related matters from those pertaining to DMIs. The individual contributions are numerically evaluated and, when possible, their design and outcome are compared with the related work presented in this chapter.

2.3.1 Design principles and requirements

Here we present a list of design principles and requirements for the VCI4DMI system that will be further elaborated as research direction and specifications in the respective following chapters:

- **Integrability** with other interfaces by using voice as the gesture, which is a recurrent performer spare bandwidth.
- **Indirect acquisition** of the gestural data from the vocal sound coming from normal microphones.
- **Error-safe** control, favoring the mapping onto less "mission-critical" and error-prone musical parameters.
- **Multi-parametric** control, permitting the simultaneous control of an arbitrary number of real-valued instrument parameters, and ensuring interface robustness and consistency.
- **Low cognitive complexity** introducing a logical and natural interaction strategy and a **reduced control space dimensionality** to a maximum of two or three intermediate dimensions, which minimize the loss in the original instrument sonic potential.
- **Automatic adaptability** towards voice characteristics, control style, and instrument sonic response, which assume no prior knowledge on these and implies the use of a **generative mapping** strategy.
- **Modular** design that includes voice-dependent adaptation in the gestural controller and DMI-dependent adaptation in the mapping block, enabling independent **reusability** and **reconfiguration** of both parts.
- **Perceptual sonic analysis** to provide an instrument response closer to the human perception of the sound variation.
- **Minimal user role in interface setup** exploiting automatic and unsupervised methods that can work with limited training data and granting mapping learnability in any case.
- **Active feedback** to support learning and use of the generated mapping by numeric and graphic visualizations.
- **Runtime tuning** flexibility that supports real-time modification and personalization of the interface response.
- **Low entry barrier and high ceiling**, providing basic and advanced use modalities to supporting the continuous creativity and skills development.

Chapter 3

Vocal Gestural Controller

In this chapter we describe a novel method that learns offline how to compute a set of musical control signals from the human voice, which is flexible and adaptive to the individual performer voice characteristics. This implements the Gestural Controller (GC) component of the musical interface we introduce in this dissertation, which includes also the gestural data acquisition and preprocessing stages. The chapter starts with an overview on the human vocal apparatus, vocal techniques, and fundamentals of voice processing in HCI. We determine entry point, challenges and objectives for this section of the thesis by addressing global issues of the state-of-the-art in relation to design principles and requirements. This is followed by a detailed description and motivation of the two learning algorithms developed to determine optimal voice signal processing settings for the interface, and to generate ad-hoc GCs using Self-Organizing Map (SOM) principles. We illustrate the resulting runtime procedure to obtain the GC intermediate abstract control parameters in real-time from the voice input, applying the training stage outcomes. We conclude the chapter by evaluating the proposed method using real voice data for measurements and comparisons with other approaches.

3.1 The human voice

The background presented in the previous chapter was limited to the core science and issues of musical interfaces. In this section we introduce the human vocal apparatus and we discuss the use of voice in HCI in order to pave the way to the work and contributions described later in the chapter.

3.1.1 Voice production apparatus

The voice production apparatus of humans is a complex and unique mechanical sound generation system. Its characteristics support the production of an astonishing spectrum of sounds and expressive variations which derive from millennia of verbal communication evolution. Languages, contemplating the differences across these, use

only partial ranges of the total sonic potential of the human voice, which is used in full only by a few specifically trained vocal performers. In voice generation the diaphragm provides the energy by pushing air out from the lungs. The airflow generates a sound in the larynx by the vibration of the vocal folds. Finally the sound is filtered by the pharynx, mouth and nasal cavities. A large part of the body and a high number of muscles are involved in generating, controlling and articulating vocal sounds. This process is a sequence of respiration, phonation and resonance, color-coded respectively in pink, blue, and yellow in the anatomy of Figure 3.1.

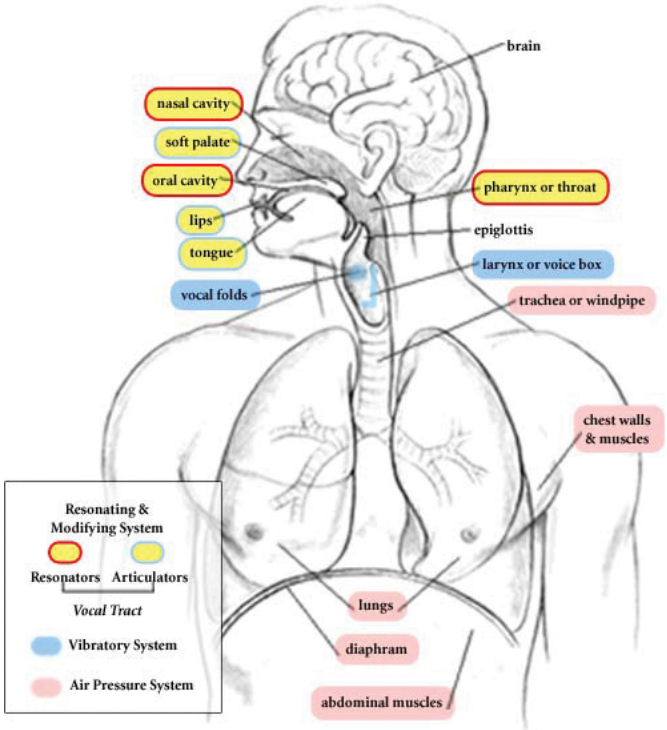

Figure 3.1: Anatomy of the human voice production apparatus with labels color-coded by function.

The voice production is historically often modeled as a source-filter system and simplified by assuming that the two components are independent (Fant, 1960). In the source, the lungs airflow serves as the supplier of continuous energy for the vocal system. With the vocal folds closed, when the sub-glottal air pressure exceeds the supra-glottal pressure by a certain threshold a temporary aperture of the vocal folds happens. These are quickly closed back by the fold tension and the suction created, via the Bernoulli effect, with the rapid airflow passing through (Van Den Berg, 1958). The continuous repetition of this process, called modal phonation, puts the vocal folds into vibration and air pulses, with interval T_{pulse}, flow into the filter component. This provides the periodic sound source for all voiced sounds such as the

vowels, with fundamental frequency $f_0 = 1/T_{pulse}$ and harmonics at its multiples. The f_0 range and the level of harmonics varies across individuals, and is affected by factors such as gender, age, and physiological condition. For speech the range typically span between 100Hz and 400Hz, while in singing it extends from 35Hz to 1500Hz, but it is usually limited to a maximum of two octaves in a single individual. The harmonic level decrease is averagely 12 dB/octave on average. The flexibility of the muscles around the vocal folds allows other modes of phonation such as whispering, ventricular, breathy, and creaky (Laver, 1980). In whispering mode the source generates aperiodic sounds with a broad spectrum. In this case the vocal folds do not vibrate as they are kept open but close together, producing an irregular and turbulent airflow in the larynx, determining the noisy perturbation that is at the base of most of the unvoiced sounds. The turbulence can also be produced directly in the mouth and in this case the vocal folds are held completely open, alike in fricative sounds. In the other cases the sound source operates in a modality halfway between modal phonation and whispering.

The pharyngeal, oral, and nasal cavities work as resonant chambers so that the vocal tract operates as an overall resonant filter, shaping the spectrum of the sound source by exciting or suppressing specific bands. The resonant frequencies, also called formant frequencies, depend on shape and length of the vocal tract. The first depends on the positioning and shaping of the articulators such as lips, jaw opening, tongue, larynx and soft palate. The distance between glottis and lips determines the second. These are changed constantly while speaking or singing, resulting in a continuous variation of the resonant filter transfer function. The first five formants determine the personal character of the voice, which varies across individuals due to mechanical differences in the vocal tract. Figure 3.2 shows the functional model of the human voice production, where the vocal folds represents boundary between the sub- and supra-glottal tracts. In Figure 3.3 we illustrate the sound spectra at the source for modal phonation and whispering, the filter transfer function split into vocal tract gain and mouth radiation impedance, and finally the resulting output spectra. The amplitudes on the vertical axis are on a logarithmic scale.

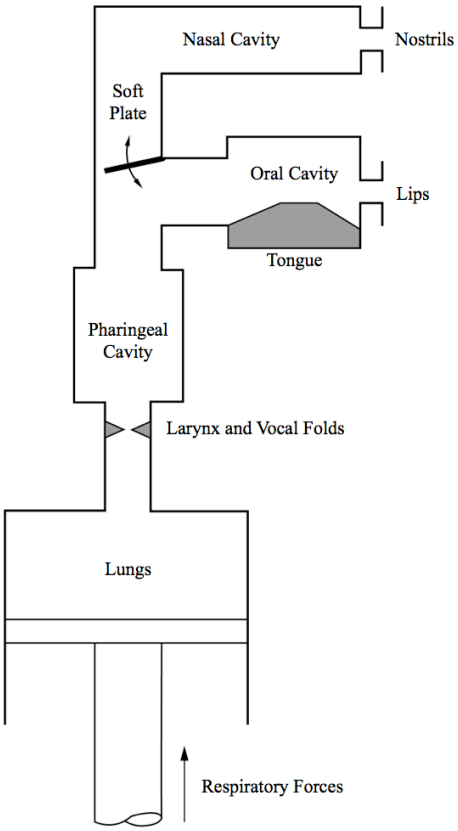

Figure 3.2: Model of the voice production apparatus, after Clark and Yallop (1995).

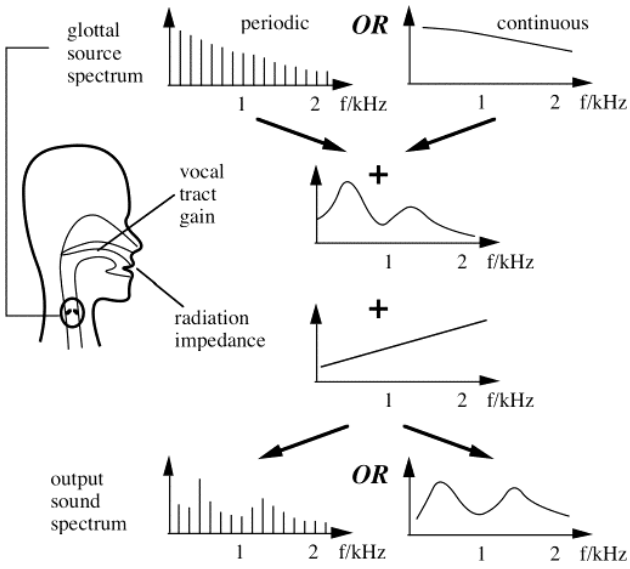

Figure 3.3: Source spectrum, filter transfer functions, output spectrum for voiced and unvoiced sounds in the source-filter model, after Epps, Smith, and Wolfe (1997).

3.1.2 Voice articulation, control and styles

The articulators, residing in the filter part of the model, can sensibly modify the sound timbre of the source. Their configuration changes dynamically in the majority of spoken sound, altering the properties of the vocal tract such as the distance of the speech organs and the stricture of the airflow. For vowels sound the configuration is static, the source is usually in modal phonation and the soft palate is usually closed forcing most of the air to pass only through the oral cavity, while if open we are generate nasalized vowel sound. The lips and opening tongue constriction determine the frequencies of the first two formants respectively, which determine generation and perception of vowel sounds (Johnson, 2008). The larynx can be lowered or raised to change the overall vocal tract length, which result in a lower or upper frequency shift of the formants. The third formant frequency is inversely proportional to the size of the cavity behind the incisors. Although there are five discrete vowels in the Latin derived alphabet, these correspond to at least to eight different phonemes, the basic units of the phonology in every language. Moreover as the lips opening and tongue position are continuous quantities, vowel sounds can vary continuously in a space, at least bi-dimensional. The vowel space, in Figure 3.4, provides an imaginary representation of a cross-section of a human head looking left, with the tongue position on the horizontal axis (second formant frequency) and the lips opening on the vertical one (first formant frequency). Discrete vowel sounds associated with specific phonemes are mapped into this space. The coordinates are obtained by averaging the position for multiple utterances across uniform speaker categories, and the figure also shows differences in phoneme positions across genders and English accents. In the vowel space any coordinate, and thus any vowel sound, can be generated, within a region bounded by vocal tract physical constraints. Moreover, in vowel utterances from a single individual there is also significant variance on both axes, which depends on several psychophysical factors. Deviations are usually smaller for the singing voice, and larger for spoken voice (Mehrabani and Hansen, 2013).

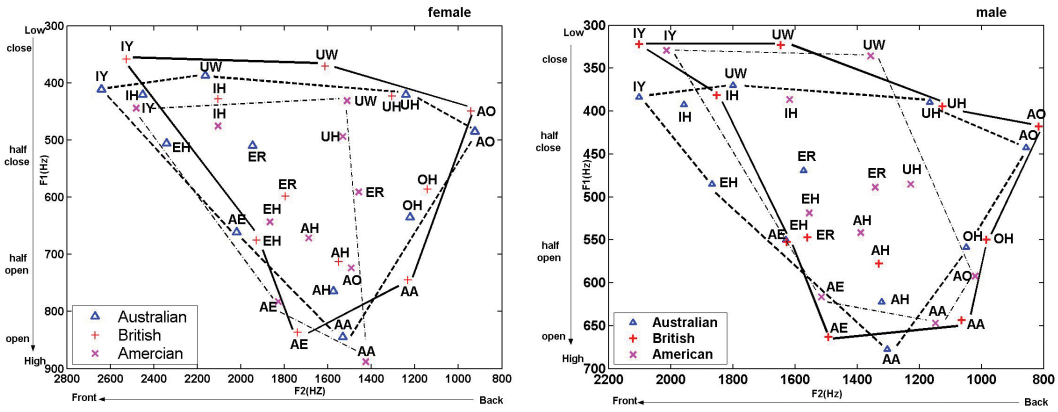

Figure 3.4: Vowel or formant space for different genders and English accents, after (Yan and Vaseghi, 2003).

For other phonemes, the articulation changes over time rapidly modifying the characteristics of the source-filter system. Simple movements in the vowel spaces determine diphthongs, a category of phonemes determined by a quick glide from the articulatory position of a vowel towards another one. More complex articulation modes include: stop articulation, in which the air flow is completely occluded by the oral and nasal tracts; nasal articulation, with the occlusion of the oral tract only; fricative or spirant articulation, presenting a continuous friction that generates a turbulent and noisy airflow; sibilants articulation, similar to the fricative but with a groove in the tongue guiding the airflow toward teeth that creates a higher pitch sound; lateral fricatives articulation, with the phenomena taking place on one or both sides of the tongue's edge; affricate articulation, which starts similar to a stop and releases into a fricative; trill articulation, with the tongue held in place but vibrating due to the airflow; flap or tap articulation, in which there is a temporary closure of the oral cavity; approximant articulations, where the vocal tract presents little obstruction; and lateral or liquids articulation, pronounced with the side of the tongue and small obstruction. Vocal sounds are classified as obstruents, mostly unvoiced, and sonorants, nearly always voiced, depending on the presence of obstruction in the vocal tract. These articulation manners determine 42 different phonemes in American English, detailed in the tree of Figure 3.5, where these are grouped by articulation and class. The number of phonemes usually differs across languages and is higher than the number of letters in the alphabet because each one is in general associated with more than a phoneme, like those consonants that can be either voiced and unvoiced. Moreover these can be also grouped into occlusives and continuants, color coded in blue and red respectively in Figure 3.5. This grouping is determined by the ability to

sustain a sound driven by a specific articulation over time. As we will discuss later, this is a key voice feature in the context of this dissertation.

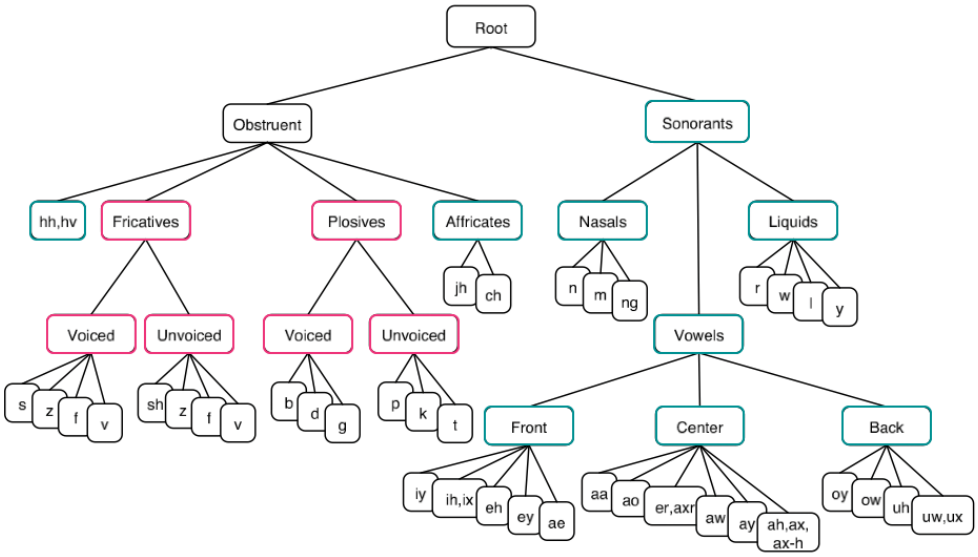

Figure 3.5: American English phoneme tree, after Dekel, Keshet, and Singer (2005), with occlusive and continuant grouping coded in blue and red.

The control over the voice apparatus is unconscious and it aims to produce specific dynamic sound rather than control the group of muscles involved in this process. The ear receives external and internal stimuli such as the bone conduction, and with the auditory system provides feedback about the voice generation activity. Thus we learn and master the use of the apparatus through a supervised reinforcement learning procedure, unconsciously adapting the control over the phonation system. Source and filter components can be controlled nearly independently. Furthermore in the source section the vocal folds vibration frequency and the energy of the pulses can be controlled separately because they are related respectively to pitch and loudness of the voice, which are two distinguishable perceptual aspects of sounds. The filter section of the vocal production apparatus has a more complex articulation structure, and the boundaries between independent and strongly correlated acoustic features are often faded and can differ across individuals. Sundberg considers the human sound generation mechanism similar to an organ, likely the most expressive acoustic instrument but also the most badly designed one, arguing that the relationship between articulatory movements and formant frequencies is very complex and difficult to control since it is a one-to-many system (Sundberg, 1987).

However at least the frequencies of the first two formants that determine the perceived vowel sound are considered independently controllable.

Vowels and voiced sound in general play a fundamental role in the traditional musical use of the human voice. These can be sustained to generate a pitch that matches a specific musical note. Trained singers turn their musical intention into sound through the vocal production apparatus in the same way trained musicians play and control sound and expression of a specific instrument. They acquire the ability to accurately control acoustic sound quality such as the loudness, the pitch stability or glissando, the vibrato, and other qualities such as the phonation, the spectral envelope, or the singer's formant (Sundberg, 2001). Moreover the singing voice quality can be further modulated with a set of vocal configurations, referred with the term *vocal register* (Henrich, 2006), which include four different modes of vocal fold oscillation namely low pulse register, low-to-mid-range pulse register, mid-to-upper-head register, and high whistle/flute register. In each register the wave generated by the glottal source has audibly different characteristics, therefore it generates different vocal timbres beside different pitch ranges.

In speech the modulation of the pitch, called *pitch contour* by linguists, has a key role as well. It determines the intonation, which may distinguish words or meaning, in pitch accent languages, while in tonal languages contributes to discriminate the various phonemes, lexically distinct variations of a single phoneme, different only in the tone of the vowel. Vocal tract articulations are used also to communicate information beyond the semantic meaning of speech. Paralinguistic and non-verbal components of the spoken voice such as the *prosody* and the *affect*, contribute to convey the emotional state of the speaker. Therefore the majority of individuals can use the voice expressively, at least in speech, even if having no musical experience or training.

The *beatboxing* is another vocal style, consisting in the vocal imitation of drums, percussive sounds, basslines, vinyl scratching, and melodies, to virtually emulate polyphonic music, and it finds its origins in the 1980's hip-hop culture. Beatbox performers make use of the largest palette of vocal sounds and vocal techniques across vocalists in order to generate a wide range of timbres. Moreover they try to cover or avoid linguistic hints that would suggest to a listener that the sound source is the human vocal tract. The extended vocal techniques often includes non-syllabic sounds, ventricular voice, falsetto, ingressive sounds, fast pitch variations, and trilling sounds (Stowell, 2010). Extensive training is essential, and also in this case the learning procedure is strongly based on the auditory feedback to match a specific non-vocal sound. Moreover placing the microphone at a closer distance from the

mouth and optionally cupping it with hands additionally extend the sound palette. Beatboxers demonstrates that the sonic potential, in terms of timbre variation, of the human voice production apparatus is much wider than what is commonly used in spoken and singing vocal styles. In fact Wishart argues that

> "From the vast array of possible sound-objects available from the human repertoire any natural language selects only a small portion and combines these phonemes into phonemic objects."
>
> – Trevor Wishart (1996).

This potential can be exploited in any voice-controlled system to enhance interaction capabilities and dimensionality, but it requires a method to capture user specific voice timbre range, which can vary more than cross-speakers phonemes also because utterance of these timbres are not formalized in any written form.

The complexity of the vocal tract is also evident looking into realistic speech synthesis models, in which Cook (1991) identifies approximately 40 articulatory control parameters. These can be controlled independently when synthesizing vocal sounds, but strong dependencies and correlations, difficult to model, may appear in the human vocal tract. These can also vary with factors such as physical characteristics and diverse phonation techniques across native languages. The characteristics of the vocal tract or sound that can be controlled consciously and separately determine the degree of freedom of any system instantaneously controlled by direct voice timbre. This an open challenge and a key issue for this thesis.

A deeper look into speech synthesis models provides us with other important information about control capabilities of the voice. A vocal synthetic sound such as a vowel, generated with constant synthesis parameters sounds unnatural and robotic. A more realistic feel is achieved applying a small amount of fundamental frequency and amplitude modulation. These, also known as *vibrato* and *tremolo*, are always present in the human voice and can be considered as unconscious slow rate Frequency Modulation (FM) and Amplitude Modulation (AM) around a target intentional value (Quatieri, 2008). Again the modulation amount can be significantly different across individuals or change with physiological factors. Vibrato and tremolo can be also a deliberate and conscious controlled choice, especially in singing, usually with a faster rate and wider amplitude. This characteristic of the vocal production apparatus suggest that even though a subject has the intention and the perception to utter with invariant articulation and constant acoustic features, some of the characteristic of the vocal sound will present a noisy behavior or modulations in amplitude and frequency.

Therefore another key aspect to use the voice for control of continuous quantities is the identification and minimization of the sources of "noise" in order to avoid propagating these down to the controlled system. Togneri presents evidences that the "spatial trajectories" of speech are four-dimensional manifolds embedded in higher dimensional spaces represented by a large set of heterogeneous low-level analytical features, related mostly to the filter part of the vocal tract (Togneri, Alder, and Attikiouzel, 1992). Even if not specifically investigating to which physical variables these four dimensions correspond to, and to what extent these can be explicitly controlled, the authors prove that vocal sounds usually embed three or four independent components. These are vital to establish a control as expressive as possible, but their extraction method requires a prior and offline analytical stage.

3.1.3 Voice processing in HCI

Despite privacy issues and inhibitions in the use of voice in spoken or singing form (Abril, 2007), the number of voice-driven systems is constantly rising and these are commonly found in consumer electronic devices. In general these applications offer users two advantages: a hand-free interaction and a faster data or commands entry. Since the prevailing use of the human voice is speaking, research efforts over the last fifty years have been mainly focused in automatically detecting the verbal contents from the voice audio signal (Rabiner, 1978) (Waibel and Lee, 1990) (Rabiner and Juang, 1993). Today the Automatic Speech Recognition (ASR) techniques find application in various scenarios in which the recognized words or sentence can be used to enter commands, search, dictate, or translate language in real-time.

In ASR the speech signal is processed at different levels. At first the voice sound signal is chunked into overlapping windows, with a length of about 10ms to 40ms in steps of 10ms to 20ms, in which the signal can be considered quasi-stationary. Each window is processed and analyzed by DSP algorithms to extract a vector of low-level features, which provides a compact representation compared to the original sequence of samples, filtering out irrelevant information. The different feature extraction techniques that can be used, detailed in Section 3.1.3.2, aim to capture either the state of the vocal apparatus or the sound characteristics relevant to the auditory system. In the first case the feature extraction models the physical characteristic of the sound source, while in the second it models the acoustic perception.

At the higher level the sequence of features vectors are used with an acoustic or physical model to determine a sequence of phonemes, which are considered the atomic elements of speech. The dominant and historically most successful approach

for this task is based on HMMs (Rabiner, 1989), which models states over a time series of features vectors, assigning to it the most likely combination of phonemes. In the next two levels phonemes are combined into words using a pronunciation model, and then words are combined into sentences using a language model. These two statistical models and their application are language, accent, and context dependent.

The output of ASR is a coarse temporal grain series of discrete choices within a limited lexicon. Moreover due to the presence of multiple processing layers, each involving complex statistical models, the response latency is in the order of seconds. Therefore ASR is far from meeting minimal requirements of any system for real-time musical interaction. Speech contains approximately 12 phonemes per second (O'Shaughnessy, 2003), and the lexicon contains typically less than 50 phonemes. Thus in ASR the phoneme recognition determines a discretization process with rate and resolution too coarse for the musical control task we aim to achieve in this thesis, which targets the simultaneous control of continuous quantities. However the low-level features, which consist of vectors of real-valued numbers, are extracted at higher rate and there still represent a quasi-continuous signal, appropriate and suitable for our musical control purposes. These are used in very different ways in the majority of the related works presented in Section 2.2, or further processed and left to the discretion of the user mapping preferences as in (Janer, 2005b) and (Kestian and Smyth, 2010).

The layering of three statistical models in ASR on one side requires a large amount of labeled training data, which usually consist of thousands of hours of speech with time aligned transcription, but on the other provides a reliable system. Besides the high word recognition accuracy, ASR systems are able to discriminate the correct output even in presence of background noises, they are robust to intra-speaker variability, and they handle inter-speaker differences (K. Stevens, 1971) (Yang, Millar, and MacLeod, 1996) providing a speaker independent system. Robustness and reliability are key factors in control interfaces but, as stated above, the higher-level statistical models of ASR are not compliant with musical control. Therefore, in the context of this thesis, providing robustness is an additional challenge that must be addressed at a lower level in the voice processing chain.

Singing, which is the secondary use of the human voice, contains both verbal and musical contents. The verbal part can be processed with ASR properly adjusted to meet specific singing attributes (Loscos, Cano, and Bonada, 1999). The musical part is often processed with a monophonic pitch extraction algorithm. Further processing of this information is used in applications such as lyrics following, automatic accompaniment generation, or in the context of Music Information Retrieval (Orio,

2006) to query databases of songs, loops, samples, or audio frames, as presented in Section 2.2. In *sub-verbal* or *non-verbal* applications the verbal contents of the voice are not relevant for the interaction and thus not recognized by the processing routines. These applications extract single or vectors of acoustic features from the voice audio signal, which are further elaborated for speaker recognition, speech/singing detection and segmentation. The real-time voice to musical device interaction systems, broadly described in Section 2.2, fall in the category of *non-verbal* vocal HCI, like the work presented in this dissertation. The machine interaction is established at abstraction level close to the sound, or timbre, of the vocal sound rather than to its verbal contents and meaning.

Apart from the voice processing and the final application, the communication channel between transmitter and receiver, mouth and microphone respectively, represents an issue. The acoustic channel is outside the control of the user and HCI system, and it can add to the original signal a considerable amount of noise. Even though some voice-processing algorithms present high noise robustness, it must be considered that external aleatory sound sources can still generate an error or an unwanted response at the system output. In general when the voice is not the sole user input modality, vocal controls are allocated to non mission-critical tasks, providing control expansion and reducing the user workload. Examples are the Direct Voice Input (DVI) system in the Eurofighter Typhoon EF200, or the voice commands for car infotainment systems. In both cases critical functions such as those for the vehicle navigation are generally not allocated to the vocal control, but left to common hand based controllers. We are faced with a similar scenario, as mentioned in the introduction, and therefore we need to extend this concept to instrument interfaces. Critical musical controls should remain mapped onto traditional tactile interfaces, while vocal control could be allocated to musical parameters that, if wrong, will less likely be perceived as an error by the listener.

3.1.3.1 Acoustic perception of voice and sound

The human auditory system, from the outer ear to the primary auditory cortex, can detect and track features of speech with higher robustness and reliability than any DSP analytical algorithm. The human hearing sense provides less numerical precision than machines, but it identifies acoustic features even when these are missing or hidden in the sound signal. The fundamental frequency f_0, at which the vocal folds vibrate in voiced sounds, is usually lower than any of the resonances of the vocal tract, and thus absent or weaker than other harmonics. The human pitch perception is

based, at least in lower frequencies, on the spacing between harmonics, which is equal to f_0, especially in the order of several hundred Hz (Goldstein, 1973), but it can represent an issue for numerical algorithms. A similar problem affects the computation of the formants. These are defined as the resonant frequencies of the vocal tract, but most of the tracking algorithms measure these from the frequency of the spectral peaks of the voice spectrum. If no harmonic falls in a resonance narrow band, a single formant doesn't generate a spectral peak and it may not be detected in such a way. The error is propagated forward since all other detected formants would have a wrong order. Humans, even in absence of a spectral peak at the resonance frequency, perceive the right formants and therefore the right vowel, due to visual cues (Mcgurk and Macdonald, 1976) (Nath and Beauchamp, 2012), and also due to the short-time memory we unconsciously use in any perception tasks to support disambiguation (Selfridge, 1959). For similar reasons, verbal communication is still possible over traditional phone lines, which have typically band pass from 300Hz to 4000Hz, a narrower bandwidth than the human voice frequency range. Moreover the loudness sensitivity of the human ear varies with the frequency, and it is most sensitive in the band between 1000Hz and 4000Hz. The loudness perception can be approximated with the logarithm of the energy. The equal-loudness contour in Figure 3.6 shows the sound pressure levels that are perceived as having equal loudness at different frequencies. The sound pressure scale is in Decibel (dB), a logarithmic scale, and the curve presents the lower values in the most sensitive band, in which the human voice usually falls.

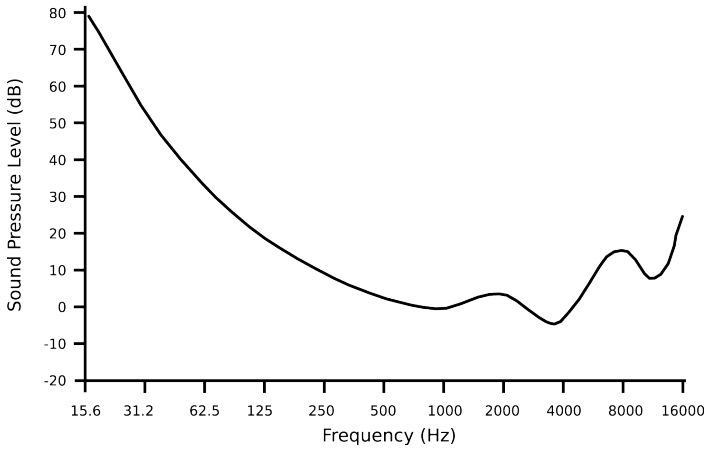

Figure 3.6: Equal-loudness contour of human hearing perception, measured empirically by Fletcher and Munson at Bell Labs in 1933.

The perception of the frequency spacing or frequency resolution of the human ear is non linear, and it decreases in the higher frequencies. Three auditory frequency scales

have been developed using different empirical subjective experimental settings: the Mel scale (S. S. Stevens, 1936), the Bark scale (Zwicker, 1961), and the Equivalent Rectangular Bandwidth (ERB) scale (B. C. J. Moore and Glasberg, 1983). These, displayed in Figure 3.7, present strong similarities, identical trend, and strong nonlinearity in the voice signal bandwidth. The Mel scale is widely utilized in voice signal analysis and the relationship to convert f_{Hz} to the linear perceptual frequency scale f_{MEL} is described in Equation 3.1.

$$f_{MEL} = 2595 \cdot \log_{10}\left(1 + \frac{f_{Hz}}{700\text{Hz}}\right) \tag{3.1}$$

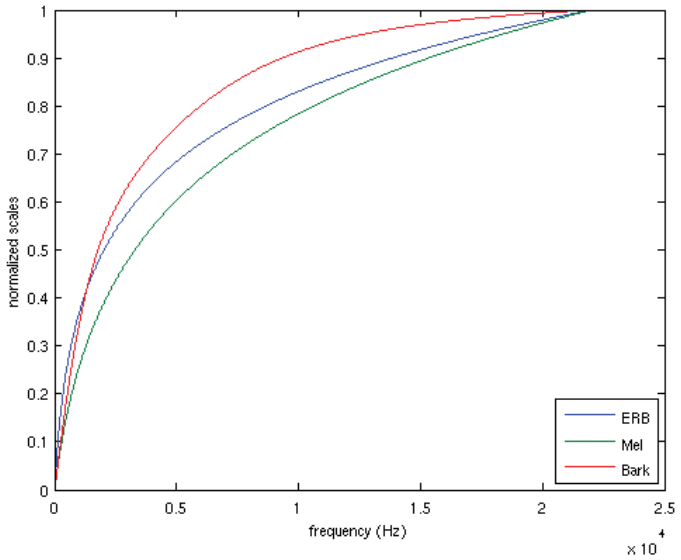

Figure 3.7: Comparison between Mel, Bark, and ERB frequency scales.

Since the human voice generation is controlled and regulated in strict relation to the hearing sense feedback, the implementation of a model of the auditory system is crucial for algorithms that analyze acoustic features. Some perceptual acoustic features of the voice can be tracked reliably in real-time, while other still require high computational load or forward-backward computation, both not suitable for real-time control applications. The auditory system can still perform in a reliable manner even when presented with voices from different individuals and in different environmental conditions such as the presence of unwanted background noise. This is in general not true for real-time analysis algorithms, which usually perform well only in specific scenarios. Therefore analytical algorithms may generate temporary wrong values, which likely determine spurious output signals or discontinuities, likely corrupting the final DMI real-valued parameter control purpose. An additional challenge is

represented by the high variability of this process, which cannot be easily generalized across users and application scenarios.

3.1.3.2 Physical and perceptual feature extraction

The voice acoustic signal is captured and transduced to an analogue electric time continuous signal $s(t)$ by a microphone, and then converted into time-discrete digital signal $s[n]$ by an Analog to Digital Converter (ADC). At this stage it can be handled, processed and analyzed as any baseband digital signal in the audio domain. For every window containing N samples, scalar or vectorial descriptors are computed. A conspicuous number of time and spectral descriptors has been developed for audio signals. Those often used with voice show relation with physical characteristics of the voice production or with aspects of the voice perception. Here we review the most common analytical techniques.

The **short time energy**, or root-mean square, in 3.2 describes the instantaneous power of the signal. The dB scale is often applied to provide a better estimation of the loudness perception $E_{dB} = 10 \cdot \log_{10}(E_s)$. It can be further elaborated using different filter curves for different bands to match the equal loudness contour. The energy can be used for voice activity detection.

$$E_s = \frac{1}{N}\sum_{n=0}^{N-1} |s[n]|^2 \tag{3.2}$$

The **zero-crossing rate** in 3.3 can be used as well to detect voice activity, and to discriminate fricative phonemes. It counts the number of time that the signal changes sign, giving a rough measurement of the signal brightness or noisiness.

$$ZCR = \frac{1}{(N-1)}\sum_{n=0}^{N-2} \mathbb{I}\{s[n]s[n+1] < 0\} \tag{3.3}$$

The **autocorrelation** measures the self-similarity of a signal with a version of itself time-shifted by k samples, described in 3.4. For periodic signals, the k value that maximizes the autocorrelation is likely to be the signal periodicity, and therefore the autocorrelation provides a measurement of f_0. However autocorrelation often detects wrong values for the reason detailed in 3.1.3.1, thus more elaborated and accurate **fundamental frequency** detection algorithms are used for voice and

instrument sounds. Puckette developed a f_0 estimator that uses amplitudes and frequencies of the constituent partials to compute a maximum-likelihood function. The presence of peaks at multiples at or near multiple of the fundamental frequency maximizes the likelihood (Puckette and Apel, 1998). The pitch estimator YIN (De Cheveigné and Kawahara, 2002) is still based on the autocorrelation method, but it introduces a set of modifications that drastically minimize the error rate such as the minimization of a difference function that is ideally null for periodic signals time shifted by $1/f_0$ periodic signals. Finally a more recent approach for f_0 estimation in monophonic signal performs multiple layers of fast lifting wavelet transform using the Haar wavelet, and for each audio frame it compares the spacing between the peaks in each wavelet to determine f_0 (Zbyszyński, Zicarelli, and Collecchia, 2013). The voice has a limited pitch range, so the value is generally considered in Hz or converted to the discrete values on the equal tempered chromatic scale.

$$R_k = \sum_{n=0}^{N-k-1} s[n]s[n+k] \tag{3.4}$$

Several **spectral descriptors** can be computed from the spectrum derived from the magnitude of the Short Time Fourier Transform (STFT) of every window such as the spectral moments, the slope, the decrease, the roll-off and variation. Moreover additional descriptors such as the noisiness, defined as the ratio between spectrum geometric and arithmetic means, can be extracted as described in (Vinet, Herrera, and Pachet, 2002). From the analysis of the frequency representation of each window it is possible to track **formants** frequencies, amplitude and bandwidth by detecting and measuring the peaks. The transformation to the spectral domain is usually based on the linear prediction, detailed later, or on the cepstral analysis in Equation 3.5, in which $c[n]$ represents the cepstrum, $\mathcal{F}$ the Discrete Fourier Transform (DFT), and $\mathcal{F}^{-1}$ the Inverse DFT (IDFT). The cepstrum, defined as the IDFT of the logarithmic magnitude of the DFT of a signal, is frequently used in speech analysis because it considers the spectrum as a quasi-periodic waveform, and taking its Fourier transform allows the separation of source and filter information. Glottal excitation is usually described by high index cepstra, while the low index models the vocal tract. Therefore these coefficients provide a compact representation of the spectral envelope and can be considered uncorrelated. The differences of spectrum computed over the phoneme /a/ using the two techniques and the deviations from the true envelope are illustrated in Figure 3.8.

$$c[n] = \mathcal{F}^{-1}\{\log|\mathcal{F}\{s[n]\}|\} = \sum_{n=0}^{N-1} \log\left(\left|\sum_{n=0}^{N-1} s[n]e^{-j\frac{2\pi}{N}kn}\right|\right) e^{j\frac{2\pi}{N}kn} \qquad (3.5)$$

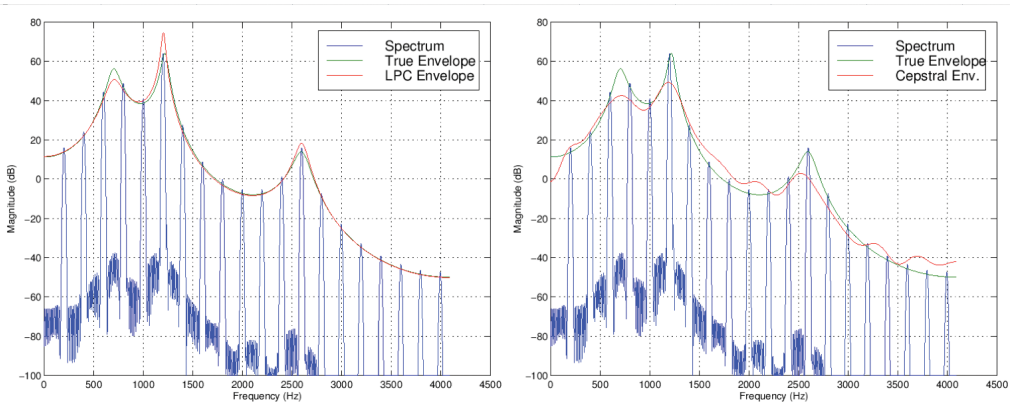

Figure 3.8: LPC and Cepstral envelopes versus DFT spectrum and True envelope.

Numerous techniques have been developed to improve capabilities of the earliest formant tracking algorithm proposed by Schafer and Rabiner (1970), based on an all-pole model characterized by second-order digital resonators. An adaptive and very robust tracking technique is proposed in (Mustafa and Bruce, 2006), but it presents a computational load too high for real-time, because it works with a single sample increment windows. An Unscented Kalman filter algorithm that provide a finely-grained tracking of the formant parameters is proposed in (Lazzarini and Timoney, 2009), while a Kalman-based autoregressive moving average modeling and inference method is presented in (Mehta, Rudoy, and Wolfe, 2012). As mentioned in the previous section the formant tracking presents implicit challenges, and for high pitch singing voice the situation can be even more critical since as little as one harmonic component may fall in the region of the first formant, making the spectral peak detection nearly impossible.

Multidimensional descriptors deriving from physical and perceptual models provide a global representation of the voice production process, and regardless of the final application, yield to better recognition, tracking or control performances when compared to the use of a set of scalar descriptors. The LPC, MFCC and Perceptual Linear Predictive (PLP) coefficients find large application in voice-based systems. The latter two are more recent and usually provide better representation and performances. The speech signal is usually preprocessed using a first order high-pass filter, which improves the overall Signal-to-Noise Ratio (SNR). This stage, called

pre-emphasis, is described in Equation 3.6, where α represent the preemphasis factor, typically equal to 0.97.

$$\dot{s}[n] = s[n] - \alpha s[n-1] \tag{3.6}$$

The **Linear Predictive Coding** coefficients computation is based on the linear prediction that is used not only for analysis of speech, but also for compression and synthesis (Itakura and Saito, 1967) (Markel, 1972) (McCandless, 1974). This analytical technique is based on the hypothesis that the voice production apparatus is a source-filter physical model where the glottal oscillations are the independent source, and the vocal tract resonances are modeled as a combination of time-invariant filters. The linear prediction coefficients a_k providing the best prediction $\hat{s}[n]$ based on the past P samples (which determines the order or number of the LPC coefficients), as in 3.7, are found minimizing the prediction error in 3.8. A single optimum solution can be always obtained using the covariance or the autocorrelation methods. The residual $e[n]$ estimates the glottal pulses, the coefficients a_k represent an all-pole model which simplifies the voice production system. The resulting LPC spectrum is obtained from the transfer function in Equation 3.9.

$$\hat{s}[n] = \sum_{k=1}^{P} a_k \cdot s[n-k] \tag{3.7}$$

$$E = \sum_{n=0}^{N-1} e[n]^2 \quad \text{where} \quad e[n] = \left(\sum_{k=1}^{P} a_k \cdot s[n-k] \right) - s[n] \tag{3.8}$$

$$\frac{S(z)}{E(z)} = \frac{1}{1 - \sum_{k=1}^{P} a_k \cdot z^{-k}} \tag{3.9}$$

The **Mel Frequency Cepstrum Coefficients** represent the spectral envelope of sounds, which is a salient component of the timbre perception (Davis and Mermelstein, 1980). The MFCC provides a perceptual modeling by parameterizing the shape of the sound, voice in this case, after warping the frequency axis to better represent the frequency perception in the human auditory system. They provide salient aspects of the spectral shape of a sound in a relatively small set of coefficients. The computation is similar to the cepstrum in Equation 3.5, but the magnitudes of the DFT are converted to a smaller number of coefficients, typically 22 for 8KHz and 40

for 16KHz sampling rates, through a bank of R triangular Mel filters. The following IDFT is replaced with a truncated Discrete Cosine Transform (DCT) to further reduce the number of coefficients to typically only 13. Equations 3.10-12 show the MFCC computation starting from the DFT of a window $S(k)$, where $V_r[k]$ represents the r^{th} triangular filter weighting function, bounded within the DFT indexes L_r and U_r. A_r is the normalization factor so that a flat DFT spectrum generates a flat Mel spectrum too. Figure 3.9 illustrates the Mel scale triangular filter bank, with applied normalization factor A_r in the bottom plot.

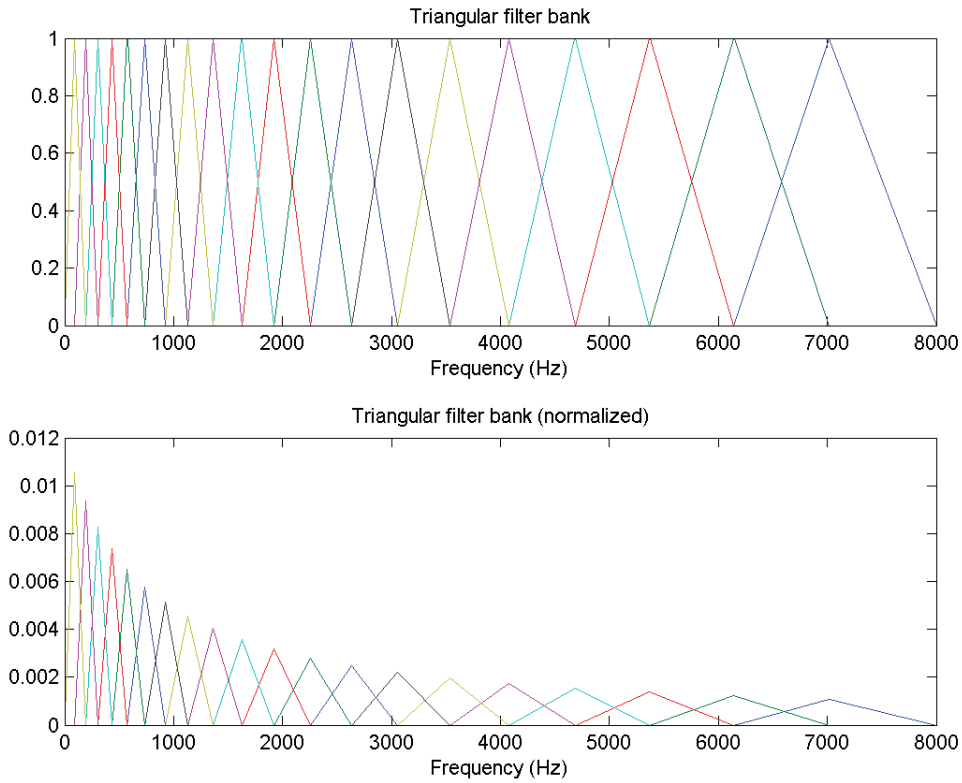

Figure 3.9: Example of Mel triangular filter bank with (top) and without (bottom) applied normalization factor.

$$MF[r] = \frac{1}{A_r} \sum_{k=L_r}^{U_r} |V_r[k] \cdot S(k)| \tag{3.10}$$

$$A_r = \sum_{k=L_r}^{U_r} |V_r[k]|^2 \tag{3.11}$$

$$MFCC[m] = \frac{1}{R}\sum_{r=1}^{R} \log\left(MF[r]\right) \cdot \cos\left[\frac{2\pi}{R}\left(r + \frac{1}{2}\right)m\right] \qquad (3.12)$$

The **Perceptual Linear Predictive** coefficients provides another compact representation of speech signal, and it uses more advanced psychophysics of the human hearing to compute an estimation of the auditory spectrum (Hermansky, 1990). At first the power spectrum of a voice frame is warped onto the Bark scale, using the approximation function in 3.13 in which $\omega = 2\pi/T$ represents the angular frequency, and then convolved to the power spectra of the critical band filter, in order to reproduce the human ear frequency resolution. Then the equal-loudness pre-emphasis compensates the uneven loudness perception at different frequencies, using the function in 3.14, which approximates the curve in Figure 3.6. The intensity to loudness conversion is approximated with the cube root, and then the output of the inverse DFT is fed into a linear predictive model, which provides an autoregressive all-pole model representing the auditory spectrum. These coefficients are in general more robust than MFCC for speaker-independent voice based applications because their computation use a 5th order all-pole model that is effective in isolating the auditory spectrum from speaker specific details. Most of the descriptors described in this section are susceptible to alterations of the short-time spectral values of the communication channel transfer function. However steady-state spectral factors have almost no impact on human perception of voice (Summerfield and Assmann, 1989). Therefore Hermansky developed the RelAtive SpecTrAl (RASTA) technique consisting in the utilization of a set of band-pass filter, one for every sub-band, in order to smooth rapid noise variations and to eliminate offsets (Hermansky et al., 1991) (Hermansky and Morgan, 1994). The resulting RASTA-PLP coefficients result in higher robustness to linear spectral distortions.

$$\Omega(\omega) = 6 \cdot \ln\left(\left(\omega/1200\,\pi\right) + \sqrt{\left(\omega/1200\pi\right)^2 + 1}\right) \qquad (3.13)$$

$$E(\omega) = \frac{(\omega^2 + 56.8\ \mathrm{E6})\omega^4}{(\omega^2 + 6.3\ \mathrm{E6})(\omega^2 + 0.38\ \mathrm{E9})(\omega^6 + 9.58\ \mathrm{E26})} \qquad (3.14)$$

Regardless of the category of features used, voice and speech processing often extend the coefficient vector adding the delta and acceleration (also known as delta-delta) coefficients, in order to provide an implicit trajectory modeling of each

coefficient. These are computed with the Equations in 3.15 and 3.16 respectively, in which c_t represents each coefficient at time t, and the delta window δ is usually equal to two. The LPC, MFCC and PLP analyze primarily the resonances of the vocal tract, which, as explained earlier, characterize mostly voiced phonemes, while they hold less information about unvoiced sounds. Moreover considering additional information such as the phase or the phonation mode does not improve the performances of voice-based HCI system (O'Shaughnessy, 2003), even though these sonic characteristics are perceivable by the human auditory system.

$$\Delta_t = \frac{\sum_{i=1}^{\delta} i(c_{t+i} - c_{t-i})}{s \sum_{i=1}^{\delta} i^2} \tag{3.15}$$

$$\Delta\Delta_t = \frac{\sum_{i=1}^{\delta} i(\Delta_{t+i} - \Delta_{t-i})}{s \sum_{i=1}^{\delta} i^2} \tag{3.16}$$

3.2 Voice as source of gestural musical control

In line with the overall principles and requirements for VCI4DMI of Section 2.3.1, we will investigate and design a mapping method that does not require specific vocal timbre to operate, but that can be automatically adapted to the user vocal control preferences and style. Only a small fraction of voice timbre range can be represented by phonemes or onomatopoeic description, therefore, with the exception of some demonstrative examples, we avoid written transcription of the vocal sound. We generalize defining only two categories of vocal sounds, *vocal-gesture* and *vocal-posture*, which are sufficient to describe the user interaction with the system for training and performance purposes. In body language non-vocal communication the terms *gesture* and *posture* describe movement and static position respectively. These identify also the presence or absence of a dynamic temporal evolution. A gesture can be considered the temporal evolution across two or more postures. In the context of musical instruments, the term posture is not common, while the term *gesture* identifies any motor action of the performer that expresses a musical intention, later realized into sound by the instrument. The gesture strongly depends on the instrument interface input modality. Therefore we define the two categories as follow:

- a *vocal-gesture* is a vocal sound in which timbre dynamically varies over time by an arbitrary speed and amount;
- a *vocal-posture* is a vocal sound which timbre does not change over time.

By the definition, only continuant vocal sound, color coded in blue in Figure 3.5, can represent a vocal-posture while occlusive sounds cannot be sustained over time. Vocal-gestures can contain any variation across continuants and occlusive vocal sounds, presenting wide timbre variation or just nuances. It can be a slow gliding from a timbre to another one or a more abrupt variation, the trajectories between and across sounds can have any form. We aim for a rational and obvious vocal-control, thus we design the VCI4DMI to respond to vocal-postures with steady instrument parameters, and to vocal-gestures with parameters variation determined by the specific mapping. The same applies for the GC that we describe in this Chapter, which convert the vocal sounds into intermediate signals for mapping purposes. Silence can be considered as a special case of vocal-posture. However the natural control and low cognitive complexity characteristics that we aim to bestow to the VCI4DMI, require that the GC output, as well as any DMI parameters and controls, are hidden to the performer who thinks, controls and gets response from the system only in sonic form. This approach is akin to the way we interact with our vocal production apparatus, which we essentially extend adding the transparent chain made out of the vocal interface plus the instrument.

The two spectrograms in Figure 3.10 are related to a vocal-posture and a vocal-gesture. In particular the posture is related to the /i/ phoneme, while the gesture represent a timbre gliding across different vowels.

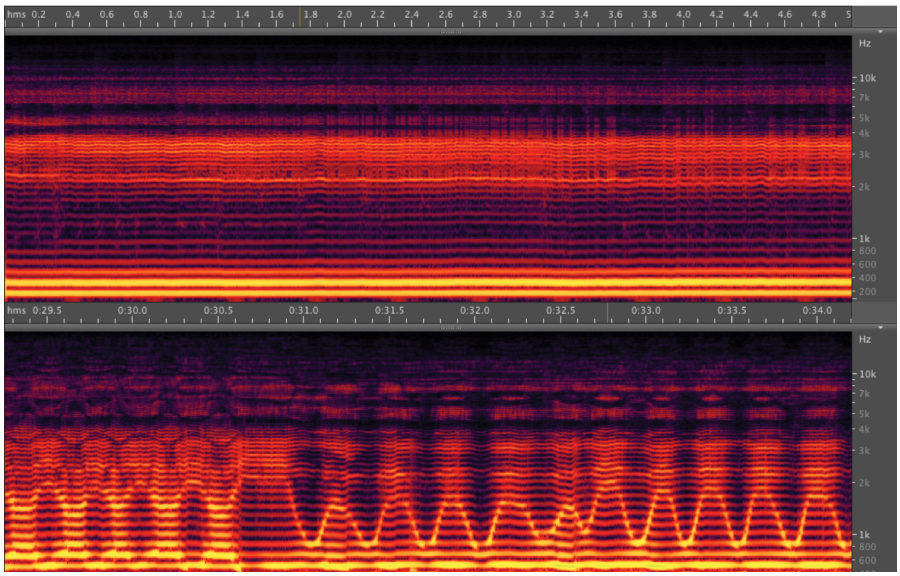

Figure 3.10: Spectrogram of an example of vocal-posture (top) and vocal-gesture (bottom).

The spectrogram, which represents the magnitudes of the DFT, provides an interpretable graphical representation of the raw sonic data, and from a visual analysis it is possible to observe that:

- in vocal-postures there are slow rate amplitude and frequency fluctuations, minor irregular burst variations and short-time noise-like variations. These are beyond the vocalist's control, who is perceptually producing a steady timbre;
- in vocal-gestures the different patterns representing different vocal timbres are evident as well as the presence of high redundancy in the data representation. In the dynamic transitions the identification of independent or multidimensional variations is not trivial.

These observations are valid also for more compact representation of the voice audio signal such as sequences of low-level features vectors computed with the techniques described in Section 3.1.3.2. Therefore naïve mapping of one or more of these features to any instrument parameter for musical control will present a noisy behavior over vocal-postures, and strictly correlated control of multiple parameters, similar to a one-to-many mapping, over vocal-gestures, neither of which is inline with the mentioned design requirements and principles.

Extracting a set of robust and independent continuous signals from the voice is the main challenge we face here. The robustness condition is important for the vocal-postures because their noisy components should be attenuated, if not rejected, in the GC output signals. The independence, or at least, statistical uncorrelation, of the GC outputs is the key factor in providing real multidimensional control, which embed the performer's musical control intention. Distinct users may present perceptible differences in their vocal tracts, and considerably distinct styles and timbre ranges in their vocal-gestures and vocal-postures. Consequently, a fixed algorithm implementing the GC functionalities can offer only a sub-optimal solution. Our goal is rather to find the optimal solution for each case through the design of an adaptive and generative mapping algorithm that retrieves the necessary information and generates a model from a set of training data provided by the user. This can also implicitly confer adaptation to other context-dependent issues such as the frequency response of the microphone, background acoustic noises, communication channel noises and offsets. In the setup phase, the user intervention and required expertise should be minimal and limited to produce simple training data, in order to set a low system entry barrier and high transparency. This implies that the ML techniques used must be unsupervised or self-supervised. However, the final system should present a

high ceiling, allowing expert users to change training and functional settings for additional mapping personalization, but also for further development and exploration of the contributions of this work.

3.2.1 Training data

The training data for the GC consists of recordings of several instances of consistent vocal-gestures and a set of vocal-postures. The latter should comprise all the continuant vocal sounds present in the gesture. Each instance of vocal-gesture should contain dynamic timbre variation, across an arbitrary number of vocal sounds. The GC component of the VCI4DMI is entirely determined from knowledge extrapolated from this user provided training data set. The temporal unfolding of the gestures in the various instances is not important and can vary. Temporal information is discarded in a later stage since we focus mainly on the geometrical and spatial unfolding of the gestural data. The rate or speed of variation in the gesture will instead have an impact since we sample the voice features at regular intervals. Both excessively slow and fast variation present shortcomings, as we will analyze later in the chapter, but relatively slow variation is preferred, since this provides more detailed representations of the gesture. We do not set a minimum amount of training data, however ML techniques benefit from larger training sets because they can refine and increase the complexity of the model. Therefore a higher number of gesture instances and longer duration or postures are likely to provide better GCs.

A basic example of training data is a vocal-gesture comprising only a glide between the vowels /a/ and /i/, to be presented in any temporal order in the different training instances. Therefore the comprised vocal-posture will be related to the steady /a/ and /i/ timbres. A more complex example is represented by gestures varying within the entire vowel space, with the associate postures represented by the collection of all steady vowel sounds. In the first case we expect the adaptive and GC to find a simple mono-dimensional control space in the data, while in the second the number of independently controllable quantities should be at least two-dimensional.

3.2.2 Parametric features computation and selection

The aim of computing robust and independent signals from the voice can be pushed upstream to the low-level features computation. Therefore we introduce prior study on which features to compute, with the related analytical settings, to better represent the requirements related to postures and gestures. In the works presented in Section

2.2 there is a lack of research-based motivation related to the choice of voice low-level features, which is instead often based on common practices in voice processing. Exceptions are found in the works of Loscos and Aussenac (2005), and of Stowell and Plumbley (2008). The first presents a preliminary study to identify the two features that better describe and implement a simple and well defined mapping problem. The second presents a more complex study, which analyzes a database of about 10 hours containing speaking, singing and beatboxing, and it considers a larger set of audio descriptors to identify robustness and amount of extra information that each feature carries. Both works present numerical and qualitative proof of the improvement given by the prior study on the low-level features to consider in the related voice-controlled systems. Here we push forward this approach in two directions:

- perform a study on the features based on the specific data from a single performer;
- extend the study to find the optimal feature computation settings and options.

Computational settings such as the sampling rate, the window size, and the step size, determine the computational cost and the extraction rate, but they also have an impact on the numerical and statistical result of the analysis of the scalar features. If we consider the vector of coefficients generated by the LPC, MFCC, and PLP, we find that the result of the analysis depends also on the value assigned to the pre-emphasis factor and to the order of the computed coefficients vector. Changing the order corresponds to a modification or increasing complexity of the underlying model to estimate. For the LPC a different order implies a different number of poles in the model of voice production. In the MFCC and PLP different orders, but also different sampling rate, determines a different number and central frequency of the filter banks approximating the auditory system. Therefore the individual coefficient may capture different aspect of the voice signal. Although there are default values for these settings, there are no criteria to establish a priori which combination of settings can provide better results in specific cases. In different application domains such as the ASR, it has been demonstrated that an appropriate tuning of the computational setting can significantly improve the performances of the system (Sanderson and Paliwal, 1997).

3.2.2.1 Quality metrics

In order to identify the optimal settings, we need to define metrics and methods to measure numerically the result we obtain with different computational parameters. We assume we have P vocal-postures in the training set, and for each window we compute a vector $\mathbf{v} = [v_1, v_2, \dots, v_C]$ containing C features or coefficients. With W_p we identify the number of analysis windows in the vocal-posture with $p = \{1,2, \dots, P\}$. To measure the noisiness of each coefficient v_{i-p} in the sequence of vectors representing each vocal-posture $\mathbf{V_p} = [\mathbf{v}_{p-1}, \mathbf{v}_{p-2}, \dots, \mathbf{v}_{p-W_P}]$ we compute the Relative Mean Difference (RMD), which is a scale invariant measurement of statistical dispersion, described in Equation 3.17. Compared to the original formula we added the modulo operator in the denominator in order to avoid negative values of the RMD generated by v_i with negative mean. The RMD in 3.17 provides an estimation of the noisiness, or robustness, of each scalar feature. Therefore we mark as noisy and reject from vectors $\mathbf{v}$, those components v_i presenting an average dispersion over the set of postures larger that a pre-defined threshold RMD_{tresh}.

$$RMD_{v_i-p} = \frac{\sum_{k=1}^{W_p} \sum_{z=1}^{W_p} |v_i[k] - v_i[z]|}{W_p \cdot \left| \sum_{k=1}^{W_p} v_i[k] \right|} \tag{3.17}$$

To measure the amount of non-redundant information in the vocal-gestures we measure the intrinsic dimensionality of $\mathbf{V_G} = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{W_G}]$, which is a matrix containing all the features vectors computed on all the frames of all the instances of vocal-gestures, for a total of W_G analysis windows. This estimates the real dimensionality of a manifold embedded in a parameter space with larger dimensionality. For example, measuring an intrinsic dimensionality equal to two means that the vocal-gesture data can be reduced to a flat surface, with arbitrary internal distribution, and therefore only two axes or components will be necessary to effectively represent unique gestural information. We tested several existing methods for the intrinsic dimensionality measurement on real and simulated data, and differences were noticed only in the decimal part of the result. However the correlation dimension (Grassberger and Procaccia, 1983) and the maximum likelihood (Levina and Bickel, 2004) methods showed better consistency and lower estimation error rate. The average of the two results determines our measure of the intrinsic dimensionality of the whole time series of feature vectors $idim(\mathbf{V_G})$. In order

to obtain true estimations, features in **v** computed with different techniques require range equalization to avoid large gaps in their absolute values.

Given a specific configuration cf_x of the setting for computing **v**, we measure the robustness and quantity of information, which we aim to maximize, with the quantity Q_{cf_x} in Equation 3.18, and we take it as an overall quality rating for each cf_x case. The first term of the Equation 3.18 estimates the quantity of non-redundant information embedded in the vocal-gestures $\mathbf{V_G}$ by averaging the intrinsic dimensionalities measures with the two methods mentioned above. The second term of the Equation 3.18 estimates the robustness of the selected features computed over the P vocal-postures by averaging the means $\mathbb{E}[RMD_{v_i-p}]$ of the single features v_i marked as non-noisy. RMD_{tresh} is the upper bound for this quantity, which tends to zero in cases with minimal noise. The modulo and logarithm operators in the numerator remap the range to positive increasing values as the noisiness decrease. The normalization with the modulo of the logarithm of the upper bound prevents the quantity from diverging. In the quality rating Q_{cf_x} we do not include the count of how many features, among the C computed, are compliant with the RMD condition because this has no relevance and direct relation with the true intrinsic dimensionality. The threshold on the RMD is usually set to 0.5. On real data the first term usually ranges in the interval [2, 5], while the formulation of the second term produces results in the lower half of that range. Therefore the quality rating Q_{cf_x} is intentionally unbalanced toward the first term, because the robustness condition has been already applied to the data, excluding the noisy features from the $idim(\mathbf{V_G})$ estimation. However we still consider the overall RMD of the robust features in the Q_{cf_x} to promote configurations that further minimize the statistical dispersion.

$$Q_{cf_x} = \left(\frac{idim_{ML}(\mathbf{V_G}) + idim_{CD}(\mathbf{V_G})}{2}\right) + \left(\frac{\left|\log\left(\frac{\sum_p \mathbb{E}[RMD_{v_i-p}]}{P}\right)\right|}{\left|\log\left(RMD_{tresh}\right)\right|}\right) \qquad (3.18)$$

3.2.2.2 Blind search algorithm

The quality measure introduced above and the *breadth-first* blind search algorithm we describe in this section do not depend on the specific composition of the vectors **v**. Our approach avoids prior decisions on feature selection, therefore we start considering a large set including those described in 3.1.3.2, with their delta and

acceleration coefficients. The exact dimensionality C of $\mathbf{v}$ depends also on the orders of the models estimated by the LPC, MFCC and PLP, which is one of the computational settings. To avoid a combinatory explosion of the cases to evaluate we vary the settings in limited ranges centered on the typical or default values. However the method we present here allows also deeper searches, impacting only the total search time. The blind search for optimal settings is performed offline, but in the final performance VCI4DMI system the features will be computed and used for mapping purposes in real-time. Therefore we favor settings that are computationally efficient such as integer power of two window sizes, rational ratios of the audio sampling rate, even number of Mel or Bark Bands. The computational parameters and the respective ranges we explore are:

- Sampling rate – 16KHz is the common sampling rate in voice processing. It is a tradeoff between computational cost and information loss at higher frequencies. Modern CPUs are powerful enough to handle voice DSP processing even at higher sampling rates. The energy of the human voice is mostly concentrated in the band below 8KHz, but components are present at up to almost 20KHz. According to Monson (2011) the reasons for neglecting the band above 8KHz are mainly historical and physical, but also the higher frequency band may have perceptual significance. The sampling rates we consider in the search are [8, 16, 24] KHz, extending and reducing the typical considered audio band by 50%.
- Window size – we consider only windows containing a number of samples equal to any integer power of two in the range [256, 4096], allowing efficient Fast Fourier Transform (FFT) computation without zero-padding. The value is also conditioned by the sampling rate and we limit the resulting audio frame to 256ms in length. Moreover the size of the window must be bigger than the step size. Values below 256 are not considered because the FFT frequency resolution is too low for the auditory scale frequency warping and related filter banks.
- Step size – this value determines how frequently a feature vector is computed and directly impacts the computational load. In order to compare the Q_{cf_x} quality ratings results we need to generate for each cf_x the same number vectors $\mathbf{v}$ from the vocal-postures and vocal-gestures, otherwise the statistical measurements will present different accuracies and thus not be comparable. Therefore the value of the step size relative to sampling duration changes with the sampling rate but it provides a fixed step size in time. We perform

the search using a step size of 16ms, as motivated in the next subsection. However the user can change this value if necessary, also in the real-time VCI4DMI system.

- Pre-emphasis – we explore four values of this parameter, in particular [0, 0.31, 0.63, 0.97]. Typical values are 0 (no pre-emphasis) and 0.97. In the proposed range we explore also two additional values providing low and mid pre-emphasis.
- Order – the number of LPC, MFCC, and PLP typically computed is in the range 12 to 14. We explore a larger span of values because this computational setting modifies the underlying estimated model, and the impact on the quality Q_{cf_x} is noticeable. We consider the range [6, 24] taking only the even values, due to some optimization in the real-time feature extraction algorithm, and with restrictions on low Nyquist frequencies that limit the maximum number of filter banks on the MFCC and PLP.

With the ranges described here we obtain 368 setting that we evaluate using the simple algorithm described by the pseudocode in Figure 3.11. The order is considered separately from the other settings into the inner loop.

```
FOR every setting
      FOR every order
            compute features on postures
            measure average RMD over postures
            identify noisy features
            compute features on gestures
            reject noisy features
            normalize features
            compute and store Qcf
      END
      compute Qcf w/ mixed LPC MFCC PLP order local max Qcf
      find and store max Qcf and settings
END
find max Qcf and display best settings
```

Figure 3.11: Blind search algorithm pseudocode for finding the voice low-level features computational parameters and the noisy features rejection maximizing the quality rating.

We do not consider all possible combinations given by different computational orders in the feature vectors, because the total number of cases would spike to about 40,000. However changing the order setting into the inner loop, keeping track of all the partial result, and measuring the partial Q_{cf_x} for LPC, MFCC, and PLP only, allows us to extend the search to extra cases given by heterogeneous orders mixing their

individual local maxima. Therefore the best settings may present individual and different order values for each low-level feature group.

3.2.2.3 Preliminary study

Before presenting the result of the search for the optimal computational settings, we briefly report the results of some preliminary studies that contributed to the refinement of the method and reduction of the search space. We started working on very large features vector that, in the worst case, had dimensionality of about 300, including delta and acceleration coefficients. Besides increasing the search time and the real-time computational load, the curse of dimensionality was an issue affecting the subsequent training stages in the GC. The level of redundancy in the original features set was excessively high. The set reduction we discuss here is based on the trends and observations on preliminary tests on a real voice data set, and not on prior theory-based decisions. The dataset was comprised of recordings collected from four individuals, two males and two females, using different microphones and ADCs. From each individual we collected 10 vocal-postures and 5 different vocal-gestures. The findings we report were later confirmed by analysis on a larger dataset, collected during the user evaluation described in Chapter 7, that we utilize later to present further results.

The delta and acceleration coefficients, as expected, do not satisfy the RMD condition in nearly 100% of the case. All the v_i present minor fluctuations over the vocal-postures as explained before. The delta coefficient, or the first derivative, which describes the rising or falling trend of the coefficient will in turn present a similar noisy behavior but with greater amplitude. The acceleration coefficients, or second derivative, follow the same behavior with even larger fluctuations. In Figure 3.12 we show the average RMD and deviation of each coefficient and the variance in **v** over a vocal-postures set of an individual vocalist. In particular we display the cases with lowest and highest overall RMD over the non-noisy features. With the different color-coding we identify the features, delta and acceleration coefficients. It is evident that in both cases the average RMD value of any delta and acceleration coefficient is well beyond the 0.5 threshold, and in some cases also beyond vertical axis visible range, which is 20 times the threshold. Any reasonable increase of the threshold did not change the outcome and this has been observed for every vocalist dataset. Therefore we discard delta and acceleration coefficients from **v** we obtaining a 66% dimensionality reduction.

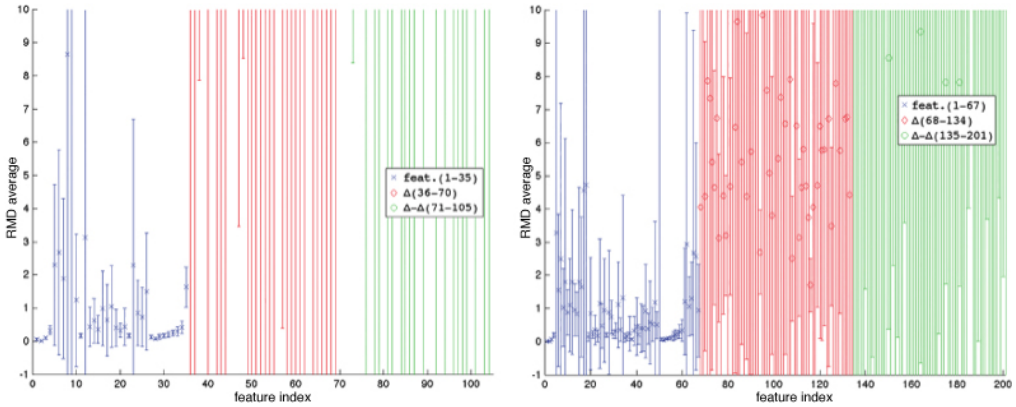

Figure 3.12: Average RMD with standard deviation for features (blue), delta (red), and acceleration (green) coefficients for worst (left) and best (right) computational settings. Delta and acceleration coefficients are always marked as noisy features and discarded.

The frequencies and amplitudes of the first four formants were computed and included in **v** using three different tracking algorithms. These were in general non-noisy, but in some cases we observed considerable jumps or discontinuities in their values, in both postures and gestures. These are due to false detections rather than variation on the formants spectral position. This behavior becomes more evident and frequent in environments with the presence of background burst noise or music. Therefore we removed formants information from **v** since those tracking errors can be propagated and magnified in the system, generating detrimental unwanted and random responses at the VCI4DMI output. However the formants information is gathered from the spectral envelope, which is still represented and used in other coefficients such as the LPC and MFCC. In Figure 3.13 there are examples of the sudden jumps and discontinuities given by the formant tracking algorithms. In the top section of the figure, the third and fourth formants are tracked over a vocal-gesture, in which the vocalist glides smoothly between voiced vowels. In the bottom section the four formants frequencies are tracked over a vocal-posture. In both cases discontinuities and false detection are evident and they affect true tracking of the voice timbre.

Short-time energy and pitch over vocal-postures turned out to be below the RMD threshold in most cases, and their tracking is not affected by significant discontinuities. However these features, related to the source part of the voice production apparatus, will be used and explicitly mapped to a side component of VCI4DMI prototype, as described in Chapter 6. Therefore to avoid overlapping controls of different parts of the system, we excluded these from the group of features considered for the generative mapping to DMI parameters.

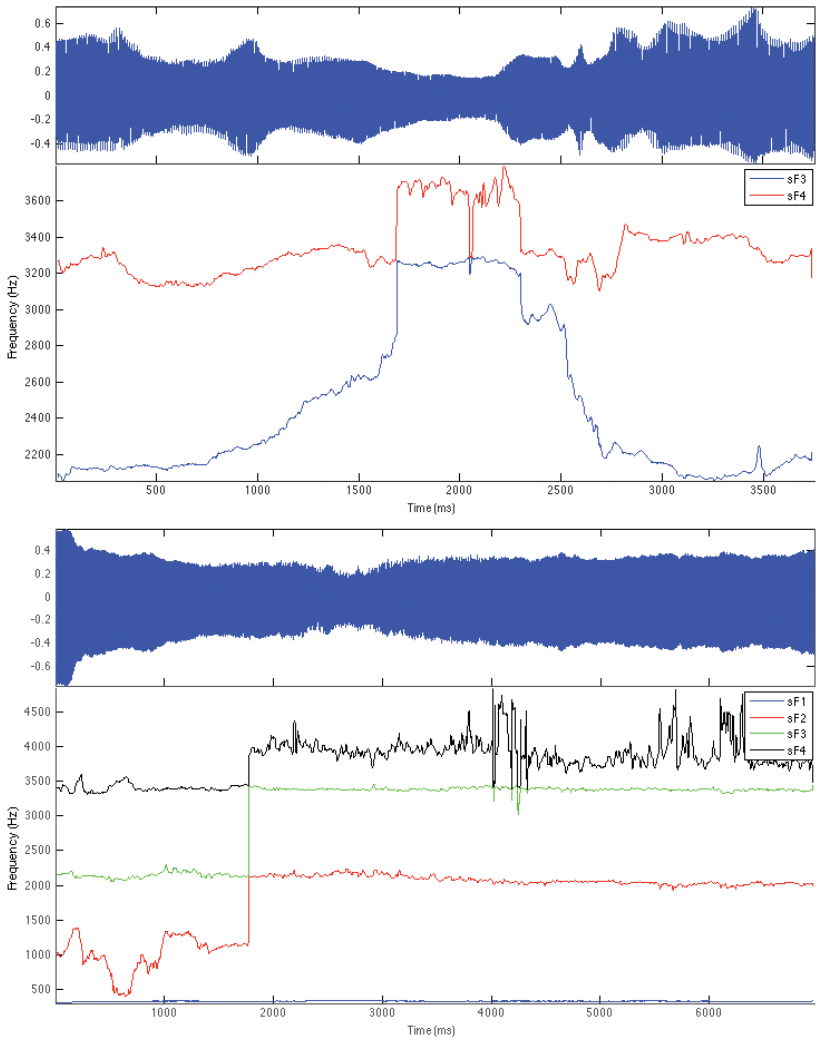

Figure 3.13: Discontinuities in the results of formant frequency tracking for an example of vocal-gesture (top) and vocal-posture (bottom). The discontinuities represent a switch in the tracked formant and make formant frequencies unreliable features for mapping purposes.

The band-pass filters in the RASTA-PLP tend to smooth rapid noise variations and to eliminate offsets, therefore this group of features shows the highest robustness over the postures. The approach is adaptive thus the filters require some time, approximately 3 to 5 seconds, to reach a stable state in presence of steady background noise or offsets. The implicit consequence is that the RASTA-PLP coefficients keep changing during this interval, even if the input audio signal is completely steady. This requires sacrificing a part of the training to bring the filters to a stable state. Nevertheless in real application scenarios the environmental noise conditions of the communication channel can change continuously and drastically. We observed that every time the feature computation is restarted after a break due to no voice activity, or after every pause of the vocalist, this adaptation process takes place again. Therefore the first few seconds of RASTA-PLP coefficients are inconsistent with

those computed later, and with the other features, which do not perform any adaptation. This has a direct impact on the VCI4DMI response consistency over time. Moreover the RASTA-PLP shows high redundancy to the PLP, and thus these are excluded from **v**.

The remaining features are spectral moments, LPC, MFCC and PLP, with a number of coefficients between 22 and 76 depending on the order computational parameter. From further testing, involving measurements and real use cases on the early prototypes of the VCI4DMI and the larger final dataset, we observed that MFCC and PLP have a higher number of non-noisy coefficients, and carry the largest fraction of information in implementing the voice-gesture mapping, usually between 70% and 95%. Moreover we tried to further reduce the feature set removing alternatively and concurrently the moments and the LPC. We evaluated the usability, with similar experiments to those described in Chapter 7. It emerged that on removing the four spectral moments the usability stayed the same, or in some cases slightly improved. We speculate that this may be due to the difference in the frequency axis, which is linear for the spectral moments and warped to the auditory scale in the MFCC and PLP. The numerical results and user evaluation presented in the following sections and chapters are thus based on vectors **v** including only LPC, MFCC, and PLP.

The fixed value of 16ms for the step size was determined by observations over results obtained for different step sizes. We did not directly compare the quality ratings for each step size, because as stated before these would not be statistically meaningful. The average smallest window we obtained in the optimal features computation settings is about 21ms. Therefore the 16ms step size is a tradeoff that guarantees the presence of window overlap of at least 25% in most cases, and avoids an excessive feature computation rate, which affects the real-time computational load.

Finally we observed the effect given by different threshold RMD_{tresh} values. Increasing it results in more features being marked as robust and passing through the selection stage, leading, as expected, to noisier DMI parameters generated by vocal-postures in the final version of the VCI4DMI but also in the early prototypes. Lowering the threshold we observed that the few features not rejected had little correlation with the leading timbre variation in the vocal-gestures. The intrinsic dimensionality decreased but not drastically. However features representative of the voice always present a certain amount of noise over the postures as we discussed earlier in this chapter. Use case tests confirmed that with very low RMD_{tresh} the system response, probed in different points of the interface processing chain, was

poorly controllable and presented several glitches. Values in the range [0.4, 0.6] provide the best tradeoff.

3.2.2.4 Results

The results presented here are based on the larger database that includes data from eleven different users. The size of the database is limited and thus we do not aim to draw generic conclusions valid across vocalists, but only recurrent trends and differences. Since we designed this method to be adaptive to voice characteristics of specific users, we expect and highlight differences across users' optimal features computation settings. In Figure 3.14 we show the results in terms of the number of robust features, average RMD of robust features, intrinsic dimensionality, and quality rating Q_{cf_x} for a specific user training data set over the 368 computational settings. For the Q_{cf_x} we show both terms that contribute to the final value. In the search range the differences between minimum and maximum for all measurement are evident and significant. The value of the intrinsic dimensionality, which has the greatest influence on the quality rating Q_{cf_x}, always presents a quasi-periodic trend as well as the number of robust features. This is due to the four nested loops in the search algorithm, with the inner one changing the order, and the outer changing the sampling rate. In general we observe that the highest spike in $idim(\mathbf{V_G})$ determines the range in which the absolute Q_{cf_x} maximum falls, which may not coincide due to the contribution of the term depending on the average RMD. However the Q_{cf_x} maximum is usually in the close neighborhood of the $idim(\mathbf{V_G})$ maximum. This indicates that generally the order and pre-emphasis, varied in the inner loops of the search algorithm, have a greater impact on the average RMD, while the other computational settings have a greater influence on the first term of Equation 3.18. This behavior is also noticeable in Figure 3.15, where we show a comparison between three quality ratings Q_{cf_x}. The first two are computed with training data of the same user but with different vocal-gestures and vocal-postures. The third training data corresponds to the same gestures and postures in the first case, but the user is different. These correspond to the Table 3.1 entries with ID 11-1, 11-2 and 6-1, in which we summarize the search results on the larger dataset, mostly derived from the user evaluation of Chapter 7. In the table the ID entry is related to the user followed by the vocal-gesture numeric IDs. The table provides a comparison for similar vowel gliding gestures, with comparable amount of training data, across 11 different users.

For user 11 we provide results also for other training data sets containing a various vocal-gestures and vocal-postures.

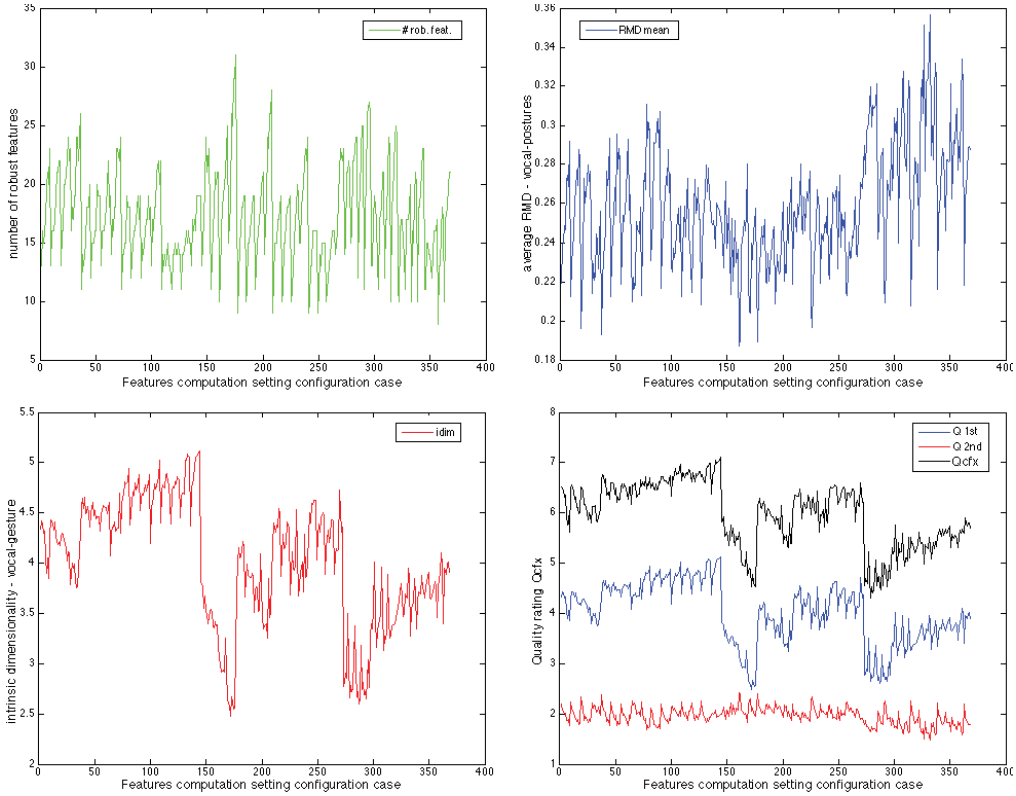

Figure 3.14: Number of robust features, average RMD of robust features over vocal-postures, intrinsic dimensionality over vocal-gestures, and quality ratings Q_{cfx} showing the two contributing terms, computed over 368 features computation setting cases for a single data set.

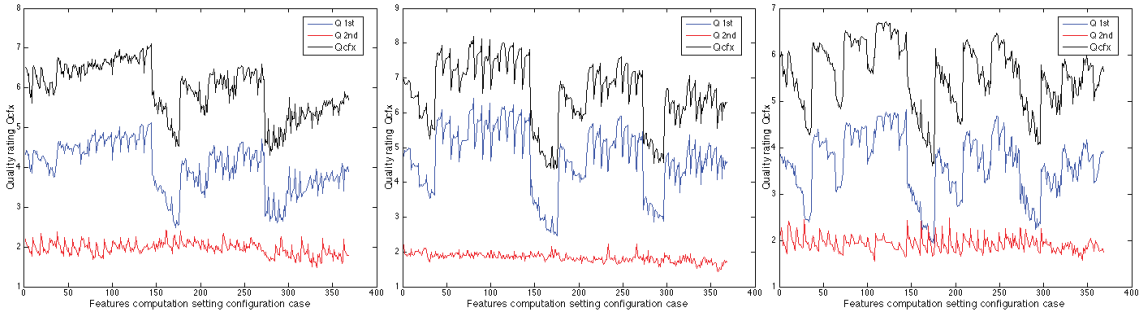

Figure 3.15: Quality ratings Q_{cfx} and contributing terms for 368 features computation setting cases on training data set 11-1, 11-2, and 6-1. The Q_{cfx} maximum, associated with optimal features computation settings and robust features selection, depends on the specific user vocal-gestures.

ID	*S. Rate*	*Win.*	*Step*	*Pree.*	*LPC*	*MFCC*	*PLP*	*idim*	*RMD*	*Qcf*
1-1	16000	512	256	0.00	8~1	6~3	6~5	3.42	0.13	6.33
2-1	24000	512	385	0.63	12~1	12~4	12~4	3.57	0.18	5.90
3-1	24000	512	385	0.00	22~1	22~4	22~7	4.15	0.28	5.97
4-1	24000	1024	358	0.31	8~1	8~6	8~5	3.69	0.16	6.28
5-1	8000	512	128	0.00	6~3	16~7	10~5	4.76	0.25	6.71
6-1	24000	512	385	0.31	12~1	12~5	12~8	4.76	0.25	6.71
7-1	24000	512	385	0.97	22~3	18~9	14~7	3.88	0.24	5.89
8-1	24000	512	385	0.63	22~1	22~10	22~11	4.82	0.26	6.71
9-1	24000	512	385	0.63	16~1	16~7	16~5	4.54	0.28	6.35
10-1	24000	512	385	0.00	18~1	18~7	18~10	4.51	0.24	6.52
11-1	24000	512	385	0.97	22~1	22~9	22~9	5.11	0.25	7.11
11-1n	24000	1024	385	0.00	18~1	18~6	18~6	4.90	0.25	6.88
11-1N	8000	256	128	0.00	16~2	16~2	16~2	4.16	0.30	5.86
11-2	24000	1024	385	0.00	22~1	22~9	22~9	6.40	0.28	8.20
11-3	24000	512	512	0.00	14~1	14~4	14~5	4.88	0.19	7.22
11-4	24000	512	385	0.97	6~3	22~7	22~11	6.24	0.27	8.09
11-5	24000	512	385	0.97	8~1	16~5	20~7	5.70	0.34	7.26

Table 3.1: Optimal features computation setting based on the maximum quality rating Q_{cf} for 13 cases. From left to right each column shows user-gesture ID, sampling rate in Hz, window size in sample, step size in sample, pre-emphasis value, order and number of robust coefficients for LPC, MFCC and PLP, intrinsic dimensionality for the vocal-gestures, average RMD for the vocal-postures, and Q_{cf} value.

In the outer most loop of the search algorithm there is the sampling rate variation, and in the intrinsic dimensionality trend, displayed in Figure 3.15, we observe three distinct segments related to different sampling rates. In general the 8KHz sampling rate often results in a low Q_{cf_x}, while within the 16KHz and 24KHz the maxima are usually very close. In general 24KHz best case slightly outperforms the 16KHz best case, also because with the largest Nyquist frequency we can compute up to two more coefficients in the MFCC and PLP. However for the real use cases we did not notice noticeable usability differences between the two sampling rates. From the numerical results in Table 3.1 and from the analysis of partial results per feature group we observe that windows of about 21ms usually provide the best results. The pre-emphasis values are varied since enhancing higher frequencies helps to increase the intrinsic dimensionality, but it also leads to higher average RMD. The order of the features is strictly dependent on the user, and usually higher orders prevail. The total number of robust feature is usually in the range [10, 20] with MFCC and PLP contributing the most. The robust LPC coefficients are only one to three, and these show a very low correlation with the robust MFCC and PLP. Thus we still include the LPC because the contribution to the intrinsic dimensionality is

often significant. Low index LPC and MFCC coefficients are usually the more robust, while for PLP coefficients low average RMD are spread across the vector. The values of the intrinsic dimensionality for the vocal-gesture 1 across the 11 users are usually between 3.5 and 5, consistent with the findings of Togneri regarding spatial trajectories of the human voice (Togneri, Alder, and Attikiouzel, 1992). User 11 is more experienced with the system and generated a larger amount of training data. In the five vocal-gestures from user 11 the intrinsic dimensionality is usually slightly higher due to better familiarity with the preparation of the training data. Moreover the vocal-gestures with ID 2 to 5 are more elaborate in terms of timbral variation than gesture 1, which includes only vowel sounds.

The method presented here is designed to eliminate features that capture noisy aspect of the voice. External noise source captured from the microphone can cause larger dispersions to the some of computed coefficients. In real use scenario we suppose that the working environment is likely not quiet and at least feedback from the speakers is present at the microphone. In the Table 3.1 entry 11-1n and 11-1N the training data has additional background music with vocals and full ensemble instrument presence. In 11-1n we used two monitor speakers at distance 0.6m from the microphone and angles of 90° and 270°. The hyper-cardioid polar pattern of the dynamic microphone provides at those angles an attenuation of about 7dB. The average RMS acoustic output from the speakers was approximately 95dB SPL. This emulates a scenario worse than usual performance stage settings, in which monitors are at a longer distance and at an angle usually close to 180°, where the microphone attenuation is double. However in the recorded signal the level of the music was about 20dB below the voice. In 11-1N we mixed music artificially in the training sound files at 9dB below the voice peaks, a higher level representing borderline scenarios. In Figure 3.16 we show a detail of the spectrograms for the equivalent vocal-postures in 11-1n and 11-1N, in which the presence of a background music signal is clearly evident. In Figure 3.17 we present the quality ratings Q_{cf_x} relative to the two training data set with noise.

In 11-1n we observe only a small drop in the overall Q_{cf_x} trend. The best computational settings are similar to the 11-1 dataset, but the null pre-emphasis and the larger analysis window clearly help to mitigate the presence of fast changing and pulsating music. In 11-1N the background noise level is higher and as expected results are remarkably different. The quality Q_{cf_x} for every case is much lower and the number of robust features is low, only 6 satisfy the average RMD threshold condition over vocal-postures. The sampling rate of 8KHz produces better results

since the Nyquist frequency up to only 4KHz removes the higher band in which the loudness of the background music is equal to or higher than the voice volume. Further test on similar cases with high-level artificially added noise showed consistency. Therefore the method presented here can adapt the computing settings also to the environment conditions besides to the vocal tract characteristics of the user. The voice data we use to train the VCI4DMI for live performances is recorded with the presence of background sounds, as similar as possible to those generated in the performance.

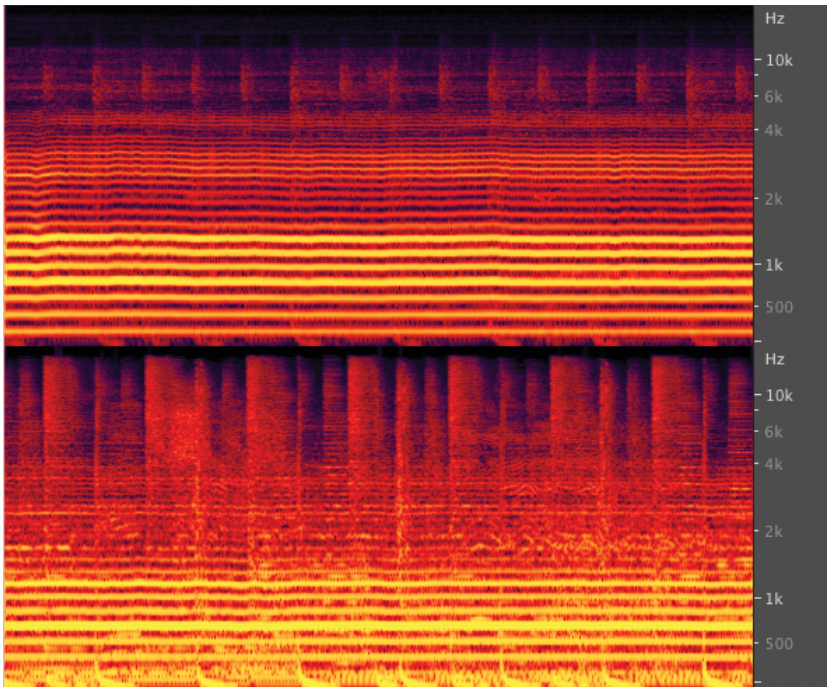

Figure 3.16: Spectrogram of examples of vocal-postures in training data set 11-1n (top) and 11-1N (bottom) with clearly visible background noise component.

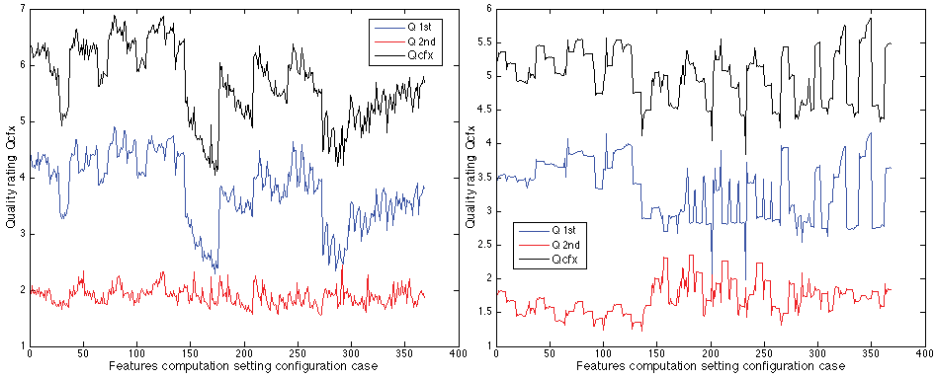

Figure 3.17: Quality ratings Q_{cfx} and contributing terms for 368 features computation setting cases on the training data set 11-1n and 11-1N with background noise. The Q_{cfx} maximum and thus the optimal features computation settings change with the increase of the background noise, showing the method adaptability to the sonic environmental conditions.

3.3 Self-organizing gestures

At this stage following after the method described above, feature vectors describe the instantaneous timbre of the voice in a high dimensional space. Noisy features have been already eliminated and the computation has been tuned to maximize the amount of information extracted from the voice. Any vocal-gesture $\mathbf{V_G} = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{W_G}]$, is therefore represented by a set of points bounded in an arbitrary convex shape with arbitrary distribution in the *gestural space*. This has high dimensionality equal to the number of robust coefficients C' in $\mathbf{v}$. The next step is for the GC to compute from the gesture a number, that we address with M, of intermediate continuous parameters in the *mapping space*, providing the essential representation of the musical control intention expressed in a voice timbre variation. This transformation must be compliant with the conditions over vocal-postures and vocal-gestures described in Section 3.2, demanding for steady and continuous GC outputs respectively. Thus we present next an adaptive method to model a GC from the training vocal-gestures.

In the literature we find three common strategies: direct mapping of all the low-level features to DMI parameters, mapping of a subset of them only, or PCA projection and direct mapping of a reduced number of principal components. The first approach is likely to provide a redundant control, and the second to present a prior removal of features that can be representative. The third considerably improves over the previous two, linearly combining the features to a reduced set, in order to map to the DMI only the most discriminative components found in the data. However the dynamics between input voice and GC output are generally not considered. A vocal-gesture consists of an arbitrary path between two coordinates in the high dimensional space. The proposed approaches simply map the coordinates to DMI parameters, without considering that in the multidimensional space the training data is likely to have non-uniform density in different regions. There is no guarantee that a perceptual linear glide between two timbres maps to a linear variation of the GC outputs, which we aim to address as well in our approach. Therefore the ideal GC transformation $g: \mathbf{R}^{C'} \rightarrow \mathbf{R}^{M}$ should preserve the manifold structure embedded in $\mathbf{V_G}$ while rearranging the samples of the training vocal-gestures in uniformly distributed regular shapes. Low density or empty sub-regions determine combinations of the GC output beyond the user's reach with vocal-gestures consistent with those provided for training.

3.3.1 Self-organizing maps as a gestural controller

To complete the VCI4DMI front-end processing implementing the GC, a non conventional application of the Kohonen SOM (Kohonen, 1982) appears to be an appropriate technique to find the desired transformation between the gestural data space and the mapping space $g: \mathbf{R}^{C'} \rightarrow \mathbf{R}^{M}$. The SOM is a particular ANN trained in an unsupervised fashion, and it is commonly used to generate a two- or three-dimensional representation, called a map or lattice, of high dimensional data. The SOM lattice is a M-dimensional hypercube composed by $(r_{SOM} \cdot M)$ output nodes O_k, each associated with a weight vector $\mathbf{w}_k = [w_1, w_2, \dots, w_{C'}]$, where r_{SOM} is the resolution of the SOM, defined as the number of nodes on each axis. The preservation of the topology of the input data is a peculiar characteristic of the SOM weights, which makes this technique suitable not only for recognition, but also for the organization and graphical display of large amounts of data. SOM finds application in the sound domain for classification, visualization, data mining and audio retrieval as presented in the survey of Ness and Tzanetakis (2009). The authors also include applications for instrument interfaces and interactive music systems. In these, the SOM is used to organize collection of sounds in a low dimensional map, which are then retrieved and reproduced in real-time by linear one-to-one mappings of sensor signals to the SOM lattice coordinates. The dimensionality of the map and sensor signals is always two except in (Odowichuk and Tzanetakis, 2012), which represents a rare case of a higher dimensional SOM application in this field. Besides dimensionality reduction and topology preservation, which implies local and global neighborhood preservation in the lower dimensional representation of the data, the SOM algorithm implicitly provides adaptation to the different densities that the data may have in different sub regions of the original data. In Figure 3.18 we show an example with the 2D PCA projection of the data and the super imposition of the 2D SOM lattice by plotting the weights of the output nodes and by connecting the neighbors. It is evident that the weights of the SOM output nodes follow the shape and topology of the most discriminant components of the training data, but these are also displaced according to the density of the data, which is far from being uniform. We propose to train the SOM with the data $\mathbf{V_G}$ and use the output lattice as a GC, as illustrated in Figure 3.19. When new high dimensional input vector $\mathbf{v_{new}} = [v_1, v_2, \dots, v_{C'}]$ of voice features is used to query the SOM, the response is the index of the node with the closest weight $\mathbf{w}_k$ to $\mathbf{v}_{new}$ in the C' Euclidean space. Normalizing the relative lattice indexes to the range [0, 1] we obtain the GC

intermediate output $\mathbf{gc_{out}} = [gc_1, gc_2, \ldots, gc_M]$ that can be used for mapping purposes, as described in Equation 3.19 and for the 2D case. In 3.19 $\|\ \ \|$ represents the Euclidean distance operator, i and j are the indexes of the 2D grid, and the number of nodes for each dimension r_{SOM} serves as the normalization factor. In general we associate to each output node O_k a M-dimensional normalized index vector $\mathbf{idx}_k$.

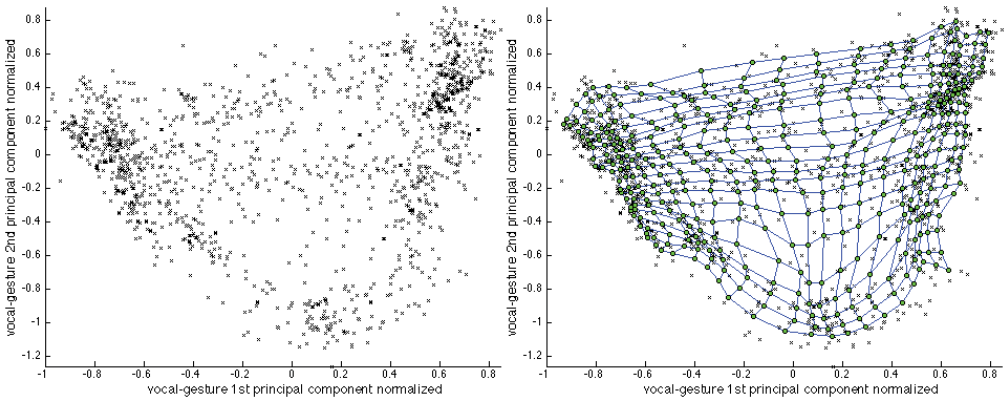

Figure 3.18: An example of SOM training data projected onto the first two principal components and normalized to the range [-1;+1] (left) and resulting trained SOM output lattice weights with neighborhood links superimposed to the training data (right).

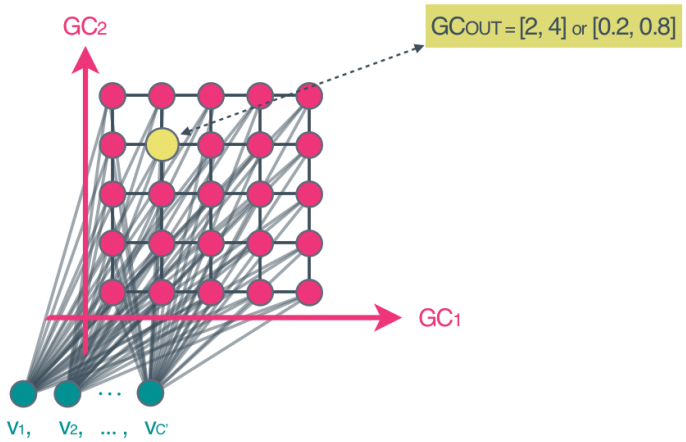

Figure 3.19: Illustration of the use of the SOM lattice as a Gestural Controller in 2D.

$$\mathbf{gc_{out}} \triangleq \frac{\arg\min_{i,j} \left\| \mathbf{w}_{O_{i,j}} - \mathbf{v_{new}} \right\|}{r_{SOM}} \tag{3.19}$$

The use of the SOM to implement a GC, in the way we described above, realizes the transformation $g: \mathbf{R}^{C'} \rightarrow \mathbf{R}^{M}$ we target because of the following characteristics:

- the number of GC output signals M is lower than the C' robust features computed from the voice, performing the necessary dimensionality reduction to extract non redundant control signal;
- the topology of the original input space is preserved so that a continuous trajectory in the higher dimensional vocal features space will be converted to discontinuity free GC output signals;
- the non-linear transformation of the dynamic range of the features to the GC output signal, estimated by the non-uniform density found in the training vocal-gestures, provides the necessary compression/expansion to obtain a pseudo-linear response to voice timbre variations.

3.3.1.1 SOM shortcomings

There are several intrinsic shortcoming in the standard SOM learning algorithm that need to be addressed here, because the outcome illustrated in Figure 3.18 and the simultaneous occurrence of all the three characteristics mentioned above are very unlikely in real scenarios. In a similar application for the remapping of voice to the timbre of instruments (Stowell and Plumbley, 2010) the SOM performance is considered unsatisfactory. The author describes the mapping result as inconsistent due to different input-output relationship obtained for every training instance with identical training set, and the control performances are poor and messy.

The SOM is an unsupervised ML technique, but the result of the training still depends on several user-defined settings: dimensionality and resolution of the output lattice and the initial and final values of the learning and attraction rates. The orientation of the lattice related to the data coordinate system is completely not determined and is likely to change at every training instance. Distortion in the global topology preservation such as folding, twisting, and discontinuities of the output lattice are common, as are failures in the local topology preservation. In Figure 3.20 we illustrate typical examples of SOM topology distortion. These characteristics of the standard SOM training method may be harmless for classification or recognition purposes, but can be lethal when the SOM lattice is used as a continuous, non-linear, and discretized representation of the training data set, as we aim in out application. In particular the topology distortions that frequently occur in the standard SOM cause:

- non monotonic GC outputs in response to monotonic trajectories in the gestural data space due to lattice twisting;

- discontinuities and jumps in the GC output due to lattice folding and edges curling, which place nodes nearby in the gestural space even if their relative SOM grid positions are far apart;
- minor inconsistency of the gesture to GC output relationship due to local topology distortion, determined by the wrong relative position of a single node in relation to its nearest neighbors.

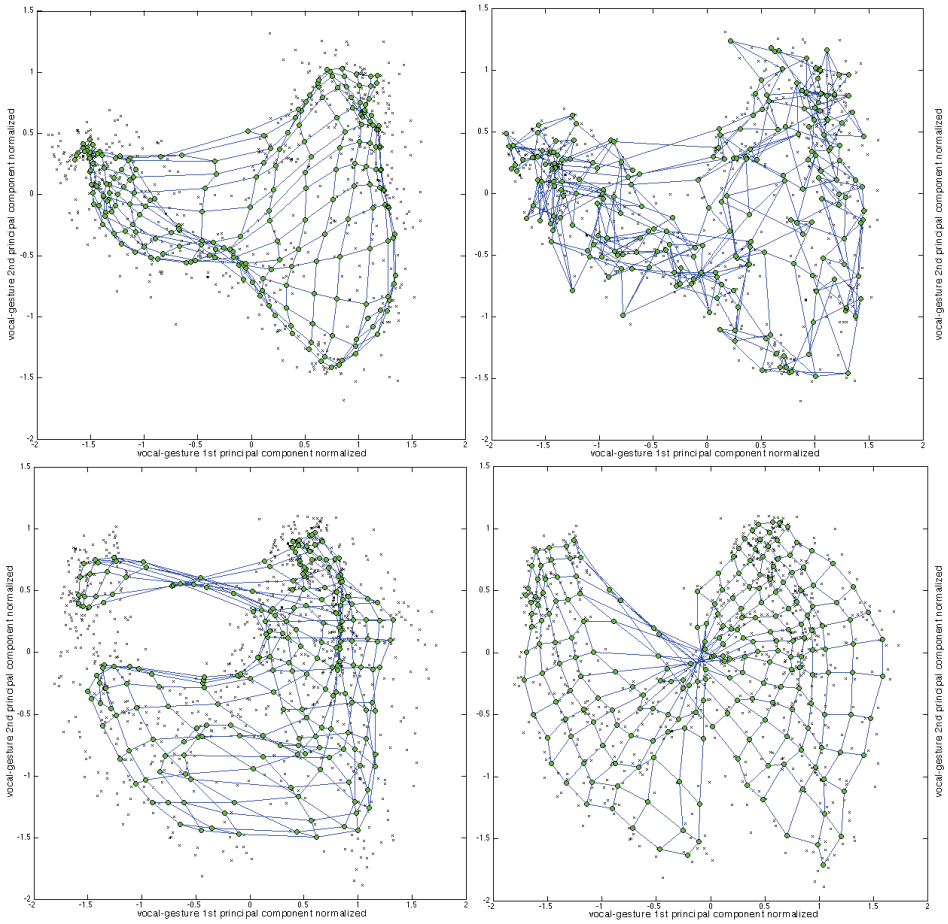

Figure 3.20: SOM output lattice topology distortion examples related to edge folding (top left), severe local distortions (top right), lattice twisting (bottom right), edge curling with folding (bottom left), displaying lattice weights superimposed to the training data normalized principal components.

Excessive or poor values of the learning and attraction rate chosen for the training are among the causes for topology distortion and poor model fitting. A low r_{SOM} results in a coarse representation of the data, while if too high local topology distortions are likely. Therefore there is a tradeoff between high-resolution and continuity of the SOM lattice. The output here is intrinsically discrete due to the finite number of nodes, therefore the concept of "continuity" has to be considered as mapping of

vectors close in the input manifold to the same or adjacent SOM nodes. One of the main causes of global topology distortions is the dualistic nature of the SOM, which simultaneously reduces the dimensionality and preserves the topology (Kiviluoto, 1996). The selection of the value of M considerably smaller than the embedded dimensionality forces the SOM algorithm to perform a severe dimensionality reduction task, which usually results in folding, twisting, or curling of the lattice.

The proposed SOM modified algorithm, that we call Self-Organizing Gestures and introduced in the next section, addresses drawbacks of the SOM that prevent the use of the lattice as a GC. Moreover since we aim to minimize the user interaction in the VCI4DMI setup, we derive the appropriate SOM settings directly from the training data that help to further reduce the probability of shortcomings appearing in the trained lattice. The SOM issues discussed in this section are also observed also in three or higher dimensional output lattices. In the chapter we will display mostly 2D cases because they are easier to visualize and interpret.

3.3.1.2 Self-organizing gesture training algorithm

Before training the SOM, we introduce a prior step to reduce the number of dimension in $\mathbf{V_G}$ to a number equal to the integer part of $M = \lfloor idim(\mathbf{V_G}) \rfloor$. We thus free the SOM from the dimensionality reduction task, which is one of the main sources of drawbacks of the SOM method used for GC, and we use it only to find the non-linear transformation between two iso-dimensional spaces. This approach is comparable to the isometric SOM (Guan, Feris, and Turk, 2006), which showed benefits in a different domain such as 3D hand posture estimation. A variety of methods have been proposed in the literature for dimensionality reduction, but a technique outperforms others only if specific characteristics are found in the data (Van Der Maaten, Postma, and Van Den Herik, 2009). In general advanced dimensionality reduction techniques provide only a small improvement over simple techniques such as PCA. Since features vectors are computed from the voice every 16ms, we observe that gestures are likely to produce manifold, or embedded shapes in the high dimensional space. To make up for the discarded temporal information in $\mathbf{V_G}$, we want to preserve at least locally and globally the pairwise distances between the original space and lower dimensional space, measuring the distance along the manifold. The orthogonal linear transformation offered by PCA in the presence of an embedded manifold corrupts the original data organization if the dimensionality reduction is excessive. Therefore we chose the Non-Linear Dimensionality Reduction (NLDR) Isomap (Tenenbaum, Silva, and Langford, 2000) which is a classical Multi

Dimensional Scaling (MDS) that uses the geodesic distance instead of the Euclidean. The low-dimensional Isomap data reconstruction respects the geodesic distance, which is measured along the manifold, of the original data, and the technique can discover manifolds of any dimensionality. Moreover for Euclidean manifolds the Isomap algorithm ensures asymptotic convergence to the real structure of the data. As in most of the dimensionality reduction techniques, also Isomap ranks the components of the lower dimensional reconstruction by their variability or quantity of discriminative information. Effective dimensionality reduction was also obtained with the Locally Linear Embedding (LLE) technique (Roweis and Saul, 2000), which has a lower computational complexity, but it presents drawbacks with low amount of training data, because it requires high density on the embedded manifold. After the NLDR the training data $\mathbf{V_G}^*$, with M dimensional entries $\mathbf{v}^* = [v_1, v_2, \dots, v_M]$, are centered to the axis origin and normalized to the range [-1, +1].

The dimensionality of the SOM is set to M and we relate the resolution and the number of training iterations to the number of elements in $\mathbf{V_G}^*$, which is equal to the total number of analysis windows of the vocal-gestures W_G, as in Equations 3.20 and 3.21. The operator in $\lfloor\ \rceil$ represents the nearest integer. The value of r_{SOM} determines the complexity of the model we want to estimate therefore and in 3.20 we relate it to the size of the dataset. The more vocal-gestures we have the higher is the complexity of the model we can estimate without under-fitting issues. The relationship is logarithmic with base M, which implies that a 3D SOM requires a larger set of training data to increase the r_{SOM} compared to a 2D SOM, although the resulting total number of nodes is higher. The factor 1.5 was derived empirically from experiments on the real vocal-gesture database in order to minimize topology distortions. Similarly the number of training iterations t_{max} in 3.21 is related linearly to all the quantities involved in the SOM settings, determining longer training for more complex models.

$$r_{SOM} = \lfloor 1.5 \cdot \log_M(W_G) \rceil \tag{3.20}$$

$$t_{max} = r_{SOM} \cdot W_G \cdot M \tag{3.21}$$

The output lattice is composed of ${r_{SOM}}^M$ output nodes O_k with weights $\mathbf{w}_k$, organized in a uniform grid with 2^M vertices. These are represented by values of the $\mathbf{gc_{out}}$ in which each component is the minimum or maximum value, 0 and 1 respectively in this case. Despite the specific DMI mapping, these particular $\mathbf{gc_{out}}$

will coincide with some case specific boundary of the DMI control. In order to provide a logical and natural vocal control we associate the $\mathbf{gc_{out}}$ of the vertices to the 2^M gestural extrema a_{xt} that better encompass the specific reduced vocal gestural space $\mathbf{V_G}^*$. These are likely to coincide with the most diverse vocal timbre presented to the system for training. We define the 2^M extrema a_{xt} as those points in $\mathbf{V_G}^*$ that maximize the sum of their inter-distances and the sum of their distances from the origin in Equation 3.22.

$$\arg \max_{\{a_{xt-1},\dots,a_{xt-2M}\}} \left(\frac{\sum_{k=1}^{2^M} \sum_{z=1}^{2^M} \|a_k - a_z\|}{2} + \sum_{k=1}^{2^M} \|a_k\| \right) \qquad (3.22)$$

The algorithm that searches for the 2^M extrema is implemented with the following steps:

1. for each quadrant, set equal absolute value boundaries orthogonal to each axis, and progressively reduce the quadrant extension until only one point a_{xt} lays in each reduced quadrant;
2. compute and store the sum of their inter-distances and the sum of their distances from the origin for the obtained 2^M extrema a_{xt};
3. rotate at fine angular steps the data around the origin and repeat 1 and 2, until the full data rotation is completed;
4. pick the rotation angle α_{opt} that maximizes the sum of distances, apply the rotation to $\mathbf{V_G}^*$ and store the obtained 2^M extrema a_{xt}.

In rare cases no points fall into a quadrant and thus we set the relative a_{xt} to the origin. After finding the extrema we determine the convex hull that includes all the gestural training samples in $\mathbf{V_G}^*$. In Figure 3.21 we show two examples of gestural training data before and the α_{opt}, highlighting the detected extrema in the four quadrants, and the convex shape bounding the training data. The example on top can be interpreted easily due to a lower number of training samples.

The $\mathbf{w}_k$ relative to the 2^M lattice vertices are initialized at the position of the extrema a_{xt}. For the remaining $\mathbf{w}_k$ the initialization has little or no effect on the final result, so we initialize them in a uniform grid within the gestural data to express graphically the effect we achieve with the proposed modified algorithm. The SOM is trained for t_{max} iterations selecting a random $\mathbf{v_{rand}}^*$ from $\mathbf{V_G}^*$ and updating all the weights $\mathbf{w}_k$ with the standard rule described by Equations 3.23 and 3.24.

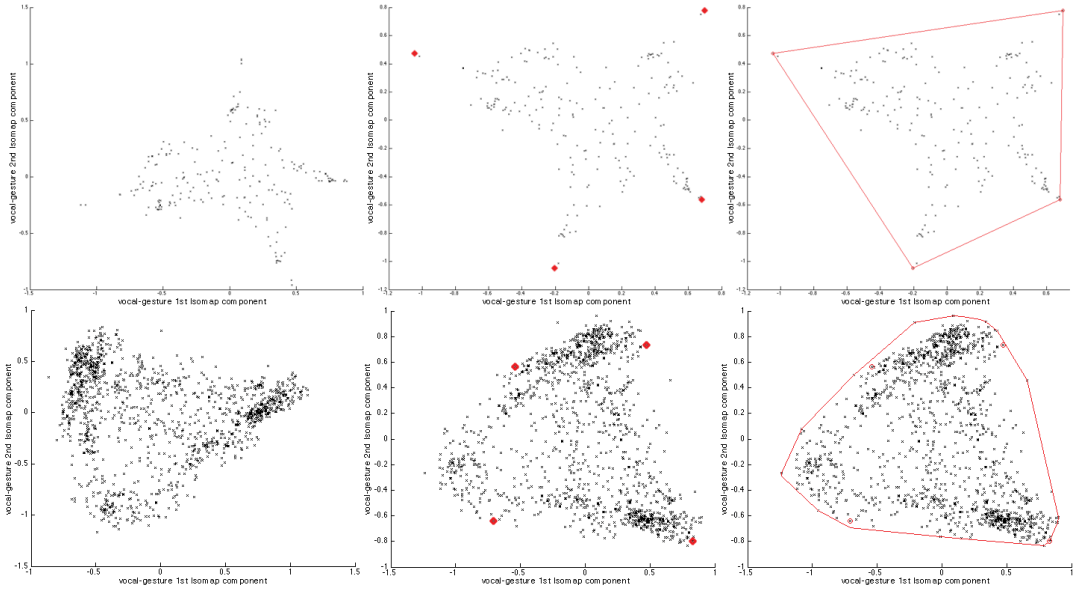

Figure 3.21: Examples of vocal-gesture training data after Isomap dimensionality reduction (left), rotation and extrema detection (center), bounding convex hull (right).

$$\mathbf{w}_k(t+1) = \mathbf{w}_k(t) + \mu(t) \cdot \Theta(t,k,z) \cdot \left(\mathbf{v_{rand}}^* - \mathbf{w}_k(t)\right) \tag{3.23}$$

$$\Theta(t,k,z) = \exp\left(\frac{-\|O_z - O_k\|^2}{2 \cdot \sigma(t)^2}\right) \tag{3.24}$$

In 3.23 and 3.24 $\mu(t)$ and $\sigma(t)$ are the learning rate and neighborhood parameter rate (or attraction rate) that decrease linearly with the time, $\Theta(t,k,z)$ is the neighborhood function and z is the index of the node O_z with weights $\mathbf{w}_z$ closest to $\mathbf{v_{rand}}^*$. The numerator of the exponential in the neighborhood function depends on the squared Euclidean distance between O_z and O_k in the SOM lattice output grid. We modify the standard training process by:

- applying a slower update for the $\mathbf{w}_k$ of the 2^M vertices using the half of the learning rate $\mu(t)$;
- forcing additional 2^M weight updates iteration every time a tenth of t_{max} has elapsed, using the $\mathbf{v}^*$ associated with the extrema a_{xt} instead of selecting random points, using $\mu(t)$ for all weights update, and half of the attraction rate $\sigma(t)$ only for the $\mathbf{w}_k$ of the 2^M vertices.

In addition at the end of the training process we associate to each node O_k the number U_k, that we call mass, which is the count of the entries in $\mathbf{V_G}^*$ to which $\mathbf{w}_k$ is the closer weight vector. To avoid local topology distortions we select a high number of

training iterations t_{max} by defining 3.21 in order to proceed with small updates. Therefore we can set an initial learning rate $\mu(t_0)$ relatively small, and a final attraction rate $\sigma(t_{max})$ relatively high, which prevent an excessive drifting of the node weights, and a value of $\mu(t_{max})$ at least one order of magnitude smaller than the initial. On the database we used for testing the best results were obtained with the values $\mu = [0.5, 0.01]$ and $\sigma = [1.5, 0.5]$. The modification of the SOM training we presented here aims to avoid topology distortions by applying forces at regular intervals that pull the SOM lattice vertices towards the extrema, as illustrated in Figure 3.22. This avoids the lattice getting folded, twisted, or curled, and at the same time provides a better overlap between the gestural data and the output node weights, which always fall within the bounding convex hull. Moreover the lattice vertices are stretched toward the gestural extrema maintaining the desired association between gesture and GC output at the boundaries. In Figure 3.23 there are two 2D and 3D examples of SOM training using the described method. In particular we show the weight initialization, the SOM at end of the training with weights linked to the $2M$ von Neumann neighbors, to the $(3^M - 1)$ Moore neighbors, and with the U_k mapped to the weights diameter. At the end of the training procedure if any topology distortion is detected in the output lattice, with the measurement described in Section 3.3.1.4, the training procedure is repeated and eventually the value of r_{SOM} dropped.

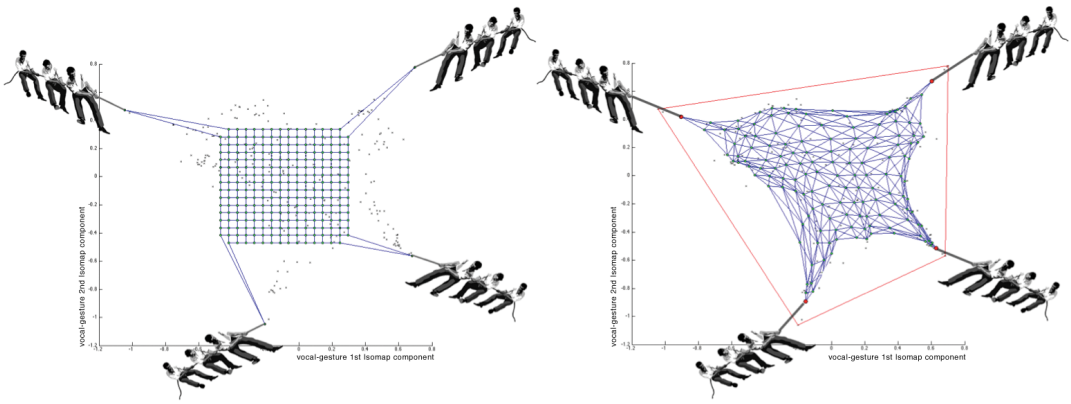

Figure 3.22: SOM lattice weights initialization with the vertices at the gestural extrema position (left) and after the training (right), with illustration of the pulling forces implemented by the modified training algorithm.

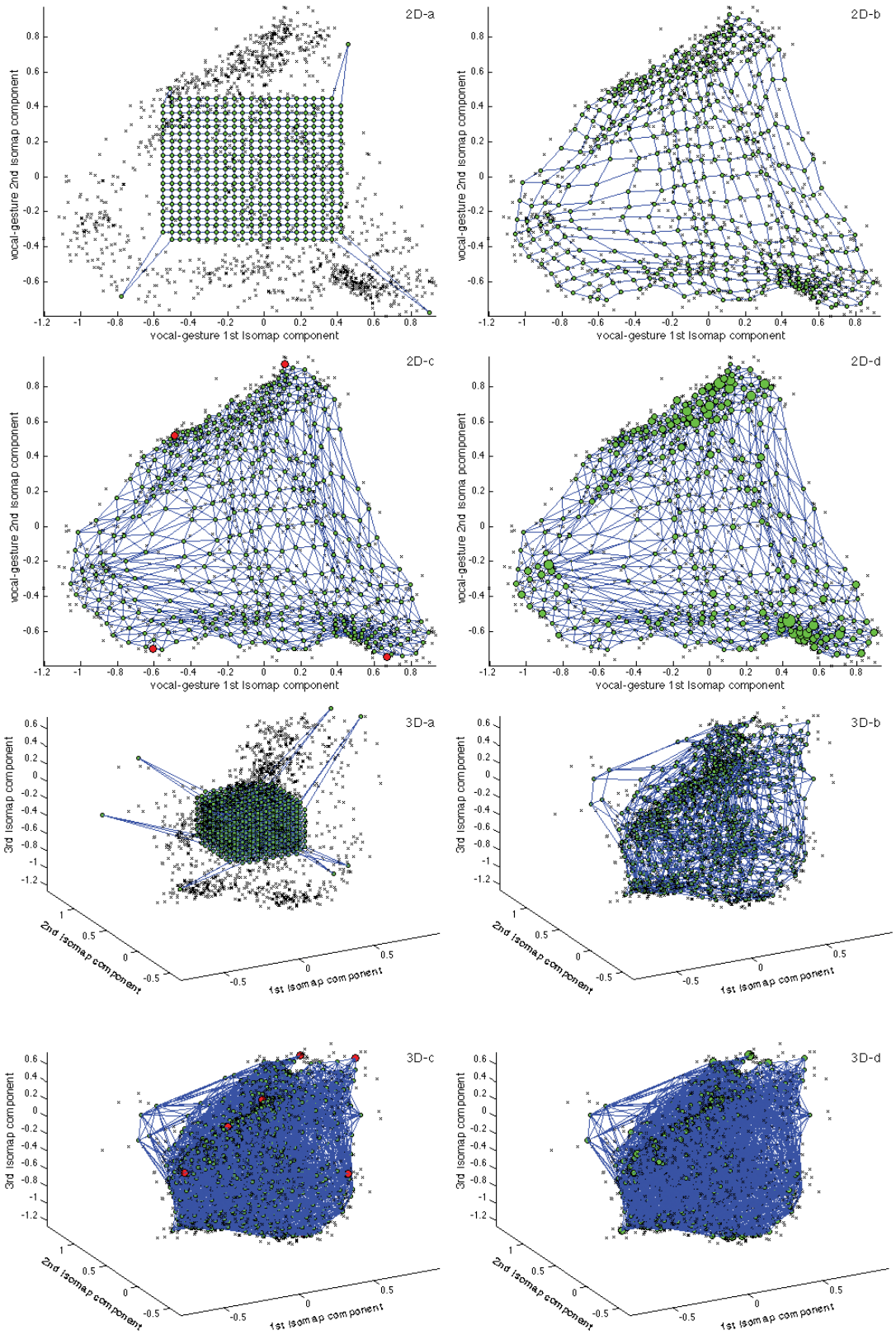

Figure 3.23: 2D and 3D examples of SOM with weight initialized (a), weights after proposed training with links on von Neumann neighbors (b), with links on Moore neighbors (c), and with node mass mapped on the weight diameter (d).

3.3.1.3 Operational modes

Here we describe the VCI4DMI interface component that implements the runtime GC using the gesture model derived from the Self-Organizing Gestures (SOG) training described above. The microphone signal is sampled and the stream is chunked into a sequence of overlapping windows. For each window a new set of M intermediate parameters $\mathbf{gc}_{\boldsymbol{out}}$ is computed as follow:

1. calculate the vector $\mathbf{v}$ with C features using the optimal settings;
2. apply the normalization factors to the different feature groups;
3. reject the noisy coefficients keeping in $\mathbf{v}$ only the C' robust features;
4. perform the Isomap NLDR found from the $\mathbf{V}_{\boldsymbol{G}}$ to obtain $\mathbf{v}^*$;
5. normalize $\mathbf{v}^*$ with the offset and scaling coefficient found from $\mathbf{V}_{\boldsymbol{G}}{}^*$;
6. rotate $\mathbf{v}^*$ by the angle α_{opt};
7. verify if $\mathbf{v}^*$ is inside the convex hull bounding all $\mathbf{V}_{\boldsymbol{G}}{}^*$ entries;
8. query the SOM output lattice with the new vocal-gesture sample $\mathbf{v}^*$ and get the related $\mathbf{gc}_{\boldsymbol{out}}$ response**.**

There are several modes to implement steps 7 and 8. We provide more choices that may fit different gestural control requirements for the VCI4DMI or for different musical interfaces. Step 8 can be conditioned to step 7, executing it only if the $\mathbf{v}^*$ is inside the convex hull. This can prevent, but it does not always guarantee, that voice timbres not included in the training set generate outputs, which can still fall somewhere within the region enclosing all the training gestural data. Another option we provide is the orthogonal projection of $\mathbf{v}^*$ to the convex hull in case this falls outside the boundaries, and then always proceed with step 8. As an alternative we can completely skip step 7. By default in the VCI4DMI we skip step 8 if the condition in 7 is not verified, but in real performances we obtained interesting interactions also with the projection mode, because it makes easier to obtain $\mathbf{gc}_{\boldsymbol{out}}$ on the lattice borders that, as we will see later, determine different timbre extrema at the output of the controlled DMI.

In step 8, the search on the SOM lattice that translates a vector $\mathbf{v}^*$ into a vector $\mathbf{gc}_{\boldsymbol{out}}$ requires the definition of a search metric, a search area and a strategy to generate the output. The Euclidean distance between the data vector and the node weights $\|\mathbf{v}^* - \mathbf{w}_k\|$ is the default the search metric, also because it is used as well to in the training phase. The output node O_k is usually the one that minimizes this distance. For a typical recognition or classification task, search can be through the

whole lattice but in the VCI4DMI GC we deal with a temporal sequence of events, and therefore we prefer to limit the search to the $(3^M - 1)$ Moore neighborhood nodes of the output O_k at the previous iteration. This approach presents two advantages: it reduces the computational load of the search, which becomes independent from the lattice size, and provides smoother transition in the GC output signals. In Figure 3.24 we illustrate this approach. The green circle is the output node at the previous iteration, the red circles represent the Moore neighborhood search space, and the smaller yellow circle is the $\mathbf{v}^*$ relative position at current and future iterations. The normalized grid indexes $\mathbf{idx}_k$, which is the GC output, of the node O_k closest to $\mathbf{v}^*$ will present a steep instantaneous transition if the search space includes the whole lattice. Instead shrinking the search to the Moore neighborhood we obtain a gradual and continuous variation of the $\mathbf{gc}_{\boldsymbol{out}}$, which moves towards $\mathbf{v}^*$ in 5 iterations along the path identified by the blue circles, generating 5 intermediate outputs compliant with the dynamics found in the training data. This in general does not introduce a delay between gesture and system response because the iteration rate is equal to 62.5Hz by default, and the voice timbre variation is usually slower. However for a SOM with high-resolution r_{SOM} it is possible to increase the rate up to 250Hz in the current implementation, which is presented in Chapter 6. This approach helps to cope with possible unwanted noise captured by the microphone, which may otherwise generate instantaneous $\mathbf{v}^*$ far from the path of the current vocal-gesture trajectory, because here we implicitly define for every node of the lattice a maximum delta in $\mathbf{gc}_{\boldsymbol{out}}$ values.

The SOM output lattice is a discrete grid and the finite value of the resolution implies in turn a limited resolution in the GC output signals too. Therefore the $\mathbf{gc}_{\boldsymbol{out}}$ vector can assume a number of unique values equal to the number of output nodes O_k, which may be small for real-valued DMI parameters mapping purposes. Moreover the limited r_{SOM} may determine a critical amplification of the vocal-posture noise in the GC out. If a $\mathbf{v}^*$ falls between two or more $\mathbf{w}_k$ the closest output node may change continuously for minimal $\mathbf{v}^*$ beyond the performer's control, generating output signal fluctuations greater than what is being produced at the input. Therefore to avoid the two issues mentioned here we interpolate to obtain continuous output using the Inverse Distance Weighting (IDW) technique in Equation 3.25 and 3.26, in which ρ is the interpolation power parameter we usually set equal to 3.

$$\mathbf{gc}_{\boldsymbol{out}} = \frac{\sum_{i=1}^{3^M} \mathbf{q}_i(\mathbf{v}^*) \circ \mathbf{idx}_i}{\sum_{i=1}^{3^M} \mathbf{q}_i(\mathbf{v}^*)} \tag{3.25}$$

$$\mathbf{q}_i(\mathbf{v}^*) = \frac{1}{\|\mathbf{v}^* - \mathbf{w}_k\|^\rho} \tag{3.26}$$

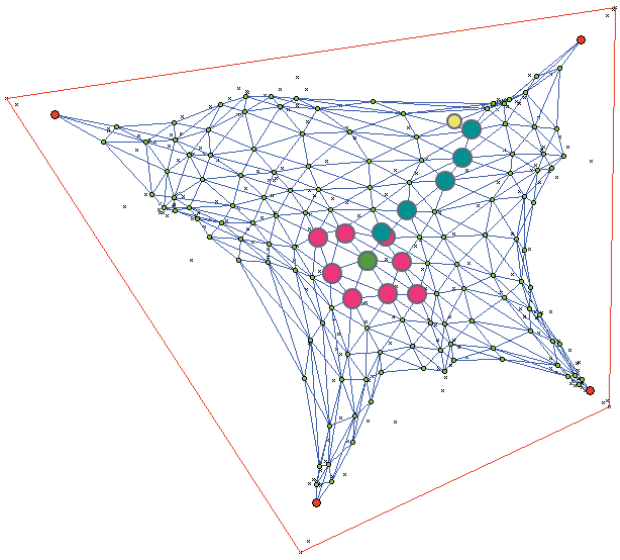

Figure 3.24: Illustration of gradual gc_{out} progression (blue) from the initial position (green) towards the target position (yellow) limiting at each iteration the search space to the Moore neighborhood (red).

If we exploit the mass of the nodes U_k that we computed at the end of the training procedure we can use as search metric the gravitational attraction force instead of the Euclidean distance, in Equation 3.27. Therefore the closer O_k would be the node producing the strongest attraction at position $\mathbf{v}^*$, and the same can be applied in case we use the IDW. The gravitational attraction depends on the Euclidean distance, decreasing with its square, and by the gravitational constant g_{const}. This search metric drastically changes the GC response to the gesture. In this situation Menzies (2002) argues that the interface main task, which behaves as a physical dynamic system, is a "dynamic control processing" rather than just an "instantaneous mapping". The attraction force as search metric perturbs the gestural space creating hills where the mass is bigger, and valleys where the mass is lower, so that a different amount and perpetuation of gestural energy is required to change $\mathbf{gc_{out}}$. The constant g_{const} can be varied to change the dynamic response of the GC and, if negative, it provides a symmetric behavior, with repulsion forces rather than attraction. Theoretically with high r_{SOM} all the U_k should be equal, but this greatly increases the probability of getting topology distortions in the SOM output lattice.

$$F_k(\mathbf{v}^*) = \frac{g_{const} \cdot U_k}{\|\mathbf{v}^* - \mathbf{w}_k\|^2} \tag{3.27}$$

3.3.1.4 Results

In this section we present the results that measure the effectiveness of the algorithm we introduced to address the drawbacks when a SOM is used as a GC. A user based evaluation of the proposed voice GC versus the design principles and requirements is presented in Section 3.4. Here we focus on the local topology distortion, global topology distortions, spread of the node masses, and training repeatability. We compare the result with those obtained with the standard SOM training algorithm, using the same training data and settings such as lattice resolution and dimensionality, number of training iterations, learning and attraction rate.

To detect local topology distortions, for every node we compute the difference of the $\mathbf{w}_k$ with the $\mathbf{w}_k$ of the $(3^M - 1)$ immediate neighbors. We compute the difference also between the relative pairs of grid index. Then for each pair O_k-neighbor compare the sign of the M components of the two differences. If all M signs are different in at least one pair, we detect a topology distortion at the node O_k. We further analyze the cases in which the number of local distortions is equal to the total number of nodes, because this is usually due only to a relative rotation of the $\mathbf{w}_k$ grid in relation to the index axis grid, which may happen only with the standard training. When two or more local topology distortions are adjacent we detect a global topology distortion. Edges excessively curled may place the weights of two vertices dangerously close. This can cause discontinuities in the GC output and should be considered as a distortion. Therefore we measure the angle of every triplet of nodes along each edge, if the maximum and the mean are higher than 90° and 45° in 2D and 60° and 110° in 3D we detect a global topology distortion.

The proposed training procedure chooses a random $\mathbf{v}^*$ point at every iteration to update the $\mathbf{w}_k$, as the original algorithm. SOM lattices trained with the same data are thus likely to be different. Therefore two GC trained with identical or similar gestural data are likely to produce very different responses. Our method aims to minimize this drawback too, and thus in the result we measure the training repeatability. This is computed as the mean of the distances between the $\mathbf{w}_k$ of nodes O_k with the same grid position that we get for two consecutive training iterations. Finally we measure the variance of the masses U_k, which reveals the capability of the SOM output lattice to adapt to the local density of the training data. The results presented in Table 3.2 are the average values obtained on 26 test cases of real vocal-gesture training data. For each training data set we repeated the same measure 10 times. Moreover these measurements were repeated for the 2D and 3D cases, which are presented separately

in the table. For the topology distortions we report the percentage of trained output lattice affected by at least one distortion.

Lattice Measure 2D cases avg	*SOM*	*SOG*	*Δ*
Local Distortion	37.96%	0.05%	-37.96
Global Distortion	78.07%	3.07%	-75
Repreatability	0.73	0.014	-0.71
Mass Variance	4.21	2.76	-1.45

Lattice Measure 3D cases avg	*SOM*	*SOG*	*Δ*
Local Distortion	20.76%	0.70%	-20.06
Global Distortion	54.23%	2.35%	-51.95
Repreatability	1.09	0.126	-0.96
Mass Variance	3.82	1.38	-2.44

Table 3.2: SOM versus SOG local distortion, global distortion, repeatability, mass variance measurement comparison, averaged over 2D and 3D test cases.

The results in Table 3.2 demonstrate that the SOG method we propose outperforms the standard SOM at least in the four aspects we report here, presenting negative deltas in every case. The drop in the local and global distortion percentage is drastic. For those few cases in which we get a distortion in the output lattice with our method, repeating the training likely provides a distortion free output. About half of the global distortions are due to curled edges and the other half is still mostly due to problems at the peripheral area of the lattice. For the standard SOM algorithm we generally do not get a distortion-free lattice because either one local or global distortion are present. The significant drop and the small absolute value of the repeatability measure indicate a consistent strategy to implement GC from training data. The mass variance measurement shows some considerable improvement too. Topology error and mass variance are generally lower in the 3D case because the lattice has a higher number of nodes, and thus can better accommodate the adaptation to the training data local shape and density. In the table the results are presented only for the 2D and 3D cases. The reasons for limiting the dimension M to a maximum of three are related to cognitive overload and user feedback issue, and will be discussed in Chapter 6.

3.3.2 Training data pre-filtering

Two issues related to the content of the vocal-gesture training data $\mathbf{V_G}$ were noticed during the training and measurement. Therefore we introduce two preliminary stages

of analysis and filtering in order to avoid these sources of potential corruption of the trained GC.

First, entries not representative of any voice timbre can fall in $\mathbf{V_G}$ if the vocal-gesture sound file is poorly edited, cut, or if it contains silence. These entries often represent outliers far from the vocal-gesture region. At first the outliers determine a widening of the convex hull bounding the gestural data, and then they may be identified as gestural extrema, having a clear impact on SOM lattice training. The GC trained with outliers in $\mathbf{V_G}$ is likely to present regions of the SOM grid difficult or impossible to reach, and thus is less representative of the performer's musical control intention. For each C' dimension in $\mathbf{V_G}$ we measure the range between the 25th and 75th quartile and each entry $\mathbf{v}^*$ beyond the double of this range is marked as an outlier and removed from the training data.

Secondly, vocal-gestural training data may present unintended vocal-postures that can bias the trained GC. When producing the gestural training data the performer often fails to vary continuously the voice timbre introducing steady pauses. This in turn causes static vocal-posture segments arising within the dynamic vocal-gestures. We observed this more frequently with novice users less familiar with the abstraction of continuous timbre variation. For example in repeated instances of a basic vocal-gesture just involving the gliding between two vowels, the steady utterance one of the vowels often represents a conspicuous amount of time. Figure 3.25 shows the spectrogram of an excerpt of a voice-gesture, in which is evident that between 4.0 and 5.5 seconds, and between 6.9 and 7.9 seconds the voice timbre does not change over time. The presence of these undesired pauses can drastically increase the density of the $\mathbf{V_G}$ data in proximity of some vocal-postures. Since the local data density in $\mathbf{V_G}$ implicitly determines the dynamic between GC input and output, the presence of postures in this context is highly detrimental, and is another possible cause leading to a GC less representative of the performer's control intention. Moreover excessively dense sub-regions increase the risk of getting local topology distortion of the SOM lattice. A proper pre-processing of the training data affected by this issue is essential. Accurate manual editing is the best option but is tedious, very time consuming, and thus not inline with the aim of minimizing the user interaction in the VCI4DMI system setup. Furthermore inaccurate cuts of the sound file can generate outliers.

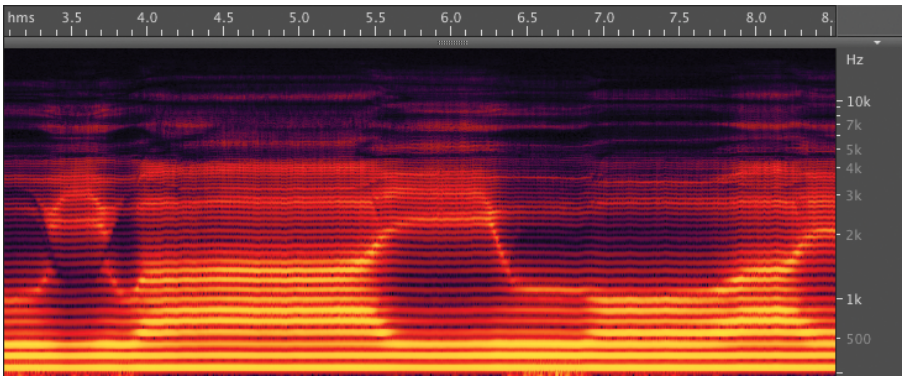

Figure 3.25: Spectrogram of an excerpt of voice-gesture that includes frequent timbre variation pauses.

Therefore we implemented the following procedure to optionally pre-filter the data in $\mathbf{V_G}$ automatically. We suppose that each unique unwanted vocal-posture determines a cluster in $\mathbf{V_G}$. Their number can be provided by the user or derived finding the number of clusters that maximize the average similarity of every data point with points in its own cluster compared to points from other clusters, as proposed in the silhouette method (Rousseeuw, 1987). Afterwards we partition the data with the k-means clustering algorithm (Seber, 1984), and we filter the data with the algorithm described in the pseudocode of Figure 3.26.

```
initialize distance_threshold at high value
FOR each cluster
      WHILE normalized average NN distance bigger than treshold
            FOR every consecutive pair of entry
                  IF distance less than distance_threshold
                        remove the second from the dataset
                  END
            END
            reduce distance_threshold
      END
END
```

Figure 3.26: Vocal gesture training data pre-filtering algorithm pseudocode.

The distance threshold is lowered at fine steps and the filtering is terminated when the normalized average nearest-neighbor distance in the cluster is below a user-defined threshold. This time we use temporal information of the training data, measuring the distance only between pairs that are consecutives. Depending on the severity of the vocal-postures presence in $\mathbf{V_G}$ the threshold value can be tuned in the range [0.15, 0.25], which provided a good pre-filtering on our database. When the number of clusters N_{cl} is detected automatically threshold is set to $((1/N_{cl}) - (1/5(N_{cl} - 1)))$. The two methods to pre-filter the data are applied on the high dimensional data $\mathbf{V_G}$

after noisy features are removed but before the Isomap NLDR. The pre-filtering was already included in all results and measurements presented in previous sections. In Figure 3.27 we show two example of pre-filtering by cluster density reduction, showing the original data, the filtered data, and the actual final training data. In order to visualize the data we performed the 2D reduction by Isomap. The two vocal-gestures are similar but from two different users. In the example on the top row the number of clusters is equal to four and entered manually, while on the bottom one the three clusters are automatically detected and set to three. It is evident how the discarded data helps to reduce the density in proximity of the center of the clusters, which is high due to the presence of the unwanted postures in the gestural data.

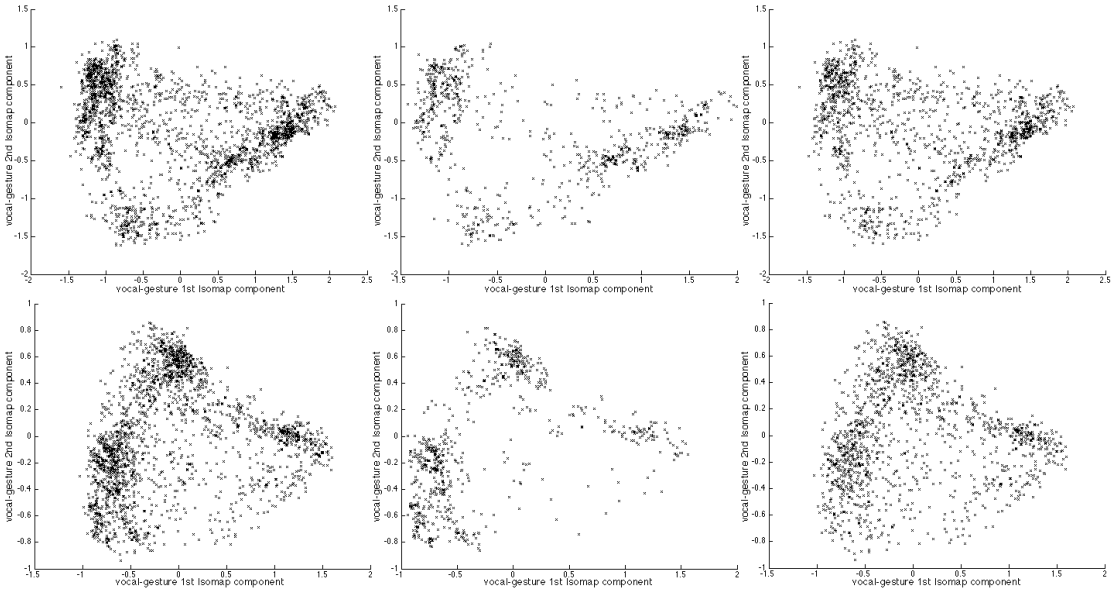

Figure 3.27: Examples vocal-gesture data pre-filtering by cluster density reduction displaying the original data (left), rejected data (center), and final training data (right).

3.3.3 Semi-supervised variant

In Section 3.3.1.2 we discussed and illustrated the importance of the gestural extrema and the related SOM lattice vertices in the GC component of the VCI4DMI. Since the training procedure is unsupervised, the user, before using the GC, has no control or knowledge about which timbre will be mapped on those key positions. Here we propose a variant of the GC training that directly allows defining the interaction required to generate the apexes of the $\mathbf{gc}_{\boldsymbol{out}}$ mapping space. The only additional inputs that this method needs from the user are the 2^M vocal-postures $\mathbf{V}_{\boldsymbol{Pxt}}$ centered in $\mathbf{v}_{Pxt}$, that will be associated with the vertices of the SOM output lattice. Unless the user has an unusual accurate control over the voice and is very experienced with the

system there is no guarantee that after the Isomap NLDR the 2^M postures will be placed well apart near the edges of the $\mathbf{V_G}^*$. Therefore even if we associate the a_{xt} with the $\mathbf{v}_{Pxt}$, SOM lattice is likely to be extended over and beyond the postures centers during the training. To ease the placement of the $\mathbf{v}_{Pxt}$ near the $\mathbf{V_G}^*$ boundaries we replace Isomap dimensionality reduction stage with a multiclass Linear Discriminant Analysis (LDA) (Rao, 1948). This supervised technique, similarly to the standard LDA, aim to minimize the "within class" variability and maximize the "between classes" variability. In the lower dimensional space data with the same label will present a tighter spatial grouping, while those with different labels will be pushed apart. The dimensionality reduction transformation, which is linear in this case, is learnt only from the 2^M $\mathbf{V_{Pxt}}$, properly labeled, and then applied to the gestural data $\mathbf{V_G}$. The gestural extrema a_{xt} are associated with the $\mathbf{v}_{Pxt}$, but we still search for the rotation angle α_{opt} that places them in the optimal position with respect to the axis. Also in this case there is no prior knowledge about which $\mathbf{v}_{Pxt}$ will be related to which SOM vertex, nor can the user specify it. The multiclass LDA aims to place the $\mathbf{v}_{Pxt}$ apart near the borders of the $\mathbf{V_G}^*$ and, even if it generally performs this task better than Isomap, there is still no guarantee about the right placement. This is due to possible inconsistencies or incompatibilities between vocal-gestures and vocal-postures provided by the user. Therefore before the SOM training we remove the $\mathbf{V_G}^*$ entries outside the convex hull determined by the $\mathbf{v}_{Pxt}$, allowing a small tolerance to prevent drastic reduction in the number of training entries. On the left plot in Figure 3.28 we observe that with Isomap NDLR the $\mathbf{v}_{Pxt}$, represented by the red circles, are not well spread apart. On the right plot we can appreciate a better separation and placement of the $\mathbf{v}_{Pxt}$ determined by the multiclass LDA dimensionality reduction, and only the entries in blue are considered for the following SOM training.

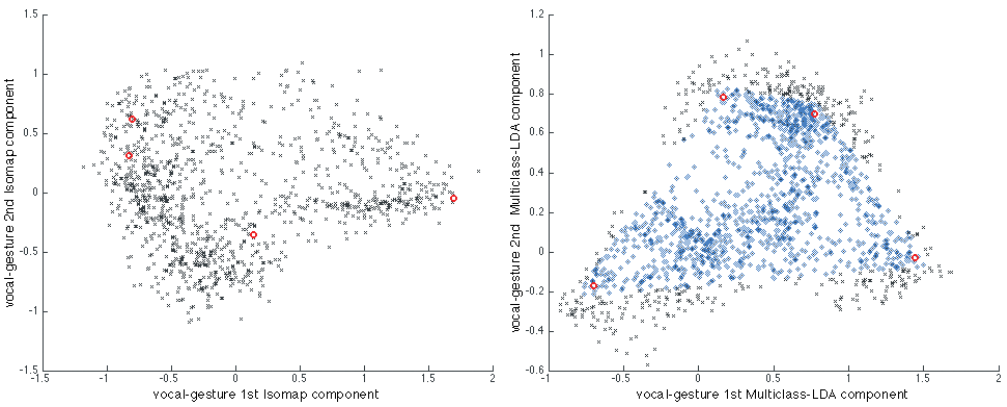

Figure 3.28: Isomap (left) and multiclass LDA (right) v_{Pix} displacement.

3.4 Evaluation and validation

To validate the proposed SOG vocal GC method we present next a performance comparison with other gestural controllers implemented with different techniques, similar to those introduced in the related work. In particular we measure and compare the independence, continuity, GC output space spread and coverage of the $\mathbf{gc}_{\boldsymbol{out}}$ signals computed from real cases of vocal-gestures. The stability, or invariance, of the $\mathbf{gc}_{\boldsymbol{out}}$ is compared over sets of real vocal-postures, measuring the standard deviation of the output signals. The output signal continuity is estimated measuring the average Euclidean distance in the GC output space determined by consecutive voice frames. We partition the GC output space into 256 and 512 discrete sub-regions for the 2D and 3D cases respectively, and for each we track the output count over vocal-gestures. For the space coverage we consider the percentage sub-regions with non-zero count, while for the space spread we compute the standard deviation of the output counts. As the estimation of the $\mathbf{gc}_{\boldsymbol{out}}$ signals independence we compute the distance correlation, defined in Equations 3.28-30, which depends on the distance covariance, equivalent to the Brownian covariance (Gretton, Fukumizu, and Sriperumbudur, 2009). The distance correlation measures the statistical dependence between two random vectors which do not necessarily share the same dimensionality (Székely, Rizzo, and Bakirov, 2007), thus it allows the measurement of the independence of a scalar signal or feature in relation to the remaining vectorial set. It ranges in [0, +1] and the lower value implies statistical independence. In Equations 3.28-30 the operator $\| \quad \|$ represents the Euclidean norm, while (X, Y), (X', Y'), and (X'', Y'') are independent and identical distributed random variables, not necessarily sharing same dimensionality.

$$\mathrm{dCor}(X, Y) = \mathrm{dCov}(X, Y) / \sqrt{\mathrm{dVar}(X) \cdot \mathrm{dVar}(Y)} \tag{3.28}$$

$$\mathrm{dVar}^2(X) \coloneqq \mathrm{E}[\|X - X'\|^2] + \mathrm{E}^2[\|X - X'\|] - 2\mathrm{E}[\|X - X'\| \cdot \|X - X''\|] \tag{3.29}$$

$$\mathrm{dCov}^2(X, Y) \coloneqq \mathrm{cov}(\|X - X'\|, \|Y - Y'\|) - 2\mathrm{cov}(\|X - X''\|, \|Y - Y''\|) \tag{3.30}$$

The five GCs that we consider in this study present progressively more elaborated solutions to compute the $\mathbf{gc}_{\boldsymbol{out}}$, roughly emulating a broad range of methods found in the related work, and gradually assessing the benefits of the

strategies presented in this chapter. The different GCs that we consider generate the output signals by:

1. selection of most independent MFCC (GC-1);
2. selection of most independent robust LPC-MFCC-PLP (GC-2);
3. PCA projection of MFCC (GC-3);
4. PCA projection of robust LPC-MFCC-PLP (GC-4);
5. SOG based GC method with Isomap NLDR of robust LPC-MFCC-PLP (GC-5 - SOG).

The selection of most independent features in the first two GCs is based on the distance correlation, and to obtain statistically comparable results we used the same optimal low-level feature computation settings for all the GCs, derived from our blind-search algorithm. These already introduce improvements over the features computation, filtering and selection techniques presented in previous works. The output signals of the GCs are normalized to the range [0, +1]. Identical vocal-gestures and vocal-postures were used to determine or train the different GCs. For the SOG based GC we used the most unsupervised and default options for data pre-filtering and system training. Moreover identical and training compliant vocal-gestures and vocal-postures were used for all performance measurements, with an overall duration of about 30 seconds for the different postures and 30 seconds for the gestures, mostly derived from the user evaluations presented in Chapter 7.

The results in Table 3.3 are presented separately for 2D and 3D GC cases. The 2D GCs results are derived and averaged from the same 15 cases of Section 3.2.2.4, excluding the two cases with background noise for statistical significance. The 3D GCs results are averaged over a different dataset, which includes only 6 cases from the users, 4 in total. The dataset used for this validation contains gestures mainly including sonorants vowels gilding, and provided by users with limited familiarity with our novel system. However these facts have low relevance here because we are comparing different vocal GC approaches to highlight the gains of our contributions rather than measuring the absolute ratings.

The results of Table 3.3 show that the SOG vocal GC method overall outperforms the other GC implementations, presenting best or close to the best results for all measurements. The trained output lattices included an average of 391 nodes for 2D cases considered and 1010 for the 3D cases. The $\mathbf{gc}_{\boldsymbol{out}}$ signals present low statistical dependence, with values of distance correlation as little cases GC-1 and GC-2, in which the features selection was directly based on the independence measurement. Great improvements are shown for the GC output space coverage and

spread, while the other GC techniques present larger non-reachable sub-regions detrimental for full expressivity in the interface DMI mapping stage.

GCout Measure - 2D cases avg	*GC-1*	*GC-2*	*GC-3*	*GC-4*	*GC-5 - SOG*
gest. independence (dCor)	0.19463	**0.16993**	0.33269	0.29505	0.18641
gest. continuity (mean distance)	0.10553	0.11328	0.07367	0.07102	**0.04966**
gest. space coverage (%)	56.875	55.4948	55.625	57.8125	**84.4792**
gest. coverage spread (count std)	14.7245	13.6361	13.4279	12.2285	**6.21326**
post. stability (std)	0.06952	0.06732	0.04023	**0.03595**	0.03908

GCout Measure - 3D cases avg	*GC-1*	*GC-2*	*GC-3*	*GC-4*	*GC-5 - SOG*
gest. independence (dCor)	0.1976	0.28217	0.24127	0.23479	**0.16579**
gest. continuity (mean distance)	0.17167	0.17418	0.13744	0.14107	**0.12076**
gest. space coverage (%)	33.4961	32.8451	33.9518	35.9701	**58.5547**
gest. coverage spread (count std)	18.5319	15.1238	13.9554	12.6171	**7.65857**
post. stability (std)	0.07033	0.06894	0.05028	**0.04447**	0.04816

Table 3.3: Comparison of independence, continuity, coverage, and spread over vocal-gestures, plus stability over vocal-postures, measured for the output signals of five different vocal GC, averaged over a database of 2D and 3D cases. Lower numbers indicates better results except for the coverage percentage (best results in bold with grey background).

The coverage is still not close to 100%, especially for the 3D case, because when capturing the data we did not ask users for full space coverage, as in the other test presented Chapter 7. The improved spatial spread, visible in the first example of Figure 3.29, did not degrade $\mathbf{gc}_{out}$ continuity or stability, which are rather improved as well. The low values for the continuity measurements imply a reduction of discontinuity of the signals at the output of the GC. These are shown in the second example of Figure 3.29. The stability over vocal-postures for the SOG vocal GC is low as in other cases, and in the DMI control component of the interface, as well as in the VCI4DMI prototype, we integrate further techniques to improve the DMI parameter stability.

The different GCs considered in this study were briefly tested in the preliminary study and these presented sufficient usability and controllability. We explored other GCs based on Independent Component Analysis (ICA) technique (Comon, 1994; Hyvarinen, 1999) to learn the extraction of independent signals from the large set of LPC, MFCC, and PLP features. The ICA based GC outperformed the other implementations in most of the measurements included in Table 3.3, but it presented severe limits in the usability and in the understanding of the mapping between voice timbre and GC output space positions. These issues determined the exclusion of this technique for the development of the VCI4DMI. The ICA supposes that the

observations are a linear combination of arbitrary and independent sources. We speculate that the issues we experience may be due to the non-linear combination of distinct acoustic voice characteristics into the low-level features we use here.

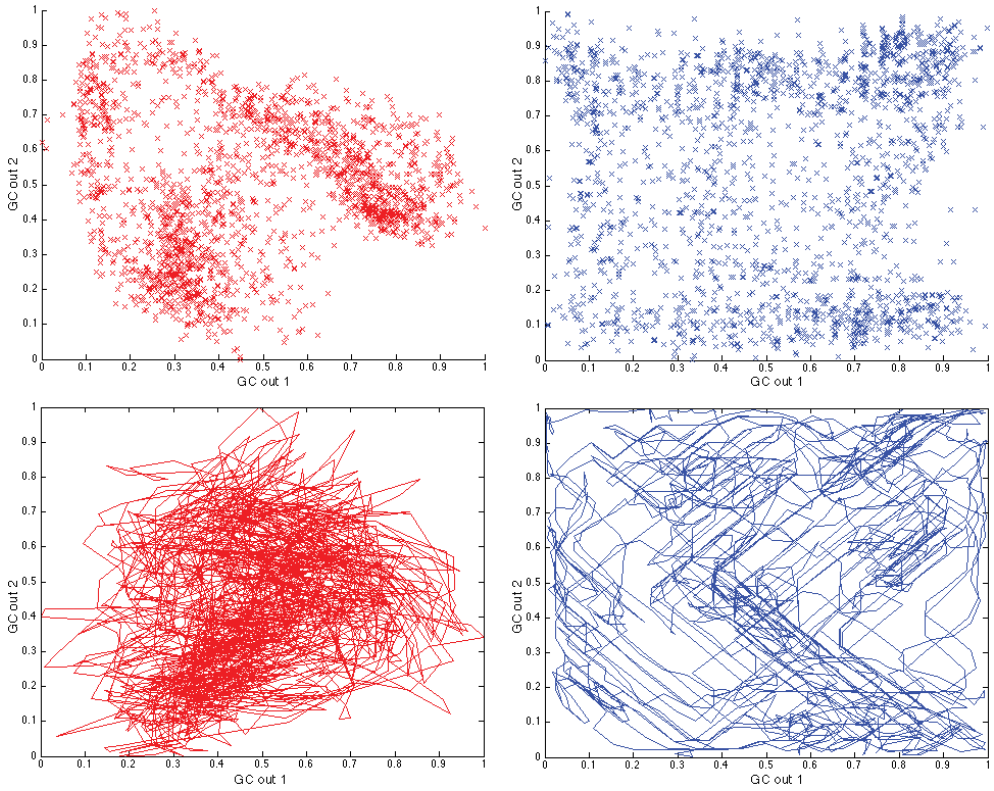

Figure 3.29: Comparisons between worst gestural controller case (red) and proposed SOG based gestural controller (blue) for outputs space mapping spread (top) and output signals trajectories continuity (bottom) for two different 2D validation dataset.

3.5 Summary

In accordance with the overall principles and requirements for the VCI4DMI of Section 2.3.1, in this chapter we introduced and motivated our novel adaptive and generative method to tune the gestural data computation and to implement an ad-hoc gestural controller from user provided vocal training data. In particular, after describing the voice production apparatus and voice perception, we presented the procedure to compute robust and continuous data from the performer's voice, representative of the musical control intention, and we refined the computational settings towards noise rejection and quantity of information enhancement. Then we introduced a gestural controller based on a model of the gestural data learnt with a SOM lattice, trained with a modified procedure, that looks at the spatial unfolding properties of the voice derived data. The GC is capable of responding to new voice

input accordingly to the model derived from the training vocal-gestures. The numerical evaluation showed consistency with the design principles and requirements. Moreover the results demonstrate that the proposed method outperforms other previously proposed solutions. The training procedure relies completely on unsupervised techniques, thus the user effort in setting up this component of the VCI4DMI is limited to providing a set of vocal-posture and vocal-gesture audio examples. In presenting and developing the contributions of this chapter we aimed to generalize underlying concepts so that they can find further application in different musical interfaces or in non-musical contexts. In Figure 3.30 we illustrate the summary of the training procedure and functional part of the GC for the VCI4DMI presented in this chapter.

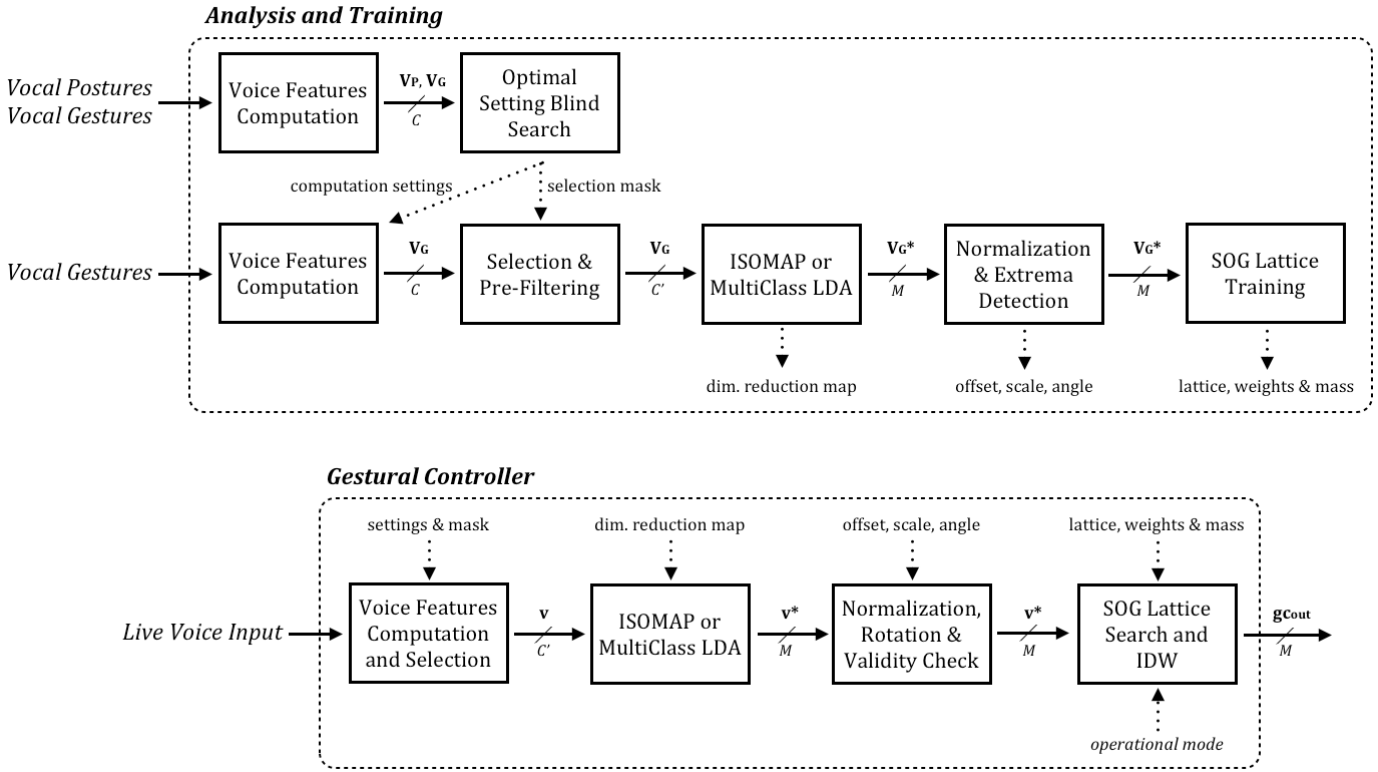

Figure 3.30: Illustrated summary of training procedure and functional part of GC for the VCI4DMI.

Chapter 4

Sonic Control of Digital Musical Instruments

This chapter is dedicated to the digital musical instrument control strategy developed to implement a non-trivial interaction between the few signals extracted from the voice and an arbitrary higher number of DMI parameters. Herein the method we propose takes advantage of specific instrument sonic maps, automatically computed beforehand, for adapting and reducing the dimensions of the control space. At first in the chapter we describe the milestones in the evolution to modern DMI and synthesis techniques. We classify and examine DMI input parameters in relation to musical and sonic effects they determine in the output sound to identify those most appropriates for vocal control. Then we present a generic DMI taxonomy for the development of categories requiring different analytical procedures to determine the specific parameters-to-sound relationship. It follows the description of the analysis methods for each category and the description of the case-specific sonic maps, taking timbre perception principles into consideration. Finally we describe the strategy to use the high dimensional sonic maps to achieve a few-to-many mapping with theoretically no losses in the sonic capability of the DMI, which is central to the VCI4DMI for providing a natural control over an arbitrary number of DMI real-valued parameters. The chapter ends with discussing results and measurements over a set of real DMIs.

4.1 Digital musical instruments and control

Modern electronic or digital musical instrument are the result of a century of evolution and integration of innovations from different fields. The way performers play and interact with these devices is on one side conditioned by centuries of practices with acoustic instrument, while on the other it evolves continuously to keep up with the new potential provided by instrument designers. Here we briefly review the short history of sound synthesis for musical instruments and we discuss the musical meaning of different type of controls.

4.1.1 From electrophones to modern DMI

Music has been part of all human cultures across the globe since more than 50,000 years, and it exists only through musical instrument and players. Prehistoric flutes, made out of carved bones and about 35,000 years old, have been found in several places across central Europe and archeologists consider these the earliest artifacts made from durable materials that can be called primitive musical instruments. The *Divje Babe Flute*, in Figure 4.1, found in Slovenia is the oldest instrument, and it hails from the Paleolithic, back to 43,000 years ago. Prior to that it is believed that generic tools such as stones, or clapping of hands were used to create rhythms as an early form of music. Musical instruments have dramatically evolved into a wide variety of forms across time and cultures (May, 1983) and although there are no reliable methods to ascertain the precise chronology of musical instruments within the human history, it is possible to classify these accurately following the Hornbostel–Sachs classification, which is divided into four macro categories: *idiophones*, *membranophones*, *chordophones* and *aerophones* (Von Hornbostel and Sachs, 1914). In the early twentieth century, with the advent and spread of electroacoustic, electric, and later electronic or digital musical instrument the Hornbostel–Sachs classification became incomplete since it included only those that today we call acoustic instruments. Therefore in 1940 Sachs added the *electrophones* category that includes:

> "51. Instruments having electric action (e.g. pipe organ with electrically controlled solenoid air valves);
>
> 52. Instruments having electrical amplification such as the Neo-Bechstein piano of 1931, which had 18 microphones built into it;
>
> 53. Radioelectric instruments: instruments in which sound is produced by electrical means."
>
> – Curt Sachs (Sachs, 1940)

Today, with the view over more than a century of *electrophones* and in order to maintain the original classification consistency, ethnomusicologists such as Ellingson (1979) and Kartomi (Kartomi, 1990) propose that only the sub-category 53 should remain in there, while the others should be placed respectively within *aerophones* and *chordophones*. The *Telharmonium*, in Figure 4.1, is the first non-acoustic instrument in history that presents both a sound generation mechanism and musical interface. It was a 200ton electric organ developed by Thaddeus Cahill starting from 1897, that would inspire 30 years later the *Hammond* organs. It electromechanically generates

an electric signal which creates musical sounds, by tonewheels for additive synthesis, and is then projected by primitive loudspeakers. Tonewheels were originally invented for radio communication purposes, and later adopted in electromechanical organs. Application and hacking of technologies designed for different purposes is a recurrent pattern in DMIs history. Earlier, Elisha Gray in 1876 realized the *Musical Telegraph*, the first electrically generated sound synthesis, transmitted over telephone lines, based on of self-vibrating electromagnetic circuits, representing a prime form of oscillators. One year later Thomas Edison invented the phonograph, the first device that could record and reproduce sounds. Even though it was neither a musical instrument nor electrical, it finally allowed sounds to travel across time and space. Sound recording devices such as tape and then samplers, will later gain a great importance for DMIs. The triode *audion*, the first amplifying vacuum tube introduced in 1906 by Lee DeForest, enabled the implementation of electronic oscillators, the basic building blocks of sound synthesizers. In 1928 Fritz Pfleumer invented the first magnetic tape sound recorder, inspired by the 1898 Valdemar Poulsen magnetic wire recorder. In 1935 it was commercialized by the German *Magnetophon*, innovating with its portability and cut and paste contents editing capability. The tape recorder was the essential device for the *Musique Concrète* artists. Indeed in the 1930's "electronic music" was defined for the first time as "electronically produced sounds recorded on tape and arranged by the composer to form a musical composition" (Dictionary.com, n.d.). In 1937 Harald Bode introduced the *Waarbo Formant Organ*, the first polyphonic synthesizer, while Evgeny Murzin invented the *ANS Synthesizer*, a photoelectronic musical instrument, generating a sound from a drawn spectrogram. The *Novachord,* by Laurens Hammond in 1939, was the first music synthesizer commercially manufactured. In 1948 Hugh Le Caine presented the *Electronic Sackbut*, the first voltage-controlled synthesizer, a control method that later would become a standard.

Sound generation in the early twentieth century was widely based on the additive synthesis technique, consisting of sine waves summation. This method, based on the Fourier theorem principles, had the drawback of requiring a high number of independent sinusoidal oscillators to achieve interesting timbres, and thus it presents difficulties for dynamic control. It was abandoned, though recently revitalized, when simple electronic active circuits started to be used to generate non-sinusoidal audio waves. Due to a richer spectrum, these allow the synthesis of complex timbres with a lower numbers of oscillators. Therefore the era of analog synthesis started with Raymond Scott who in 1951 introduced the first remarkable fully analog electronic step sequencer, containing 16 independent oscillators and tone circuits, and in 1956

he presented the *Clavivox*, a synthesizer sub assembled by Robert Moog. Voltage controlled amplifiers, oscillators, and filters became the basic building blocks of most analog synthesizers. In 1957 RCA designers Herbert Belar and Harry Olson built the *Mark II Music Synthesizer*, installed at the Columbia Princeton Music Centre. Composed of an array of analog synthesis components it had the size of a room and could manage four notes variable polyphony. To change the timbre it was required to modify the module interconnections and it generated sound only by programming through a punched paper tape. In 1964 Robert Moog displayed the *Moog* modular analog synthesizer at the Audio Engineering Society convention, which was a breakthrough with its small size and intuitive use. The work of Moog was remarkable for the sound of his machines but also because it was oriented toward making devices portable and accessible to musicians, not only to engineers. He realized the *Minimoog* in 1970, integrating all modules of the earlier versions in a single device with a built in keyboard, which gained high popularity in live performance and in pop music.

New methods of sound generation started to emerge such as the pioneering work of Iannis Xenakis who in 1959 introduced a primitive form of granular synthesis in his composition, using analog tone generators and tape splicing. In 1975 Curtis Roads implemented the first granular synthesizer using MUSIC V. Real-time would be achieved only in 1986 by Barry Truax. Different synthesis techniques based on grains, elementary sound units, will later gain larger reputation and application (Schwarz, 2006) such as the corpus based concatenative synthesis (Schwarz, 2000) and audio mosaicing. In 1967 John Chowning discovered the FM synthesis algorithm (Chowning, 1973), which can generates rich and complex timbres with a low number of oscillators. It was then exclusively licensed to Yamaha and included in the 1983 DX-7, the first stand-alone digital synthesizer. Sample based sound synthesis became a mature technique with the Fairlight CMI, the first groundbreaking polyphonic digital sampler, introduced by Peter Vogel and Kim Ryrie in 1978.

Physical modeling synthesis was first introduced in 1969 by Lejaren Hiller and Pierre Ruiz, using a finite difference model of a vibrating string to generate plucked and struck string tones (Hiller and Ruiz, 1971). In 1979 Claude Cadoz, Annie Luciani, and Jean-Loup Florens developed *CORDIS-ANIMA*, a physical description language for musical instruments based on networks of masses and springs (Cadoz, Luciani, and Florens, 1993). In 1992 Julius Orion Smith III developed a simple and efficient delay-line structures to model wave propagation in objects such as strings and acoustic tubes (J. O. Smith, 1992), and he cooperated with Yamaha for its integration in the VL-1, the first commercial physical model synthesizer, released in 1994. In 1991 Jean-Marie Adrien introduced the modal methods for physical

modeling synthesis, which relies on the decomposition of the dynamics of a system into modes, each of which oscillates at a given natural frequency (Adrien, 1991), later implemented in the *Modalys/MOSAIC* environment by Joseph Derek Morrison (Morrison and Adrien, 1993).

Microprocessors, microcontrollers, and CPUs started to be integrated in DMIs to provide additional control functionalities, starting with fast memory save and recall of machine state as there is in the 1977 *Prophet-5* from Sequential Circuits. This trend led to more complex devices such as the 1988 Korg M1, the first complete music workstation with an on-board MIDI sequencer and a rich palette of sound, with the unprecedented all time record of 250,000 units sold worldwide. Today the trend continues using general-purpose personal computers and the hardware implementation platform, and offering virtual devices implemented in software. High functionality integration is offered by DAWs that followed the approach of the 1993 Steinberg *Cubase Audio*, which provided 8 tracks of recording and playback with DSP effects built-in on the Atari *Falcon-030*, using only native hardware. In the same time span a variety of audio effects and signal processing techniques were developed, often in the context of radio broadcasting. These were introduced as an adjunct part of a DMI or as a separate unit to manipulate sounds from other sources. Standalone effect processors proliferated because they are more flexible and suited also for the processing of the sound of acoustic instruments, voice, or recorded sounds.

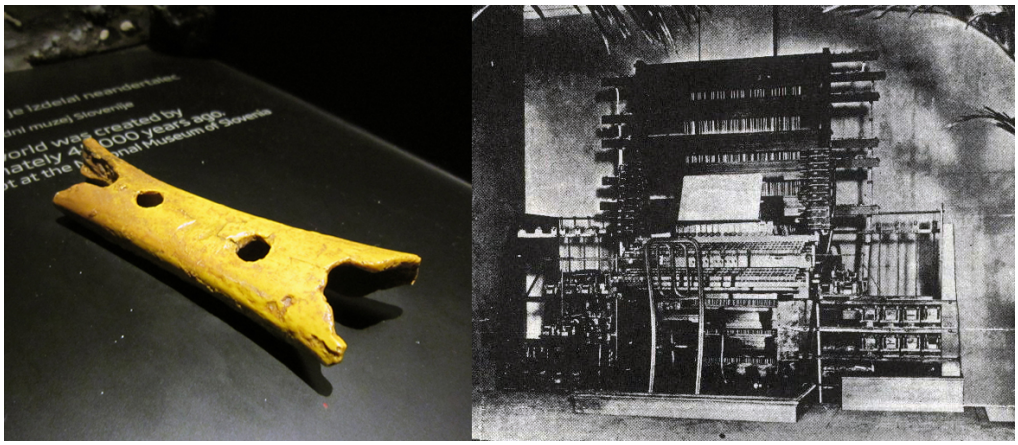

Figure 4.1: Divje Babe Flute (left) and Telharmonium (right) respectively the first acoustic and electronic musical instruments in history.

4.1.2 Control flow

Musical instruments can be modeled as a flow of information processing that converts gestural parameters into musical parameters through an intermediate

transformation stage involving only technical parameters (Kvifte and Jensenius, 2006), usually hidden and irrelevant to performers. These, at every stage, are represented or perceived as continuous quantities, which become discrete when the variations are tied to fixed step quantities such as the pitch on the chromatic scales. Regardless of the specific acoustic instrument or synthesis technique, the input parameters determine loudness, pitch, timbre and duration of the generated sound. As Kvifte and Jensenius suggest, these are described by different measurement modes, respectively ordinal, interval, nominal and ratio levels. For DMIs the technical, or synthesis, parameters are very relevant because they are not fixed as they are in specific acoustic instruments. The possibility to change phyisical and geometrical properties of an acoustic resonator during a performance are extremely limited, while for sound synthesis there are no limits on runtime modeling and variables tuning. The wide and continuous timbre morphing potential is a key characteristic of modern DMIs, while the range and modulation rates of pitch, duration, and loudness are broader. Therefore besides triggering and modulating musical parameters, performers can design sound timbres and textures in real-time. The possibility to link synthesis variables to parametric envelopes and Low Frequency Oscillation (LFO) for example, facilitate the creation of evolving sounds rich in dynamics. In acoustic instruments it is simple to relate gestural parameters to discrete and continuous musical parameters, while DMIs expose possibly hundreds of variables in the synthesis algorithm, also called technical parameters, which may have little, variable, or no correlation with the musical output, unless the user is deeply familiar with the specific device.

In the previous chapter we described a method to extract intermediate parameters from an ad-hoc vocal GC. These are continuous so that mapping is possible on both real-valued and discrete DMI control variables, while the output of the GC has a theoretical limit equal to the real dimensionality of the vocal-gesture trajectories. Therefore the challenges we face in developing the mapping strategy for the VCI4DMI are to:

- provide a reliable and non-error-prone control of musical parameters determined by the controlled DMI technical parameters;
- implement a non-trivial control beyond the simple map of individual components of the GC output to synthesis variables, while maximizing the number of simultaneously controllable DMI parameters;
- automatically adapt the interface to the specific target instrument behavior to provide a posterior co-design (Cook, 2001) between

instrument and interface, and to hide the intermediate information processing stages to the user.

4.1.3 Critical parameters

The VCI4DMI requires the simultaneous control of multiple real-valued and time-continuous parameters, as mentioned in the previous chapters. Despite the specific synthesis technique and the mapping strategy we adopt, the variation of DMI parameters modifies the sound generation process, which in turn impacts on different musical parameters. Traditional music, regardless of the genre, is largely built on the fundamental elements of rhythm, harmony and melody. Listeners, even if not musically trained, have a large musical experience that conditions the way they experience the pieces unfolding. As Meyer (1956) claims, generating, suspending, prolonging or violating listeners' expectations about the fundamental elements is a key feature of musical compositions and improvisations (Narmour, 1992). Moreover melody, harmony and, to a certain extent, rhythm unfold in a two-dimensional discrete time-frequency domain. On one axis the grid is determined by the pitch scale, and on the other by the temporal gird, formed by integer fraction of the tempo. A musical event generated in a wrong grid position is easily perceived as a performer's error because it corrupts one or more musical fundamental structures of a piece. We consider "critical" and less suitable for vocal control those DMI parameters directly linked with discrete musical parameters in the time or frequency domains. Therefore when the voice is not the exclusive input modality of an interface system, we favor the mapping of the voice to real-valued DMI parameters that affects exclusively timbral properties of the musical parameters. Examples are morphing the spectral envelope or the brightness of a synthesizer's timbre, the depth of a modulation, or the feedback of a tape delay.

4.1.4 Few-to-many mapping trend

With the complexity of the synthesis model continuously increasing, the numbers of controllable parameters on DMIs related to variables in the sound generation and processing algorithms is continuously expanding. For instance, the modification of synthesis patches in a real or emulated analog synthesizer is easily feasible by adjusting the few dozen synthesis variables that regulate the two oscillators. The same operation is more challenging with the hundreds of variables in physical modeling synthesizers. Moreover parameters can present strong correlations or

mutual dependencies that change their effect on the resulting output sound. Examples are the number of harmonics of an oscillator followed by a low pass filter, and the amplitude plus frequency control of the modulator in FM synthesis. In the first case the audible effect of increasing the harmonic content strongly depends on the cutoff frequency of the filter, while in the second the timbre variation produced by increasing the modulator frequency depends on the amplitude parameter. Dealing with a high number of DMI parameters with potential no or little effect on the output sound is admissible, even if not desirable, in composition and production context, where the user can patiently adjust one parameter at a time to exactly tune the device for the desired sound. In live performances the expressivity of control is usually much more significant than a fine and precise sound synthesis tuning.

In the linear one-to-many mapping, the control space dimensionality is reduced at the expense of large losses in achievable parameter combinations. For instance mapping two synthesis parameters to two standard MIDI control changes, with 7-bit resolution or 128 unique values each, we can set up to 16,384 unique combinations or sounds. When mapping these together to the same MIDI control change we can achieve only 128 different combinations, determining a 99% loss over the potential set. Designers often package more meaningful one-to-many mappings with the synthesis patches, often called "macro" parameters, using different linear transformation for each DMI parameter controlled by the same control signal. This strategy provides expressive sound morphing, but the manual definition of each scaling and offset value requires a deep knowledge of the specific synthesis process, not inline with the principle of minimal interaction and expertise for the VI4DMI setup.

The necessity for a reduced set of control parameters, with higher relation to musical or perceptual features of the sound has motivated and driven several works in recent years. These enable controlling a number of DMI parameters with fewer interface control signals derived from the performer's gesture, limiting the decline in expressivity and sonic potential. The *Manifold Interface* (Choi, Bargar, and Goudeseune, 1995) helps users to identify, through visualizations and real-time gestural control, the sub-regions of the parameter spaces associated with the desired acoustic results, to shrink the mapping region to a less extended area. Interactive evolution and genetic algorithms principles are used in (Dahlstedt, 2001) to simplify the generation of new sound objects. These are associated with a large set of synthesis parameters and generated by genetic mutations from a selected pair of parents, from which the new object is likely to inherit sonic characteristics. The *Metasurface* (Bencina, 2005) implements two-to-many mapping by distributing

instrument parameters snapshot on a surface first, and then interpolating between these using a technique based on the Voronoi tessellation, that ensures parameters continuity when moving across the parameters presets. The *Modulation Matrix* (Brandtsegg, Saue, and Johansen, 2011) provide control plus automatic modulation on a large number of parameters achieved with self-modifying mapping system combined with a dynamic interpolation scheme, implemented efficiently by its inherent sparse matrix structure property. Principles of catastrophe theory and dynamical systems are applied to musical interface in (Gaffney and Smyth, 2013) to expand generate additional mapping parameters, modeling the complex dynamics regulating physical phenomena in acoustic instruments.

4.2 Sound timbre and perceptual control

We discussed the performers' need for a transparent DMI control strategy, so that their gestures can be directly related to musical parameters rather than technical, synthesis or processing variables. Directly controlling perceptual sound aspects is possible in certain scenarios. These can be derived from a detailed knowledge of the synthesis process, estimated by an analysis of the relationship between instruments input and output, or by directly accessing knowledge about the organization of sound material in certain synthesis techniques. Next we briefly present characteristics and limitations of these systems and we introduce the principles of human timbre perception in more detail. This approach can shift the interaction feel from synthesis parameters directly to perceptual features of the sound. This is a key aspect in the development of this dissertation because the input modality of the interface, the voice, is produced directly targeting specific timbres rather than targeting physical configurations of the vocal tract. Therefore the techniques developed here have their roots in natural timbral control strategies while making the DMI control space implicit rather than explicit.

4.2.1 Timbre, descriptors and perception

Timbre is often considered "the psychoacoustician's multidimensional waste-basket category for everything that cannot be labeled pitch or loudness" (McAdams and Bergman, 1979), or the sound attribute that allows us to distinguish two sounds with equal pitch and loudness. Since it includes multiple objective and subjective characteristics of the sound, a simple measurement or standardized numerical

representation is not possible. Indeed, as mentioned before, timbre can be measured only at nominal level, to distinguish and separate different sounds, but it is not possible to order it on an absolute mono-dimensional scale, unless we target single characteristics falling into the domain of the timbre such as the tonal character, brightness, noisiness, color, attack and decay, color glide, microintonation, vibrato, or tremolo. In Section 3.1.3.1 we introduced principles of sound perception in the human auditory system. In particular we observed the non-linearity of the frequency and loudness perceptual scales. The Seneff (1988) auditory model, illustrated in Figure 4.2, captures the essential characteristics of the cochlea and hair cells response to sound pressure waves. In the cochlea the sound is divided into separate critical frequency bands, each producing an output on a separate nervous channel. In the hair cell synapse the final probability of firing depends on the processing chain in which the half wave rectification is followed by the short-term adaptation, reduction of synchrony and fast automatic gain control.

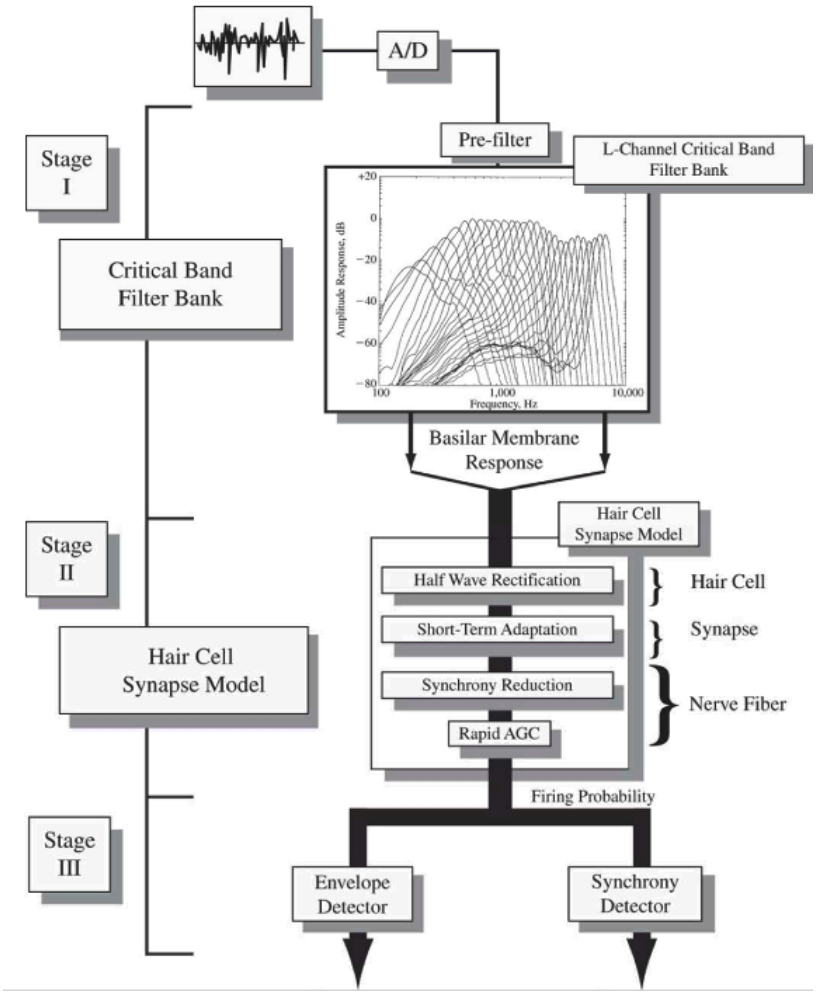

Figure 4.2: Seneff auditory model diagram.

Perceptual sound information is then derived from the grouped firing probability going through the envelope and synchronous detectors. Thus the perception of timbre strongly depends on the spectral contents, often approximated with the spectrum, and the time envelope of sounds. Schouten (1968) recognized five main timbre acoustic parameters, ranked in relation to contemporary music (Erickson, 1975): tonal to noise-like character, spectral envelope, time envelope, spectral envelope and fundamental frequency dynamics, and onset. The spectral envelope is often approximated with the power spectrum, or further reduced and adapted to the auditory system computing energy of the 25 Bark critical bands. The phase component of the spectrum is ignored because, according to the Ohm's acoustic law (Ohm, 1843) (Von Helmholtz, 1954), the ear is phase deaf, although this is a very crude generalization. The four waves in Figure 4.3 are generated with identical frequency components but different phase relation. Despite the strong differences in the time domain representation, these tones are perceptually very similar. The time envelope is modeled, but also implemented in DMIs, as a sequence of four segments, namely Attack, Decay, Sustain, and Release (ADSR), as illustrated in Figure 4.4. The attack, decay, and release segments are described by a time value to linearly or exponentially increase or decrease the amplitude between two values, while the sustain is represented by an amplitude value and it can last an arbitrary amount of time.

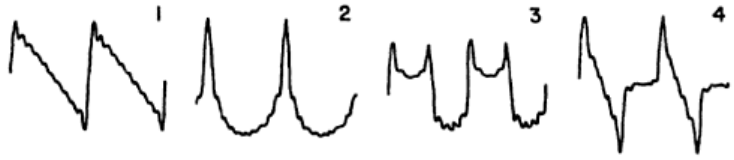

Figure 4.3: Four waveforms generated with different phase relations of identical frequency components whose timbres perception is similar, after (Plomp, 1976).

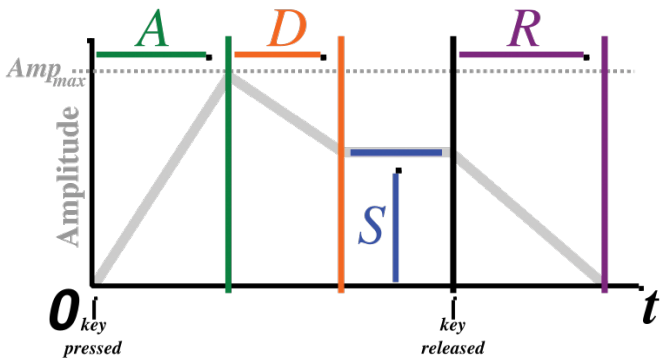

Figure 4.4: Illustration of ADSR envelope.

The amplitude envelope of real sounds can differ across the critical bands, as supported by the Seneff model, and in modern sound synthesis this is emulated applying independent ADSR envelopes to different parameters such as to the cutoff frequency of filters besides the amplitude of the oscillators. Other timbral characteristics such as the brightness or the noisiness are usually computed from the spectral centroid and geometric mean.

Despite the high dimensionality of the timbre spaces, listening tests have demonstrated that this space can be reduced down to two or three dimensions, using MDS techniques (Grey, 1977). These results have been replicated and extended several times since Grey's original work. Results demonstrate high consistency with those of the pioneering work of Grey, across a variety of different musical or sonic contexts. McAdams and Cunible (1992) argue that the data in the reduced timbre space should maintain the proximity of the original space, therefore NLDR techniques based on non-Euclidean distance measurements provide a better representation than linear MDS. Moreover they demonstrate the correspondence of the three axis of the reduced space with the spectral energy distribution, the onset characteristics, and the variation of the spectral distribution over time. However there is no model to represent the interaction between the different attributes characterizing the timbre, especially because this can be context and subject dependent. Moreover, in these studies the listeners were always exposed to a relatively small database of sounds, and thus in larger contexts results of the timbre MDS is likely to have higher dimensionality.

Risset and Wessel (1999) in reviewing more than a century of multidisciplinary works on timbre description and perception, argue that the adopted representation and studies lack of dynamics and context consideration. They observe that, for instance, in a single acoustic instrument the dynamics and pitch variation can be wide, and the listening condition can be affected by heavy reverberation of the hall or distortions low-quality loudspeakers. Regardless of these aspects, source and timbre can be always unequivocally identified, implying a perception constancy beyond specific circumstances, and supposing the existence of an invariant physical acoustic characteristic mediating a given timbre. In their exploration of the timbre by analysis and synthesis Risset and Wessel recognize the centrality of the timbre in modern music compositions even though the high dimensionality and blurry definition still confound a scientific representation.

4.2.2 Control-to-timbre mapping and interactive sound maps

In this section we present existing strategies for direct mapping of the interface control data to instrument timbre attributes. The mapping directly to the synthesis control parameter space, which is nonetheless still necessary when real-time synthesis has to be executed, will be implicit. The link between the two spaces, GC output and timbre respectively, needs to be determined, but the strategy can differ across synthesis techniques.

Subjective timbre dissimilarities between 24 orchestral instruments were measured and organized by Wessel (1979) in a reduced 2D space by MDS, which was then used to control an additive synthesizer. The generation of the related parameters interpolates between those of the original 24 patches. This strategy allows the generation of a large range of timbres not present in the listening tests, and implements a drastic reduction of the control space, while providing perceptually meaningful control dimensions. A more complex approach which does not require subjective listening is described in (Jehan and Schoner, 2001), in which the computerized perceptual analysis of sounds is composed by pitch, loudness, plus the timbre descriptors brightness, noisiness and energy of the bark critical bands. These were used to develop a synthesis engine that models and predicts acoustic instrument timbres. The method is based on cluster-weighted modeling to approximate synthesis parameters from timbre descriptions, while it can also predict a timbre given a new set of parameters. The similarity arrangement of sonic material into a Gaussian kernels space, used to implement the control for a variety of DMI, demonstrated the relevance and benefit in music composition and performance expressivity of space modeling graphical representations and dimensionality reduction techniques (Momeni and Wessel, 2003).

The strategies considering the perceptual characteristic of the timbre within the gesture-to-parameters transformation are generalized by the multi-layer mapping chain proposed by Arfib et al. (2002). The introduction of the intermediate perceptual space within the modular process improves sensitivity and efficiency of the mapping. The navigation of a high dimensional, continuous, and complex sonic space that allows timbre micro variations is mapped to a 2D controller in (Van Nort and Wanderley, 2007). The authors propose time-invariant and perceptually repeatable mappings, as well as dynamic and less sensitive mappings in a 4D sonic space. A larger set of intuitive and natural timbre descriptions are exposed and used to control sound synthesis in (Poscic and Krekovic, 2013). The system is based on fuzzy models

manually defined by expert users using if-then-else rules, but allows a novice user to effectively control the synthesis algorithm just by using visual programming.

Sound synthesis techniques based on a database of heterogeneous samples such as audio mosaicing and concatenative synthesis (Schwarz, 2000), can intrinsically provide control close to the timbre perceptual characteristics of the output, because the available sounds are often analyzed, arranged and retrieved according to psychoacoustic features. These techniques sequence sound chunks drawn from a database, and their temporal length is sufficient to preserve most characteristics of the original timbre, with the exception of the time envelope. The database is organized and visually represented in 2D or 3D sonic maps, and gestures related to instantaneous coordinates of the space determine the sequencing of different sound frames into evolving timbre and textures. In *MoSievius* (Lazier and Cook, 2003) the MFCC analysis of the sound frames in the database provides the spatial representation for the audio mosaicing engine, and for querying the database in real time the system uses the analysis of the live audio input or a symbolic description of the target sound. Longer segments of the live audio input are considered to compute the descriptors of the query (Schnell, Cifuentes, and Lambert, 2010). This approach includes characteristics of the temporal evolution of the sound in the analysis and retrieval, providing more pertinent sound classification and organization. Similarly, bipolar and higher level timbre descriptors are used to implement a sound browser, which organizes textural sounds on a 2D map (Grill, 2012). The descriptors, including timbral, temporal and structural characteristics of the sound are dimensionaly reduced using the t-distributed stochastic neighbor embedding technique. In *CataRT* (Schwarz et al., 2006), a real-time corpus based concatenative synthesis system, smaller sound grains are first extracted from a corpus of sound, then analyzed and finally concatenated to generate the output sound, which presents a wide timbre potential. The grains are organized and visualized in a 2D color-coded sound descriptor space, and the individual components can be associated with descriptors selected from a large set. The interaction for the real-time grain selection is directly established over the sound descriptor space, and implemented relating the gestural data to the grain selection in different modalities (Schwarz, 2012).

The low dimensional representation of sound databases or synthesis patches is recently gaining popularity also in commercial products, because it eases the use of complex DMI, and speeds up the generation of new configurations, especially for

low-expertise users. In Arturia[2] software synthesizers, the factory presets are organized in a 2D space and color-coded by instrument category. From the space, in Figure 4.5, the user browses the presets with neighborhood perceptual relevance and new patches are generated interpolating a selectable subset of patches in a user-defined position.

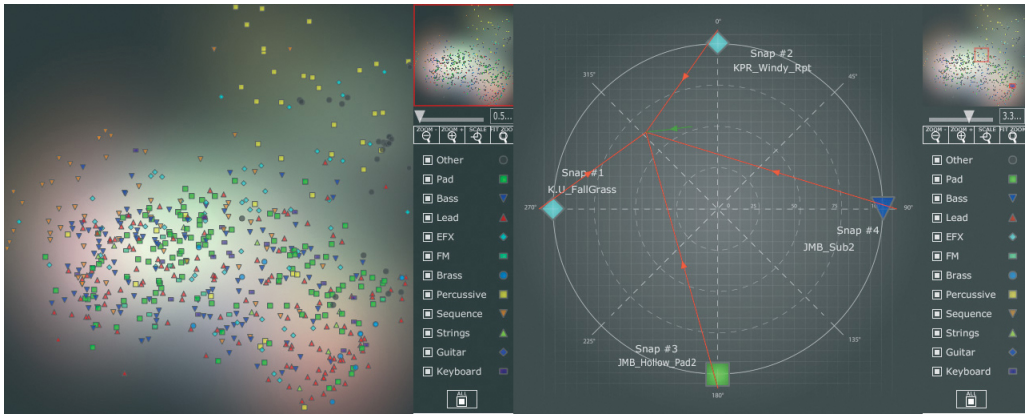

Figure 4.5: Organization of synthesis patch in timbre space and generation by preset interpolation provided in Arturia software DMIs.

4.3 Modeling DMI sonic response for timbre control

The VCI4DMI aims to provide natural control over an arbitrary number of DMI real-valued parameters, which determine the timbre of the generated sound. Voice production directly aims for the timbre rather than for vocal tract articulation, thus a natural DMI control requires identical characteristics and interaction at perceptual sonic level, hiding the technical parameters from the performer. A specific synthesis algorithm plus the subsets of fixed and variable parameters determine a univocal sonic space enclosing all the potential timbre variation. A limitation of sound generators is represented by their weak or missing relationship to sound perception models, which determines a sound object oriented interaction rather model oriented (Wishart, 1996). The generation of ad-hoc models for the selected synthesizer and parameters, centered on the sound perception, has a key role. The techniques for timbre analysis and mapping presented in the previous section are not sufficiently generic in their theoretical development and practical implementation for the application in practical DMI design. Sound models that embrace an end-to-end description can provide compression of the parameter space, expose only sonic

[2] http://www.arturia.com

related parameters, and represent the distinctive behavior that generates a specific narrow range of sound (Wyse, 2005). In this section we define our generic strategy to derive the sonic space for any DMI without any prior knowledge about the particular synthesis model. The DMI is considered as a black box converting algorithm variables into perceptual sound features. At runtime, retrieving the synthesis variables from the sonic space provides a strong intrinsic reduction of instrument parameter space, that prevents cognitive overload of the interface, and at the same time provides consistency, continuity and coherence (Garnett and Goudeseune, 1999). Moreover in order to cover more scenarios, the proposed method is extended to include DMIs that only process input sound signals rather than generate sounds themselves such as, for example, audio effects processors.

The works reviewed in Section 4.2.2 are built upon perceptual timbre dissimilarities of different instruments, even though these may be synthetized by the same device or algorithm. The low dimensional representation of the timbre space is sufficient to discriminate, identify and recognize distinct timbres. However in the context of this thesis, an instrument with most of its synthesis variables fixed and the remaining parameters under the performer's control may generate only minor timbre nuances, overlapping with others or clustering too tightly in the proposed low dimensional timbre spaces. A prior reduction or a manual selection of the components of the sonic space is inadequate because we assume no prior knowledge of the DMI internal algorithm. Therefore we favor an analysis method that first considers the timbre in a high dimensional space, and in a second stage finds the non-linear reduction that maximizes the discrimination across sonic results despite the absolute timbre breadth. Moreover we do not consider the timbre only as an instantaneous feature of the generated sound, but we can also include, when necessary, dynamics that characterize the transient phase or the temporal variations of sound textures.

4.3.1 Generic DMI model

Despite the synthesis algorithm, the implementation, the sound generating or processing properties any deterministic DMI can be modeled with Equations 4.1-4. The set I_{all} contains parameters of a specific instrument, and it includes the subset I of P parameters which are the target of the real-time interface control. The perceived timbre $\zeta(\mathbf{i}, T)$ over a time interval T, generated by the unique combination of parameter $\mathbf{i}$, depends on the T-tuple of DMI outputs $d(\mathbf{i}, t)$ and the auditory system

transfer function A(). Equation 4.3 is valid for DMIs that generate sounds, while 4.4 is related to DMIs that process sounds, and thus depending also on the input signal s.

$$\mathrm{I} = \{\mathrm{i}_1, \mathrm{i}_2, \dots, \mathrm{i}_P\} \subseteq \mathrm{I}_{all} \tag{4.1}$$

$$\mathbf{i} = [i_1, i_2, \dots, i_P] \tag{4.2}$$

$$\zeta(\mathbf{i}, T) \approx \mathrm{A}(d(\mathbf{i}, 1), d(\mathbf{i}, 2), \dots, d(\mathbf{i}, T)) \tag{4.3}$$

$$\zeta(\mathbf{i}, T, s) \approx \mathrm{A}(d(\mathbf{i}, 1, s), d(\mathbf{i}, 2, s), \dots, d(\mathbf{i}, T, s)) \tag{4.4}$$

Without loss of generality, we consider each parameter i in the range $[0,1]$. The set of unique combination of input parameters, in Equation 4.5, has cardinality B equal to the product of the number of unique values that each parameter can assume, in Equation 4.6. This in turn depends on user-defined limited range and on real-valued parameter resolution. In Equation 4.6 the symbol $|\ |$ represents the cardinality operator.

$$\mathbf{I} = [\mathbf{i}_1, \mathbf{i}_2, \dots, \mathbf{i}_B] \tag{4.5}$$

$$B = |\mathbf{I}| = \prod_{j=1}^{B} |\mathrm{i}_j| = \prod_{j=1}^{B} \frac{\max(\mathrm{i}_j) - \min(\mathrm{i}_j)}{\mathrm{resolution}(\mathrm{i}_j)} \tag{4.6}$$

To fully characterize a DMI we relate each parameter vector $\mathbf{i}_\mathrm{j}$ to a timbre descriptor vector $\mathbf{d}_\mathrm{j}$ with dimensionality A, which are coefficients that provide a detailed representation of the output sound timbre. Therefore two matrices $\mathbf{I}$ and $\mathbf{D}$, with different dimensionality but equal number of univocally related entries B, represent the model describing the sonic space generated by a given DMI. The vectors $\mathbf{d}_\mathrm{j}$ are related to the $\zeta(\mathbf{i}_j, T)$ and numerically computed by extracting timbre perceptual descriptors from the sequence $\left(d(\mathbf{i}_j, 1), d(\mathbf{i}_j, 2), \dots, d(\mathbf{i}_j, T)\right)$, as in 4.7 where f represents the timbre descriptor analytical function.

$$\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_B] = [f(\zeta(\mathbf{i}_1, T)), f(\zeta(\mathbf{i}_2, T)), \dots, f(\zeta(\mathbf{i}_B, T))] \tag{4.7}$$

4.3.2 DMI taxonomy for perceptual sonic response analysis

In order to develop a holistic analysis method that allows interfacing the VCI4DMI to most instruments, we present next a taxonomy for DMIs based on the sonic characteristic and high level relation between parameters and output, easily identifiable by users. The goal is to have a variety of different analytical strategies for different cases, while at the same time avoiding proposing too many categories that would be difficult to associate to specific DMIs by inexperienced users. Moreover the selection of the analysis type can be also in accordance to specific sonic characteristics that the performer wishes to control.

DMIs presenting major aleatory or stochastic components in the algorithm are excluded from this study because for these cases the parameters vectors $\mathbf{i}_j$ and the timbre descriptor vectors $\mathbf{d}_j$ cannot be uniquely related. Minor stochastic components such as the presence of secondary noisy modulation can still be included taking the average of the perceptual descriptors computed over a longer analysis time interval T. DMIs with causality relation between past inputs and current output, such as those embedding dynamic models, are still theoretically compliant with the DMIs model we adopt here, but the set of unique parameter combinations **I** must be extended accordingly attaching at each unique present-time combination, all the possible sequence of past parameters within a finite time interval τ, as in Equation 4.8. This can easily result in a combinatory explosion so that the cardinality of **I** is extremely high, which has an impact on analysis total time, real-time computational load and memory occupancy.

$$\mathbf{i} = [i_1(t), i_2(t), \dots, i_P(t), \dots, i_1(t-\tau), i_2(t-\tau), \dots, i_P(t-\tau)] \qquad (4.8)$$

The first branch that we consider in the DMI taxonomy discriminates between sound generators and sound processors. All sound synthesis devices belong to the first category, while in the second we have devices such as equalizers, filters, compressors, reverberators, delays, phasers and flangers. Synthesizers are often equipped with simple effects that can be applied to the output sound, while effects are often composed by the cascade of two or more basic effects. Therefore we define the macro categories, to include the composite devices, as follow:

- sound generators: any chain of sound synthesis plus sound processing devices that generates an output sound given as the input control parameters only;

- sound processors: any chain of sound processing devices that, independently of the control parameters, always require an input audio signal to present sound at the output.

The presence of a sustain phase in the global amplitude ADSR envelope discriminate between sustained and decaying sound generators. In the first case after the triggering signal such as a note-on message, the sound is generated indefinitely until another signal, a note-off message, stops the sustain phase and starts the release phase, so that the sound is terminated. In the second case the decay phase sets the amplitude back to zero, therefore note-off messages are irrelevant and the total duration time of the sound is fixed by the attack time plus decay time. The acoustic equivalents for the sustained category are wind and bowed strings instruments, while plucked strings and percussive instrument are typically decaying. We can further fork two categories by the constant or modulated characteristic of the timbre during the sustained phase. While certain DMIs produce a clearly steady timbre over the sustain phase, with invariant amplitude and phases of frequency components, and an identical waveform for every cycle, others generate periodic fluctuations of timbre perceptual characteristics due to the presence of explicit or implicit LFO, or aperiodic texture-like variation.

Sound processors, despite their actual implementation, are perfectly described by a Finite Impulse Response (FIR) filter or by an Infinite Impulse Response (IIR) filter, usually with a decay factor in the feedback loop. Therefore these devices can be fully characterized by equivalent representations in the time and frequency domain. However the parameters they expose and the perceptual modifications they impart to the input sound are in one or the other of these domains. Therefore we introduce the categories of time domain processors such as delays and reverberators, and the category of frequency domain sound processors such as filters and saturators. The second category can be further divided in the same way as the sustained generator group, because the alteration of the input spectrum can be the either constant or varied over time. For analysis of the time domain processors this difference is not significant so that we hold them in a single category. We include hybrid sound processors or borderline cases such as phasers and flangers, in the frequency domain group because the specific analytical technique we use address better the low frequency modulations that they generate. The taxonomy proposed in this section is illustrated in Figure 4.6.

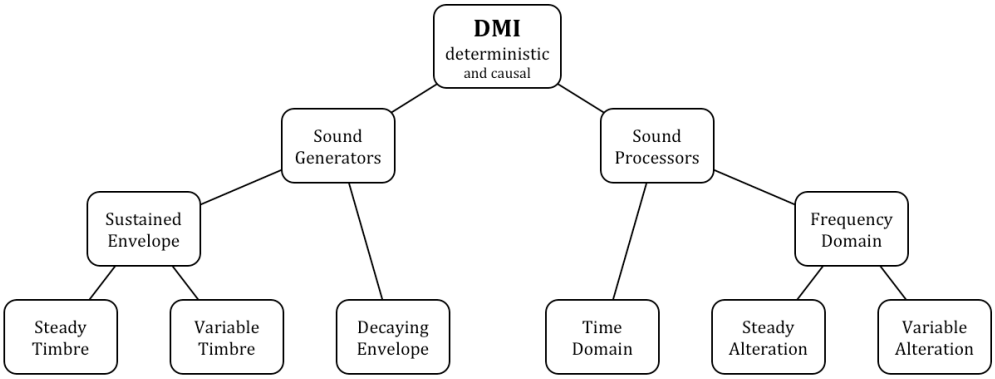

Figure 4.6: DMI taxonomy tree.

4.3.3 DMI parameters to sonic space analysis

In this section we introduce methods to compute the sonic space of specific DMIs, covering the six categories at the bottom of the taxonomy tree in Figure 4.6. In the prototype detailed in Chapter 6, the analytical procedures we describe next are completely automated and present interfaces compatible with any software or hardware DMI. The user interaction is limited to the selection of the analysis mode, which is determined by the DMI position in the taxonomy tree, by the selected control parameters effect on timbre perceptual characteristics, and by preference on interface control aspects. Unlike similar works, here the aim is not to fully characterize the timbre of an instrument, but to capture in the sonic space those perceptual aspects that change with variation of the DMI control parameters.

4.3.3.1 Sound generators

For all generator cases the timbre descriptor vectors $\mathbf{d}_\mathrm{j}$ are computed processing a set of timbral descriptors computed for a sequence of overlapping windows within a time interval T, as generalized in 4.7 and detailed in 4.9.

$$\mathbf{d}_\mathrm{j} = f\left(\zeta(\mathbf{i}_j, T)\right) \approx f_{mode}\left(f_{desc}\left(d(\mathbf{i}_j, 1)\right), \ldots, f_{desc}(d(\mathbf{i}_j, T))\right) \qquad (4.9)$$

In Equation 4.9 $f_{mode}()$ is specific to the analysis mode while $f_{desc}()$ computes the following descriptors: loudness of the 25 Bark critical bands and optionally spectral moments and spectral flatness, as described in Section 3.1.3.2. The centroid and the flatness approximate the brightness and noisiness of the timbre respectively. The number of bark bands is flexible and based on the formula in 4.10 (Traunmüller,

1990), while the loudness of each band is estimated applying the equal loudness contours and the cube root, as described for the PLP computation.

$$bark = \left(\left(\frac{26.81 \cdot f_{Hz}}{1960 + f_{Hz}}\right) - 0.53\right) + a$$
$$\text{where } \begin{cases} bark < 2 & a = 0.15 \cdot (2 - bark) \\ 2 \leq bark \leq 20.1 & a = 0 \\ bark > 20.1 & a = 0.22 \cdot (bark - 20.1) \end{cases} \tag{4.10}$$

4.3.3.1.1 Steady timbre

In this analysis case, after triggering the sound generation and once attack and decay phases terminate, we maintain the sustain phase to the end of the analysis interval. Then for each parameter combination $\mathbf{i}_j$ we send the parameters unique combination to the DMI, compute the descriptors over a sequence of windows, and we set the relative $\mathbf{d}_j$ to the mean of the descriptors, as in 4.11.

$$\mathbf{d}_j = \frac{1}{W_T}\sum_{k=1}^{W_T} \mathbf{d}_{j_k} \quad \text{where } \mathbf{d}_{j_k} = f_{desc}\left(d(\mathbf{i}_j, k)\right) \tag{4.11}$$

$$T = \frac{(step \cdot W_T) + (win - step)}{SR} \tag{4.12}$$

In 4.11 the W_T is the number of analysis windows fitting the analysis interval T, that in turn depend on the window size and step, by default at 4096 and 1024 respectively, and by the sampling rate SR. Equation 4.12 expresses the relation between W_T and T. Ideally for invariant timbres, sampling a single window would be sufficient, but this may be affected by the generation of minor measurement noise, because we cannot synchronize sound generation and analysis with sample precision. Thus for every $\mathbf{i}_j$ the relative alignment of the window to the DMI waveform will be random. Taking the average over a small number of overlapping windows, typically 8 to 32, has little impact to the overall analysis time but it minimizes the measurement noise.

4.3.3.1.2 Variable timbre

The procedure to compute the timbre descriptors $\mathbf{d}_{j_k}$ over a sequence of windows is identical to the one described above, but in general we sample a larger number of

windows. In this case, for each descriptor in $\mathbf{d}_{j_k}$, $f_{mode}()$ computes mean, range and oscillation frequency. The range is simply computed by the difference between the maximum and the minimum found in the sequence of the $\mathbf{d}_{j_k}$. The oscillation frequency is estimated by autocorrelation, while the maximum and minimum detectable periods are computed as in Equations 4.13 and 4.14. To provide a more robust detection we assume that at least two complete periods must fall within the autocorrelation window. The dimensionality of the resulting $\mathbf{d}_j$ is therefore three times the number of descriptors, as in 4.15. If no sharp peaks are detected in the autocorrelation, the oscillation of a specific feature is likely not periodic, and we set it to the median oscillation frequency of the other features the specific $\mathbf{d}_j$. This workaround avoids the generation of false large variabilities in the overall features oscillation frequency across the whole parameters set **I**, which may otherwise undermine the following dimensionality reduction stage. However if more than half of the total autocorrelation show no sharp peaks, we conclude that the overall timbre variation is aperiodic, and we remove the frequency component from 4.15.

$$f_{Hz}^{min} = \frac{2}{T} \tag{4.13}$$

$$f_{Hz}^{max} = \min\left(\frac{2 \cdot SR}{step}, \frac{SR}{win}\right) \tag{4.14}$$

$$\mathbf{d}_j = [\mathbf{d}_j^{mean}, \mathbf{d}_j^{range}, \mathbf{d}_j^{freq}] \tag{4.15}$$

4.3.3.1.3 Decaying envelope

Without a sustain phase the sequence of descriptor vectors $\mathbf{d}_{j_k}$ is computed from the beginning of the attack phase to the end of the decay. The analysis interval T has to be equal to the sum of the attack and decay times. If these change within the parameter combination set **I**, T has to be equal at least to the sum of the maximum attack and decay values. In this case for the $\mathbf{d}_j$ we consider the entire temporal sequence of the descriptors over the amplitude envelope, as in Equation 4.16. Here the curse of dimensionality could be mitigated with increasing the analysis step size and reducing the number of Bark critical bands.

$$\mathbf{d}_j = [\mathbf{d}_{j_1}, \mathbf{d}_{j_2}, \ldots, \mathbf{d}_{j_{W_T}}] \tag{4.16}$$

For sustained instruments, if the parameters in **I** also determine sonic or duration changes also in the attack, decay or release phases, it is possible to run two separate analyses. The first looks at the timbre during the sustain phase, while the second measures the temporal evolution of the descriptors over the attack, decay and release phase, sending the note-off message before the end of the decay phase, so that the release phase will start straight after. Again to prevent excessive dimensionality, the number of Bark critical bands can be lowered, and does not necessary have to be equal in the two analytical stages. At the end the two timbre descriptor matrices **D** can be combined in a single one, extending the features set for each entry associated with a vector $\mathbf{i}_j$. Other borderline situations are represented by DMI with sustained steady timbre but with control parameters enabling an LFO, thus the variable timbre analysis mode should be used, and vice versa, sustained variable timbre with parameters not chaining depth and period of oscillations can be analyzed with the first analysis mode. Moreover the envelope analysis mode does not necessarily have to extend over the whole ADSR, but can be focused into a smaller temporal sub window. In Figure 4.7 there are several examples of timbre descriptors $\mathbf{d}_{j_k}$ relative to a single $\mathbf{i}_j$, computed over a sequence of window before the respective $f_{mode}()$ compression, where each colored line represents a single descriptor. On the first row there are examples of sustained steady timbres, where is evident, except for the left case, minor fluctuations of the descriptors around a central value. The second row shows variable periodic timbres in which most descriptors are oscillating at the same fundamental low frequency. The third row shows the descriptors over decaying envelopes with sharp attack and on the fourth there are variable aperiodic texture-like timbres. In most cases it is possible to observe that different descriptors $\mathbf{d}_{j_k}$ can present different temporal trends within a single parameters set $\mathbf{i}_j$.

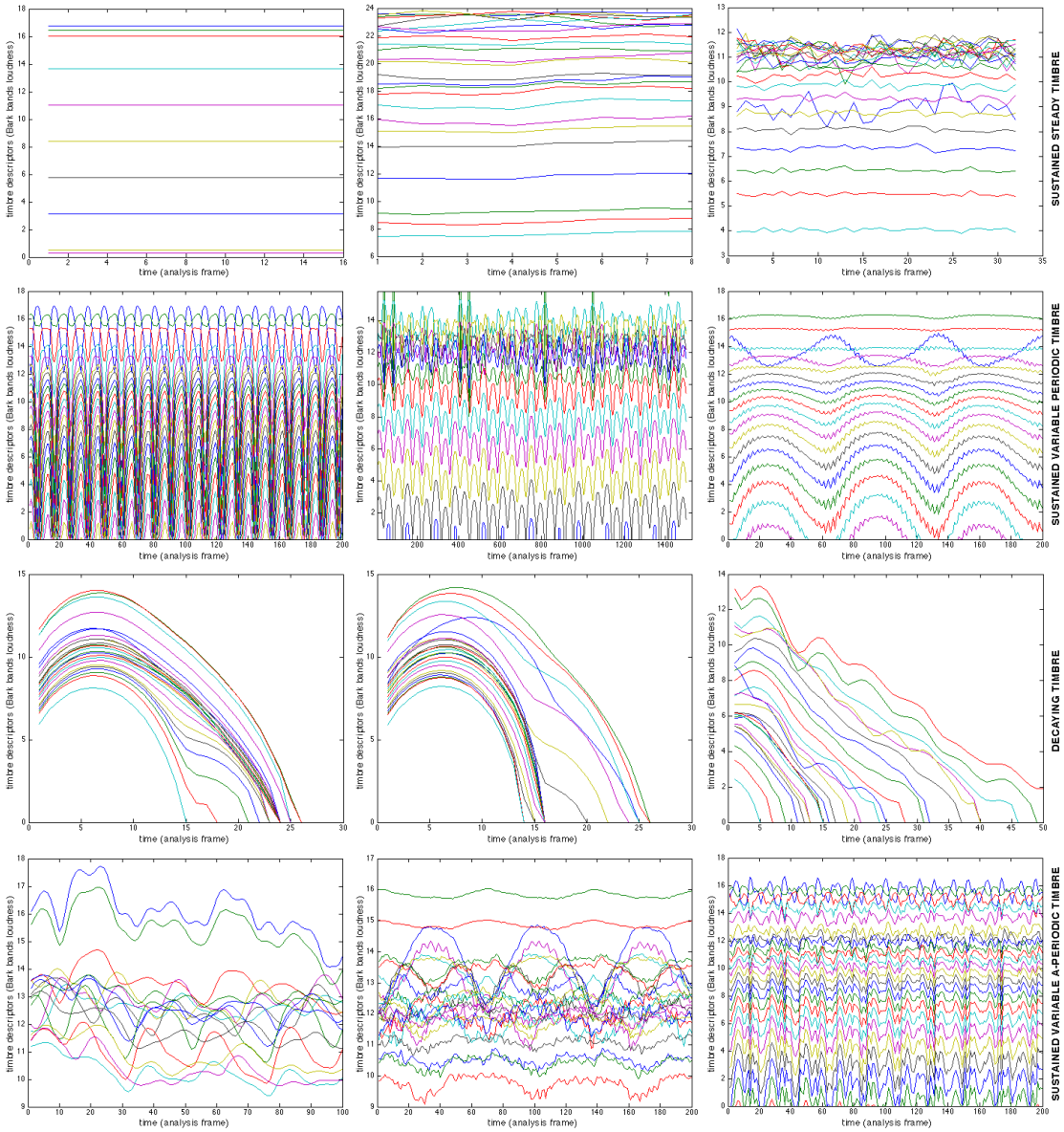

Figure 4.7: Color-coded Bark bands loudness describing the timbre over a sequence of analysis frames. Three examples on each row for sustained steady timbre, sustained variable periodic timbre, decaying timbre, sustained variable aperiodic timbre.

4.3.3.2 Sound Processors

Sound processor devices are usually less complex than generators and expose fewer control parameters, which are related to signal processing variables usually perceptually closer to the acoustic variation they impart to the output sound. Techniques to accurately measure the transfer function or the frequency response of a digital filter exist, but these representations do not hold significant perceptual relevance. The concept of timbre for a sound processor is ambiguous since they do not generate any sound, but the output signal is perceptually different from the input

one. Some characteristics of the input timbre can be smoothed, boosted or modulated. In other cases we have the clear perception that the sound is placed in a different acoustic environment. The equivalent of the timbre for a sound processor can be defined only in relation to an input sound, because the perceptual alteration depends on original sound characteristics too. For instance the perceived acoustic effect of a low pass filter applied on a bass guitar and on a violin is drastically different, even though the cutoff frequency is identical. Therefore we hold the same methodology introduced in this chapter for deriving a model of the sonic response, determined by a variation of a subset of the input parameters. We still imply no prior knowledge about the device but we need to define an input signal, with a time invariant timbre, to run the analysis on the output. Unless the user provides a specific input signal, we use generic test signals to fully characterize the sound processor response.

4.3.3.2.1 Frequency domain steady and variable alteration

For sound processors with perceptual relevance in the frequency domain we use input signal with full spectrum such as white, pink or brown noise, in order to analyze the instrument response at every frequency. White noise has a flat spectrum in which all frequency components are presented to the digital filter with equal energy. The power density of the pink noise falls off at -10 dB/octave and is proportional to $1/f_{Hz}$. The spectrum is flat on a logarithmic scale since bands with width of equal intervals, such as octaves, have equal power. Pink noise is perceptually flat on the frequency domain and therefore preferred for the analysis. Brown noise has a steeper power density decrease, equal to -6 dB/octave, or proportional to $1/f_{Hz}^2$. A sound processor with fixed noise signal at the input can be considered and analyzed as a sound generator. The analytical procedures described in 4.3.3.1.1 and 4.3.3.1.2 for steady and variable timbres can be applied here without modification. In this context we consider only frequency domain sound processors without ADSR envelopes, but the same method used for decaying sound generators can be extended and applied. External strategies are used for changing or modulating over time a sound effect amount such as send and return channels or dry and wet mix. Averaging the descriptors for each $\mathbf{i}_j$ over a larger analysis interval T is here crucial due to the implicit noisy characteristics of the input and output.

4.3.3.2.2 Time domain

Sound processors in this category generate delayed and filtered replicas of the input signal. The characteristic of the output sound changes over time also with fixed input parameters, and may become stable only after some time if the input signal is invariant. Thus the techniques presented above are neither computationally and perceptually adequate. These effects are used to simulate acoustic environments, generating auditory cues suggesting a certain position, distance, movement of the sound source within a space characterized by sound reflections. Even though more efficient implementations exist, the output sound is the result of the convolution in the time domain of an Impulse Response (IR) with the input signal. An IR characterizes a acoustic environment from the perspective of one location, and in digital effects the control parameters allows for the modification of environmental properties such as virtually moving the walls apart, changing the surface material, or morphing the resulting IR. Auditory perception of the sound source distance is based on intensity, spectrum, and binaural cues, and in closed space the direct to reverberant energy ratio is an additional location cue, where the direct sound is considered to be that which arrives within the first 2-3ms only for impulse-like sounds (Chowning, 1971; Zahorik, Brungart, and Bronkhorst, 2005). A room acoustic response is determined by size, shape and material, and is perceived from the characteristics of the early reflections and late reverberation. The transition instant between early and late reflections is dependent on the room characteristics as well. Moreover the reflections can also contribute to color the perception of the original timbre (Brüggen, 2001). Reflections arriving with delay greater than 100ms are interpreted as different sounds, thus we perceive an echo. With a lower delay we are in presence of a reverberation because the original and reflected sounds blend together and are perceived as a single prolonged sound.

All these perceptual characteristics are embedded in the IR, and also visible in the time domain graphical representation. For the analysis of this DMI category we start measuring the IR for every parameter combination $\mathbf{i}_j$. The input test signal is a Dirac or Kronecker delta, depending on the analogue or digital measuring domain, and we record the output, at audio sampling rate, for a duration T, which is set to the longest IR resulting from any parameter combination in $\mathbf{I}$ or to a user selected temporal range. We set the mix of the sound processor to 50% dry and 50% wet so that the original impulse is present at the output, in order to use it to perfectly align, with sample precision in the digital domain, all the recorded IR. The descriptor vectors $\mathbf{d}_j$ are set to the loudness of the whole IR approximated by the cube root of

the energy, but to reduce the excessive dimensionality we perform a down-sampling of the IR recording to 8KHz from the original sampling rate. As an alternative for reverberators it is possible to compute a compact set of features from the IR that includes: the total energy, the T_{60}, amplitude and temporal position of the maximum, number of peaks, slope and intercept of the decay line. The T_{60} is the time required for the reflections to decay 60 dB. We observed that these descriptors generated a sonic representation poorly discriminative in the high dimensional space in **D**, when the IR variations are minimal. The same is observed with delay-like sound effects, which mostly present zero samples and few Dirac delta replicas in the IR, especially if measured within a digital system. In these two scenarios using the down-sampled IR leads to a better sonic representation. In Figure 4.8 there are six examples of descriptors $\mathbf{d}_j$ related to the loudness of the whole IR down-sampled at 8KHz, for a reverberator and simple delay with different input parameters, on top and bottom row respectively.

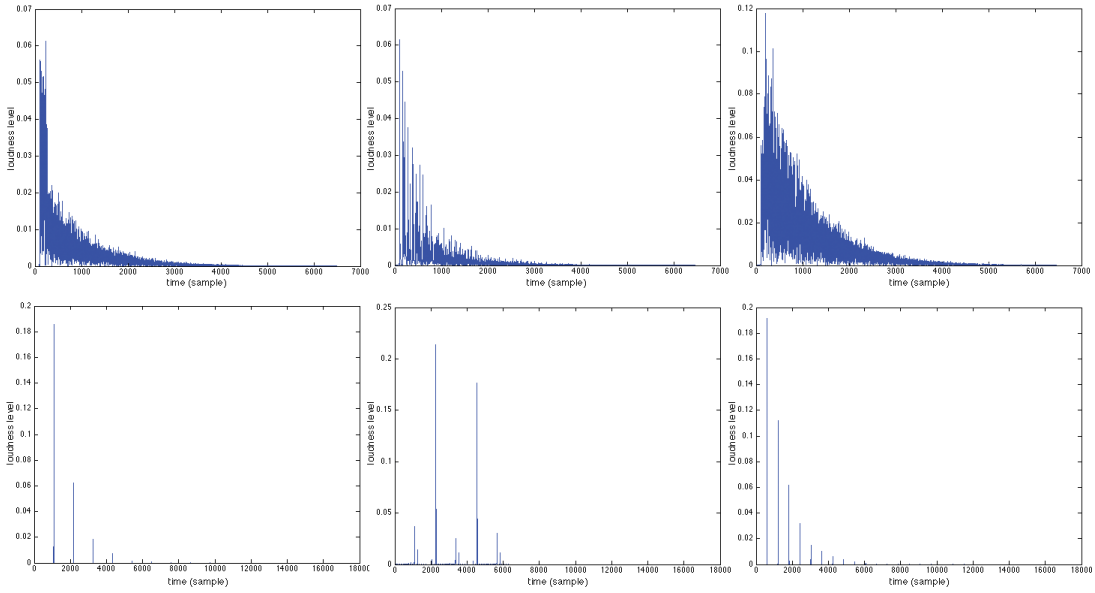

Figure 4.8: Impulse response loudness for a reverberator (top) and a delay (bottom) with different input parameters.

4.3.3.3 Channel, pitch and velocity

DMIs usually support at least a stereo output, while the analysis we described is based on a single channel. However the outputs are either derived from stereo imaging techniques applied to a mono signal or by independent and parallel synthesis or processing chains, but attached to the same control parameters. In both cases the two signals present differences, but from individual listening the timbre perception is

identical. Thus we analyze only a single channel by default, but in a different context the user can run the analysis on the sum of the individual channel signals, or on each signal individually and then combine the two descriptor matrices **D**.

Synthesizers are often played via a piano-like keyboard, and therefore the generated sound depends also on which key is depressed and on the pressure applied on the key. In the MIDI protocol, these represent pitch and velocity parameters respectively. If a DMI exposes these parameters, in the analysis we set them to the typical default values, the note middle C and the velocity 100 out of a maximum of 127. As an alternative, the user can define these reasonably in the middle of the range used in performance. In both cases how representative the sonic space is when pitch and velocity are different can represent a theoretical and practical issue. Previously we discussed how the timbre embraces all acoustic characteristics that are not pitch and loudness, however running the analysis with a pitch a few octaves higher or lower returns quite different sonic spaces. The scientific debate on how to understand the pitch and timbre relationship has been going on for a century. Schoenberg (1922) considers the pitch as a fundamental dimension of the timbre, while according to Von Helmholtz (1954) these are completely separate and independent entities. Other studies suggest that the spectral energy distribution, which determines the timbre, and the pitch are independent in most cases. However this statement holds in harmonic tones in which the pitch is clearly perceived, but is not true for inharmonic spectrum in which the pitch is uncertain or multiple and is determined by the balance of the spectral components (Plomp and Steeneken, 1971; Risset, 1978a, 1978b). We use a single and fixed pitch sonic representation even if this is varied in performance. As an alternative multiple analysis can be performed for different pitch ranges, followed by dynamic selection of the mapping derived from the **D** associated with the currently played chromatic scale interval.

To emulate the behavior of acoustic instruments, velocity is usually mapped to the maximum value of the ADSR amplitude in order to generate louder sound when the key is depressed with more energy. However acoustic instruments generally respond nonlinearly to higher amounts of energy, which usually results in the presence of additional higher order harmonics. Thus often the velocity is also mapped to a filter envelope or other parameters. Variation of the loudness does not greatly affect the timbre perception, while the filter cutoff frequency does. Therefore the sonic space we compute is representative if the velocity is mapped only to loudness related parameters, or if in the performance the velocity values are maintained relatively close to those used in the analysis phase.

4.4 From sonic space to parameter space

The sonic space derived with the methodology presented above is high dimensional and unsuitable to implement musical control. Appropriate dimensionality reduction is vital because, as we described in the previous chapter, the vocal GC provides a number of control signals drastically lower than the dimensionality of **D**. Although with a GC presenting more outputs, the cognitive overhead to interact with spaces of four or more dimensions is not trivial to manage (Van Nort, 2009), even repeating identical gestures to produce similar sounds is challenging. If we reduce the sonic space **D** to a number of components lower than the number of parameters in **I**, and we retrieve the DMI parameter vectors **i** from **D**, then we obtain a dimensionality reduction of the control space plus the adaptation to the parameters-to-sound characteristics of the instrument, without loss in parameter combinations.

4.4.1 Low dimensional sonic space

To measure the intrinsic dimensionality of the sonic descriptor space **D** and to reduce it to $\mathbf{D}^*$, in which the B entries are M dimensional as in Equation 4.17, we use the same techniques presented for the voice data. The use and preservation of the geodesic distance in the reduced space, guaranteed by Isomap NLDR, provides a drastic improvement in the representation of the timbre data, also repeating the early MDS experiments of Grey and McAdams (Burgoyne and McAdams, 2007, 2008).

$$\mathbf{D}^* = [\mathbf{d}^*_1, \mathbf{d}^*_2, \dots, \mathbf{d}^*_B] = \begin{bmatrix} d^*_{1,1} & \dots & d^*_{B,M} \\ \dots & \dots & \dots \\ d^*_{1,M} & \dots & d^*_{B,M} \end{bmatrix} \tag{4.17}$$

Over a set of 34 DMIs analyzed with different modes, we observed that the intrinsic dimensionality of the sonic space is always below 7, and between 2 and 3 in most cases. The intrinsic dimensionality is usually higher when we analyze the entire ADR envelope for decaying sound generators. The dimensionality is independent of the number of DMI control parameters P which ranged between 2 and 8 in our study. However the low dimensional timbre space provided by Isomap concentrates more variance, or energy, in the first few components, which are more discriminative than those obtained with PCA or other techniques. An example of this recurrent observation is in Figure 4.9, in which the histograms show the percentage of variance distribution on each dimension of the lower dimensional space computed with PCA

and Isomap, on left and right respectively, starting from the same high dimensional timbral data. In Figure 4.10 there is a simple but representative example of timbre space derived from a single DMI parameter. The perceptual variation of a filter with variable cutoff frequency applied to a square wave oscillator is reduced by PCA, on the left, to a three dimensional curve, while Isomap, on the right, rearranges the data on a straight mono-dimensional line, where 99% of the variance is concentrated on the first component. The number of entries in the descriptor matrix **D** can be extremely high, as well as the dimensionality, and these are usually higher than those in the vocal-gesture low-level feature matrix $\mathbf{V_G}$. This implies higher Isomap complexity, but it does not affect the VCI4DMI real-time computational cost because on the DMI mapping side no new high dimensional vectors need to be reduced.

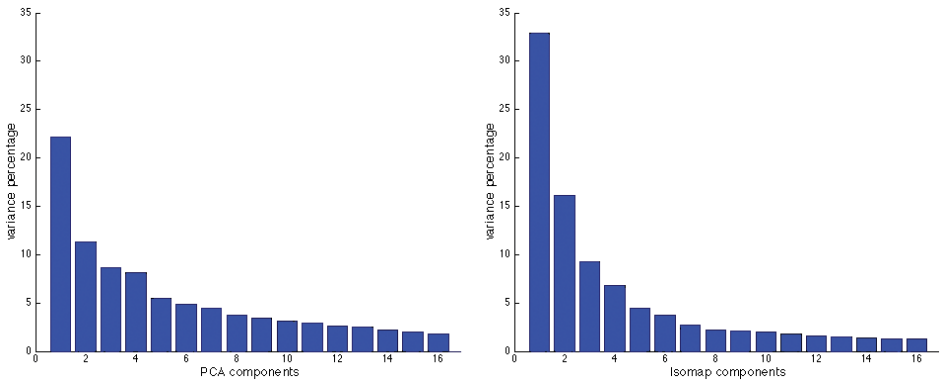

Figure 4.9: Distribution of percentage of total variance for PCA (left) and Isomap (right) individual components applied to reduce the dimensionality of the same timbre data.

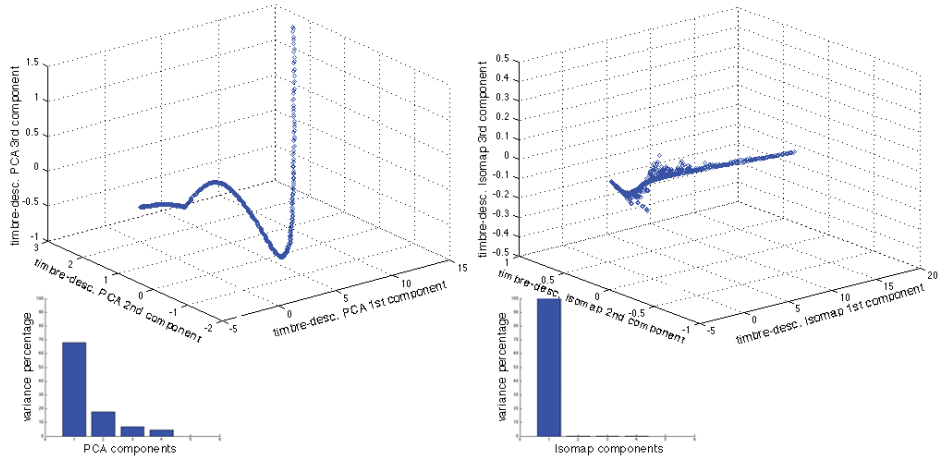

Figure 4.10: PCA vs. Isomap reduction of timbre space generated by single DMI parameter variation, with reduced sonic space (top) and components variance distribution (bottom).

4.4.2 DMI parameters retrieval

The gesture, voice in our case, can generate a vector of DMI parameters from the sonic space simply by using a properly scaled GC output $\mathbf{gc_{out}}$ as a search coordinate in the reduced descriptor space $\mathbf{D}^*$, both M dimensional, and picking the parameter vector $\mathbf{i}$ associated to the closer reduced descriptor vector $\mathbf{d}^*$. In Figure 4.11 we show six examples of DMI sonic spaces reduced to their three principal Isomap components.

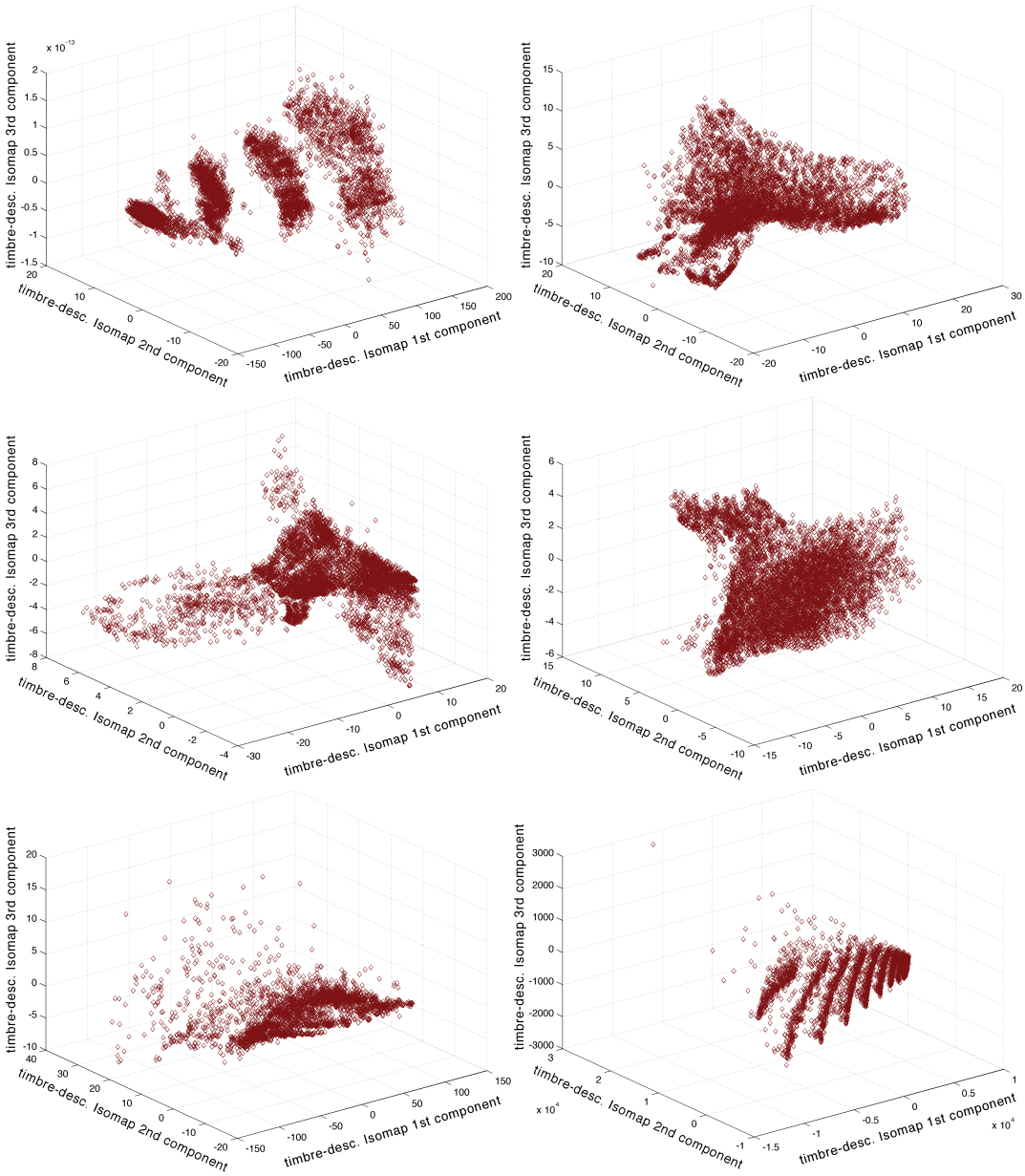

Figure 4.11: Examples of DMI sonic space reduced to the 3 principal Isomap components. The top four spaces are related to synthesizers subject to the variation of four to five parameters, the last two are related to reverberator and low pass filter parameters variation.

For simple cases is possible to heuristically find the relationship between perceptual timbre, visual sonic space representation, and DMI parameters. For example in the low pass filter sonic space of Figure 4.11, generated changing only cutoff frequency and resonance of a low pass filter, and analyzed with a relative low parameter resolution, the resonances span along the clustered parallel arches, each representing a constant cutoff frequency. The DMI control method described above is intrinsically affected by a serious drawback. There is no guarantee that the relationship between perceptual timbre and parameters is bijective. Even though our model creates a unique reduced descriptor $\mathbf{d}_j^*$ for each parameters unique combination $\mathbf{i}_j$, the relation may not be unique in the other direction, which is used here to retrieve the DMI parameters. Different combinations can generate identical or similar sounds, with relative coordinates very close in the descriptor space **D**, but far in the parameter space **I**. In theory this is not an issue because the sound remains equal even if the parameters are different. However in most DMI algorithm implementations, wide step parameter variation generates output glitches, unless the instantaneous and large value parameter variations are applied to synthesis subsections not contributing to the output timbre at the moment. For example in the serial modulation chain of oscillators of a FM synthesizer, if the amplitude of a modulator is zero, the parameter of the upstream oscillators can be arbitrarily varied without changing the synthesized sound. Moreover, as we will detail later, to smooth the VCI4DMI output for the DMI we interpolate linearly in the time domain between consecutive values at a higher rate than the parameter generation. The non-bijective relation is thus even more detrimental here because for a short interval during the temporal interpolation, the DMI parameters are likely to determine a timbre inconsistent with the target one. This shortcoming is addressed by the specific mapping technique that we present in the next chapter.

4.4.3 Analysis parameters resolution and spatial interpolation

The total time T_{total}, in 4.18, to measure the descriptor matrix **D** is linearly proportional to the number of unique combinations B, which in turn depends on the range and resolution of individual parameters, as in 4.6. For the time domain sound processors the term T in Equation 4.6 is equal to the maximal length of the IR, while for all other modes is described in Equation 4.12. The term T_{adj} is the interval that we wait before starting sampling analysis windows after the DMI receives a new parameter vector $\mathbf{i}_j$. This interval, manually defined, allows the sonic output to

stabilize or in ADSR mode it allows the tail of the sound at the previous step to terminate. The T_{sync} is the interval necessary to the automatic analysis chain, described in Chapter 6, to exchange synchronization acknowledge message, usually small and outside the user control.

$$T_{total} = (T + T_{adj} + T_{sync}) \cdot B \tag{4.18}$$

Considering three full range parameters with 7-bit MIDI resolution, we have more than two millions unique combinations. Considering only a single 1024 samples analysis window and no additional adjustment or sync time, it requires more than 12 hours to measure **D**. Moreover large numbers of parameter unique combinations impact on the VCI4DMI real-time memory occupation and computational load. Unless the user aims for a full characterization of the DMI, it is possible to reduce the analysis time without losing DMI control precision by limiting the parameter resolution in the analytical stage and interpolating at the output. We apply the IDW, so that the parameter vector $\mathbf{i}_{\boldsymbol{out}}$ depends on the values of the N neighbors $\mathbf{i}_j$, determined and weighted by the relative distances in the reduced sonic space $\mathbf{D}^*$, described by Equations 4.19 and 4.20. The strategy to determine the N neighbors and the function $m()$ that maps the GC output to the reduced space $\mathbf{D}^*$ is presented in the next chapter.

$$\mathbf{i}_{\boldsymbol{out}} = \frac{\sum_{i=1}^{N} \mathbf{q}_i\left(m(\mathbf{gc_{out}})\right) \circ \mathbf{i}_i}{\sum_{i=1}^{N} \mathbf{q}_i(\mathbf{d}^*)} \tag{4.19}$$

$$\mathbf{q}_i(\mathbf{d}^*) = \frac{1}{\left\| m(\mathbf{gc_{out}}) - \mathbf{d}_i^* \right\|^{\rho}} \tag{4.20}$$

The IDW worsens the effect of the non-bijective parameters-to-sound relationship. This is illustrated in Figure 4.12, where the coordinates determined by the mapping of the GC output $\mathbf{gc_{out}}$, in blue, falls in the middle of the four $\mathbf{d}^*$ neighbors, in red, which generate similar sounds but with entirely different parameter combination, which we always consider in the scaled range [0,1]. In this case the IDW output returns a parameters set $\mathbf{i}_{\boldsymbol{out}}$ that is far from all the $\mathbf{i}_j$ associated with the neighbors, and it likely generates a sound inconsistent with that position in the sonic space. The IDW interpolation between sonic and parameter space assumes that a sound in a specific sonic space position inherits characteristics of the immediate neighbors, but it

requires that these are relative neighbors in the parameter space too, which is not always guaranteed. This issue is addressed as well within the mapping technique in the next chapter.

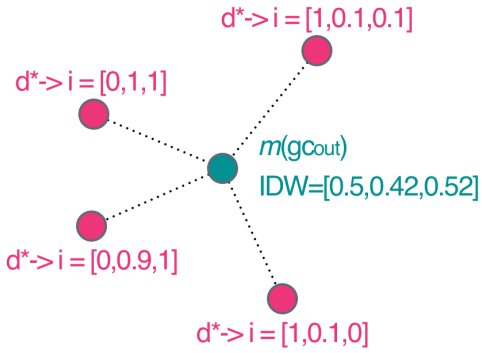

Figure 4.12: Illustration of the detrimental effect of the non-bijective sound-to-parameters relationship on IDW interpolation between sonic and parameter spaces. Different parameter sets with very different values generate similar sounds, but interpolated values between those parameter sets may be in radically different parts of the sound space as represented by the sound descriptors.

In real use cases we observed that the IDW technique is effective to make up the limited analysis resolution. If the analysis time is an issue and needs to be limited, it is advisable to reduce the parameters resolution but still allow adequate time per combination to get a precise timbre representation, allowing longer adjustment time and more sampling windows. Limiting the total time by just reducing the analysis time per parameter combination T, may generate a less precise and noisy measurement later reflected in the sonic spaces. Regardless of the number of DMI variable parameters, we usually do not get control or mapping improvements with more than 10K entries in the descriptor matrix **D**, which provide reasonable analysis time and limited computational load in the runtime VCI4DMI. Due to the interpolation we obtain satisfactory usability also with **D** containing as few as 1k entries. On modern CPUs, common synthesis algorithms demand only a small fraction of the computational power, and therefore it is theoretically possible to shorten the analysis time to a small fraction by running offline at a rate higher than the real-time audio. However here we consider the DMIs as a black box over which we have no control and access to internal algorithms, which requires that the analysis be done at the audio rate at which the DMI generates sound samples.

4.5 Evaluation and validation

In order to validate our DMI analysis method and the strategy to reduce the instrument control space, next we present results for a set of 34 different DMIs implemented in 14 different state-of-the-art commercial software digital instruments. The instruments included in the set, as well as the parameters selection and settings for the perceptual sonic analysis were selected for their sound aesthetic characteristic, breadth of timbre morphing, and complexity of multi-parametric control challenging to achieve with traditional interfaces. Most of these were also used in performances context described in Chapter 6. Although in this section we draw generic conclusions, the results are presented for individual instruments, because a statistical summary here would be misleading due to incompatible DMI spaces sizes, entries, parameters, and analysis. Instead, we identify trends in the extended set of measurements.

In Table 4.1 we present the DMI evaluation set, detailing for each case the generator/processor type, the synthesis/processing algorithm, the number of variable parameters, analysis mode, number of descriptors, analysis frame per parameter combination and resulting number of entries in **D** and **I**. Each instrument is associated with a numeric ID that will be used to address the specific DMIs in other tables. Table 4.1 also includes the intrinsic dimensionality of the timbre descriptor space **D**, the number of Isomap components holding 95% of the total variance of $\mathbf{D}^*$, the maximum correlation found between any descriptor and any parameter within **D** and **I**, and their relative indexes. As expected the results are strictly dependent on the specific DMI context, and this is confirmed as well from the further measurements detailed later. There is no relationship between numbers of parameters and characteristics of the sonic space, providing motivation for our strategy of providing DMI dependent mappings.

The intrinsic dimensionality measure and Isomap dimensions variance shows that a number of components in **D** drastically lower than the space dimensionality, usually below 4 and not related to the number of control parameters, is sufficient to represent the timbre variation of the DMI sound. The curse of dimensionality is evident when a high number of descriptors per state are included in **D** such as for the decaying timbre and IR analysis modes. However in most cases forcing the reduction to two or three dimensions still provides a usable mapping.

DMI ID	*type*	*algorithm description*	*# variable parameters*	*analysis mode*	***d*** *&* ***D*** *dimensionalities*	*analysis* ***d*** *per* ***i***	***D*** *and* ***I*** *cardinality*	*idim(**D**)*	***D*** *Isomap reduced 95% variance dim*	*max param to* ***D*** *correlation*	*max correlation param. and desc. idx*
1	generator	FM	6	steady	25-25	8	6399	5.01	9	0.80	2-25
2	generator	virtual modular	4	steady	24-24	100	1296	2.61	1	0.93	1-23
3	generator	PCM smp. based	4	steady	25-25	16	3584	3.49	3	0.85	2-15
4	generator	FM	6	steady	27-27	50	4840	2.95	1	0.77	1-27
5	generator	virtual modular	2	steady	21-27	16	2601	2.99	8	0.26	1-6
6	generator	virtual analog	3	steady	15-15	16	528	2.15	3	0.91	3-1
7	generator	granular	3	steady	25-25	20	1001	2.49	8	0.82	3-18
8	generator	virtual modular	4	steady	24-24	25	1125	2.67	1	0.93	1-23
9	generator	FM	3	steady	25-25	32	1331	3.76	1	0.97	2-15
10	generator	FM	3	steady	24-24	64	1248	2.28	1	0.89	2-7
11	generator	FM	3	steady	25-25	32	616	2.66	1	0.90	2-18
12	generator	physical model	3	decaying	30-1500	50	125	40.2	16	0.95	3-28
13	generator	wavetable based	7	decaying	25-1250	50	11664	6.03	357	0.90	6-1
14	generator	granular	4	decaying	16-1600	100	4116	11.3	716	0.57	2-4
15	generator	wavetable based	4	decaying	13-650	50	5324	3.78	2	0.98	4-1
16	generator	PCM smp. based	3	decaying	22-1100	50	385	2.39	8	0.99	2-2
17	generator	virtual analog	3	decaying	25-750	30	715	4.37	128	0.99	3-2
18	generator	virtual analog	4	decaying	25-1000	40	576	7.03	77	0.91	1-20
19	generator	granular	4	decaying	25-800	32	3402	40.8	385	0.87	3-24
20	generator	wavetable based	4	decaying	25-875	35	900	6.02	193	0.92	3-16
21	generator	physical model	8	decaying	25-1000	40	14400	67.2	4	0.92	3-2
22	generator	FM	6	variable	30-60	50	4840	2.82	1	0.77	5-27
23	generator	FM	4	variable	29-58	200	1000	2.47	6	0.95	1-20
24	generator	virtual analog	3	variable	29-87	75	252	2.52	8	0.93	1-19
25	generator	physical model	3	variable	27-81	1500	343	3.10	1	0.74	3-16
26	processor	parametric EQ	4	fq. steady	24-24	16	270	4.98	3	0.99	3-1
27	processor	pysical resonator	4	fq. envel.	16-512	32	648	4.70	10	0.86	3-3
28	processor	flanger	2	fq. steady	28-28	16	378	2.28	1	0.97	1-19
29	processor	virtual amp	8	fq. steady	25-25	32	7200	4.09	4	0.80	7-10
30	processor	low pass	2	fq. steady	28-28	16	378	2.28	1	0.97	1-19
31	processor	delay & reverb	3	time IR	24k-24k	1	288	23.9	93	0.97	n.a.
32	processor	delay	2	time IR	16k-16k	1	315	61.2	27	0.45	n.a.
33	processor	delay & reverb	3	time IR ft.	7-7	1	220	1.39	6	0.90	2-3
34	processor	reverb	3	time IR ft.	7-7	1	1330	1.83	1	0.84	1-2

Table 4.1: Characteristics of the DMI evaluation set, including parameters-to-sound analysis settings, and sonic space dimensionality.

The maximum correlation parameter-descriptor is usually close to 1. Such a value represents almost a direct sonic control of a perceptual timbre dimension. This measurement, properly extended, can provide useful feedback on parameters-to-sound relationships. Further, we observed that parameters might also have a low maximum correlation with any timbre descriptors, which indicates control space dimensions with little sonic relation, which we aim to hide from the user.

Additional measurements for the DMIs sonic spaces Isomap reduction to 2D and 3D are detailed in Table A.1 in the Appendix A, and summarized in Figures 4.13-15 below. With few exceptions and regardless of the dimensionality of **D** the lower dimensional sonic space holds a large percentage of the total information, measured as the components' variance. This is usually approximately 80% for 2D and 90% for 3D, visible in the rising trend in the left plot of Figure 4.13 and detailed in the first column of Table A.1. These suggest that the drastic dimensionality reduction results in only a small loss in sonic space discrimination. After the reduction some entries may overlap with others descriptor vectors. Browsing the space with a fine step allows us to find at least a coordinate nearest neighbor to every $\mathbf{d}_j^*$ in $\mathbf{D}^*$, and all parameter combinations in **I** can be retrieved, but such a fine step resolution is unlikely in a musical interface. The practical parameter loss is measured using a larger step resolution, and results are illustrated in the right plot of Figure 4.13 and detailed in the second column of Table A.1. In particular we sampled the space over a grid that divides each component of the sonic space into 128 identical intervals. The rising trend in the plots shows that losses are lower when the sonic control is implemented in the 3D reduction rather than 2D. Losses are still high for some DMI cases and need to be addressed with a more elaborate mapping strategy.

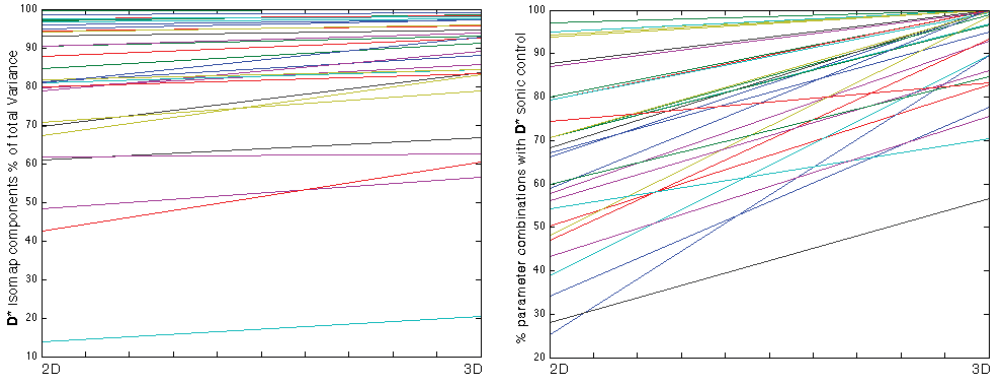

Figure 4.13: 2D and 3D Isomap reduced sonic space percentages of total variance (left) and percentage of combinations covered controlling the DMI from the sonic space (right). The rising trend shows that the 3D cases outperform the 2D ones, especially for the space coverage. Each colored line represents a different DMI case.

In the left plot of Figure 4.14 and in the third column of Table A.1 we show the average distance of every $\mathbf{d}_j^*$ from its nearest neighbor $\mathbf{d}_{j-NN}^*$ in the original and reduced spaces. The distance measurement is normalized to the space dimensionality and thus the two numbers are comparable. As expected, average distance decreases with the spatial dimensionality reduction, which clusters the data. We repeated the

same measurement computing the average normalized distance between the parameters vectors $\mathbf{i}_j$ and $\mathbf{i}_{j-NN}$ related to the neighbor pairs in $\mathbf{D}^*$. The results, illustrated in the right plot of Figure 4.14 and detailed in the fourth column of Table A.1, show an opposite trend that demonstrates two potential drawbacks of this control strategy. First, the absolute average distance in the parameter space is generally larger than the one in the sonic space, reflecting the situation when similar sounds may be generated with very different parameters sets. Secondly, the average distance in the parameter space also shows an opposite trend, increasing while reducing the dimensionality. This suggests that when reducing the dimensionality the neighbors of $\mathbf{d}_j^*$ might change, and this adversely affects the overall non-bijective parameters-to-sound relationship.

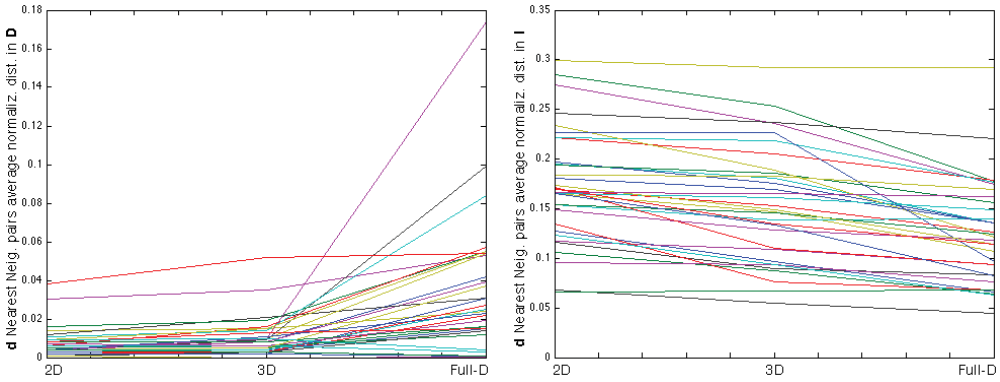

Figure 4.14: Pairs of nearest neighbor entries in sonic space (2D, 3D, and full dimensional) normalized average distance (left), and normalized average distance of the related parameter vectors (right). The opposite trend in the two plot shows that the dimensionality reduction contributes to clustering the data and worsening the non-bijective parameters-to-sound relationship issue. Each colored line represents a different DMI case.

We mentioned that the non-bijective parameter-to-sound relationship issue is more problematic when using IDW output interpolation for the DMI parameters, as demonstrated in the results detailed in the last column of Table A.1 and illustrated in Figure 4.15. The falling trend in the plot shows that the difference between the parameter outputs interpolated using distance weights from original sonic space and from the reduced space is generally higher for lower reduction. This has been measured by generating 1000 random coordinates in $\mathbf{D}$, projecting these also to $\mathbf{D}^*$, and using the set of points for interpolating the parameter output from both spaces. The interpolation here is performed with a number of entries equal to 1% of the space cardinality. The DMI control from reduced sonic space is possible and provides effective reduction of the control space for any instrument. The drawbacks identified

here are addressed in the generative and case-specific mapping method presented in the next chapter, which also addresses other issues arising from the arbitrary distribution of the sonic space. Moreover we will introduce a technique to limit the generation of parameter discontinuities due to the non-bijective relationship $\mathbf{d}_j^*$ to $\mathbf{i}_j$.

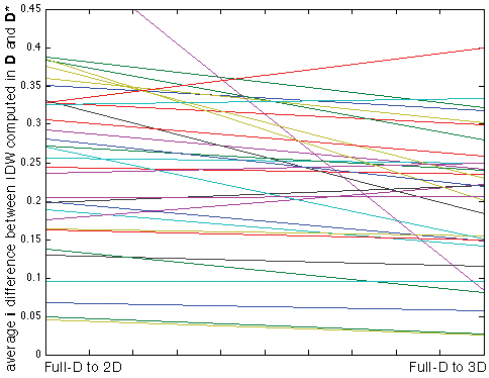

Figure 4.15: Average difference between IDW parameters interpolation using distance weights from original sonic space and from reduced space, for 2D and 3D cases. The differences are high in absolute value and the trend is falling in the majority of cases, showing that the non-bijective parameters-to-sound relationship affects the IDW precision, which is worst for more extreme dimensionality reduction. Each colored line represents a different DMI case.

4.6 Summary

In this chapter we presented a holistic framework to generate a model of the perceptual sonic response of an instrument, determined by the variation of a subset of input parameters. We introduced our DMI taxonomy to provide adequate analytical strategies depending on the DMI characteristics. Moreover we described the control strategy we developed for the VCI4DMI, which is based on the target instrument timbre space and provides adaptation to the parameters-to-sound characteristics and dimensionality reduction of the control space with no combination loss. The strategy is compliant with our interface principles and requirements, because it provides natural and expressive control over an arbitrary number of DMI parameters with the limited number of control signals that the vocal GC provides. We note, however that the sonic spaces generated with the proposed method do not necessarily require the presence of a voice-controlled system. These can be exploited in generic interfaces of any kind for reducing and adapting the control space, or for other analysis-synthesis purposes. In Figure 4.16 we illustrate the summary of the analytical procedure introduced in this chapter.

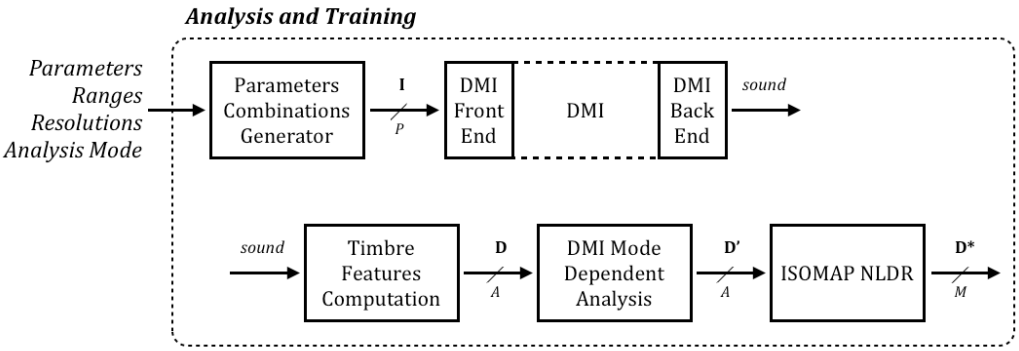

Figure 4.16: Illustrated summary of the DMI parameters-to-sound analysis procedure.

Chapter 5

Mapping and Search in the Sonic Space

In this chapter we describe the mapping strategy that we develop to adapt the projection of the GC output onto the sonic space of a specific DMI, from which we retrieve control parameters. The intermediate mapping layer that we describe here maximizes the overlap between two heterogeneous spaces with arbitrary shape and distributions, representing the voice and the instrument sound respectively. This linearizes the response between vocal-gestural control and variation of the perceptual characteristics of the DMI output. Moreover here we address issues related to the non-unique relationship between DMI parameters and output sound, which implicitly affects any synthesis control strategy based on sonic representations. We begin the chapter with motivating principles and by describing the method for rearranging the data in the sonic space, computed as described in the previous chapter. The projection to the sonic space is then implemented with an ANN trained to model the nonlinear regression from the redistributed space back to the original one. It follows the descriptions of a set of operational modes and parameter retrieval strategies that we developed for the VCI4DMI, which aims to minimize the implicit drawback of the non-bijective relationship between control and sound in DMIs. Finally, we present numerical results and observations to validate the proposed method over a set of cases with real DMIs.

5.1 Intermediate standard mapping layer

In Chapter 3 we showed that vocal-gestures are continuous multidimensional trajectories, while in Chapter 4 we represented the DMIs response with a perceptual sonic space, used to implement transparent device control. Vocal and sonic spaces, represented by $\mathbf{V_G}$ and $\mathbf{D}$ respectively, besides being computed with different techniques, are very likely to differ in dimensionality, number of elements, enclosing shape, and internal data distribution. In the VCI4DMI a proper transition between the two spaces is crucial to provide expressivity, non-trivial control, natural and linear response. However $\mathbf{V_G}$ and $\mathbf{D}$ are case-dependent, and these can vary within same user and instrument, when providing a different set of training-vocal gestures or

targeting other control parameters in the same DMI. Therefore we develop a generative and unsupervised mapping technique to learn the specific transformation from the vocal-gesture set $\mathbf{V_G}$ to the instrument sonic response $\mathbf{D}$, in accordance with our adaptive, generic, and automatic setup principles for the interface.

A mapping function $\mathbf{V_G}$ to $\mathbf{D}$ ties a specific control, expressed in voice timbre variation, with a specific DMI. Moreover due to the arbitrary characteristic of the data in these spaces, definition and computation of the mapping function can be challenging. Therefore we separate the mapping problem into two halves (Wanderley, Schnell, and Rovan, 1998), finding individual transformations of $\mathbf{V_G}$ and $\mathbf{D}$ into isomorphic spaces with fixed and identical distribution. This allows a simple one-to-one mapping to project coordinates of one space onto the other one, and it splits the VCI4DMI learning process into two separated parts, permitting the use of specific voice maps with different instrument maps and vice versa, resulting in a further simplification of the interface setup. A uniformly distributed hypercube represents the simplest choice for the intermediate transformation stage. This has dimensionality M to which voice and DMI data have been already reduced via Isomap, as described in the previous chapters. In Figure 5.1 we illustrate the principle of mapping from the vocal-gesture lower dimensional space $\mathbf{V_G^*}$ to the sonic lower dimensional space $\mathbf{D}^*$ through the intermediate space. The two mapping stages transform vocal-postures into stationary sonic coordinates and DMI parameters, and transform vocal-gestures into trajectories in the sonic space that in turn vary and modulate DMI parameters.

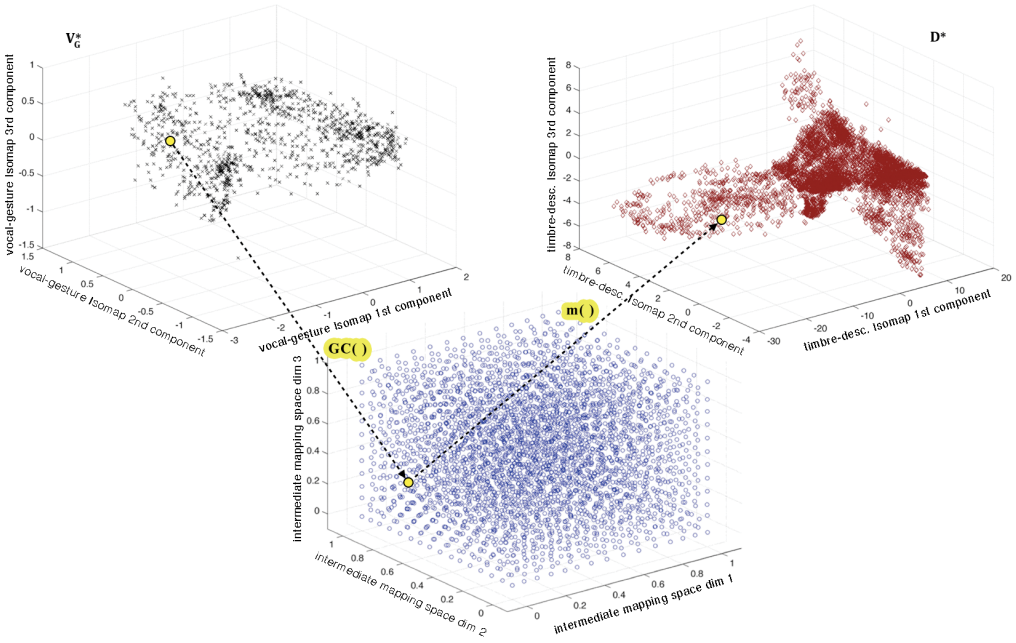

Figure 5.1: Illustration of the mapping flow from lower dimensional vocal-gesture space V_G^* to sonic space D^* through transformations to an intermediate uniformly distributed hypercube.

The SOG method presented in Chapter 3 already implements a redistribution of the vocal-gesture spatial data into a uniformly distributed hypercube, because the particular application and training of the SOM adapts the output lattice to the local topology and density of the data. The interpolated relative position on the output lattice is already a coordinate into the intermediate layer space. Therefore this chapter is dedicated to the learning procedure to derive the DMI specific mapping function $m()$ in 4.19, which maps the GC output $\mathbf{gc_{out}}$ into the DMI reduced sonic descriptors space $\mathbf{D}^*$, which is the missing block in the VCI4DMI chain. The mapping of the two spaces to the same intermediate shape and distribution are conceptually diverse, and thus achieved with different techniques. Firstly, for the voice data the transformation is direct while the VCI4DMI requires the DMI case-specific sonic space mapping function $m()$ to implement the inverse conversion from the uniform hypercube back to the original sonic space, as illustrated in Figure 5.1. The model learned from the voice considers the possible variability that human generated data can present. Moreover it has to respond to new data not presented in the training examples according to the knowledge derived during the learning phase. On the instrument side we have a deterministic digital machine, fully analyzed and characterized, used within the parameters limits priory defined. For the runtime interface there is no new or unknown DMI sonic data, and there are no outliers or undesired entries in the sonic data we use to learn the mapping. The direct projection of the GC output onto the uniform redistributed sonic space $\mathbf{D}_U^*$ can be used for parameter retrieval. However since we interpolate with the IDW method in the output in the parameters space $\mathbf{I}$ using the distances in the sonic space as weights, reverting back to the original $\mathbf{D}^*$ we obtain weights more accurate and coherent than in $\mathbf{D}_U^*$. This is also important because the interpolation from $\mathbf{D}_U^*$ can further adversely affect the non-bijective parameter-to-sound relationship.

5.1.1 Linear perceptual response

If we suppose that the mapping $m()$ is any linear function and it translates $\mathbf{gc_{out}}$ coordinates into $\mathbf{D}^*$, the resulting system will probably present:

- sub-regions with entries in $\mathbf{D}^*$ not reachable for any value of $\mathbf{gc_{out}}$, or a set of $\mathbf{d}^*$ not nearest neighbors to any projected $\mathbf{gc_{out}}$;
- ranges of $\mathbf{gc_{out}}$ projected outside $\mathbf{D}^*$, or projections with a relatively distant nearest neighbor $\mathbf{d}^*$;

- linear and constant rate $\mathbf{gc_{out}}$ trajectories generating a non uniform perceptual sonic variation of the DMI output, or trajectories that are projected to $\mathbf{D}^*$ traverse sub-regions with different $\mathbf{d}^*$ density.

These shortcomings are due to the arbitrary shape and irregular distribution of the reduced sonic space $\mathbf{D}^*$, clearly visible in the examples of Figure 4.11. Browsing $\mathbf{D}^*$ to retrieve the related DMI parameters **i**, with a simple 2D touchpad mapped with linear scale and offset, would result in different sensitivity and unresponsive areas across the surface, detrimental to any musical application of this control strategy. We address these issues at first finding a homotopic transformation of $\mathbf{D}^*$ into the uniform distributed space $\mathbf{D}_{\boldsymbol{U}}^*$, and then the nonlinear DMI specific mapping function $m()$ is implemented with an ANN, trained with $\mathbf{D}_{\boldsymbol{U}}^*$ and $\mathbf{D}^*$ as input-output pairs. Thus the ANN learns the inverse of the homotopic transformation that distributes the arbitrary sonic space into a uniformly distributed hypercube.

In the example shown in Figure 4.10, the analysis is related to a single DMI parameter, which is the cutoff frequency of the low pass filter applied to a waveform with lower energy in the higher harmonics. The relationship between cutoff frequency, on the horizontal axis normalized in [0,1], and the perceptual principal component of the Isomap on the vertical axis, is nonlinear, as represented on the left plot in Figure 5.2. We observe that for cutoff frequency within [0.6,1] the perceptual variation is small, while it is considerably higher in the range [0,0.2], which accurately represents the user's perception when mapping this parameter on a linear fader. As expected, we observe that the principal component of the Isomap is highly correlated with the sound brightness, inverted by the Isomap on the principal component axis. For this simple case in which $\mathbf{D}^*$ is mono dimensional, the three shortcomings presented above can be addressed by a $m()$ derived from the distribution of the reduced sonic space, represented by the histogram in Figure 5.2. There the black line represents the integration of the histogram complement, defined as the difference of each bin value with the maximum bin value, and it represents the inverse of $m()$. This mapping function projects the control signal, in the range [0,1], on the vertical axis of the histogram plot, to a spatial coordinate of $\mathbf{D}^*$, in the horizontal axis, used to search the nearest reduced sonic entry $\mathbf{d}^*$ and retrieve the associated parameter set **i**. With this method we obtain a linear relation between the adapted control parameter and the perceptual feature variation, clearly visible in the right plot of Figure 5.2. The mapping described here is represented in Equations 5.1-3, in which d_1^* and gc_{out1} are the mono dimensional coordinates of $\mathbf{D}^*$ and scalar

output of the GC respectively. The mapping implemented in this manner determines a relationship between gestural controller and reduced sonic space coordinates proportional to the local data density in $\mathbf{D}^*$. The variation across the gc_{out1} range [0,1] results in larger step increase in d_1^* where the local density is high, and smaller where the local density is low. Regions of the reduced sonic space $\mathbf{D}^*$ with high density are determined by many DMI parameters producing very similar sounds and should be browsed at faster pace since the perceptual variation they determine is minimal and vice versa.

$$m^{-1}(d_1^*) = \int hist^{COMP}(d_1^*) \cdot dd_1^* \tag{5.1}$$

$$hist^{COMP}(x) = \max_x(hist(x)) - hist(x) \tag{5.2}$$

$$d_1^* = m(gc_{out1}) \tag{5.3}$$

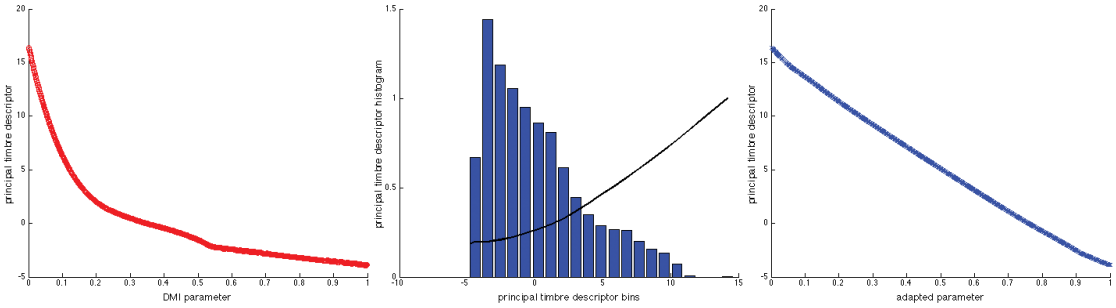

Figure 5.2: Linearization of the perceptual mapping in a single dimension. The DMI parameter versus perceptual principal descriptor (left), adapted control parameter versus perceptual principal descriptor (right), sonic space principal component of the Isomap histogram and mapping function (center).

5.1.2 Sonic descriptors spatial neighborhood coherent redistribution

The mapping approach described above is equivalent to redistributing the data in $\mathbf{D}^*$ using the rank-transform method to uniform distribution, and learning the inverse of this transformation, which represents the DMI specific mapping function $m()$. The rank-transformation changes the coordinate d_1^* into $d_{1-U}^* = z$, in which z is the rank of that d_1^* within the entire set $\mathbf{D}^*$, so that d_1^* is the z^{th} largest value in $\mathbf{D}^*$. The change of coordinate system presents a different range, irrelevant for the mapping

purposes, but preserves the neighborhood of the original space, which is crucial here. The rank-transform is scalar, so for multidimensional $\mathbf{D}^*$ it requires a separate application for each component, but it generates joint uniformly distributed data if and only if each component is statistical independent from the others. Two examples of rank-transformation applied to 2D computer generated data with 16000 entries are illustrated in Figure 5.3. In the top case the data is drawn from a multivariate Gaussian random generator and the rank-transformation has rearranged the data into a uniformly distributed square. This is due to the statistical independence of Gaussian distribution components, and those of any linear combination of Gaussians. In the bottom case the data is drawn from three different Gaussian multivariate generators, with non-statistically independent components. The rank-transformation helps to spread the data but is still far from presenting uniform distribution.

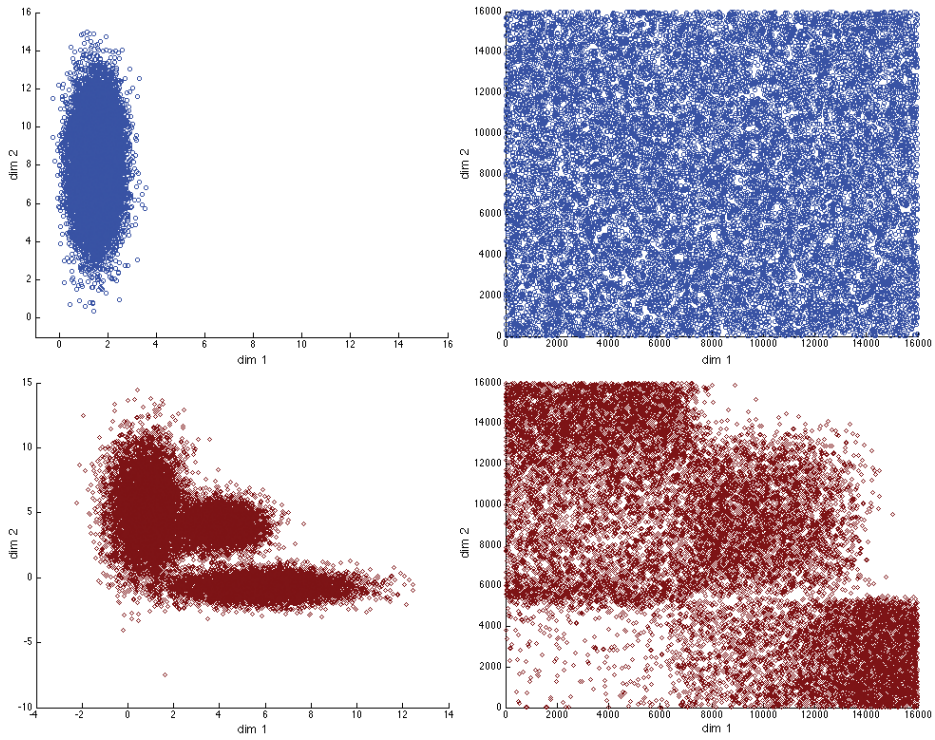

Figure 5.3: Two examples of rank transform applied to Gaussian data (top) and non-Gaussian data (bottom).

Since we cannot assume statistical independence of the $\mathbf{D}^*$ Isomap components, we further rearrange the data to obtain a uniform distribution using a technique similar to the constrained centroidal Voronoi tessellation mesh generator (Nguyen et al., 2009) and to the *unispiring* physical algorithm (Lallemand and Schwarz, 2011). In both methods, a convergent iterative algorithm implements the data redistribution process. In the first case every entry is iteratively repositioned to the center of mass of the

Voronoi tessellation region. In the second case the data is moved according to a repulsive force based on the length of the Delaunay triangulation edges. The Delaunay triangulation and the Voronoi tessellation provide rigid structures that are iteratively deformed but not modified, thus both methods guarantee a coherent transformation in preserving the neighborhood of the original space. Both methods assume that the data is bounded in an arbitrary convex hull, which is a M dimensional hypercube in our case. Data points moved outside the boundary are orthogonally projected on the boundary before the next iteration. This in turn may harm the preservation of the original neighborhood, particularly when the rank-transformed data results in clusters near the boundaries. In order to avoid this shortcoming, we adopt these two simple stratagems:

- at initialization scale the data so that there is a tolerance margin between the data edges and the bounding hull;
- at every iteration shrink the data towards the hull center if a relevant percentage of entries are moved outside the boundaries.

We set the boundary to the M dimensional hypercube with unitary side, and the rank transformed data is scaled reducing the absolute maximum and minimum to 0.9 and 0.1 respectively. The algorithm ends when the sum of the iteration total data movement or the data covariance measure λ is below a certain threshold. The latter, introduced in (Nguyen et al., 2009) and defined by Equations 5.4-5, tends to 0 for uniformly distributed data. We evaluate the performances of the uniform redistribution algorithm by measuring λ and the percentage of entries $\mathbf{d}^*_{j-U}$ that do not present the same neighbors as $\mathbf{d}^*_j$, using the same method we adopted for the SOM topology distortion detection.

$$\lambda = \sqrt[2]{\left(B \frac{\sum_{j=1}^{B} \gamma_j^2}{\left(\sum_{j=1}^{B} \gamma_j\right)^2} - 1 \right)} \tag{5.4}$$

$$\gamma_j = \min_{z=1,\dots,B, z\neq j} \left| \mathbf{d}^*_j - \mathbf{d}^*_z \right| \tag{5.5}$$

In Figure 5.4 we show examples of 2D and 3D reduced sonic spaces $\mathbf{D}^*$ redistributed to uniformly distributed square and cube through the intermediate rank-transform step. In the plots on the right in Figure 5.4, it is evident that the rearrangement

algorithm generates a distribution almost perfectly uniform. For these two cases in Figure 5.5 we show some intermediate snapshot during the redistribution algorithm.

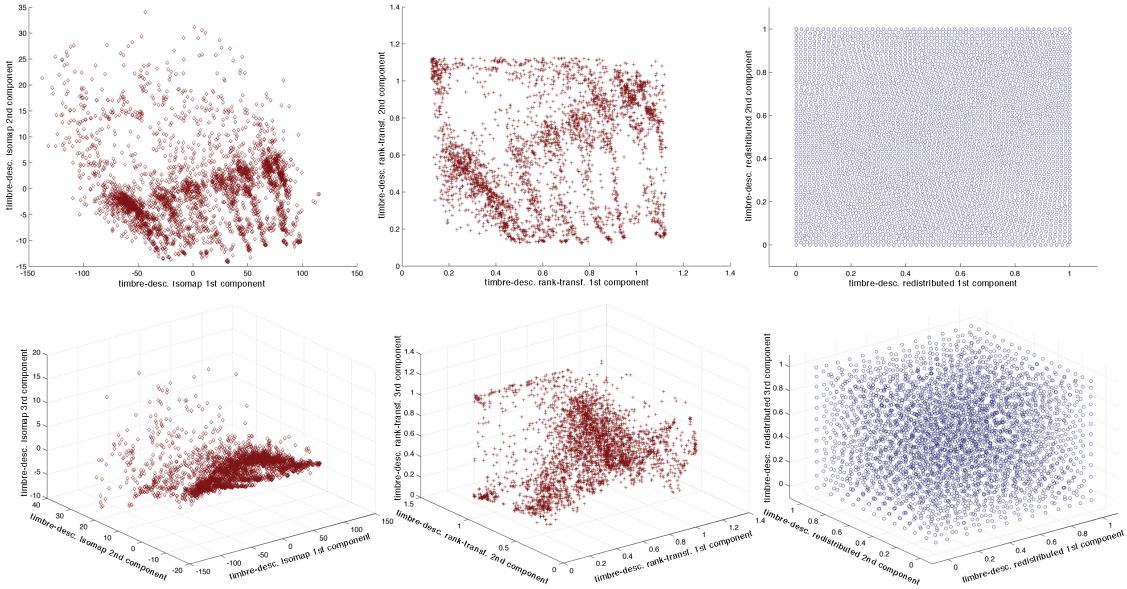

Figure 5.4: Sonic spaces (left) redistributed to uniform square and cube (right) through the intermediate rank-transform step (center), for 2D (top) and 3D (bottom) examples.

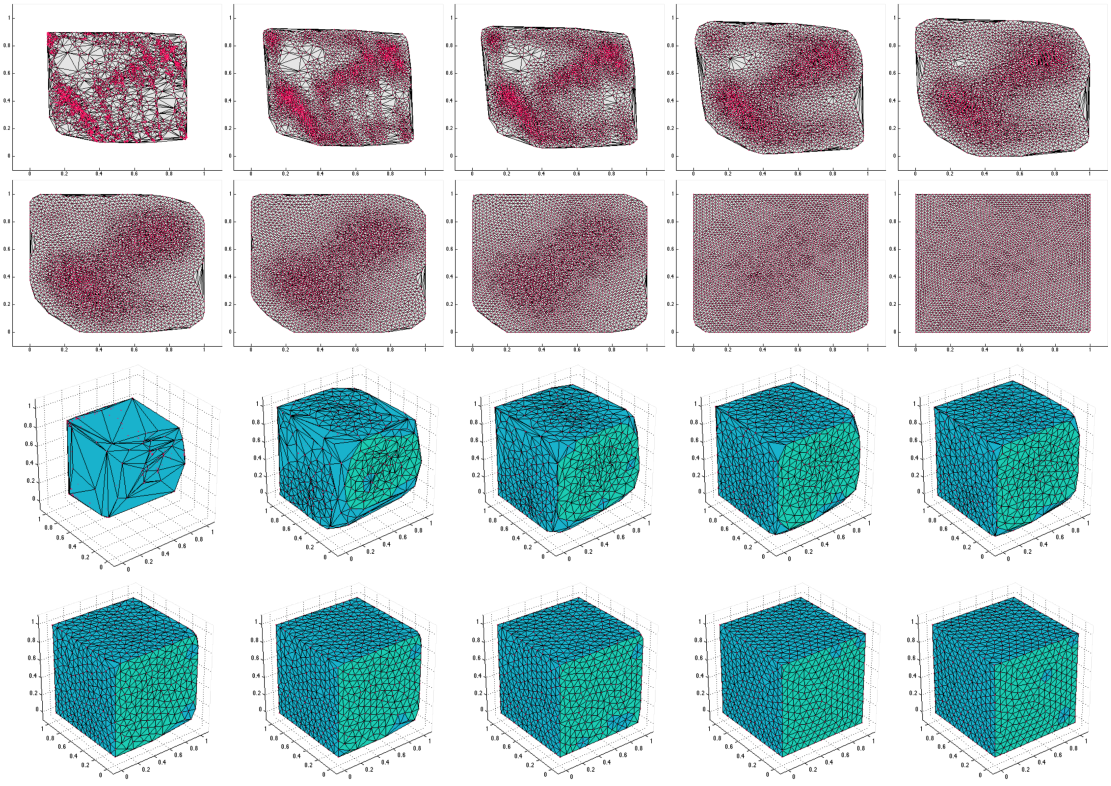

Figure 5.5: Reading from left to right, top to bottom, these figures show the intermediate snapshots of the uniform redistribution algorithm for sonic spaces in 2D (top two rows) and 3D (bottom two rows).

5.1.3 Mapping function modeling with ANN

After the redistribution of the reduced sonic space $\mathbf{D}^*$ to the uniform $\mathbf{D}_{\boldsymbol{U}}^*$, the mapping function $m()$ can be redefined as in 5.6.

$$\mathbf{D}^* = m(\mathbf{D}_{\boldsymbol{U}}^*) \approx NN(\mathbf{D}_{\boldsymbol{U}}^*) + \mathbf{E} \tag{5.6}$$

We use a neural network to learn the nonlinear transformation determined by the sequence of rank transformation plus uniform redistribution, so that $m()$ is approximated by the ANN input-output learned function $NN()$ except for an error component $\mathbf{E}$, that we aim to minimize. In particular we use a feed forward network with sigmoid activation functions in the hidden layers, while the neurons in the output layer present linear activation functions. The network is trained with the Levenberg-Marquardt back propagation algorithm (Levenberg, 1944; Marquardt, 1963), using $\mathbf{D}_{\boldsymbol{U}}^*$ and $\mathbf{D}^*$ for the input and output pairs respectively, thus the network presents an equal number of input and output neurons. If the training accuracy target, measured with the Mean-Squared Error (MSE), is not met within a time and maximum number of epoch limits, we gradually grow the network and repeat the training. We start with a single hidden layer with $M + 1$ neurons, and we grow by one unit at a time. After a limit of $M + 15$ we start adding neurons to the second hidden layer and reset the first hidden layer to $M + 1$ neurons. The minimum MSE that the ANN can reach is case dependent, so for certain $\mathbf{D}^*$ the goal value cannot be achieved. Therefore after a user-defined maximum number of training instances, we select the ANN configuration with the minimum MSE, as in Equation 5.7. The MSE measures the square root of the average distance from the original position that the $NN()$ determines projecting all entries from $\mathbf{D}_{\boldsymbol{U}}^*$ to $\mathbf{D}^*$. Moreover, as an additional measurement of the $NN()$ training quality, we generate a test set of random coordinates, uniformly distributed, which emulates all possible combination of a M dimensional MIDI controller. These are fed to the mapping function and we measure the percentage of $\mathbf{D}^*$ loss entries, which are those never being the nearest neighbor of any projected coordinate in the test set.

$$MSE = \frac{\sum_{j=1}^{B} \left| \mathbf{d}_j^* - NN(\mathbf{d}_{j-U}^*) \right|^2}{B} \tag{5.7}$$

In Figure 5.6 we show three spaces $\mathbf{D}^*$ next to the their related $NN(\mathbf{D}_{\boldsymbol{U}}^*)$, which show similar overall structure and shape, while minor local differences still exist. When the MSE and the percentage of loss entries are too high, it is possible to cope with the poor mapping performances by bypassing the mapping function $m()$ and implementing the DMI parameter retrieval $\mathbf{i}$ directly in the uniform sonic space $\mathbf{D}_{\boldsymbol{U}}^*$. As mentioned before, in this way IDW interpolation distance weights, used to improve poor the DMI analysis parameter resolution, would be less accurate because they are measured in $\mathbf{D}_{\boldsymbol{U}}^*$ rather than in $\mathbf{D}^*$. The gradual growth of the ANN, the training accuracy measurement, and the automatic configuration we presented here, frees the user from the definition of the network characteristics, which if inappropriate can lead to under-fitting and over-fitting issues. Moreover these provide a metric to evaluate, prior to use, the computed mapping function goodness.

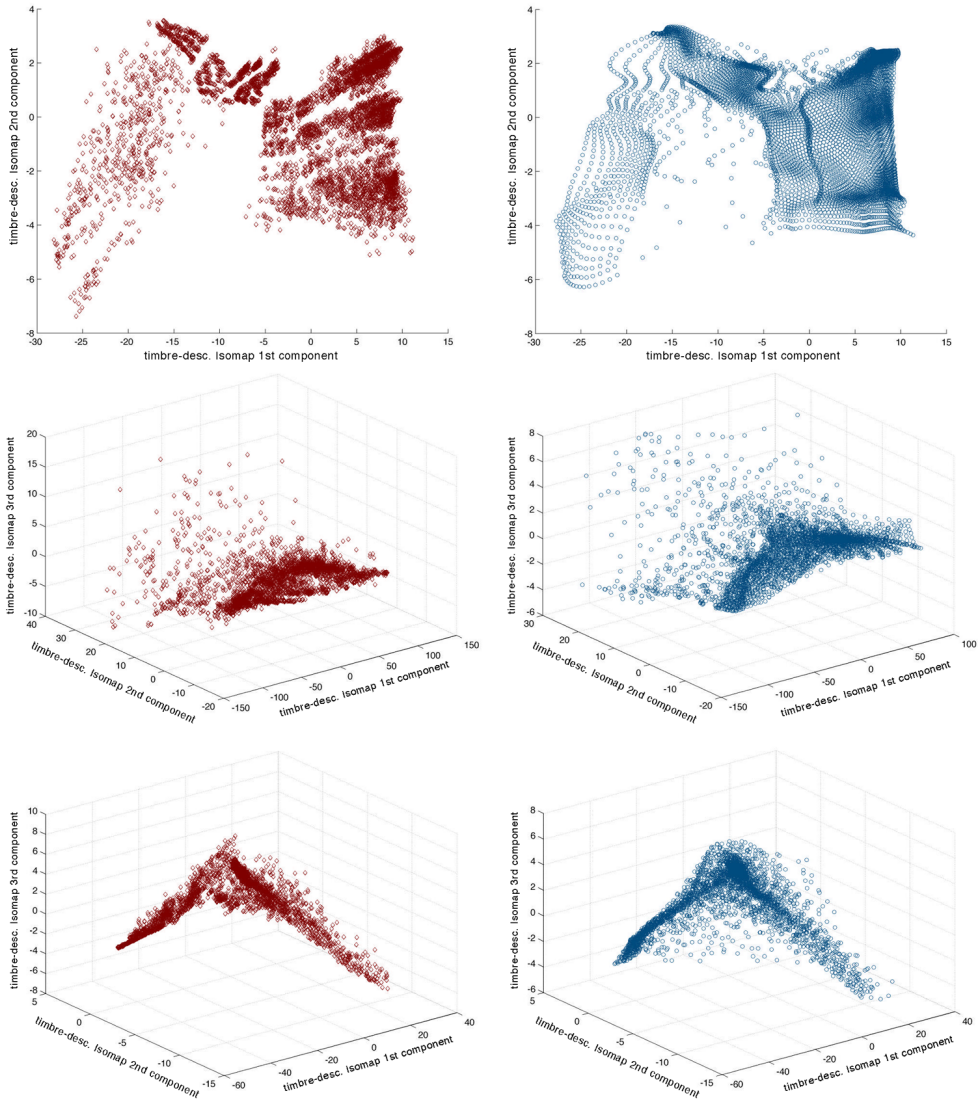

Figure 5.6: Examples of sonic space (left) versus trained ANN sonic space projection of the uniformly redistributed space (right).

5.2 Search space and discontinuity-free parameter retrieval

The mapping function, implemented by the ANN, converts the GC output $\mathbf{gc_{out}}$ into coordinates of the reduced sonic space $\mathbf{D}^*$ that we use to determine the closest sonic descriptor $\mathbf{d}_j^*$, and to retrieve the associated DMI parameters vector $\mathbf{i}_j$, as in 5.8. The non-bijective parameter-to-sound relationship may result in discontinuities of the **i** stream, or parameter values generating wrong sounds in relation to specific sonic space positions when IDW interpolation is applied to the output, as in Equation 4.19.

$$\mathbf{i_j} \ : \ \underset{j}{\operatorname{argmin}} \left| \mathbf{d}_j^* - m(\mathbf{gc_{out}}) \right| \tag{5.8}$$

In a sound driven synthesis system Puckette (2004) addresses this intrinsic shortcoming of DMI control strategies implemented in the sound domain, limiting the instantaneous transitions n the sound space only to those entries that ensure parameters continuity. Thus at every iteration we can determine a restricted search space $\mathbf{D}_{rst}^*$ in $\mathbf{D}^*$ including those $\mathbf{d}_j^*$ with related parameters set $\mathbf{i}_j$ minimizing the Euclidean distance with the current $\mathbf{i}_{out}$, as described by Equation 5.9. The cardinality of $\mathbf{D}_{rst}^*$ depends on the number of parameters in **I**.

$$\mathbf{D}_{rst}^* \ni \mathbf{d}_j^* \rightarrow \mathbf{i}_j \ : \ \underset{j, j \neq out}{\operatorname{argmin}} \left| \mathbf{i}_j - \mathbf{i}_{out} \right| \tag{5.9}$$

5.2.1 Parameters continuity and usability tradeoff

The method proposed by Puckette ensures perfect parameters continuity but at the same time it can introduce a usability drawback. The limited search subspace in $\mathbf{D}^*$ is determined by neighborhood relationships in **I** so that each element in the restricted search space $\mathbf{D}_{rst}^*$ can be potentially located anywhere in the sonic space. These can be clustered in small nearby areas, dislocated far apart, or organized on a straight line. Therefore specific entries or sub-regions of the sonic space may not be reachable, or determine a trapping state difficult to breakout of. The first issue can be admissible because other and near entries in $\mathbf{D}^*$ are likely to be reachable and generate similar sounds, while the second is detrimental for the interface usability. This approach implicitly limits the possible transitions in $\mathbf{D}^*$ linking every $\mathbf{d}_j^*$ to only a few other sonic space entries, unknown to the user. Moreover there is no guarantee pertaining

to the omni-directionality of the output transitions for each entry $\mathbf{d}_j^*$, which in turn determines possible unresponsive states of the interface. In Figure 5.7 we show examples of a sonic space in which the red point represents the closer $\mathbf{d}_j^*$ and the green circles represent $\mathbf{D}_{rst}^*$. It is evident that green entries far from the red one are unlikely to be the next nearest $\mathbf{d}_j^*$, because these would require an exact instantaneous GC output transition with large step variation. Moreover from the bottom representation of uniform redistributed sonic space it is clear that the interface will be unresponsive if the GC output moves towards a region in which there are no green entries.

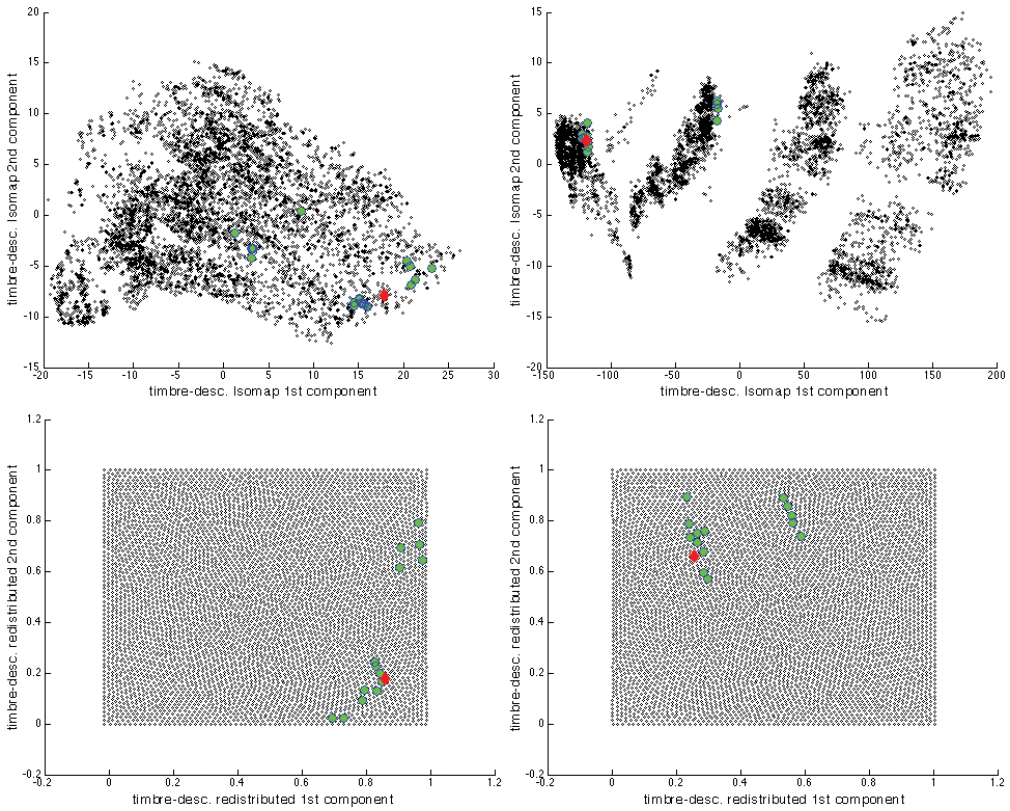

Figure 5.7: Sonic spaces with the green entries representing the nearest neighbor of the red entry in the DMI parameter space visualized over two examples (left and right) of original space (top) and redistributed space (bottom).

In order to address this drawback, besides providing to the performer the real-time visualization of the spaces, as in Figure 5.7, we allow minor discontinuities in the DMI parameters stream while including more elements in the restricted sonic search space $\mathbf{D}_{rst}^*$. Thus we extend the search to those entries $\mathbf{d}_j^*$ whose related parameters set in $\mathbf{I}$ are within a user-defined radial distance i_{rad} from the current $\mathbf{i}_{out}$, as in 5.10.

$$\mathbf{D}^*_{rst} \ni \mathbf{d}^*_j \rightarrow \mathbf{i}_j \ : \ \left|\mathbf{i}_j - \mathbf{i}_{\boldsymbol{out}}\right| < i_{rad} \qquad (5.10)$$

This approach provides a tradeoff between continuity and usability of the system, and the number of entries in the restricted sonic search space $\mathbf{D}^*_{rst}$ grows rapidly with decimal increase of i_{rad} (the parameters in $\mathbf{I}$ are normalized in [0,1]), thus the potential discontinuities in the $\mathbf{i}_{\boldsymbol{out}}$ stream are still limited. This still depends on the cardinality of the space and the DMI parameter resolution used in the analytical stage. In Figure 5.8 we show from left to right the different $\mathbf{D}^*_{rst}$ for i_{rad} equal to 0.15, 0.25, and 0.35 respectively. In the example of Figure 5.9 we show the restricted search space $\mathbf{D}^*_{rst}$ over the related uniform redistributed sonic space as well, for values of i_{rad} equal to 0.25, and 0.35. Both examples demonstrate how the value of i_{rad} determines the responsiveness or sensitivity of the interface, which can be tuned according to the performer preferences and specific sonic space characteristics. Finally we propose an additional solution in which the i_{rad} is dynamically determined from the instantaneous variation of the voice reduced vector $\mathbf{v}^*$, measured by the Euclidean distance from the value at the previous iteration, and multiplied by a user-definable coefficient. With this strategy, vocal-postures determine $\mathbf{D}^*_{rst}$ including only the currently closest entry $\mathbf{d}^*_j$, while the size of $\mathbf{D}^*_{rst}$ grows linearly with the rate of the timbre variation in vocal-gestures.

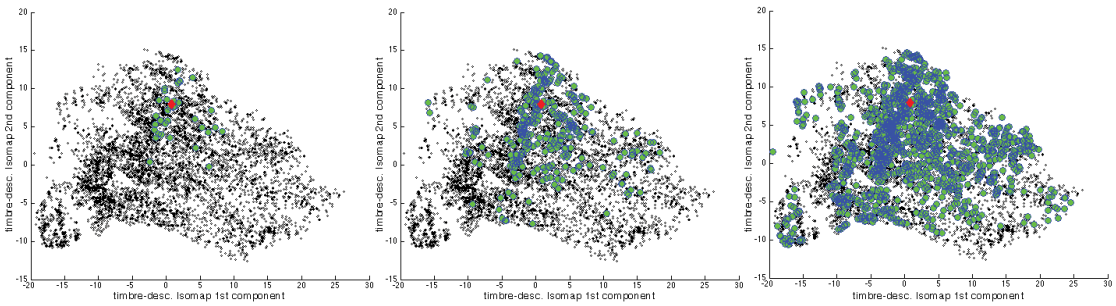

Figure 5.8: Restricted sonic search space, in green, for increasing values of the maximum radial distance.

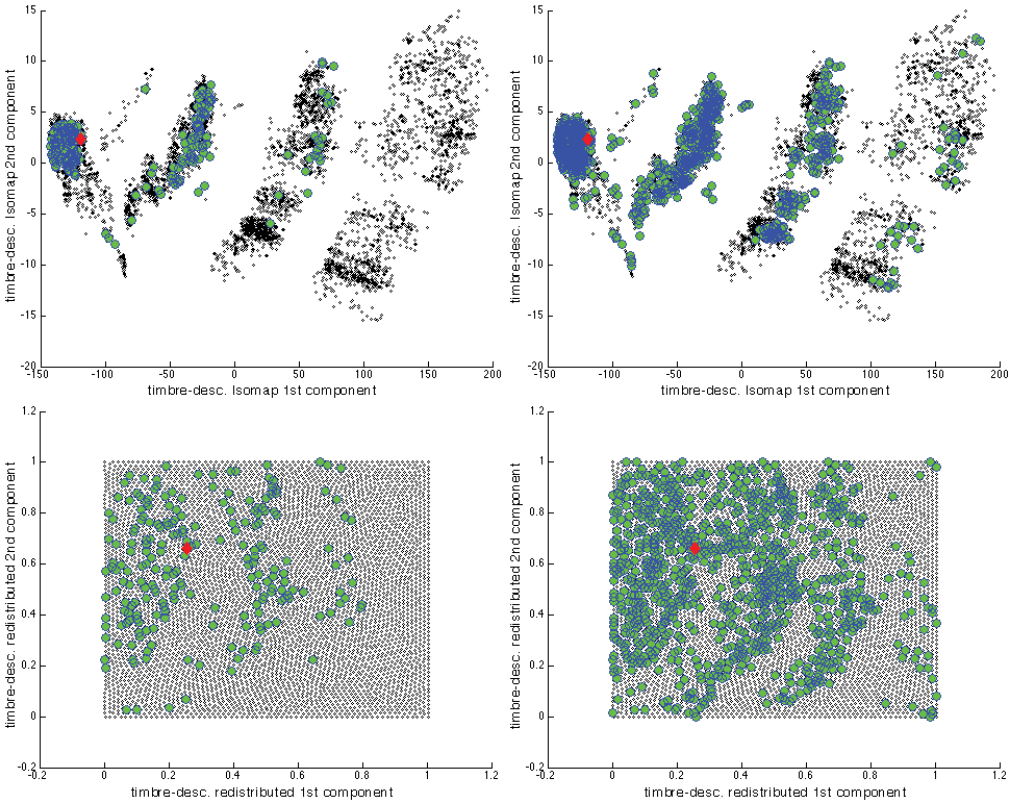

Figure 5.9: Restricted sonic space, in green with blue borders, for different maximum radial distance over original space (top) and uniformly redistributed space (bottom). The blue regions are determined by high density of restricted sonic space entries.

5.2.2 Parameters kernel function

When parameter-to-sound mapping is highly non-bijective and there are multiple clusters of identical sound generated by heterogeneous combinations of parameters, effective browsing of the sound space $\mathbf{D}^*$ may still be an issue with the strategy proposed above because of possible scattered restricted sonic space $\mathbf{D}^*_{rst}$, unless we set a large i_{rad} and allow potential parameter discontinuities. As an alternative in this case we include information derived from the DMI parameters in the descriptor vector $\mathbf{d}$ before NLDR stage. Although not related to any perceptual aspect of the sound timbre, this is the only possible approach to spatially separate identical sounds generated with different $\mathbf{i}$. In order to minimize the impact on the perceptual representation of $\mathbf{D}^*$, we include in the descriptor vector $\mathbf{d}_j$ a single scalar derived from the parameter combination $\mathbf{i}_j$, similar to the kernel methods in ML, used to increase the dimensionality of poorly discriminative features (Shawe-Taylor and Cristianini, 2004). We chose the non-linear kernel function in 5.11, to extend the descriptor vector in 5.12, because it provides clearly distinct values for distant

combinations across the whole parameter space. This is visible in Figure 5.10, in which the surface represents the output of the kernel function for two parameters, always in the normalized range [0,1]. In Figure 5.11 we show the differences in the sonic space determined by including parameters related scalar in $\mathbf{d}_j$, which contributes to separate and spread the dense clusters, more likely affected by the non-bijective parameters-to-sound issue. In Equation 5.10 low index n parameters have a slightly bigger impact on the kernel function value $K(\mathbf{i}_j)$. Thus we sort the parameter index by their maximum correlation with any descriptor in the sonic space **D**. This allows for the parameter least correlated with the output sound, and likely less discriminative, to contribute the most to the $K(\mathbf{i}_j)$ and vice versa.

$$K(\mathbf{i}) = \sum_{n=1}^{P} \sqrt[(n+1)]{i_n^{(n+1)} + \frac{1}{(n+1)}} \tag{5.11}$$

$$\mathbf{d}_\mathrm{j} = \left[f_{mode}\left(f_{desc}\left(d(\mathbf{i}_j, 1) \right), \ldots, f_{desc}(d(\mathbf{i}_j, T)) \right); K(\mathbf{i}_j) \right] \tag{5.12}$$

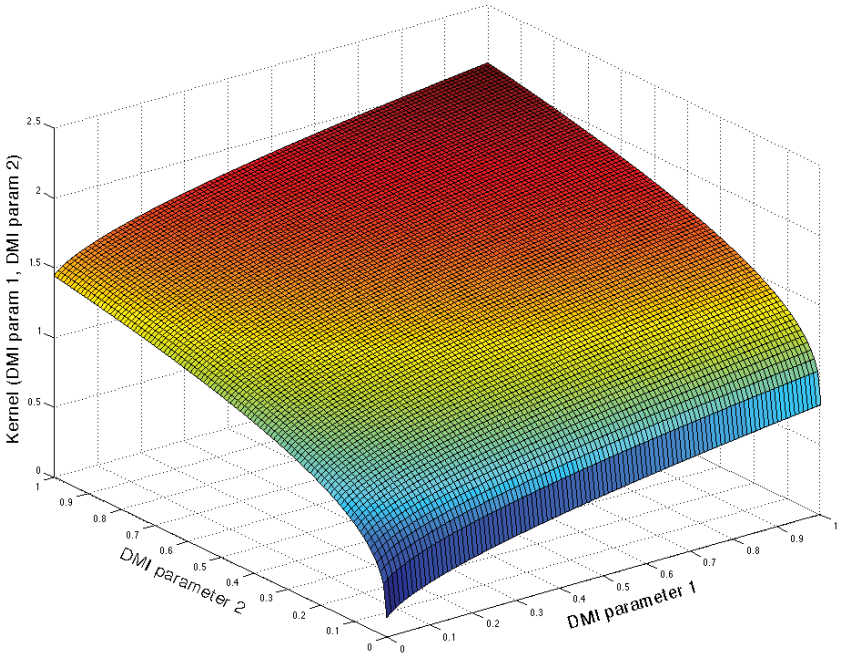

Figure 5.10: Surface representing the scalar generated by the kernel function for two DMI parameters full range variation.

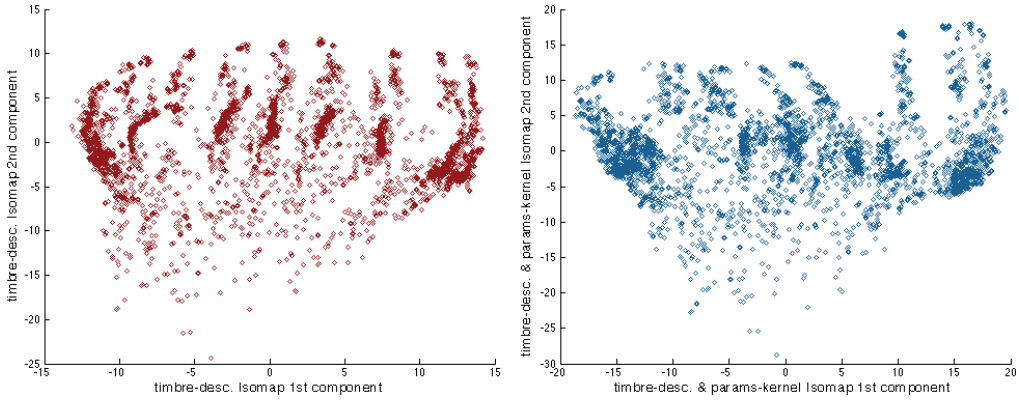

Figure 5.11: Comparison between low dimensional sonic spaces without (left) and with (right) DMI parameter kernel scalar in the high dimensional descriptor vector. The better discriminability can be seen in the sonic space that included the kernel scalar.

5.2.3 Operational modes

Below we summarize the steps implemented in the VCI4DMI to derive the instrument control parameters $\mathbf{i_{out}}$ during runtime from the gestural controller $\mathbf{gc_{out}}$, including the different mapping and parameters retrieval options described in this chapter, which are implemented and available in the prototype. The outer level list entries represent the sequential steps while the inner level list includes the alternative options for each step.

1. Set the restricted search space $\mathbf{D}^*_{rst}$ equal to:
    a. $\mathbf{D}^*$ to consider the whole space and accept potential discontinuities;
    b. the related nearest neighbor in $\mathbf{I}$ to the previous $\mathbf{i_{out}}$, as in 5.9;
    c. the related entries in $\mathbf{I}$ within i_{rad} from the previous $\mathbf{i_{out}}$, as in 5.10:
        i. i_{rad} has a fixed value;
        ii. i_{rad} is dynamic and linked to the instantaneous $\mathbf{v}^*$ variation;
2. derive the mapping coordinates from $\mathbf{gc_{out}}$:
    a. using the ANN mapping function $m(\mathbf{gc_{out}})$;
    b. use $\mathbf{gc_{out}}$ as mapping coordinates and replace $\mathbf{D}^*$ with the $\mathbf{D}^*_U$ space (in this case also $\mathbf{D}^*_{rst}$ is in $\mathbf{D}^*_U$);
    c. with linear scale and offset applied to $\mathbf{gc_{out}}$ components (for comparison only);
3. compute the Euclidean distance between the mapping coordinates and the entries in $\mathbf{D}^*_{rst}$ and set $\mathbf{i_{out}}$ to:
    a. the $\mathbf{i}_j$ relative to the $\mathbf{d}_j$ nearest to the mapping coordinates, as in 5.8;
    b. the IDW interpolation of the $\mathbf{D}^*_{rst}$ entries as in 4.19.

To improve the implementation efficiency, step 1 of the following iteration is always computed straight after step 3 of the current iteration, exploiting the idle time before the next GC output arrival. The computational complexity depends on the number of entries B in the reduced sonic space $\mathbf{D}^*$, and from the combination of options in the above steps. Option 1.a requires B Euclidean distance to be computed in step 3, but these are only 2D or 3D. Case 1.b and 1.c requires lower Euclidean distances to be computed in step 3, but within step 1 it needs B Euclidean distances in $\mathbf{I}$, a space likely higher than 3D, plus B comparison operation, thus it requires a higher overall computational cost. The restricted search space $\mathbf{D}^*_{rst}$ is handled by entries index only, thus in step 2 the swap between $\mathbf{D}^*$ and $\mathbf{D}^*_{\boldsymbol{U}}$ has no cost. Finally the IDW in 3.b requires only a small computational load increase compared to 3.a because both step 3 alternatives require the computation of the Euclidean distance with the set $\mathbf{D}^*_{rst}$.

5.3 Evaluation and validation

For the evaluation and validation of the generative DMI mapping strategy presented in this chapter we use the same DMI set presented in Section 4.5 and the respective analyzed sonic spaces. In the tables below the instruments are identified with the same numeric ID defined in Table 4.1. Here as well the measurements are detailed for each DMI, due to the inconsistent numbers and scales that make statistical summary meaningless.

In Figure 5.12 we show the uniformity of sonic spaces, measuring the data covariance measure λ as in Equation 5.4. This is detailed in the first column of Table A.2 in Appendix A, and perfect uniform distributions have values of λ close to 0. We observe a sensible λ increase from $\mathbf{D}$ to $\mathbf{D}^*$, which is more evident for the 2D case, supporting again that the drastic NLDR contributes to cluster the data, while here we aim for evenly spaced entries in $\mathbf{D}^*$. However the redistributed and reduced sonic spaces $\mathbf{D}^*_{\boldsymbol{U}}$, show λ values close to 0, which are similar across the set, and not dependent on space cardinality due to the termination condition of the iterative redistribution algorithm, identical for every DMI. The second column of Table A.2 shows a percentage of redistribution errors clearly below 1%, supporting the reliability of the method. Errors were measured verifying the neighborhood relationship between original and transformed spaces with the same technique used for the detection of SOM topology distortions. As expected the few errors that do occur are usually near the vertices of the uniform redistributed space.

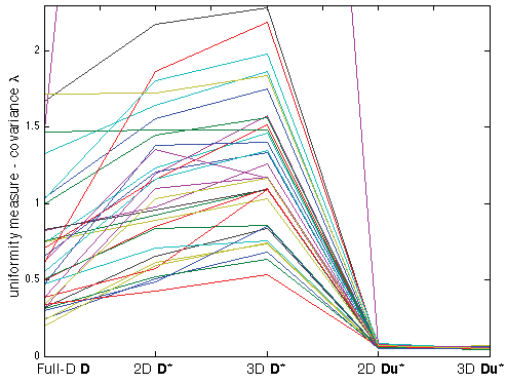

Figure 5.12: Uniformity of different sonic spaces estimated measuring the data covariance λ. The rising trend of the first half shows how the dimensionality reduction worsens the original uniformity, the falling trend in the second half shows that the redistribution algorithm generates almost perfectly uniform distributed sonic spaces. Each colored line represents a different DMI case.

In the third column of Table A.2 we show the ANN configuration that minimized the MSE of the regression $\mathbf{D}_{\boldsymbol{U}}^{*}$ to $\mathbf{D}^{*}$. The network configuration and minimum MSE are case dependent, but in general two layers of hidden neurons provide better performances. Additional hidden layers provided little or no gain hence not justifying the increase in the computational load that this would require in the runtime VCI4DMI mapping. The complexity of the transformation that the ANN tries to estimate is essentially related to the non-uniformity of $\mathbf{D}^{*}$. In general, when the uniformity measure λ of the reduced sonic space is bigger than 0.5 the MSE is significant but we verified that the overall system is still usable. When λ is above 1 the ANN fails in learning the inverse redistribution, but the VCI4DMI can still be used skipping the ANN mapping function $m()$ and projecting the GC output directly onto $\mathbf{D}_{\boldsymbol{U}}^{*}$, and accepting lower the IDW precision. In the remaining results we detail in this section, the few cases showing low mapping performances are usually associated with highly clustered original sonic spaces, denoted by large value of λ.

In Figure 5.13 and in the fourth and fifth columns of Table A.2 we display the percentage of obtainable $\mathbf{d}_{j}^{*}$, thus parameter combinations $\mathbf{i}_{j}$, projecting the entries in $\mathbf{D}_{\boldsymbol{U}}^{*}$ back to $\mathbf{D}^{*}$ with the ANN mapping function $m()$ and finding the nearest entry in the reduced sonic space. In general the percentage is particularly low, especially for 2D cases, but this measure is mostly associable with the absolute precision of the trained ANN. We repeat the same measurement with settings similar to a real use-case, using coordinates derived from a grid with resolution 1/128 on each unitary range component. We projected these with $m()$ onto $\mathbf{D}^{*}$ and searched for the nearest

$\mathbf{d}_j^*$. The results, in the sixth column, show that in general we lose less than 10% of the total space spanned by the parameter combinations. Finally we measured the linearity of the sonic response, approximated by the average line fitting residual for all reduced principal descriptors, which are the coordinates of the $\mathbf{d}_j^*$, obtained feeding the mapping function $m()$ with the coordinates of the diagonals of the intermediate GC space. The residuals are generally low, except for high λ cases, and this suggests good linearity between GC space and sonic response of the proposed mapping method.

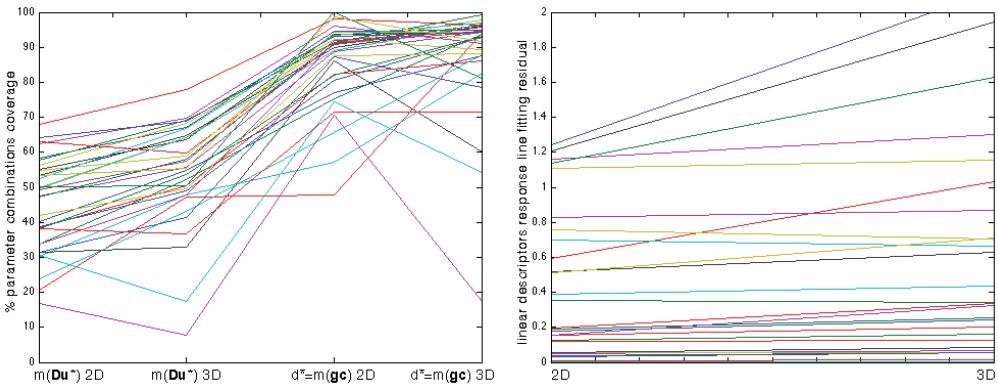

Figure 5.13: Percentage of unique DMI parameter combination obtainable with ANN projection of $\mathbf{D_u}^*$ onto $\mathbf{D}^*$ and with the ANN used for sonic space control, clearly above 80% in most case (left), average line fitting residual for all reduced principal descriptors, lower than 0.2 in most cases, implying linearity between control and sonic response (right). Each colored line represents a different DMI case and results are shown for 2D and 3D cases.

In Figure 5.14 we illustrate percentage of obtained parameters, parameter continuity and sonic space coverage spread for approximately 2 minutes of mapping from data coming from a GC emulator with a rate of 100 $\mathbf{gc_{out}}$ per second. The results are detailed in Tables A.3-4 in appendix A. In particular we replicated typical output trajectories of the vocal GC, evenly covering the intermediate mapping space, and this is fed to the DMI mapping block. The percentage of obtained parameters is evaluated as above. For parameter continuity we consider the average distance between consecutive parameters set $\mathbf{i}_{out}$, and the spread is measured counting the standard deviation of the nearest neighbor count of each $\mathbf{d}_j^*$. The study demonstrates that limiting the search space to $\mathbf{D}_{rst}^*$ minimizes the discontinuities in the parameters generation, but it also shows that there is a tradeoff between continuity and usability, estimated with coverage percentage and spread deviation. Therefore we repeat this test for different values of i_{rad} to demonstrate the effect of different cardinality of $\mathbf{D}_{rst}^*$ on the three measurements. Further, this has been repeated for the 2D and 3D

sonic mappings using the ANN-based mapping function $m()$ for the results in the left plots of Figures 5.14-16 and in Table A.3, while in the right plot of Figure 5.14-16 and in Table A.4 the GC output projection and the search are directly onto $\mathbf{D}_{\boldsymbol{U}}^{*}$. Moreover in both cases we did not use the IDW at the output, which often eases the eventual DMI parameter output discontinuity. The individual DMIs analysis settings, especially those related to the parameter resolutions, result in drastically different number entries in $\mathbf{D}$ and in the restricted search space $\mathbf{D}_{rst}^{*}$ for the same value of i_{rad}.

From the results we observe that increasing i_{rad} the space coverage grows and the spread decreases, resulting in a sonic space mapping easier to navigate. Moreover, increases in the coverage are more evident and in cases presenting low percentage for small i_{rad} it is evident how the system gets trapped in few states or entries of $\mathbf{D}^{*}$. The coverage can eventually decrease slightly due to the random component in the test input data generation. The parameter continuity is usually good as the average distance of consecutive parameter vectors is small and lower than i_{rad}, which represents the upper bound. The average distance grows with i_{rad} due to the non-bijective parameter-to-sound relationship, or to a rapid gesture variation. Comparing the left and right plots of Figures 5.14-16 and in Tables A.3-4 we conclude that the second simplified mapping method often results in improved usability due to higher coverage with lower spread, but this comes at the expense of the IDW precision, which is more critical when the cardinality of $\mathbf{D}^{*}$ is low. The results demonstrate that the method we introduced in this chapter can provide a balance between parameter reach and discontinuity to address the intrinsic problem of the non-bijective parameter-to-sound relationship. Moreover comparing the results of the 2D and 3D cases we observe that often the performance increase provided by the additional dimension is minimal. Thus the 2D mapping offers an almost equally expressive interface requiring less cognitive attention, as discussed later in Chapter 7.

Finally in Table 5.1 we show the improvement on the results presented in this section of adding the DMI parameters kernel scalar $K(\mathbf{i}_j)$ of Equation 5.11 in $\mathbf{D}$. The table only includes a subset of DMI evaluation cases, limited to the 2D reductions, where the improvements are usually more critical. The effects on the data covariance measure λ are conflicting, but on average, adding the kernel scalar determines a $\mathbf{D}_{\boldsymbol{U}}^{*}$ to $\mathbf{D}^{*}$ transformation that is better modeled by the ANN. We obtain a lower MSE that, as explained before, reflects higher performances in all further measurements. However with this stratagem we allow in the space $\mathbf{D}$ one non-sonic descriptor, which is not in total accordance with the interface principles. Hence the kernel should be included only if the usability is excessively poor and limited.

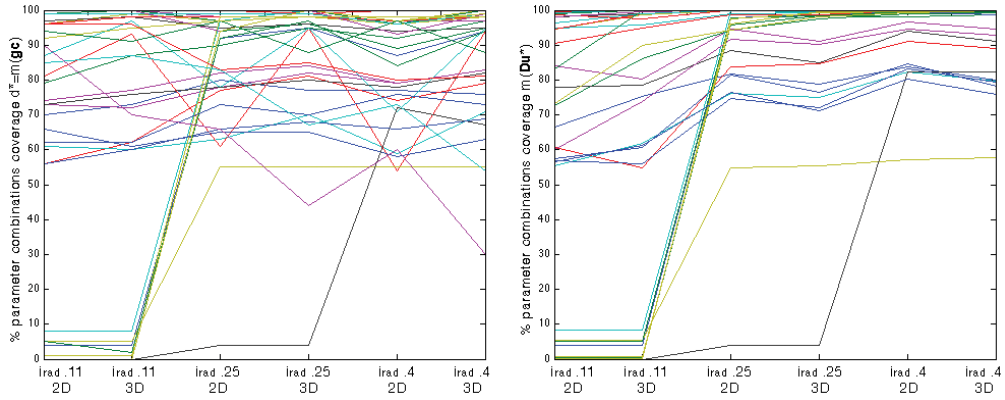

Figure 5.14: Percentage of obtained parameters for 2 minutes of simulated use, with different values of i_{rad} and different mapping dimensionality using the ANN-based mapping function (left), and the mapping directly onto the uniformly redistributed space (right). The coverage increases with larger i_{rad} and it is above 80% in most cases. Each colored line represents a different DMI case.

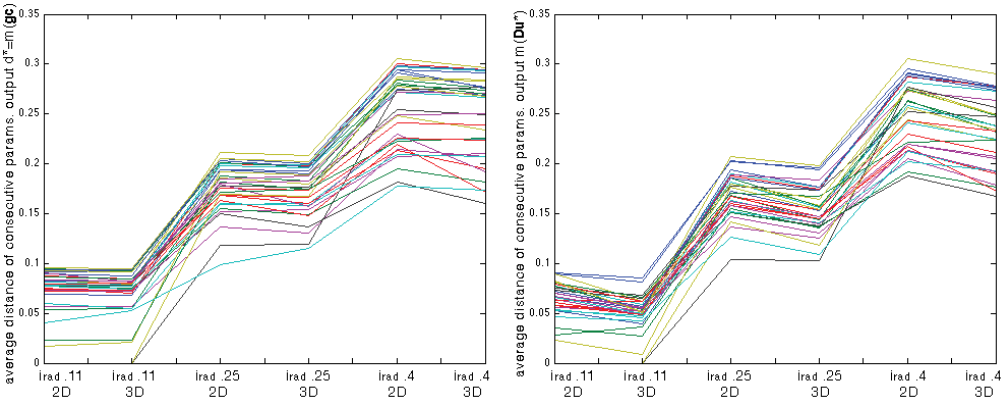

Figure 5.15: Parameters continuity obtained for 2 minutes of simulated use, with different values of i_{rad} and different mapping dimensionality using the ANN-based mapping function (left), and the mapping directly onto the uniformly redistributed space (right). The results represent the average distance between consecutive parameters vector, and these show that continuity worsens with larger i_{rad}, but values are generally small and lower than the upper bound i_{rad}. Each colored line represents a different DMI case.

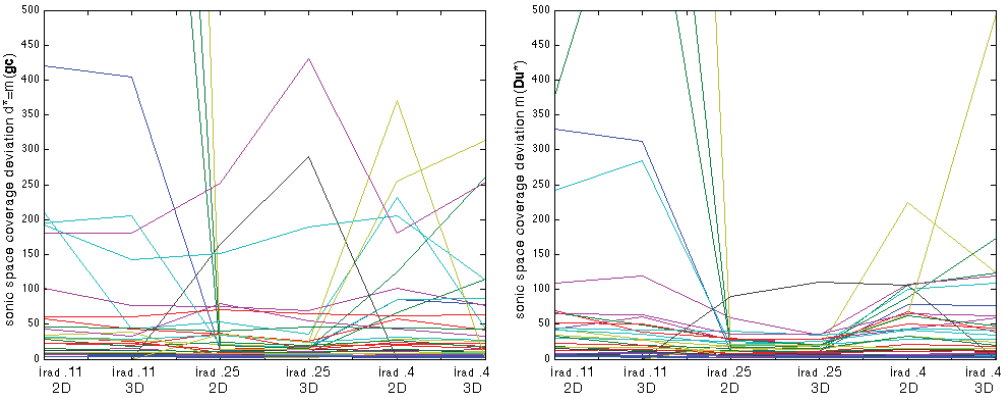

Figure 5.16: Space coverage spread of obtained parameters for 2 minutes of simulated use, with different values of i_{rad}, different mapping dimensionality using the ANN-based mapping function (left) and the mapping directly onto the uniformly redistributed space (right). The spread values are generally low indicating a sonic space mapping easy to navigate and in general these decrease with larger i_{rad} when using the ANN-based mapping function. Each colored line represents a different DMI case.

DMI ID	*uniformity covariance λ*			*redistrib. error %*	*ANN hidden layers neurons and MSE*		*% param. m(**Du***)*	*% param. **d***=m(**gc**) sonic control*	*linear descriptror response residual*
	D	***D****	***Du****		*cfg*	*MSE*			
7	0.833	0.955	0.055	0.019	19-6	0.94	55.2	93.8	0.055
7k	0.831	1.104	0.048	0.015	14-7	0.19	60.8	93.3	0.088
8	0.652	1.204	0.058	0.064	9-6	11.2	38.8	80.7	0.181
8k	0.592	1.222	0.046	0.044	12-6	2.17	49.2	89.0	0.234
10	0.709	1.160	0.055	0.045	17-4	4.29	20.5	47.9	0.193
10k	0.480	0.910	0.050	0.011	15-4	0.63	60.8	93.7	0.205
11	0.553	1.159	0.060	0.100	15-6	25	23.9	65.6	0.391
11k	0.451	1.158	0.050	0.055	14-7	4.86	44.6	82.0	0.463
14	0.316	0.653	0.053	0.008	19-7	48.7	53.3	89.9	0.515
14k	0.404	0.808	0.049	0.007	18-6	0.06	63.5	94.9	0.066

Table 5.1: Result improvements for a subset of DMI case and limited to the 2D reduction when including the parameters kernel scalar in the high dimensional sonic space before the mapping computation.

5.4 Summary

The procedure to derive the sonic space mapping function and the techniques to retrieve DMI parameters complete the training and functional components of VCI4DMI. In accordance with the principles and requirements expressed in Section 2.3.1, in this chapter we presented a strategy to provide reduced and adapted control over the sonic spaces of specific instruments with linear responses between controllers and perceptual sound. We motivated and illustrated the redistribution of the space into a uniformly distributed hypercube that we use as intermediate and standard mapping layer between the voice and instrument domains. An ANN models the sonic space redistribution process and implements the coordinate conversion from the GC output to the lower dimensional sonic space for each specific instrument. We presented a tradeoff for the DMI parameter retrieval where there is a balance between discontinuity-free parameters and the explorability of the sonic space, including a functional option to dynamically link the maximum allowed parameters step to the voice input. Finally we proposed a strategy based on a kernel function for the DMI parameters to ease the non-bijective shortcoming of the parameters-to-sound relationship. The DMI mapping realized in this way provides automatic adaptation and posterior co-design between instrument and controller. Moreover the intermediate mapping layer vocal-gesture and sonic spaces are iso-dimensional and isomorphic and are perfectly overlapped, maximizing the breadth of explorable sonic space given a set of vocal-gestures and target DMI parameters. With a certain tolerance continuous trajectories in the vocal space determine continuous trajectories not only in the sonic space but also in the parameter space. The analysis and training procedures are unsupervised so that user intervention is not required. The system is modular and thus different voice and DMI components can be independently joined in the VCI4DMI, promoting high reusability of analysis and training procedure outcomes. In Figure 5.12 we illustrate the summary of the training procedure and functional part of the DMI mapping for the VCI4DMI presented in this chapter.

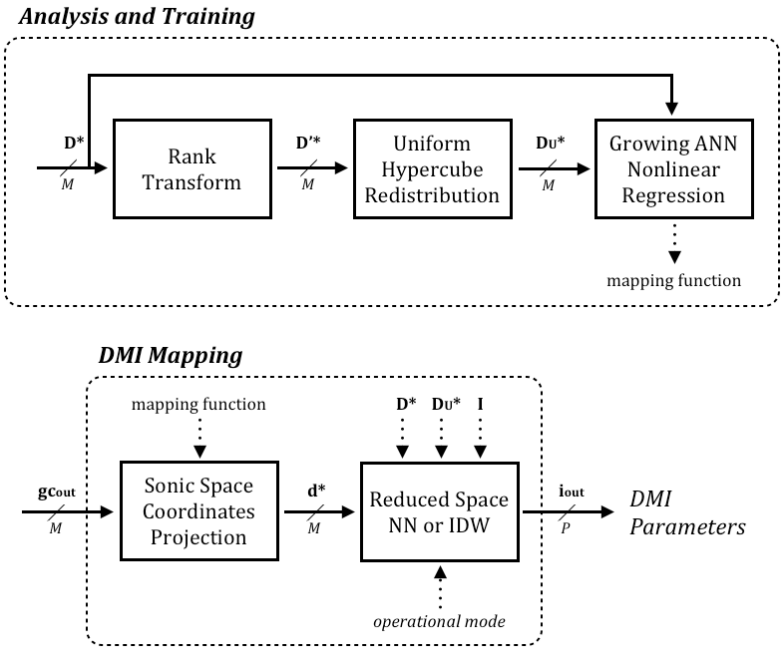

Figure 5.17: Illustrated summary of training procedure and functional part of the DMI mapping through sonic space components for the VCI4DMI.

Chapter 6

Functional Prototype

This chapter is dedicated to the proof-of-concept prototype of the VCI4DMI that we developed in the framework of this thesis. The open-source prototype integrates the adaptive and generative mapping methods described in the previous chapters, and it provides a platform for further exploration of the novel instrument mapping techniques and for the users evaluation of the proposed interface. We start the chapter introducing the VCI4DMI prototype structure and software implementation for training and runtime components, which has been optimized and refined for reliable use in live performances. Then we present the user perspective on the setup workflow and the available options to meet specific performers' preferences. Then follows the discussion regarding the audience point of view on the vocal interface. The chapter ends with the description of a solo musical performance exclusively based on the VCI4DMI, for which we developed a specific prototype version and a custom hardware controller.

6.1 Prototype Implementation

The open-source[3] LGPLv3[4] functional prototype is implemented in a set of Max/MSP[5] patches, Max for Live[6] devices, and MATLAB[7] functions, which cooperates concurrently to implement the online instrument analysis, the offline trainings for the vocal GC and the DMI mapping components, and the runtime integrated interface. The prototype partitioning is visible in Figure 6.1, where we illustrate the overall functional diagram of the entire VCI4DMI system, and we indicate the implementation language and partition entity on the top right of each functional block.

---

[3] http://stefanofasciani.com/vci4dmi.html
[4] https://www.gnu.org/licenses/lgpl.html
[5] http://cycling74.com/products/max/
[6] https://www.ableton.com/en/live/max-for-live/
[7] http://www.mathworks.com/products/matlab/

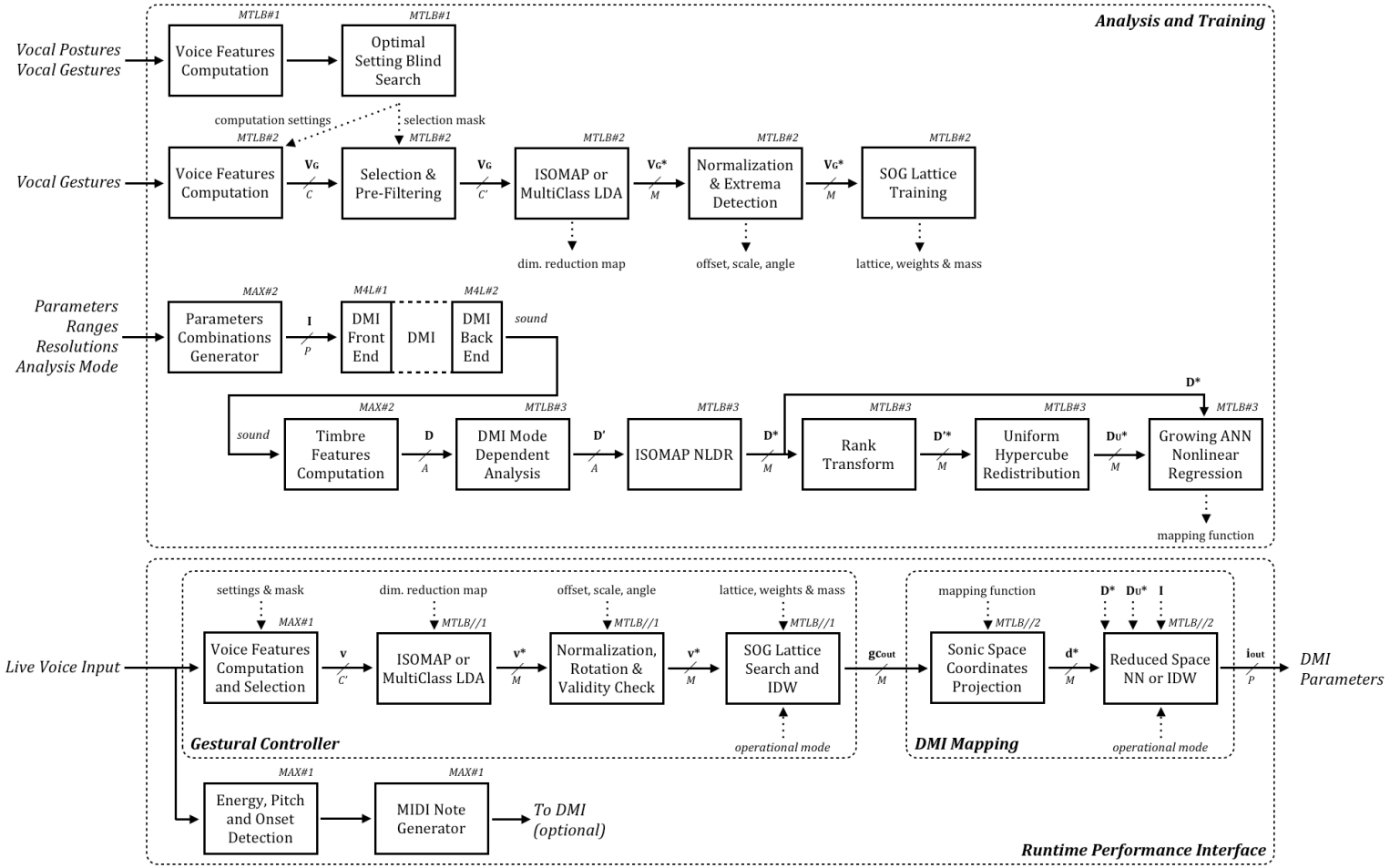

Figure 6.1: VCI4DMI overall functional diagram with implementation language and entity on the top right of each block.

6.1.1 DMI front-end and back-end

To provide the VCI4DMI with unrestricted DMI interfacing capability for analysis and real-time control purposes we developed front-end and back-end devices that wrap any instrument and expose a fixed communication protocol to the VCI4DMI system. The front-end is the bridge for the DMI parameters and it is necessary for analysis and runtime control phases. The back-end routes the DMI sound signal to the analysis system and can be removed during the real-time vocal control. The front-end and back-end components are integrated in a DAW, which supports and eases the interfacing of the VCI4DMI to any software plugin and hardware DMI, including any arbitrary chain of these. Front-end and back-end are developed as devices for Ableton Live[8], which is one of the most popular state-of-the-art DAWs. The implementation is based on in Max for Live that permits control with an identical protocol any parameters of DMIs hosted in Live. The VCI4DMI prototype core communicates with DMIs via the Open Sound Control (OSC) protocol, which is not yet supported in most consumer hardware or software instruments, so the front-end also provides an ad-hoc conversion stage.

We developed two front-end devices, one for sound generators and the other for sound processors. Both target a specific device hosted in Live and drive up to 8 DMI parameters. These are received via OSC and set instantaneously on the DMI if coming from the online analysis system, while they are linearly interpolated every 5ms (can be increased up to 1ms) between two consecutive control vectors $\mathbf{i_{out}}$. The interpolation ramp time is equal to the system $\mathbf{i_{out}}$ output rate, which in turn depends on the voice analysis step size. The sound generator front-end also provides a bridge to the online analysis system to trigger the device, converting signals from OSC into MIDI note-on and note-off, including specific pitch and velocity. The sound processor front-end also generates the input signal for the instrument analysis and it is compliant with the analysis modes, selecting between Dirac impulse, white, pink, or brown noise. Both front-end devices are transparent to the Live incoming MIDI and audio signals, allowing runtime device control without blocking the DAW track dataflow.

The back-end device, only needed for the analysis stage, forwards the DMI stereo output to the analysis system through a TCP network stream, which transmits PCM uncompressed audio with low-latency. Both front-end and back-end exchange

[8] https://www.ableton.com

data outside Live only though network based protocols, and therefore the analysis and the runtime control system can be hosted on the same machine, using the virtual local loopback, or on separated remote machines. Moreover, thanks to the audio and MIDI routing features of Live, it is possible to interface the VCI4DMI also to external hardware DMIs. For this purpose we developed an additional Max for Live bridge device that routes and maps the front-end parameters to a MIDI hardware port as control change messages. For the analysis stage the back-end is fed with the input stereo channel connected to the DMI audio output. In Figures 6.2-3 we show screenshots of the sound generator front-end and sound processor front-end, targeting respectively a VST external plugin and a Live native device, both followed by the DMI back-end.

Figure 6.2: Max for Live sound generator front-end and back-end, respectively on the left and on the right side of a VST plugin DMI hosted in Live. Front-end and back-end devices wrap the DMI enabling control and data exchange with the VCI4DMI core system for analysis and runtime control purposes.

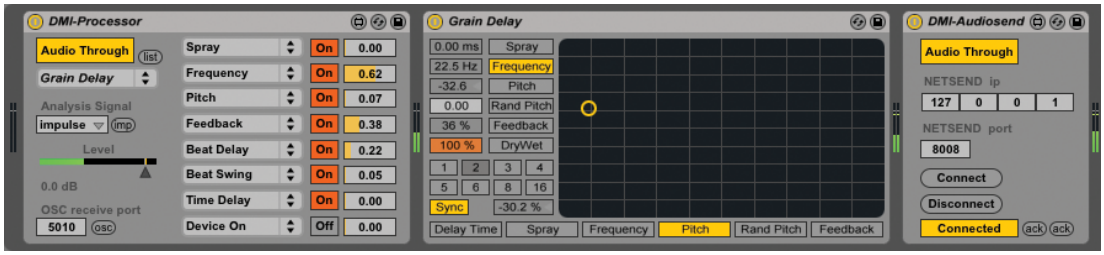

Figure 6.3: Max for Live sound processor front-end and back-end, respectively on the left and on the right side of a native Live audio effect. Front-end and back-end devices wrap the DMI enabling control and data exchange with the VCI4DMI core system for analysis and runtime control purposes.

6.1.2 Online DMI analysis

The online analytical stage is implemented in a Max/MSP patch, in accordance with the instrument analysis modes in Chapter 4. It communicates with the front-ends via OSC and receives the DMI audio output from the back-end via TCP streaming. The

patch GUI, shown in Figure 6.4, allows enabling up to eight DMI parameters and defines the respective ranges and resolutions. The system provides four analysis modes: two for generators and two for processors. The branching between steady and variable timbres occurs in the following offline analysis stage, bringing to six the total mode count as in the taxonomy of Chapter 4. The users can modify the default analytical settings that include window size, window step, analysis timing, IR maximum duration, number of analysis windows per state, analysis pitch and velocity. Feedback provided prior to the analysis process shows the resulting number of parameters unique combination, the estimation of total analysis time, and detectable period range for variable timbre analysis. The timbre descriptors are computed online and includes the loudness of the Bark critical bands, energy, brightness or spectral centroid, noisiness, spectral deviation, spectral skewness, and these can be individually disabled from the related plot bar, which in turn gets darkened. The Bark critical bands can be replaced with the MFCC, and the number of bands can be expanded or reduced in both cases. Alternatively the system allows the use of external timbre analytical tools by generating a corpus of wave files, equal in length, containing the DMI audio outputs associated with the unique combinations in **I**. For sound processors analyzed in the time domain using the Dirac impulse as input signal, descriptors are not computed online, while we the store the entire IR sampled at audio rate for each parameter combination. These are analyzed only later in the offline stage, because we require a pre-processing step in which all IR recordings are aligned to the input Dirac impulse with single sample precision. The system supports a test mode in which the descriptors are computed and visualized but not stored, while the DMI parameters faders in the GUI become interactive. The analytical settings are stored into recallable presets. Once the analysis system is configured, the process runs automatically and without supervision. The matrix of parameters unique combinations **I** is generated as soon as the analysis starts, while descriptor vectors are accumulated progressively in **D**. Finally **I** and **D** are exported to files for the following offline analysis and DMI mapping training.

Since dynamic programming is not provided in Max/MSP and Max for Live we set a maximum number of DMI parameters that the front-end and analysis patch can support. The current limit of eight parameters has been sufficient for any DMI analysis that we used in real performance and user evaluation scenarios. However this limit can be extended by replicating internal modules of the patches, while the rest of the system can support any number of DMI parameters.

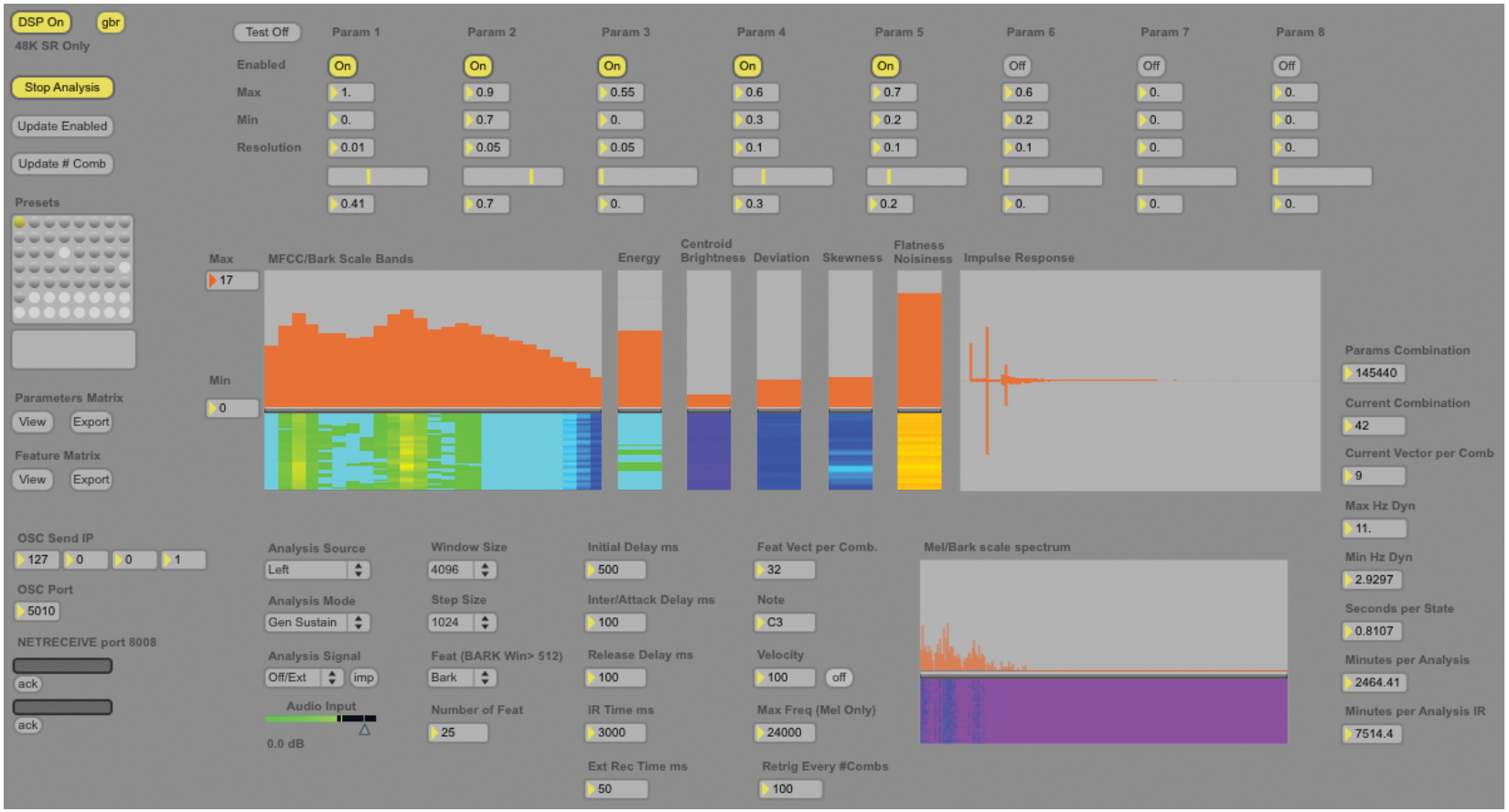

Figure 6.4: DMI online analysis GUI Max/MSP patch for selecting DMI parameters ranges and step resolutions, for selecting the analysis mode and options, and for visualizing the timbre descriptors.

6.1.2.1 DMI analysis synchronization

The DMI online analysis involves four different devices that exchange a considerable amount of data with different protocols over diverse channels, which include general-purpose computer networks that present variable latency. The synchronization of the analysis procedure is central to ensure that each **d** only contains descriptors computed at the right time over the DMI output generated by the right **i**. Therefore we implement the set of synchronization messages, propagated across the devices, ensuring that in the analysis sequential flow an operation is executed only if its upstream dependencies have been completed. This is illustrated in Figure 6.4 where the solid and dotted lines represent data and sync messages respectively. The Max/MSP DMI online analysis patch sends **i** via OSC to the front-end, which updates the DMI and waits for a Max for Live internal update acknowledgement. This is then propagated to the back-end within the Max for Live environment, and forwarded to the Max/MSP DMI online analysis patch through a positive impulse over an additional and dedicated audio channel, streamed synchronously with the two used for the DMI stereo output. When Max/MSP receives the update acknowledgment message, the timbre descriptor computation starts, and when completed a new **i** is sent to the front-end and the sequence restarts. Additional waiting intervals inserted after the parameters update acknowledge message or after the analysis completion message can be defined to handle the analysis exclusion of attack, decay, and release phases. For analysis modes that require an additional event trigger for every **i** such as the impulse generation for the processors time analysis mode, and note-on/off for decaying timbre generators, the synchronization system presents an additional internal loop, visible in Figure 6.4. The DMI online analysis patch triggers the event only after receiving the parameters update message, and in turn the **d** analysis waits for the trigger acknowledgement as well, propagated back with a negative impulse on the additional TCP streaming channel.

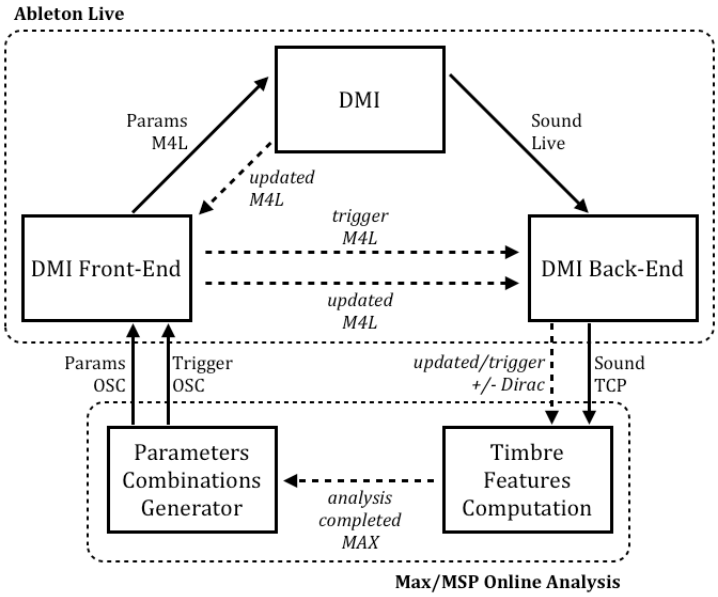

Figure 6.5: Illustration of the DMI online analysis synchronization strategy where solid lines and dotted lines represent data and sync messages respectively. This strategy allows the various components to stay synchronized even when running on different machines.

6.1.3 Offline DMI analysis and mapping training

The timbre descriptor matrix **D** computed during the analysis stage with the patch described in the previous section contains multiple descriptor vectors $\mathbf{d}_j$ for each unique parameter combination $\mathbf{i}_j$. The mode dependent **D** postprocessing and the mapping $m()$ ANN training, described in Chapters 4 and 5, are implemented in a MATLAB offline function which does not require further data exchange or synchronization with the DMI. We provide one version for the time domain sound processors and another for other modes. The arguments of the functions include the **D** and **I** files path, DMI parameters kernel flag, and the timbre descriptor matrix **D** postprocessing analysis mode identifier. Moreover for the time domain version the optional reverb IR features computation flag is required, while for the other version the input arguments include the number of descriptor vectors $\mathbf{d}_j$ per $\mathbf{i}_j$, analysis window and step size. These return mapping function, sonic space, DMI parameters neighborhood relationship, and runtime DMI mapping pre-computed data into structures, which can be directly loaded into the runtime VCI4DMI, supporting the operational modes described in 5.2.3. If the intrinsic dimensionality of **D** is higher than two, the function computes and includes in the output structure reduction and mapping functions for both 2D and 3D cases, which present differences in the uniform redistribution process as well as in the ANN based $m()$. This supports the

interfacing of the DMI with vocal GC with two or three independent output signals. During the analysis and mapping computation the MATLAB function provides initial, intermediate, and final visualizations of the sonic data and of the ANN output, similar to those in Chapters 4 and 5. Moreover the function provides training feedback information and measurements such as those in Sections 4.5 and 5.3.

6.1.4 Offline Voice analysis and GC training

Two separates offline MATLAB functions implement the blind search for the optimal voice low-level features computation settings and the SOG training that returns the vocal GC. The first function takes as input the path where the vocal-gesture and vocal-posture files are stored, plus the number of posture files. The different instances of vocal-gestures are concatenated and included in a single file, while the vocal-postures are kept in separate files because statistical measurements are computed first on the basis of individual postures. For each case across the 368 different settings explored in the blind search algorithm, the function outputs information on RMD, robust features, intrinsic dimensionality of the vocal-gesture $idim(\mathbf{V_G})$, and quality rating Q_{cf_x} for each feature group and for the overall low-level features set. The function outputs and stores in files the computation settings related to the best Q_{cf_x} case, the relative robust feature mask, the feature group normalization coefficients, vocal-gesture features matrix $\mathbf{V_G}$ and the P vocal-postures features matrix $\mathbf{V_p}$ computed with the best settings. Overall Q_{cf_x} trends are plotted at the end of the low-level vocal features optimal computation settings blind search algorithm.

The vocal GC is computed in a second independent function that implements the SOG training procedure. The input arguments include the path containing the blind search result files, and optional parameters to manually define the GC dimensionality M, the number of cluster in $\mathbf{V_G}$, and SOM lattice training parameters. Moreover we provide a separate function for the semi-supervised variant in which the multiclass LDA replaces Isomap NLDR. This requires the user to define the number of postures to be associated with the 2^M vertices. During the SOG training the function provided data and output lattice plots, similar to those included in Chapter 3. Moreover the intrinsic dimensionality of $\mathbf{V_G}$, percentage of data rejected after pre-filtering, gesture extrema detection, SOM configuration, and details on output lattice topology distortions are displayed as well. In the rare chance of output lattice with one or more topology distortions, the training is repeated. The outcome of the training is saved into a data structure compatible with the runtime VCI4DMI, supporting the

operational modes discussed in Section 3.3.1.3. The data structure includes the settings to compute the low-level voice features, the scaling and normalization coefficients, the dimensionality reduction map, the data rotation matrix, the bounding convex hull, the output lattice weights, masses, and neighborhood indexes, which are necessary to implement the vocal GC. When the blind search is not needed, the SOG training input file can be computed online using the runtime Max/MSP patch presented in the next section, which supports both live voice and file input.

6.1.5 Runtime Interface

The runtime VCI4DMI that maps in real-time a training compliant vocal-gesture onto DMI specific target parameters, is based on information embedded in the two data structures generated by the respective vocal GC and DMI mapping training functions. The runtime VCI4DMI prototype is implemented concurrently by a real-time MATLAB function, which includes two parallel jobs, a Max/MSP patch, and the Max for Live DMI front-end, which provides the final interfacing layer with the instrument. In Figure 6.6 we illustrate the implementation distributed over the three different environments.

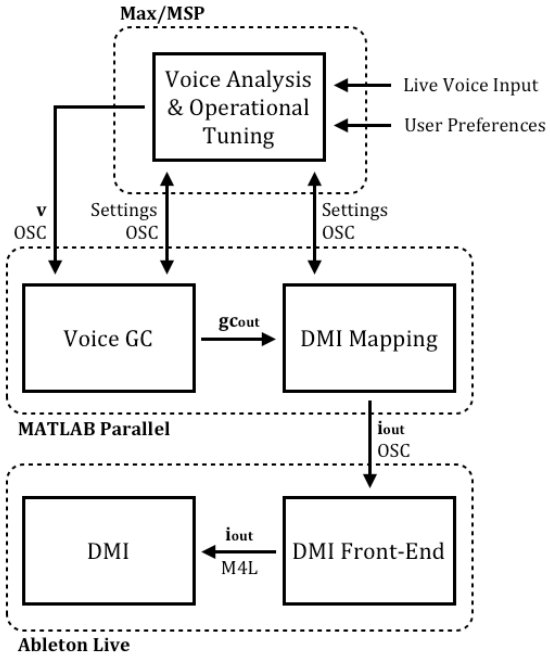

Figure 6.6: Illustration of the implementation distribution of the runtime part of the VCI4DMI prototype.

The core of the VCI4DMI is implemented in the two MATLAB parallel jobs, while the Max/MSP patch provides real-time voice analysis and the GUI to configure and tune the interface response. The clear separation of the runtime VCI4DMI into independent entities aims to exploit the computational power offered by the modern multi-core CPUs. The communication between the blocks, including those between the MATLAB internal parallel jobs, are based on protocols compatible with computer networks so that the prototype can support task distribution on separate local or remote machines, desirable with weak CPUs or in remote network musical performances.

6.1.5.1 Vocal GC and DMI mapping

The runtime VCI4DMI MATLAB function takes as input arguments the vocal GC and DMI mapping memory structure as input. At the beginning the system starts two separates parallel jobs, and initializes OSC server on the voice GC side and clients on both sides. Then it performs the initialization routine to set up the internal working configuration, compliant with the current operational mode, and it sends a sequence of OSC messages to the runtime Max/MSP patch to configure the low-level features computation settings, update the GUI related to the operational mode and to other functional options. These are always stored and updated in the two data structures during the interface execution. After initialization the two parallel jobs wait for incoming messages. If the OSC message contains operational settings, the voice GC is immediately updated and the new configuration updated in the related data structure, while if the settings are related to the DMI mapping part, the message is propagated to the second parallel job, which updates the system accordingly. When the OSC message contains voice data the GC performs the sequence of operations to compute the stream gestural controller output sample vectors $\mathbf{gc_{out}}$, which is propagated to the second parallel job that performs the DMI specific mapping and parameters output $\mathbf{i_{out}}$, are sent to the Max for Live front-end via OSC. The runtime endless loop is terminated with a specific OSC message, which determines the conclusion of the parallel jobs and returns the voice GC and DMI mapping data structures containing updated settings and operational modes, so that the specific VCI4DMI can be stored and executed again later.

The computation from the voice instantaneous low-level features vector $\mathbf{v}$ to the DMI parameters set $\mathbf{i_{out}}$ may appear mostly sequential, but the parallel implementation provides an overall speed improvement up of a factor of 1.9x compared to the sequential version, almost doubling the $\mathbf{i_{out}}$ rate limit. In the vocal

GC and DMI mapping components a consistent amount of computation is related to the update of the search spaces, which can be performed independently and in parallel. Moreover these are pre-computed for the next iteration as soon as the current $\mathbf{gc_{out}}$ or $\mathbf{i_{out}}$ messages are sent, in order to exploit efficiently the incoming message waiting time. Internal configuration update for the two components, due to OSC operational setting messages, are executed in parallel as well. Finally, having the OSC server and DMI clients running separately reduces the OSC overhead, since send and receive can happen simultaneously. The implementation includes minimal synchronization barriers when the two jobs exchange data, and the OSC server integrates a message pipe. The system detects and warns when computational overload occurs, monitoring the quantity of OSC voice data messages in the pipe.

Interface visual feedback is optionally provided by sending the $\mathbf{gc_{out}}$ to the runtime Max/MSP patch via OSC, which is represented in the M dimensional intermediate mapping space. Moreover a non-parallel version of runtime function can provide a simplified and interactive visualization of the reduced voice vector $\mathbf{v}^*$ projected over the SOG output lattice weights $\mathbf{w}_k$, displaying closer output node O_k and the bounding convex hull, plus the DMI sonic space $\mathbf{D}^*$ with $m(\mathbf{gc_{out}})$, closer entry $\mathbf{d}_j^*$, and restricted search space $\mathbf{D}_{Red}^*$, as in the 2D example of Figure 6.7. These two visualizations have a severe effect on the computational load, especially for the 3D case, and are mostly used only in the early stages when the performer is becoming familiar with and adjusting the generated mapping.

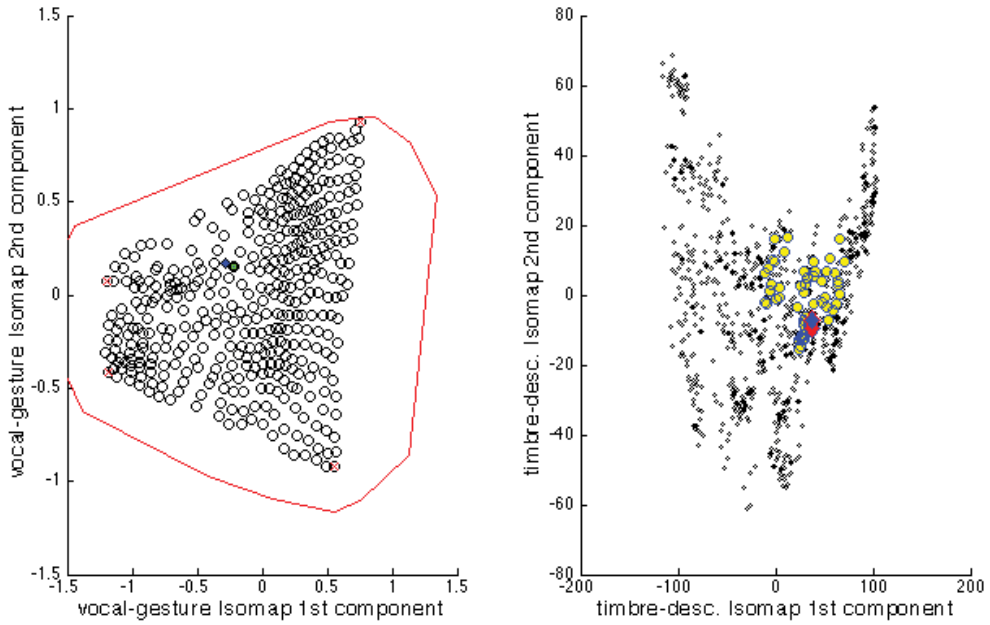

Figure 6.7: Simplified real-time user-feedback visualizations of the SOG lattice (left) and sonic space (right) implemented in the runtime non-parallel version of the prototype.

6.1.5.2 Voice analysis and runtime interface GUI

The computation of the low-level features from the live voice input is implemented in a Max/MSP patch, which streams vectors of the descriptors to MATLAB via OSC. The patch provides a GUI displaying VCI4DMI operational settings and options. These can be adjusted at runtime resulting in a modification of the data processing flow within the patch itself or in the concurrent MATLAB routine. The Max/MSP patch has full control of the MATLAB part, which runs on the in background and is transparent to the user. The prototype integrates additional functionalities, described in this section, for the effective application in real performances. In Figure 6.8 we show the GUI of the Max/MSP patch divided into the 12 sub sections detailed below.

a. System settings including IP address and ports for OSC client and server, low-level features computation configuration, set by the MATLAB runtime function or manually imported/exported. The RMD threshold can be defined for online robust features analysis.
b. Audio input and output channel selection, optional input and output from/to file, basic audio processing including gain, compressor, two-pole two-zero filter with graphical interface, and spectral visualizations of the signal before and after the processing.
c. Low-level features first order IIR low pass filter coefficient to further smooth the interface response and improve the stability of the output over vocal-postures. Coefficient values above 0.95 determine a slower and less responsive tracking of the voice timbre variation.
d. Four operational modes are provided:
    1. Runtime control, in which the computed $\mathbf{v}$ are sent via OSC to MATLAB routines that implement the vocal interface. Options allow enabling and disabling MATLAB visual feedback, local $\mathbf{gc_{out}}$ and $\mathbf{i_{out}}$ feedback, internal recording of $\mathbf{v}$, $\mathbf{gc_{out}}$ and $\mathbf{i_{out}}$ streams.
    2. Testing mode with voice low-level features computed and only visualized on screen, plus optional RMD and robust features online calculation and visualization at regular intervals.
    3. Vocal-postures online learning mode, to detect robust features based on average RMD measurement over a set of vocal-postures with fixed features computation settings. Export $\mathbf{V_p}$ and noisy feature mask files for the SOG offline training.
    4. Vocal-gestures online recording mode for generating and exporting the $\mathbf{V_G}$ file for the SOG offline training.

e. Vocal GC operational mode settings and options for changing and adjusting the response of the GC at runtime. The search can be extended to the whole output lattice or restricted to the neighbors only, with optional gravitational attraction search metric, with user defined gravitational constant. The convex hull validity check can be enabled, disabled, and optionally used to orthogonally project on the boundary vectors falling outside the hull. Two scaling factors are available to expand and contract the convex hull and the SOG lattice. The status of the interpolation by IDW can be changed as well as the power parameter. The user can change the voice to sonic space mapping by inverting the output ranges of the individual $\mathbf{gc_{out}}$. An optional adaptive mode is available to modify the GC after every performer's voice interruption. In particular we shift the SOG lattice weights to match the new input vocal-posture position $\mathbf{v}^*$ with the closer lattice vertex. Finally the GC can be bypassed to browse the DMI sonic space directly with the low-level voice features.

f. DMI mapping operational mode settings and options change sensitivity and response of the instrument dependent interface component. The mapping function $m()$ can be set to the nonlinear ANN, direct linear scaling plus offset, or bypassed browsing directly the uniformly redistributed space $\mathbf{D}_U^*$. The restricted search space $\mathbf{D}_{rst}^*$ can be set to the entire $\mathbf{D}^*$, to the first or second neighborhood level or within a distance i_{rad} from the previous $\mathbf{i_{out}}$ in $\mathbf{I}$. The DMI parameters IDW interpolation status can be changed as well as the power parameter. Finally the DMI mapping can be bypassed linking $\mathbf{gc_{out}}$ directly to the instrument parameters. The distance i_{rad} as well as the IDW interpolation exponent can be dynamically related to the instantaneous $\mathbf{v}^*$ variation, to improve the DMI output parameters stability over voice-postures.

g. Two timed gates are provided to perform or inhibit the mapping based on the live input voice characteristics. The first is based on energy level and optionally also on the detection of voiced sound. The second works on the low-level voice features flux, which is the instantaneous Euclidean distance of two consecutive $\mathbf{v}$. For both gates the user can set the threshold, monitor the real-time value, and set the minimum true condition time interval to open the gate. The interface can still react when just background noise at the input generates a $\mathbf{v}^*$ inside the convex hull, thus the gates provide a temporary interruption of the interface mapping when voice is not present at the input,

or when the voice is invariant. The timed gate is effective with burst noise such as background music, and if properly tuned it blocks short transients and loud inputs, unlikely to be voice signals, from generating $\mathbf{i_{out}}$.

h. A default combination of DMI parameters can be optionally forced on the DMI, via the MATLAB and DMI front-end chain, when the energy gate is closed. The user can specify the time interval that takes to linearly glide from the last $\mathbf{i_{out}}$ to the defaults. This functional option grants additional dynamism to the interface.

i. For performance exclusively based on the VCI4DMI we provide a simple method to trigger DMI sounds, generating MIDI notes from the voice input by tracking energy, pitch, and onset. These voice characteristics are not used within the interface, and as discussed in Chapter 3, are controlled independently from the supra-glottal or filter component of the voice production apparatus. The note pitch, velocity and note duration can be manually fixed or derived from the live voice tracking. These are sent via MIDI to the instrument and the voice onset detection determines the transmission of the note-on message. These settings are stored in the structure containing the vocal GC data.

j. Real-time $\mathbf{gc_{out}}$ and $\mathbf{i_{out}}$ visual feedback normalized to [0,1]. The first is mapped into 2D unitary square coordinates, with the third dimension mapped to the pointer diameter. The second is mapped onto 8 individual vertical faders.

k. Visualization of low-level scalar and vectorial voice features, with short time history, maximum and minimum values. When the RMD measurement is enabled, the related value for each coefficient is represented by a horizontal yellow segment for every bar, with the lower and higher vertical positions representing respectively the zero and the threshold values. The noisy features mask can be manually defined from the individual plot bars, which in turn gets darkened.

l. Plot and mask reset, plus settings, divider, mask file import/export.

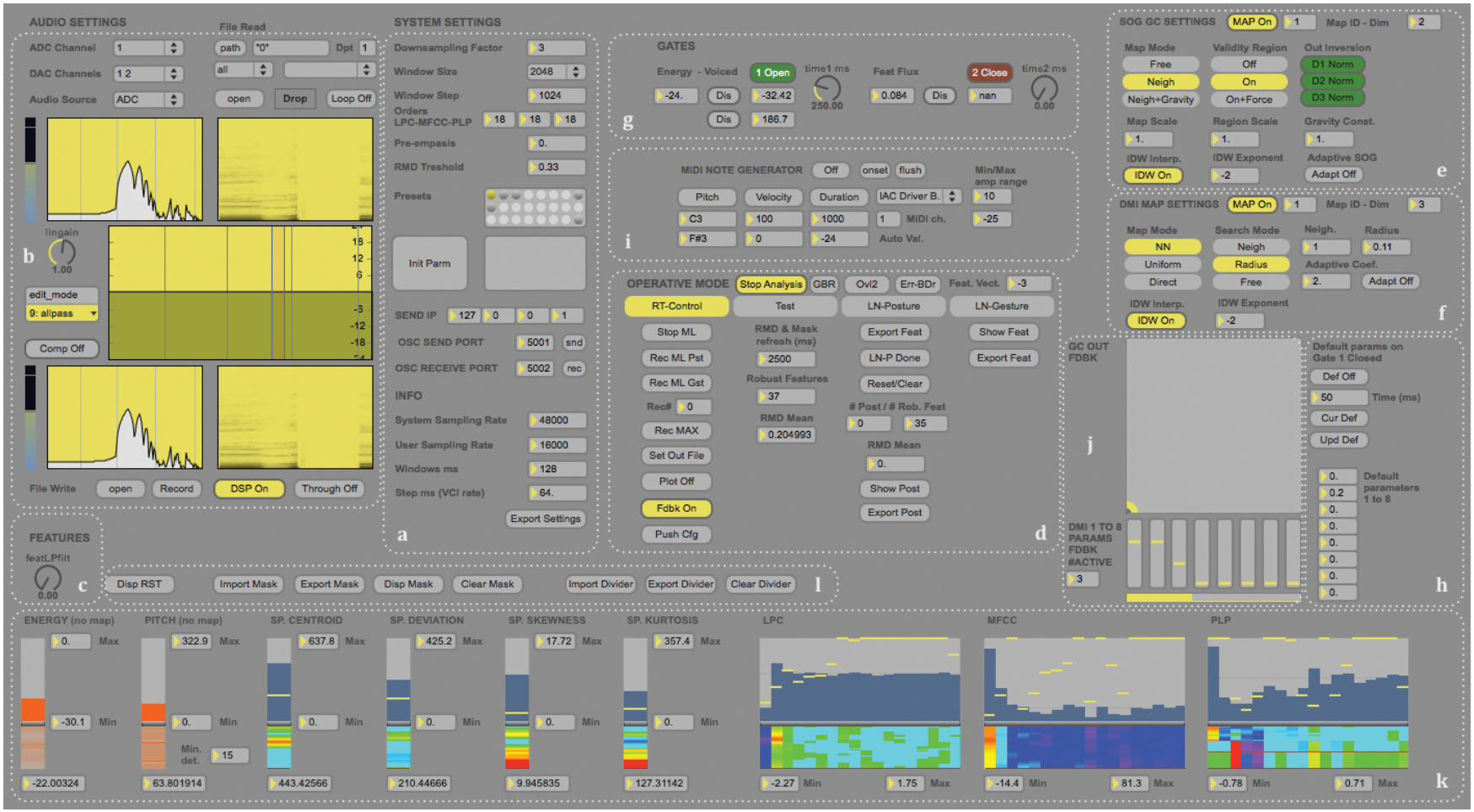

Figure 6.8: Voice analysis and runtime interface GUI Max/MSP patch, each labeled component is described in the text.

6.1.6 Libraries, Toolboxes and Externals

Besides the native functions offered by each implementation platform, the prototype depends on the external libraries, toolboxes, and externals listed below. These are integrated in their original versions or specifically modified for the VCI4DMI.

- MATLAB
    - Neural Network Toolbox[9] for generation, training and map of feed forward ANN. The function that computes the trained network output is designed for data batch processing and in the prototype this has been properly optimized to meet the hard real-time constraints of the interface.
    - Parallel Computing Toolbox[10] for the concurrent execution of the VCI4DMI runtime components, implemented in separate and independent parallel jobs.
    - Dimensionality Reduction Toolbox[11] (Van der Maaten and Hinton, 2008) used for Isomap dimensionality reduction and data projection to the low dimensional space, which has been partially optimized for real-time mapping of the voice high dimensional data.
    - Rastamat[12] library for MFCC and PLP computation and used as reference for the porting of the auditory spectrum and the Bark critical bands in Max/MSP.
    - Distmesh[13] library (Persson and Strang, 2004) used to redistribute arbitrary mesh into specific distribution and shapes, modified to meet specific termination conditions of the redistribution algorithm.
    - Kohonen's Self-Organizing Map[14] used for standard SOM comparison and then modified for the customized implementation of the SOG training algorithm.
    - Oscmex[15] library for sending and receiving OSC messages in real-time between MATLAB, Max/MSP and Max for Live.

---

[9] http://www.mathworks.com/products/neural-network/
[10] http://www.mathworks.com/products/parallel-computing/
[11] http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html
[12] http://labrosa.ee.columbia.edu/matlab/rastamat/
[13] http://persson.berkeley.edu/distmesh/
[14] http://home.wlu.edu/~levys/software/som/
[15] http://sourceforge.jp/projects/sfnet_oscmex/

- Multiclass LDA[16] for the multiple class linear discriminant analysis used in the semi-supervised SOG variant.

- Max/MSP
  - Ircam FTM[17] (Schnell et al., 2005), Gabor (Schnell and Schwarz, 2005) and MnM (Bevilacqua, Müller, and Schnell, 2005) shared library providing externals for real-time audio analysis, matrix/vector data manipulation and visualization. It supports the porting of functions between Max/MSP and MATLAB, implemented in a similar code flow and providing identical results.
  - Analyzer~[18] (Jehan, 2001), a FFT-based perceptual analysis external, used for real-time parametric onset detection and loudness tracking.
  - Netsend~[19] external for high quality and low-latency uncompressed multichannel audio TCP and UDP network streaming.
  - OSC-route[20] for OSC messages dispatching.

Other MATLAB third party functions used include *inhull*[21] for verification of data position respect with to a multidimensional hull, *medoultierfilt*[22] for parametric outlier removal from multidimensional data based on median and quartiles, a modified version of *T60*[23] for computing reverberators descriptors from the IR.

The prototype integrates a collection of externals and abstraction that we developed for the VCI4DMI implementation. These are included in the prototype repository and implement functionalities that can find use in different applications.

- FTM Max/MSP Externals: LPC to cepstrum, eigenvalues and eigenvectors, formants tracking, Levinson-Durbin recursion, RASTA adaptive filtering, RMD, polynomial roots, unique parameter combinations matrix generator.
- FTM and Max/MSP Abstractions: PLP, RASTA-PLP, MFCC, spectral flux, spectral moments, auditory frequency scale conversions, auditory spectrum, critical Bark bands, loudness of Bark bands, GC emulator.

---

16 http://www.mathworks.com/matlabcentral/fileexchange/31760-multiclass-lda/content/LDA.m

17 http://ftm.ircam.fr

18 http://web.media.mit.edu/~tristan/maxmsp.html

19 http://www.remu.fr/sound-delta/netsend~/

20 http://cnmat.berkeley.edu/patch/4029

21 http://www.mathworks.com/matlabcentral/fileexchange/10226-inhull/content/inhull.m

22 http://www.mathworks.com/matlabcentral/fileexchange/12958-medoultierfilt/content/medoutlierfilt.m

23 http://www.mathworks.com/matlabcentral/fileexchange/1212-t60-m/content/t60.m

6.1.7 Computational performance and memory profiling

The runtime VCI4DMI presents a variable computational load that limits the maximum $\mathbf{v}$ to $\mathbf{i_{out}}$ rate, and is related to the complexity of the voice and instrument models used in the system. These in turn depend on the size of vocal training data and DMI analysis option provided by the user. The time required to complete the system offline training depends on the running platform processing power. The results we present in this section are based on a MacBookPro8,2 MD322xx/A equipped with 2.4GHz quad-core Intel i7, 1.3GHz bus speed, 16GB of dynamic memory, and running OSX 10.7.5. The execution times are estimated without using the additional GPU co-processing power, and without the concurrent execution on the operating system of other CPU demanding applications.

The blind search for the low-level features computational setting over 368 cases, plus 48 mixed order cases, takes in MATLAB about 48 minutes for a data set composed of 67 seconds of vocal-gestures and 10 vocal-postures with a total duration of 69 seconds. With this dataset the SOG training takes about 9 minutes for a 2D lattice with r_{SOM} equal to 22 and a total of 484 output nodes and approximately 53 minutes for a 3D lattice with r_{SOM} equal to 10 and a total of 1000 output nodes. Within the SOG training total time an average 69% is spent for the SOM modified training, 16% for Isomap, 5% for the extrema search, and 2% for the data pre filtering. The data structure containing the vocal GC data is roughly 500KB for the 2D case and 600KB for the 3D case, in which the greatest portion of 400KB is used to store the shortest path neighborhood graph of the Isomap dimensionality reduction. For the variant with multiclass LDA the time spent for dimensionality reduction as well as the memory occupation of the vocal GC data structure is lower.

The control and synchronization with the DMI and the computation of the timbre descriptor in the DMI online analysis, performed at audio rate, usually requires a small fraction of the computational capacity of a single core. The complexity varies mostly with the window and step size. These are equal to 4096 and 16 samples respectively in the worst and most unlikely settings, in which we reach 100% single core load for 48KHz sampling rate. In Chapter 4 we related the DMI online analysis total time to the size of $\mathbf{I}$, which in turn also affects the execution time of the offline DMI mapping. Thanks to the parameter interpolation, analyses with $\mathbf{I}$ including from 1K to 5K are generally sufficient to model the DMI and provide accurate sonic control. Here we consider a more complex case with a decaying sound generator analyzed for 11.5K parameter combinations, 50 analysis windows of 4096 samples each at 512 step size for each $\mathbf{i}$, with T_{adj} equal to 100ms, requiring approximately 2

hours and 23 minutes for the online analysis. The decaying sound generator case presents the highest load for the dimensionality reduction stage since the dimensionality of **D** is the highest. For this case the training took 27 minutes to generate the mapping for a 2D GC, while for the 3D it required 34 minutes. An average 53% of the time is spent for the ANN nonlinear regression, 19% for the uniform data redistribution, 8% for Isomap, and 3% on the mode dependent analysis. The output structure containing the data for 2D and 3D mappings is 2.7MB in size.

The system training execution can be longer than the two cases discussed above if the training data size is larger. However larger datasets would only be necessary for high-complexity cases, while on average the amount of vocal data and DMI analyzed parameter combinations is about 40% lower than detailed above, requiring half of the execution time for the training. The dynamic memory consumption generally does not exceed 3GB. The MATLAB offline training functions are not optimized so that CPU and memory consumption can be sensibly reduced eliminating plots, removing intermediate evaluation routines, removing obsolete data from memory more frequently, optimizing loops, and parallelizing the computation, which is largely possible such as for the blind search. We estimate that there is enough room to reduce the execution time of each function by 50% to 70%. This can be further lowered downgrading the precision or termination condition in several components of the training procedure. The double float precision only used in MATLAB, is not essential and the porting to 32bit platforms is possible without performance decrease.

For the runtime, the VCI4DMI Max/MSP patch computational complexity is dominated by the real-time vocal features computation and data visualizations, which can contribute an extra 50%. With the visualizations disabled and the patch stripped to the bare minimum, as described later in Section 6.3, the real-time constraints are satisfied on a single core system computing a new **v** every 1ms and using the most complex low-level features settings from the blind search range (highest sampling rate, largest window, highest feature order). The load estimation also includes the audio channel pre-processing, the OSC data transmission, onset, pitch, and energy tracking for the MIDI note generator.

The computational complexity of the vocal GC and DMI mapping implemented in MATLAB depends on the complexity of the system embedded in the data structures returned from the training algorithms. However these represent the bottleneck of the system, with the Isomap reduction and ANN $m()$ contributing the most to the computational complexity. With the vocal GC and DMI mapping derived from the two high complexity range cases discussed before in the learning part, we obtain a maximum **v** to $\mathbf{i}_{\mathbf{out}}$ rate of 5ms for the 2D case, and 10ms for the 3D case.

For this case the Isomap dimensionality reduction presents a neighborhood graph matrix with 3.5M entries, and the ANN has respectively 20 and 7 neurons in the two hidden layers. For these cases we profiled the joint GC and mapping execution over 10 minutes of alternate use of the 2D and 3D configurations, with input gates always open and running with the most complex operational mode. The results indicate that the Isomap reduction is responsible for 26% of the total load, mostly due to the internal shortest path search Dijkstra (1959) algorithm. The ANN $m()$ contributes 12% of the load, the convex hull validity check contributes 8% especially from the 3D cases, the OSC reception and transmission accounts for 12% and 7% respectively, the Euclidean distances used mostly in the DMI mapping routines comprise 10% of the load, the $\mathbf{v}$ normalizations contributes slightly less than 2%, and the remaining 23% is due to other computations accounting for less than 1% each. The semi-supervised GC variant with the multiclass LDA dimensionality reduction results in a lighter GC load. The computational load profiling supports the fair partitioning of the computational cost between the vocal GC and DMI mapping parallel jobs. These results are obtained with Isomap and ANN mapping functions optimized for single input and real-time computation. The original versions provided in the respective toolboxes present higher computational load halving the maximum $\mathbf{v}$ to $\mathbf{i_{out}}$ rate. The OSC implementation in the MATLAB is not efficient compared to other programing environments, so there is potential for further computational load reduction.

The computational limit of the vocal GC and DMI mapping is satisfactory for voice input since the acoustic characteristic of the voice is quasi-stationary below 10ms to 20ms. The latency of the system is given by the sum of the voice analysis window and step lengths, plus a variable OSC network transmission delay, and it is typically below 24ms. This does not include the additional DMI-dependent time for the output sound to reflect the changes determined by updated input parameters.

6.2 User and audience perspective

In addition to numerical evaluation and proof-of-concept, the functional prototype of the VCI4DMI provides a valuable platform for the user evaluation of the interface principles and practical implementation, which is central in interface design. The voice-controlled interface presented in this thesis, as well as the software implementation, are the result of several cycles of iterative design (Nielsen, 1993) in which the usability issues, limitations and shortcomings, identified in the early versions of the prototypes, were progressively fixed and refined. The current version

of the system still has room for improvements, but provides usability, stability and features sufficient for use in real musical performance. The open-source code, the wide set of learning options and runtime modality settings promote the use of the system as a mapping exploration tool for researchers and musicians, and permits extensibility through the integration of internal modules in other musical interfaces.

6.2.1 User perspective and workflow

The minimization of the user interaction and expertise required to setup ad hoc vocal interface was among the design principles for VCI4DMI. The setup workflow, from the point of view of users familiar with the concept of voice timbre variation as a musical control gesture, requires the user to:

1. record several instances of vocal-gestures;
2. record vocal-postures with timbres included in the vocal-gestures set;
3. train the vocal GC selecting the unsupervised or semi-supervised mode, and optionally define the output dimensionality;
4. select the DMI target parameters ranges, resolution and analysis mode;
5. run the DMI analysis and train the DMI mapping component of the interface.

These steps require time but no action or input from the user in defining the mapping, and compared to supervised ML methods requiring coherent input-output examples, the preparation of the training set is relatively simple and quick. Although the two training procedures provide visual and numerical feedback pertaining to learning outcome and precision, the user cannot predict how the voice will be mapped to the GC output and in turn to the DMI sonic space. Therefore the price to pay for the use of unsupervised ML is the need for practice and familiarization with the automatically generated adaptive mapping. For this purpose the visual feedbacks play an essential role, as expected and verified in the user evaluation and described in the next chapter. Visualizing the interactive SOG lattice, the sonic space and $\mathbf{gc_{out}}$ signals facilitate the performer process of understanding the voice to DMI sonic space relationship. The DMI parameter visual feedback is less significant, especially when more than three parameters are involved. The GUI runtime options, in particular the inversion of individual $\mathbf{gc_{out}}$ ranges and the adjustment of the i_{rad}, are essential to tune the VCI4DMI response according to the needs of the performer. After practicing and memorizing the mapping, performers are in general able to control the DMI with only the output sound as feedback. Moreover the numerous runtime operational options allow gradual adjustment of the interface between two extreme behaviors: the

univocal relationship between voice and instrument timbres, and timbre variations as trajectories and directions for the browsing of the sonic space. The interface modular design promotes the reusability of the vocal GC or DMI mapping in different vocal interfaces, and this can further accelerate the system setup.

6.2.1.1 Limitations and drawbacks

The SOG and DMI mapping method, as well as the prototype implementation, support any dimensionality M for the uniformly distribute hypercube representing the intermediate layer connecting the two components of the VCI4DMI. The control benefits given by higher M values are traded off by higher training time and runtime computational load, limiting the maximum voice features $\mathbf{v}$ to DMI parameters $\mathbf{i_{out}}$ rate. The main issue with M bigger than three is the inability to provide effective and cohesive visual feedback, crucial for learning and using the mapping. From the early use cases we observed a cognitive overload in the interface use and severe difficulties in interpreting the generated mapping, not in line with natural control and concurrent use design principles. With M equal or larger than four we experienced overall usability and control poorer than the 2D and 3D systems trained with the same data set. This relates to the HCI principle that a key interface issue is the tight matching between control and perceptual structure of the task, thus the simply addition of more degrees of freedom does not necessarily result in any improvement (Jacob et al., 1994). In the prototype we limit M to three at most, which still provides rich and complex DMI control, due to the adaptation and effective control dimensionality reduction granted by the control strategy implemented in the perceptual sonic space.

The vocal GC can also respond to vocal timbres not presented in the training set, and this can represent a shortcoming. Also, since we discarded the temporal unfolding information and we focused on the spatial distribution of the data, for the GC it is sufficient that the input sound generates a coordinate within the convex hull boundary. This can be considered a drawback or a creative improvisation potential of the interface, but it is intrinsic in the gestural data analytical approach we took in this work, which provides low latency and a higher degree of freedom compared to other methods based on input recognition tied to the temporal evolution of the voice timbre.

6.2.2 Audience perspective

In Chapter 2 we discussed that when novel DMIs and NIMEs are used in live performances, the audience needs to understand the relationship between gestures and

sound first to then recognize the skills of the performer. This is perceived as an embodied phenomenon, in which confidence and naturalness of the interaction determine the intimate incorporation of the instrument, considered as an extension of the performer (Fyans and Gurevich, 2011). The intelligibility and coherence of DMI interfaces are often issues for the audience. Moreover poorly visible interface bodies or the use of generic devices such as tablets and laptops, can result in a perception by the audience of lack of active creation and of visible causal link between performer and sonic result (Zadel and Scavone, 2006). Puckette, whose work contributed widely in bringing laptops on musical stages, recognizes the necessity of gestural legibility to support the audience understanding in live performances (Puckette, 1991). Visual and corporeal aspects are "extra-musical requirements" in performances, but if considered from the audience perspective, they make the playing action more committed, convincing and effective (Schloss, 2003).

The VCI4DMI can present further performance perception issues because the voice is not a common musical interface input modality, furthermore the audience is not exposed to the gesture since the voice does not reach the listener through the loudspeakers. Moreover, in this context the interface is an abstraction, which only exists in the form of a software algorithm. There is no physical device capturing visible gestures besides a standard microphone. Finally during performances the VCI4DMI can be rapidly reconfigured, resulting in a drastic change of the voice to instrument relationship. Despite these issues, in performances exclusively based on the VCI4DMI we observed that the unusual interface input modality strongly captured the audience willingness to invest effort in understanding the interface control principles. Moreover the absence of a physical interface or general-purpose non-musical devices allowed the audience to focus on the visible performer vocal-gesture. Although the voice is not audible, visual cues are still available and play an essential role. The relationship between mouth opening and DMI sound generation control was immediately evident. To a certain extent voice can be seen rather than heard (Mcgurk and Macdonald, 1976; Rosenblum and Saldaña, 1996) and the relationship between sound and mouth shape has been included in some musical interfaces and interactive systems (McLean, Shin, and Ng, 2013). Thus also in this case the audience can go through the gradual process of associating different vocal sounds, related to specific performer's facial and body expressions, with instrument timbre response, and then acknowledge the performer's musical control skills.

6.3 Performance version

Next we present an additional version of the runtime part of the prototype developed to meet the tighter requirements of a solo performance in which the VCI4DMI controls a set of DMIs hosted in Live. The DAW offers a powerful platform for musical performance, so we improve the integration between prototype and Live to facilitate configuration and use of complex setup. Moreover we reduce the system controls and feedback to fit into a minimalistic wrist controller to avoid laptop interaction on stage.

6.3.1 Bank of settings and fast reconfiguration

The prototype presented in 6.1, and in particular the runtime VCI4DMI, is effective for the vocal control of a single DMI, but it presents usability issues when the user wants to change while performing the target DMI or switch to a GC trained with a different vocal-gestures set. This required restarting the execution of the MATLAB runtime component with different vocal GC and DMI mapping data structures plus the modification of OSC settings in the target DMI front-end. In order to minimize reconfiguration time, we implemented a performance-oriented prototype version that at startup loads a bank of vocal GC data structures and a bank of DMI mapping data structures, with individual and independent OSC clients. The same vocal GC or DMI mapping data structure can be indefinitely replicated in the input banks and configured with different operational modes and settings. From the Max/MSP GUI the user can switch the vocal GC, the target DMI mapping, or both. This generates an OSC message for the MATLAB part that determines the temporary interruption of the vocal analysis in the runtime patch, the loading and of the new data structures, the internal vocal GC and DMI mapping components reconfiguration, the update of GUI features computation settings in the Max/MSP patch via OSC messages. When the sequence is completed the voice analysis is restarted and the newly configured VCI4DMI is ready for use. This process has been optimized and interface interruption for the reconfiguration takes less than 500ms. The two banks updated with modes and options set via the GUI are returned to the user upon execution termination. In Figure 6.9 we show the performance version of the voice analysis and runtime interface GUI Max/MSP patch, in which we added the extra performance control features, dedicated hardware system controller communication, pitch scale filter for the MIDI note generator, and excluded CPU consuming data visualizations.

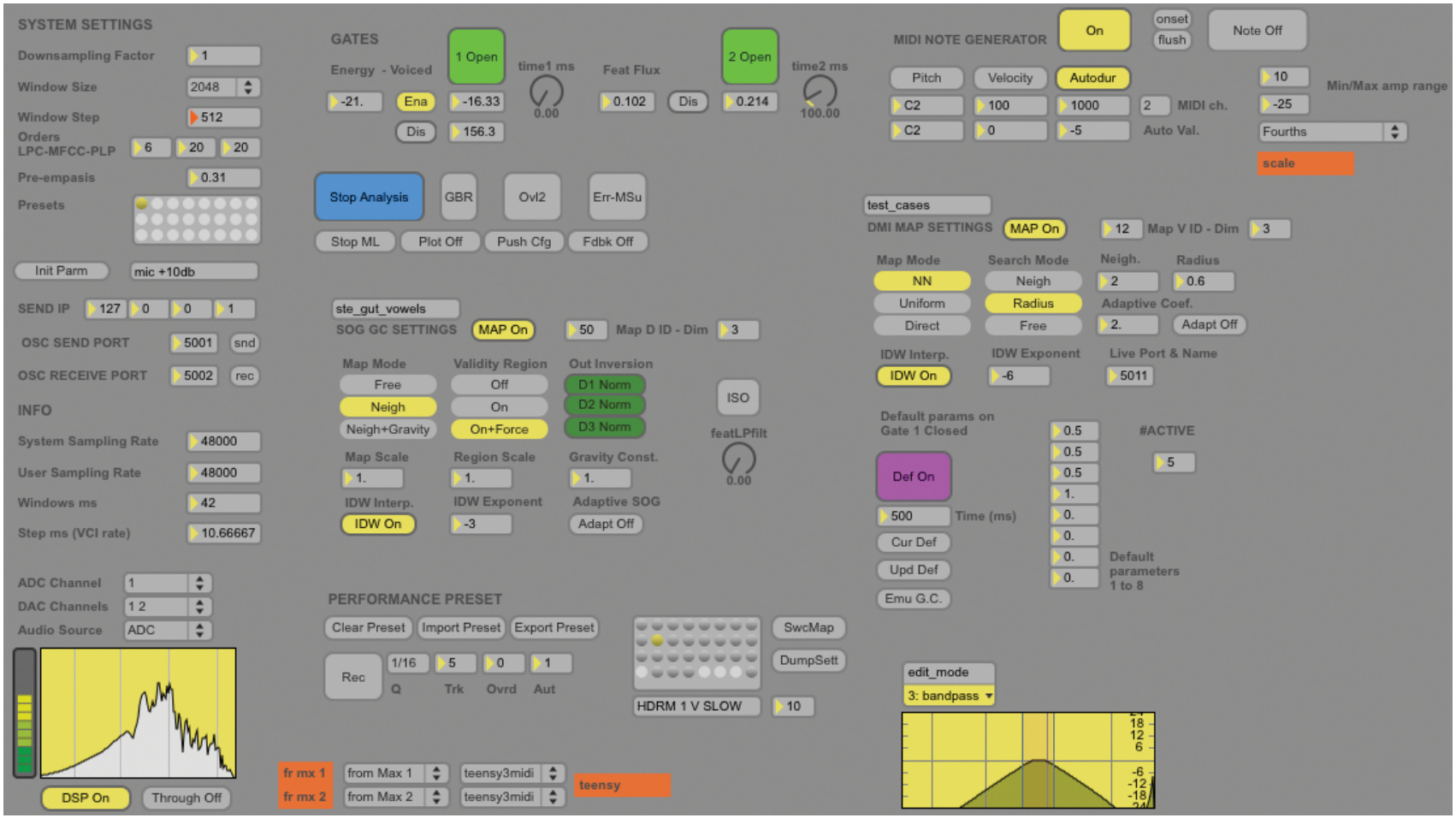

Figure 6.9: Performance version of the voice analysis and runtime interface GUI Max/MSP patch.

6.3.2 Performance setup

The live performance "One at a Time by Voice"[24] that we presented at the NTU/ADM Symposium on Sound and Interactivity 2013 concert and at the 2014 Margaret Guthman Musical Instrument Competition, aimed to demonstrate the musical control potential of the VCI4DMI. The performer builds an improvised composition interacting with a set of DMI hosted in Live using the voice as the sole input modality. The VCI4DMI is meant to be an extension to traditional controllers rather than an alternative, but this performance excludes other musical interfaces to facilitate the audience understanding of the mapping control strategy. In the setup we exploit Live DAW features by adding some features to the Max/MSP runtime patch. The user can store and recall working presets associated with:

- a pair of vocal GC and DMI mapping from the bank loaded in MATLAB, determining a target DMI instrument in Live;
- target a track in Live that receives the MIDI messages generated in the Max/MSP patch from the live voice input;
- perform Live track recording, MIDI quantization and overdub.

The set of instruments hosted in Live and controlled with the VCI4DMI includes 11 synthesizers, 4 effects and 6 elements of a drum set. The performer changes the vocal interaction mode and currently controlled instrument just by pressing the preset button on the hardware interface. The DAW loops on four bars reproducing control parameters and events generated from the VCI4DMI and optionally recorded on each track. Thus the performer can "layer up" a musical piece interacting and recording one instrument at a time using the voice exclusively. The selection of a head-worn microphone in the performance, visible in the two stills of Figure 6.10, leaves the hands of the performer mostly free, which is a conscious choice to emphasize the possible concurrent use of the VCI4DMI with other hand-based interfaces.

[24] Demo and performance videos available at http://stefanofasciani.com/vci4dmi.html

Figure 6.10: “One at a Time by Voice” performed live at the NTU/ADM Symposium on Sound and Interactivity 2013 concert (left) and using the wrist-mounted special-purpose hardware interface at the 2014 Georgia Tech Margaret Guthman Musical Instrument Competition (right).

6.3.3 Minimalistic wrist controller

In the performance setup on the left of Figure 6.10 the laptop provides the essential feedback on the VCI4DMI status while the different interface preset recall and DAW recording options were mapped to the buttons of a standard MIDI controller. In order to remove the dependence on the laptop setup in performance, we developed a minimalistic wrist controller that provides sufficient system control and feedback. The core of the hardware, in Figure 6.11, is a Teensy 3.0[25] USB development board that features a 48MHz ARM Cortex-M4 32bit microcontroller, with peripherals, development environment, and libraries compatible with Arduino. The wrist controller is equipped with four push buttons, one rotary encoder with push button, one vibration motor, and a 160x128 pixel TFT LCF screen. The firmware running on the board implements a class compliant USB-MIDI device. The bidirectional communication with the Max/MSP runtime VCI4DMI patch is exclusively based on MIDI note-on/off and control change messages. A rotary encoder is used to browse the performance presets, identified on the LCD by a string of 5 characters, while the preset is selected and enabled by pressing the encoder push button. Three push buttons toggle the enabled/disabled state of the DAW track recording, the voice driven MIDI note generator, and the DMI default parameters on gate close. The

[25] https://www.pjrc.com/store/teensy3.html

fourth button triggers a user defined MIDI note, allowing the use of the vocal control for DMI parameters generation only. On the screen the five colored LED-like indicators provides feedback on the working status of the interface and of the selected preset, the gates on the input voice in the Max/MSP patch, the voice driven MIDI note generator, the DMI default parameters on gate close, and the DAW recording. The note-on active state of the MIDI note generator is mapped on the vibration motor and on the border of the top screen region. The background of the screen turns red when errors or problems are detected within the system, such as overloading. A time progress indicator on screen, directly controlled from Live, informs the performer on the current position in the DAW four bar loop. Finally the live $\mathbf{gc_{out}}$ is mapped to the 2D coordinate and color gradient of the square in the top section of the LCD.

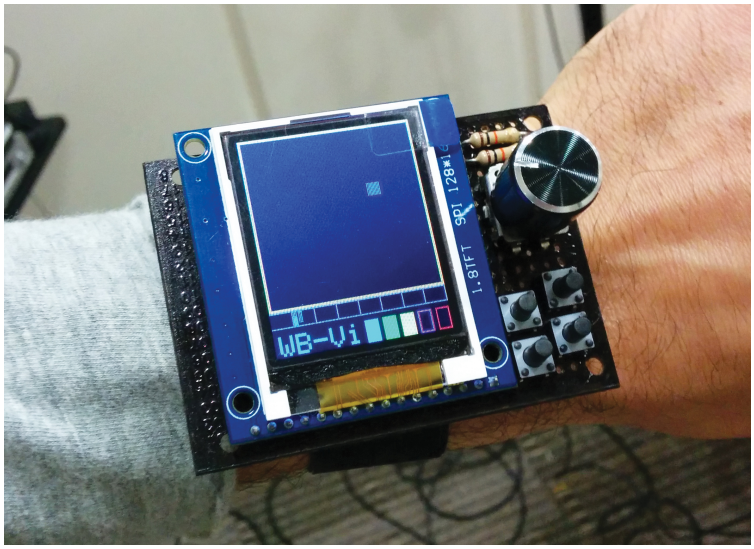

Figure 6.11: Minimalistic wrist device that provides all the core functionality and feedback necessary for performing with the VCI4DMI.

6.4 Summary

In this chapter we presented the implementation of the functional proof-of-concept prototype, which includes the offline learning algorithms and online mapping for the VCI4DMI system described in this thesis. We detailed technical aspects of the distributed software implementation, based on different development environment, and we discussed for each component the necessary input data and generated output results. In addition we presented the optimization strategy and the resulting computational complexity for a typical high demanding interfacing case. We analyzed the system from the user point of view, reviewing the procedure to set up ad hoc DMI interfaces responding to specific user vocal-gestures, highlighting

limitations and drawbacks. Moreover we briefly discussed the audience perspective on musical performances involving our novel interface. We concluded the chapter presenting a solo performance exclusively based on the VCI4DMI for which we developed a dedicated version of the prototype and a wrist physical controller to meet the tighter performance demands for musical control and instrument set interaction.

Chapter 7

User Evaluation

In this chapter we present the results of a study that involves users in evaluating the VCI4DMI system. Expert musicians and performers were invited to discover, explore, and perform with our novel interface, identify advantages and limitations of the vocal interaction, and discuss possible benefits of using the system in their specific performance scenarios. We begin the chapter briefly discussing methods and open issues in the evaluation of musical instruments and interfaces. Then we illustrate our specific methodology starting from the participant selection, going through the experimental setup and procedure, the interview guidelines, and finally the analysis strategy of the collected data. Then follows the presentation of qualitative and quantitative results of the formal study, including observations and key discoveries. The chapter ends with a discussion on findings versus initial expectations in performing with the VCI4DMI, highlighting drawbacks and limitations.

7.1 Evaluation of instruments and interfaces

A systematic strategy for the evaluation of musical instruments and controllers is challenging and previous studies in this the field still present conflicts, discrepancies and limited scope (Fels, 2004). The adoption of HCI evaluation methods are effective mostly for interfaces comparison within specific tasks, which are application and context dependent (Orio, Schnell, and Wanderley, 2001). These typically require quantitative empirical demonstrations about the functional advantages introduced by the new design. Timing and rhythm of tasks, as well as feedback interactions, are peculiar in musical context and have no obvious parallel in HCI. These represent additional challenges in determining solid experimental procedures. The majority of recent work on novel interfaces generally present informal evaluations or sometimes none at all (Stowell, 2010; Marquez-Borbon et al., 2011). Although highly innovative, most interfaces have been designed to fit specific inventor needs and ideas, making their comparison often meaningless. Bounding the evaluation span to a set of pre defined tasks, typical for HCI input devices, may restrict the view to original and artistic use of the interface, blending the different concepts of

controllability and expressivity (Dobrian and Koppelman, 2006). The advantages and drawbacks of each interface strongly depend on the individual evaluator background and aim. The design and evaluation of a DMI or musical interface involves several stakeholders. Performer, audience, designer, and eventually manufacturer, may have different evaluation goals and methodologies to measure enjoyment, playability, robustness and specifications achievement (O'Modhrain, 2011). Their perspectives affect different phases of the design cycle. The performer is the most important stakeholder because of his exclusive access to intention, result and interaction experience. In performer-centered evaluations the recruitment of subjects can be an issue if the available participant population is small, especially when the design addresses users with specific backgrounds and expertise (Wanderley and Orio, 2002). Therefore the evaluation strategy for small sample size should be focused on the identification of trends and patterns, since statistical measurement or generalization can be misleading or inconsistent. Moreover short evaluation sessions are sufficient to assess only traditional usability factors, while evaluating creative and exploratory affordances require observations over longer period (Gelineck and Serafin, 2009).

The design of an interface is the rationale outcome of an engineering process, often driven by creative design. The final musical and aesthetic results can be subjective and criticized, but scientific methodology to verify the initial engineering principles is a key factor (Wanderley and Depalle, 2004). Therefore formal evaluations, qualitative or quantitative, are fundamental because they generalize the test outcomes and provide the basis to other researchers for further improvements and innovations. In the context of this thesis the user evaluation is complementary to the numerical quantitative results presented in the previous chapters and extended here. The user-centered evaluations are essential to determine how a novel interface is received and whether it enables creativity and expressivity, how it imposes or suggests new modes of thinking, interacting, and organize time or texture in music. Learnability, control efficiency, output diversity, reproducibility, virtuosity, linearity, and predictability are characteristics that contribute to make a "good" instrument or interface, and should be considered in design and evaluation phases (Jordà, 2004a; Jordà, 2004b). An instrument or interface expressivity is hard to quantify because it is highly subjective and depends on the performers control aim and output perception. In order to allow musicians to perform expressively, nuances must be transferred into different parameter domains and behavior modes (Malloch et al., 2006). Expressivity depends on and can be associated with the range of available choices, rather than with the number of controllable instrument parameters (Clarke, 1988), and it is also related to the dimensionality of the interface gestural control (Pressing, 1990). Further this is

also strongly influenced by the selection of the controlled DMI when evaluating interfaces only (Gelineck and Serafin, 2009). The lack of performer's experience measurement in the evaluation methodologies can be addressed with non-intrusive monitoring of physiological data such as electroencephalography, electromyography, and galvanic skin response sensing (Kiefer, Collins, and Fitzpatrick, 2008). Formal user-centered evaluations do not guarantee effective musical interface design, though it can certainly identify poor ones, or eventual drawbacks and limitations. Moreover it enables user-centered iterative design (Norman and Draper, 1986) when considering feedback, further task requirements, and solicits for additional functional features in refining and re-implementing the interface.

7.2 Method

The VCI4DMI is neither an instrument nor an interface, but an unsupervised framework to implement generative mappings between voice and DMI. The specific interface depends on the training data that the user has provided and on the characteristic of the controlled DMI. Such systems are challenging to validate due to the lack of evaluative criteria (Marquez-Borbon et al., 2011). In the user evaluation, comparison with other voice driven system is impractical because related prototypes are not distributed and because they are too different in scope and aims from the novel and unique features of the VCI4DMI. Finding participants with previous experience in vocal control is highly unlikely too. Our evaluation method is not limited to measurements and observation of use-cases and specific tasks, but it includes free exploration of the system, from which we attempt to identify user experience, embodied interaction, value-sensitive design, and affective computing (Harrison, Tatar, and Sengers, 2007) by analyzing the audio-video recordings and the interface activity logs. The qualitative evaluation is mainly based on the analysis of the recorded individual open-ended in-depth interviews. Each subject was interviewed before and after the using the VCI4DMI to determine the individual perspective on the system, aiming to identify pattern and trends, as well as limitations and directions for improvement.

7.2.1 Participants recruitment and selection

In qualitative research, the selection of appropriate size and composition of the sample involved in the study is critical. For phenomenological research, which

describes a lived experience, a range from 6 to 12 participants has been reported as sufficient due to the emergence of thematic redundancy after hearing half of the subjects (Thomas and Pollio, 2004). A more recent trend, with practicality issues, does not pre-establish the sample size but ends the study only when reaching theoretical saturation in the results (Guest, Bunce, and Johnson, 2006). Qualitative studies are mainly focused on a small number of participants who represent the phenomena of interest, but participants should be purposely selected to represent rich knowledge about the study questions (Gubrium, 2012). In our study we fixed the initial sample size to 10 subjects but we recruited 6 additional participants for further investigation if results turned out to be conflicting or unclear. The relatively low sample size favors the ability to probe participants' deeper and protracted use of the VCI4DMI, and increases the pertinence of interview data with the interface experience.

The VCI4DMI has been designed to provide an additional layer of control to musicians using DMIs in their performances, which span from players utilizing minor audio processing to the sound of their acoustic instruments, to DJs and live remixers mixing and filtering pre recorded sound material, including songs and samples libraries. We recruited participants from this wide category, favoring experienced professionals or semi-professional musicians, with no formal training in singing, age ranging between 21 and 65, and we selected the most heterogeneous sample that can provide multiple perspectives while being representative of the general potential user population (Highhouse and Gillespie, 2009). We recruited participants by snowball sampling (Goodman, 1961) and posting an advertisement on bulletin boards of local communities of electronic musicians, music technologists, DJs and composers. The advertisement introduced the research context, summarized the study protocol, but did not include specific details on the subject of the study other than the generic title "voice-controlled interface for digital musical instrument", to prevent participant biasing. Interested subjects were asked to provide a brief profile, including their performance practices and relevant experiences, to determine the most appropriate and diversified participants sample. The recruitment lasted for approximately 2 months while the 10 individual evaluation sessions were conducted within 3 weeks.

7.2.2 Experimental setup

Identical setup and environmental conditions were used for the 10 evaluation sessions. To let the participants feel comfortable and ease eventual inhibition we ran the interviews and evaluations in a soundproof closed room, with a camcorder placed

in a corner and facing the side of the participants. The VCI4DMI used in the evaluation is slightly precedent to the final version presented in Chapter 6, but it includes identical functionalities, control capabilities, support for mapping banks, and it was derived from preliminary user-centered design iterations. To facilitate the exploration and interaction with our novel system the GUI grouped and highlighted the sections to adjust different components of the system, and these were also mapped to a hardware interface with labels and LED feedback, as in Figure 7.1. Eight different DMIs, hosted in Live, were selected for this study, covering a broad range of possibilities. The DMI analysis and mapping computation were executed beforehand. This avoided spending a significant fraction of the sessions for instrument selection and training, and it allowed the presentation of an identical experimental setup to all participants. However we illustrated to participants the principles and a demo of the DMI parameter-to-sound perceptual analysis and of the control strategy implemented starting from the DMI specific sonic space. Further, participants were likely to be new to the abstraction of simultaneous parameter control through dimensionally reduced sonic space, so we included simple DMI cases with a low number of variable parameters and space entries. The experimental instrument set is presented in Table 7.1, which includes DMI details, type, analysis mode, number of controlled parameters and entries in the sonic or parameter spaces.

ID	*DMI Description*	*Type*	*Analysis Mode*	*# Params*	*Space Entries*
1	Wavetable-Based Synth	Generator	Decaying	4	5324
2	Wavetable-Based Synth	Generator	Decaying	5	11664
3	FM Synth	Generator	Steady Timbre	6	4840
4	Virtual Modular Synth	Generator	Steady Timbre	4	1296
5	Granular Synth	Generator	Variable Timbre	5	7116
6	Delay+Reverb Chain	Processor	Time IR	3	388
7	Low Pass Filter	Processor	Freq. Steady Timbre	2	478
8	Guitar Amp Emulator	Processor	Freq. Steady Timbre	5	7200

Table 7.1: Characteristics of the DMIs pre-analyzed and available for use in the evaluation sessions.

The sound generators could be triggered with an external piano-like MIDI keyboard, with the VCI4DMI note generator, or with midi clips triggered from a dedicated controller, used as well to launch a selection of music loops for the sound processors input. Hence the setup included a keyboard and a clip launching button matrix. Users were free to configure and perform with the available devices and the DAW according to their personal preferences, as well as to simultaneously use the vocal interface and hand-based controllers. A large 46-inch screen was used to display the

interface visual feedback and the DAW, while the VCI4DMI GUI was displayed on a smaller 15-inch laptop screen. Finally a head-worn microphone facilitated the simultaneous use of hand-based controllers for playing the DMIs or tuning the VCI4DMI. The experimental setup is visible in Figure 7.2, in which the additional microphone and laptop are only for interviewing and audio recording purposes.

The participants provided vocal-gestures and vocal-postures for training two different 2D vocal GCs, reducing the dimensionality with Isomap and multiclass LDA respectively, and one Isomap based 3D vocal GC. Moreover we reduced by a factor of 4 the number of configurations considered in the features setting blind search to minimize the training time. The VCI4DMI hardware controller allowed the user to select and pair any available vocal GC and DMI mapping. As in the performance version, the VCI4DMI used for evaluation supports runtime GC and mapping reconfiguration, operational modes and functional options store and recall, automatic DAW MIDI and OSC messages routing to the specific DMI.

Figure 7.1: Hardware interface with labels and LED feedback to control and tune the VCI4DMI, including vocal GC and DMI mapping selection, operational modes, and additional functional options.

Figure 7.2: User evaluations experimental setup.

7.2.3 Protocol

In each user evaluation session we followed the protocol summarized below, developed by running two preliminary sessions with additional subjects for which we did not include the collected data in the study.

1. Brief illustration of the evaluation session protocol, participant's rights, study aim and research context.
2. Start audio-video recording.
3. First interview.
4. Description and demo of the overall VCI4DMI for vocal simultaneous real-valued parameters control.
5. Introduction of vocal-posture and vocal-gesture concepts, explanation of vocal GC principles.
6. Recording and editing of voice training data set and vocal GC training.
7. Illustration of the DMI analysis method and demo of reduced sonic space control with a 2D pad.
8. Description and demo of the interface visual feedback, and main runtime mapping tuning options.

9. Illustration of the set of available DMIs, vocal GCs, and hardware interfaces, and instrument triggering/control modes.
10. Assisted use and familiarization with the system.
11. Free practicing and system exploration.
12. Select the preferred pair of vocal GC and DMI mapping with the related operational modes and functional options and repeat twice the following sequence of tests:
    a. select 4 different vocal-postures aiming for 4 different steady DMI sonic outputs, and maintain each posture for at least 4 seconds in 3 non-consecutive attempts;
    b. select and perform 2 different gestures and repeat each 3 times aiming for identical DMI output sonic variation;
    c. generate as many as possible different DMI sonic outputs varying the voice timbre within 60 seconds of use.
13. Further free exploration and performing.
14. Second interview.
15. Stop audio-video recording, save recordings and interface logs.

We limited the duration of the evaluation session to a maximum of 90 minutes, dedicating roughly 15-20 minutes for each interview, and at least 30 minutes for the VCI4DMI free use and exploration. System explanation, illustration, and demonstration are essential for the participant to understanding the novel features of the system. A critical issue for the appropriate training of the system was the introduction of the concept of vocal-gesture and the GC principles. We pointed out to the participants that the temporal unfolding of the voice timbre variation is not considered in the system, while the focus is on the spatial distribution. We illustrated this abstraction showing graphical representations, reduced to 2D, of other vocal-gestures that ideally fill an arbitrary shape. Audio examples were presented as well. We suggested, but not forced, participants to use vowels as vocal-postures and instances of vowel gilding as vocal-gestures, easier for new users. In the assisted use we asked participants to become familiar with the 2D vocal GC first, looking at the GC visual feedback on screen, and only later focused on the DMI mapping component. During the free exploration, participants were left alone testing and performing with the system in the way they preferred. We intervened only when the participants asked for further clarifications or when we noticed them having major difficulties in using the system.

7.2.4 Interviews aim and guideline

The aims of the two interviews held during the sessions are distinct. In the first the objective is to outline the participants' profile, their relevant experience in musical performances involving DMIs, and their performance practice. In particular we focus on the interfaces components, mapping, and usability constraints. Moreover we question the subjects about personal perspective, ideas, and experience with any voice driven system, not necessarily in the musical context. The second interview represents the core of the qualitative evaluation, and it aims to highlight understanding, experience, and perceptions of advantages and drawbacks related to the VCI4DMI. In particular we encouraged the subjects to criticize the interface they used in the light of their musical expertise, and we discuss the application in performance contexts with which they are familiar as player or audience.

In both in-depth face-to-face interviews we used the same discovery-oriented strategy (Kvale, 1996; Boyce and Neale, 2006), asking open-ended questions that allow the interviewer to deeply investigate participants perspective and experience with the novel interface, and the respondent to freely answer questions using their own words. The semi-structured and conversational format of the interviews allows for unexpected digressions to divert the planned question sequence and follow the participant's interest or knowledge. These are often significant and productive for the aim of the study. However the interviewer has to guarantee that the conversation does not flow towards irrelevant topics. The interviewer has to be an active listener, flexible, responsive, able to reshape questions interpreting the previous responses, and seek clarity of the answers. The audio-video recording allows the interviewer to focus only on the conversation rather than registering the responses of the participants. In the first interview we start briefly explaining the purposes of the interview and asking some introductory icebreaker questions to get the ball rolling. Both interviews conclude with the interviewer summarizing the main points understood from the participant's responses. Summaries of the pre-planned interview questions are listed below.

First interview:

a. Personal information.
b. Roles and experiences in computer/electronic music, music technology, musical instruments.
c. Nature of, practices and preferences, in live musical performances.
d. Instruments and equipment used in live performances.

e. Details and motivation about instrument interfaces, input modality, gesture capturing sensors, and interaction established with the instruments.
f. Motivation and description of eventual musical control delegated to automations or pre recorded/programmed sequencers.
g. Advantages and limitations of performance instrument setup, especially related to interface characteristics.
h. Brainstorming on solutions to allocate extra musical intentions, if any, or to overcome recurrent interface limitations.
i. Use of voice in performances e.g. singing or communications.
j. Previous experiences and reason for using/not-using voice driven system.
k. Major concerns, limitations and advantages with voice driven systems.
l. Imagine and describe an ideal voice-to-DMI interaction system.
m. Additional thoughts on interview topics.

Second interview:

a. Description of the system used, using personal/informal terminology.
b. List advantages, limitations, and drawbacks of the interface.
c. Changes and improvements in the VCI4DMI system.
d. Feelings on vocal interaction and control toward DMI sound characteristics and control parameters.
e. Discussion on practicing and skills required for mastering the use of the VCI4DMI.
f. Usefulness, benefits and shortcomings of the different visual feedback.
g. Impact of system operational modes and functional options on interface response according to personal preferences.
h. VCI4DMI as alternative or extension to traditional interfaces.
i. Given the suggested improvements, illustrations of possible applications and benefits in familiar live performances contexts.
j. Interest in and detail of further use and training of the VCI4DMI with personal instrument set.
k. Additional thoughts and feedback relevant or beneficial to the project.

7.2.5 Qualitative and quantitative evaluation methods

The qualitative and quantitative evaluation of this study is based on the data recorded during the whole of the user evaluation sessions, including the audio-video recorded

interviews, the log of the VCI4DMI that tracks working modality and the streams of input voice low-level features, GC output, and DMI output parameters stream. Additionally we record the audio signals from the microphone connected to and DMIs controlled with the VCI4DMI system.

We transcribed and annotated the data from audio-video recordings of the interviews prior to the VCI4DMI usage. These were analyzed and filtered to determine the profile of each subject, which includes details on expertise relevant to the aim of this study. Moreover from each transcription we extract the participants' opinions on three main arguments: limitations of performer-instruments interaction in musical performances, voice HCI, and possible application of voice for musical control. The detailed profiles allow us to compare and correlate the study results with eventual specific profile characteristics.

The analysis of the participants' interviews regarding their experience with the VCI4DMI represents the core of this qualitative evaluation. We started with a time-stamped transcription of the audio-video recordings. Then the transcription and video was further annotated, normalized into non-colloquial text, labeled by relevant topics, organized at higher hierarchical levels in order to synthesize the participants' answers to the key questions (Kvale and Brinkmann, 2008; Bryman, 2012), while maintaining time alignment between analysis and raw data. For the analysis we worked in two directions: we look for answers to specific arguments on interface use and experiences that define the participant conceptualization of the VCI4DMI, but the open-ended questions stimulated conversations in which participant freely expressed their thoughts across different topics. For this component of the interviews, having no prior expectation about the contents, we operated in a reverse fashion coding and grouping the annotations into more abstract categories from which we try to draw conclusions, similar to grounded theory methodology (Martin and Turner, 1986). In order to validate, and then generalize, the respondents information we used the triangulation method (Rothbauer, 2008), specific opinions and evaluation of the system had to be verified by multiple participants, preferably with substantially different profiles. For the time-stamped transcription and annotation of the videos, as well as for the definition of the analysis system we used the Multimodal Analysis Video[26] software (O'Halloran et al., 2012; O'Halloran, Tan, and E, 2014) that provides a single framework from the analysis through to the visualization of the results. In this context the analysis of open-ended interviews provides an accurate representation of the participants judgments, especially compared to a survey based

[26] http://multimodal-analysis.com/products/multimodal-analysis-video/

on Likert scales (Likert, 1932), which are common in HCI user evaluations. First the selection of the measurement scale determines the statistical method used for analyzing the data (Zumbo and Zimmerman, 1993), but when the sample size is small, statistics may have no relevance. Secondly data from evaluation scales should be understood within the context of how such ratings are made. When the respondents sample is not uniform, each participant may use a different "frame of reference" and "standard of comparison" when interpreting questions and making judgments. Striking contradictions between data collected from Likert scales and free text answers has been observed (Ogden and Lo, 2012). Therefore in our evaluation we chose open-ended interviews due to the small sample size and due to the rating scale bias determined by different participants' age, background culture, performance practices, played instrument, and exposure to this research field. Moreover it would have been difficult for participants to define a frame of reference for the VCI4DMI because vocal interfaces are not common in the musical context.

Furthermore the free practice and performing video recording, with the optional aid of the interface log, were inspected and annotated to identify eventual patterns in practices, behaviors and other exceptional and unexpected issues. Finally for the quantitative evaluation we use the interface log to compute a set of numeric measurements and ratings related to the three use-cases described in the protocol sections 12.a, 12.b, and 12.c. In particular these measure interface reliability, usability, and the repeatability, versus the interface design principles.

7.3 Results

The qualitative and quantitative results of the user evaluation are separated into four sub-topics, each presented in the following subsections.

7.3.1 Participants profile

In Table 7.2 we present the profiles of the 10 subjects that participated in this study. The fourth column summarizes their relevant musical expertise besides the live performances, where the tag builder indicated prior experiences in designing hardware or software for musical instrument, interfaces, or interactive musical systems. For performances description the tag "electronic music" includes a significant use of samples and music loops, generally related to dance music genres, while the use of pre-recorded material is very limited in "live electronics"

performances. In the last two columns we indicate if the participants are aware about DMIs and interfaces besides state-of-the-art consumer products, such as those built by the research and do-it-yourself communities, and whether they are aware of research in computer music. Several participants have multiple positions and roles in relation to music.

participant ID	*age range*	*country of origin*	*role and relation with electronic music*	*years experience*	*nature and role in live performances*	*performers configuration*	*instrument setup*	*use of voice in performances*	*DMIs, interactive systems, interfaces awarness*	*computer music research awarness*
1	21-29	India	percussion player / builder	6	percussionist	ensemble	acoustic percussions + effects	minor vocals	consumer / research prototypes	aware
2	21-29	Singapore	electric bass player / electronic music composer	4	bassist / electronic music	band duo	electric bass + effects / DAW + controllers	no	consumer	unaware
3	21-29	China	electric guitar player	12	guitarist	band	electric guitar + effect chain + sampler-looper	minor singing	consumer / DIY hacking	unaware
4	30-39	Singapore	electronic music composer / electric guitar player	5	electronic music	solo trio	DAW + controllers	minor vocals	commercial	unaware
5	21-29	Singapore	pop composer / piano-guitar-sax player	10	keyboardist	band	synthesizers + effects	minor use with vocoder	consumer	unaware
6	50-65	Scotland	avantgrade composer / builder/ keyboard player	40	keyboardist	solo band	synthesizers	no	consumer / research & DIY prototypes	contributor
7	30-39	Malaysia	electronic music composer	4	electronic music	solo duo	DAW + controllers	no	consumer	unaware
8	40-49	Switzerland	audio-visual composer / DJ / builder	8	live audio-visuals / DJ	duo	DAW + DIY software + controllers + sequencers	no	consumer / research & DIY prototypes	contributor
9	40-49	Canada	electronic music composer / builder / electric guitar player	25	guitarist / live electronics	band	electric guitar + effects / synthesizers	no	consumer / research & DIY prototypes	contributor
10	30-39	Spain	electronic music composer / DJ / builder	18	DJ / live electronics	solo	DAW + controllers + drum machine + sequencers	no	consumer / research & DIY prototypes	aware

Table 7.2: User evaluation participants' profiles.

The profiles show differentiation in expertise, performance practice, and instrument setup, though all subjects have significant experience in one form or another. Although a laptop running a DAW is a common choice in electronic music setup, these vary in implementation and functional application across subjects. All subjects reported that when performing, the voice was an unused communication channel, although some are eventually engaged in backup singing or vocoder use for limited time intervals.

In Table 7.3 we present a summary of the main topics discussed in the interviews before the VCI4DMI usage, derived labeling and normalizing the interview transcriptions. The table repeats the instrument setup field to facilitate associations and comparisons. Although there is a clear diversity in the answers, we can identify the recurrent trends listed below.

- Instrument interfaces often include only general-purpose push buttons, faders, dials, encoders, and their equivalents implemented on touchscreens. This implies that participants implement their own mappings and that in performances the hands-to-interface bandwidth is fully saturated.
- Pre-programmed dynamic automations of parameters or musical event were widely used in performances by all participants, with the exception of those including basic and non-programmable DMIs in their setup. The main reasons for using automations are their reliability, high precision, the difficulty or impossibility of manual execution, or the number of parts or variable parameters being too high for a single performer control. These issues have also been addressed with the use of pre-recorded loops and clips.
- Participants discussed interface limitations experienced in their previous work as ranging from a lack of sonic expressivity, the need to simultaneously play and tweak parameters, and poor efficiency in specific tasks related to non-quantized controls such as those mapping continuous gestures to continuous parameters. To overcome these limitations most respondents proposed smart improvements over their existing interfaces, while only a minority suggested to use other spare bandwidth channels or to increase the dimensions of the existing gestural input.
- With only one exception no one had previously made regular use of a voice-driven system, although everyone had tried at least a few times, mostly because they had accidentally stumbled across such an application on their mobile phones. Participant experiences were limited only to speech/words recognition systems. We speculate that this contributes lack of expressing

ideas about considering the voice bandwidth for easing the interfaces limitations they had experienced. The main concerns they expressed, experienced or supposed, with voice-driven systems were the poor accuracy and the difficulties posed by noisy environments.

- When imagining voice to instrument interaction, as well as when proposing solutions to limitations, we notice a correlation with individual expertise, experiences, and musical instrument skills. Approximately half of the participants proposed to use speech recognition techniques as a surrogate for driving instrument with voice commands (e.g. set parameters X to 0.6) perhaps because they were not aware of non-verbal voice HCI possibilities, and some raised appropriate concerns on the resulting latency they supposed would plague the musical use of such systems. The remaining instead proposed to track voice features to control or modulate continuous instrument parameters, which is close to the solution we propose, but had doubts in particular about the potential for loudspeaker to microphone feedback.

Two interesting observations were recorded from Participant 6 and Participant 9. The first claimed that the poor expressivity with some DMI interfaces is due to the total lack of formal and sustained training that people generally get with electronic interfaces compared to acoustic instruments, whose require years of instruction and practice to perform satisfactorily. Moreover with novel interfaces we are pioneering in exploring and defining new methods for playing and skill refinement, which simply haven't been established yet, as observed by Ryan (1992) two decades ago. The second argued that interface limitations are not only the drawback. Interfaces are important and must be transparent to audiences because this allows them to see how far a performer can go within the physical constraints and limitations of a specific musical instrument or interface, and they can then understand the music better and evaluate the virtuosity of the performer.

participant ID	*instrument setup*	*instrument interfaces*	*pre-programmed delegated control*	*limitations*	*possible solutions for liitations*	*voice HCI experience*	*voice HCI systems issues*	*possible voice instrument interaction*
1	acoustic percussions + effects	set of djembe + effect unit buttons and dials	no - fixed parameters	no effects variation, change timbre require switch djembe	capture body gesture to change effect preset or and timbre processing	mobile search and call contact / minor use / tried for fun	reliability in noisy conditions	voice commands to mixer, filters, effects
2	electric bass + effects / DAW + controllers	effects foot switch and expression pedal / push buttons	fixed parameters / programmed backing track - reliable and beyond duo live control	preset tuning only 1 param / DAW cant control all the instruments	touchscreen interface to map more parameters	address dictation to GPS navigator / often / found it by accident / safety while driving	poor accuracy, reliability in noisy conditions	voice pitch and loudness as a xy pad and manual map
3	electric guitar + effect chain + sampler-looper	foot switches, dials & touch lcd effect units + foot switches sampler	no - fixed parameters	stop playing to adjust efx params, looper+efx no simultaneous control	sit down allow 2 foot controls, eye motion mapped on sampler loop functions	mobile intelligent personal assistant / minor / found it there	poor accuracy	generic discrete voice commands
4	DAW + controllers	push buttons & touch screen	volumes, effects, backing track - reliable and concentrate on adding top layer	touchscreen latency, poor sensitivity, poor expressivity, basic interaction	holographic screen for 3D interaction with sounds mapped on shapes and colors	message dictation / minor / found it there / sometime speed up the task	privacy	voice pitchclassifier to enable different effects or trigger samples
5	synthesizers + effects	keyboard with modwheel & dials + touch screen for soundscape modulation	no - fixed parameters	change synth patch interrupt sound, or tweaking params take too long	instantaneous and tempo synced synthesis patch switching	no	poor accuracy	generic discrete voice commands - possible latency issues
6	synthesizers	keyboard with faders, hybrid continuous pitch keyboard	random process generate notes, control variance - play at higher level, novel patterns	continuous pitch control, poor expressivity, musical patterns bounded to player expertise	code and drive at high level stocastic process to generate musical controls	mobile intelligent personal assistant / minor / found it there / useful if recognize	poor accuracy	convolution voice and DMI sound / trigger sounds by voice pattern similarity / voice pitch voice to note
7	DAW + controllers	push buttons, faders, dials	half of DMIs fully presequenced - preprogram controls difficult to replicate live	limited access to params and smaples without dynamic mapping	use higher number of physical interfaces	laptop commands and dictaction / minor / looked for it / good for drawing & typing	poor accuracy	vocal command to DAW - map words to specific function, detune + 5, change waveform
8	DAW + DIY software + controllers + sequencers	pressure sensitive push buttons, touchscreen, faders, encoders	pre-programmed parameters envelope on samples - easier to perform	touchscreen no physical feedback, sequencer no complex temporal patterns	semi autonomous in sequencing - temporal pattern generators	mobile voice search / minor / tried for fun	poor accuracy	voice as source of real-valued parameters modulation / sample triggering
9	electric guitar + effects / synthesizers	effect foot switch / keyboard, breath controller, touch pads	no - fixed parameters	button and keys quantized control, poor expressive mapping - play with B music written for A	electric guitar very expressive, use as controller analyzing it sound	no	privacy and use in silent environment	voice for continuous control / note generation by singing that preserves and map expressivenes and vibrato
10	DAW + controllers + drum machine + sequencers	pressure sensitive pads, faders, dials, rotary encoders, xy pads	yes - execution of complex parts - reliable and impossible manually	large music & samples database on the fly search - laptop screen engagement	database analysis visualization - more feedback on interface	mobile voice search and message dictation / minor / found it there / useful while driving	reliability in noisy conditions / need for check and correction	database search by voice temporal patterns / voice driven sound morphing

Table 7.3: User evaluation first interview key topic responses summary.

7.3.2 VCI4DMI users qualitative evaluation

A summary of participants' answers to the key questions about the VCI4DMI experience is presented in Table 7.4, where the answers are grouped and normalized from colloquial non-technical terminology to a concise and formal language. Figures 7.3-4 show respectively footage of user evaluation sessions and a screenshot of the software used for interview transcription, annotation and analysis. In Figures 7.5-8 there are four examples of DMI vocal control, showing a correlation between vocal-gesture and timbre variation. These include spectrograms of voice and resulting instrument sound for three sound generators with sustained timbre, and one sound generator with decaying timbre. For the last example the voice note generator was enabled in the VCI4DMI.

The main findings from the user qualitative evaluation are listed below.

- Every participant described the system as a voice to multiple DMI parameters controller, showing a clear understanding of the interface main purpose, and some mentioned also the concept of adaptive mapping and training. Positive and beneficial aspects of the VCI4DMI that participants mentioned in different phases of the interview includes the high expressivity, intuitive and natural use, extra control dimension, wide and dynamic sonic range of the output, smooth response, and low application constraints. Respondents proposed the use of the VCI4DMI as an extension to traditional interfaces that provide an additional control layer in a wide range of performance contexts, but some suggested as well the use as the sole interface because of a sense that it would be "cool" and "very unusual" for spectators. Most participants recognized at least one possible application in their performance context to overcome their instrument setup limitations, discussed in the first interview. These opinions were essentially uniform across individual evaluations and in agreement with the motivation, aim, and principle of this work.
- The discussion on drawbacks and limitations of the VCI4DMI presented very diversified answers supporting the need for this study, which identifies further usability improvement. Some are contrasting with positive interface features described by other respondents. These were usually just due to inappropriate system tuning, and can be addressed easily by users familiar with the interface response to system setting variation. Most feedback about the shortcomings of the VCI4DMI included suggestions to simplify the GUI, exposing only few tuning parameters in a basic user mode, while showing the

complete set in a separate advanced view. Moreover participants suggested to further integrate the implementation and put everything into a plug-and play hardware or software box, including ready to use mappings. This does not necessarily imply reduction of the system's adaptive and learning potential. Some cross-speaker maps can be computed and provided with the system for simple control tasks, leaving the choice to run personalized system training to the user. Indeed most participants also explained that a simple two-parameter vocal control it would be a great improvement with commercial value, although would not make a great contribution to the scientific field. Finally three participants argued that adding a mapping mode that directly associates voice and DMI timbre by their similarities would further simplify usability and learnability.

- Feedback in the VCI4DMI was addressed with three different interactive graphical representations. All participants found these essential for proper learning of the specific mapping response. Most argued that when familiar with the mappings, the visual feedback is not necessary or that is distracting for performances, unless presented in a minimalistic form. Participants found the first use easy and intuitive, but recognized some issues in achieving the desired vocal control. Nevertheless users experienced improvements in just 30 minutes of free practicing, and they agree that practicing is essential to master vocal control skills as well as familiarizing oneself with the different use modalities and training procedure provided in the VCI4DMI system. However some expressed concerns about the different control skills that might be required to master for each new pair of voice GC or DMI mapping.
- Approximately two thirds of the subjects expressed the feeling of vocally interacting with a sonic object, so that the mental abstraction in the control process was voice-to-sound. Some argued that in this process they tried to establish a relationship with the GC output space first, totally ignoring DMI parameters and sound. They focused also on parameters for simple DMI mappings such as for the simple control of the frequency and resonance of the low pass filter. The remaining participants used the VCI4DMI still with a mental abstraction of voice to parameters, which may prove challenging due to the potential non-bijective parameters-to-sound relationship. This group response was due to their unfamiliarity with the available DMIs, so that they were exploring the parameters-to-sound response first. They argued that training and using the interface with their DMI setup might allow focusing on sonic response only. Participants assume that the VCI4DMI use with their

own personal DMIs might have been easier for them, and more accurately performed, although more difficult to compare for the aim of this study.

From the interview analysis we also observed that participants consider the simultaneous multi-parametric control more important than the absolute control precision for instrument expressivity in performances, which is more challenging with the VCI4DMI and may require a learning curve similar to those of some acoustic instruments. This can also provide better endurance as many objected that the interface is vocally tiring. We noticed that the machine learning, which is central to the VCI4DMI, was rarely mentioned in the interviews. We speculate that this is due to being relatively hidden in the protocol. In fact participants were assisted in the training of the vocal GC, and DMI adaptive mapping was computed beforehand and only briefly demonstrated. However we believe that the VCI4DMI approach for unsupervised and automatic mapping generation does not make apparent the ML training component especially to novice users.

Comparing the interview analyses and considering the fraction of time spent discussing specific topics, we noticed that the more experienced participants, aware of the research in this field, were more critical and spent a considerable amount of time proposing improvements or alternative interface principles. Younger and less experienced participants mostly addressed implementation issues and also provided a wider spectrum of possible application scenarios. With few exceptions the individual musical expertise did not constrain the vision of the possibilities for the system and possible benefits beyond the individuals' musical scope. In evaluating the key features, usability, and personal perspectives on the system, participants showed an overall agreement and uniformity in their responses, especially on beneficial features, control abstraction, learning curve and implementation issues.

participant ID	desctiption	pro	application and benefit	cons	technical improvements	visual feedback	sonic or parameters interaction
1	novel voice to instrument interaction system trained with individual voice	new way of expression, extra control over physical limitations, vocal sonic control	control beyond hands limit, tune processing params to change drum timbre in a continuous space	poor and unclear control of fast decaying sounds	simplify interface GUI	essential for learning, unless apparent random response, perform without when familiar	sonic interaction, parameters initially or if having control issues
2	voice vowel midi controller	simultaneous control of many parameters, resulting wide sonic variation, natural control	additional control bass distorsion parameter while playing	difficult reach and maintain certain sonic points, poor smoothness	less tuning settings label options and settings by resulting effect on interface response	useful for learning, especially GC out space, perform without when familiar	sonic for generator, parameters for processors
3	control system to shape DMI sound through voice, complete voice digital control system	additional control dimension, extended sonic range, enables creativity, many possible uses	add and modulate sound layer to guitar, good for beatboxers and for complex dj setup	system too complex for musicians with low exposure to digital systems	simplify interface GUI and provide preset for different performance context	essential for learning and use	sonic interaction
4	voice controller with endless interaction possibilities	highly dymanic and expressive, natural control, freedom in musical expression	extension or alternative, not limited to switches amd keys, use in any form of digital art	complex tuning settings, vocally tiring	simplify interface GUI and put in a box	necessary for learning and performing	both
5	trained voice controlled interface	easy to use, low entry barrier	control extension for multiple parameter control of sound effects and soundscapes	fighting against DMI loudness to hear own voice, vocally tiring	no	useful for learning, distracting when performing	parameters interaction
6	voice mapping to DMI sound of synth, dimensions in voice control dimension in DMI sound	intuitive extra control dimension, great marketing value	extra control for any performance scenario, interesting for audience if mapping is clear	voice sound change uncorrelated to the way DMI changes sound, that would be easier to learn	provide basic default mappings no training, improve posture stability, more instant gratification	essential for learning, irrelevant for just having fun, perform without when familiar	sonic interaction, both if number parameters is 2
7	voice control over parameters or sound for any DMI	enables many novel control and interaction possibilities	additional control layer or alternative to other controllers, unique/unexpected for audience	limited repeatability, some mapping difficult to use, audience may hear funny vocal sounds for control	integrate voice commands	useful for learning and performance, expecially the GC lattice interactive representation	parameters at the beginning, sonic later, especially if trained with familiar DMI
8	system learn from voice how to move into a sonic space of an analyzed DMI	smooth, very responsive, possible to tune, wide spectrum of instrument interaction	control extension, application in wide range of musical performance scenario, alternative to vocoder	challenging for non vocally trained users	less tuning settings, multiple params in multiple DMI simultaneously, faster training	important for learning, minimalistic for performing, use realtime video out	interaction to visual represenation of mapping spaces and sound
9	voice to control of params that learn characteristic of voice and DMI to generate optimal mapping	wide sonic range, touchless, unusual and cool, expand voice sonic range, commercial potential	extention to tune params instrument played by hand, or concurrently play another instrument	need to learn 2 maps, each DMI require new learning, hard to learn if DMI sound very different from voice	provide out of the box working map, then slowly train and refine response, improve response smoothness	useful for learning phase but require simplification, perform without when familiar	interaction to visual GC space during learning, then sonic interaction
10	voice to instrument mapping framework	intuitive and easy to use, training with any vocal sound and DMI, expansive, no application constraints	extra control for filters and effects for dj, trigger and modulate sound effects, sync params to singing	challenging to master vocal control skills, lessprecise than other interfaces	simplify interface GUI, software integration, standalone version	essential for learning, on occasionally in performance with multiple DMI maps	sonic interaction, especially with familiar DMI

Table 7.4: User evaluation second interview key topic responses summary.

Figure 7.3: Footage of participant engaged in VCI4DMI exploration and interviews within the evaluation sessions.

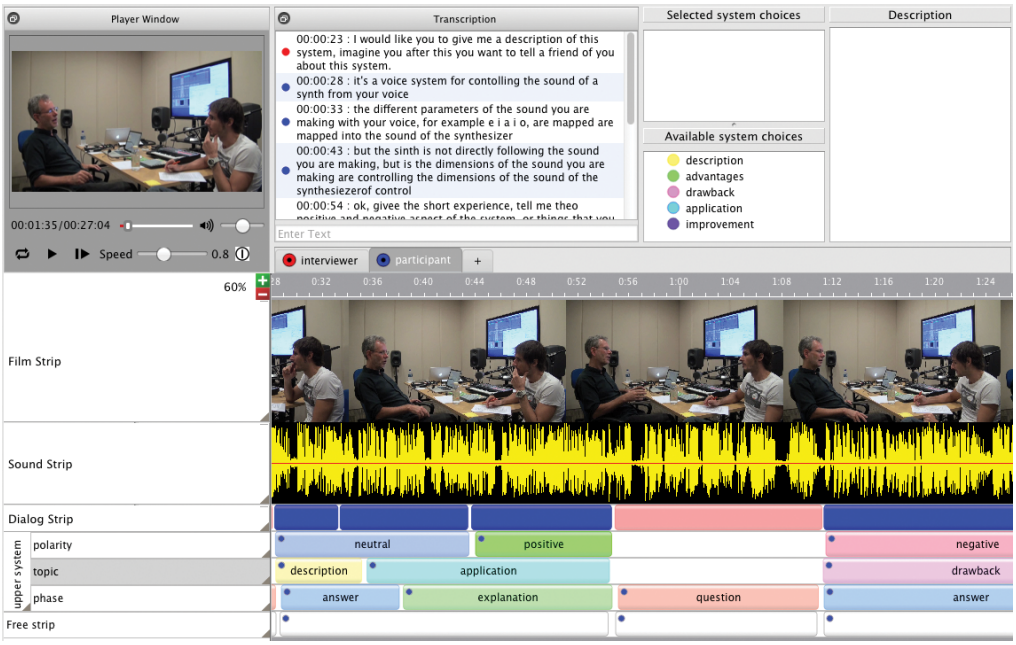

Figure 7.4: Screenshot of the software used for transcription, annotation, and analysis of the recorded interviews. Details of the transcription and selected annotation system on top, transcription and labeled annotation aligned to the video timeline on the bottom.

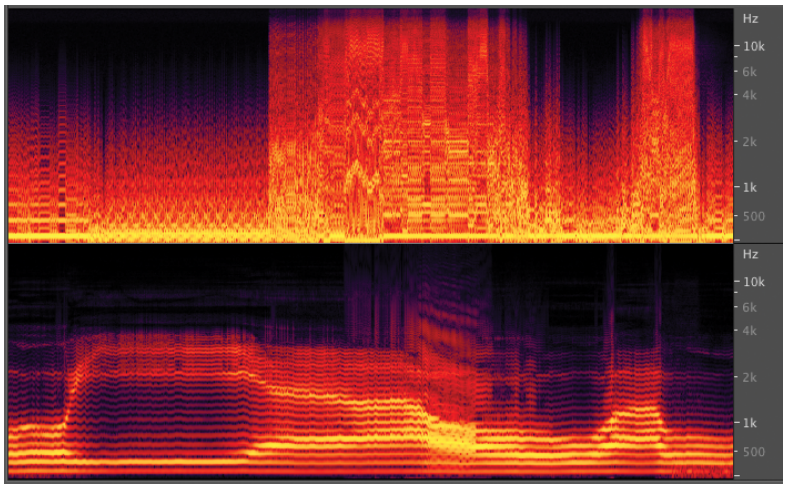

Figure 7.5: Example of DMI ID-3 vocal control, with the instrument sound spectrogram on top and the driving vocal-gesture spectrogram on bottom.

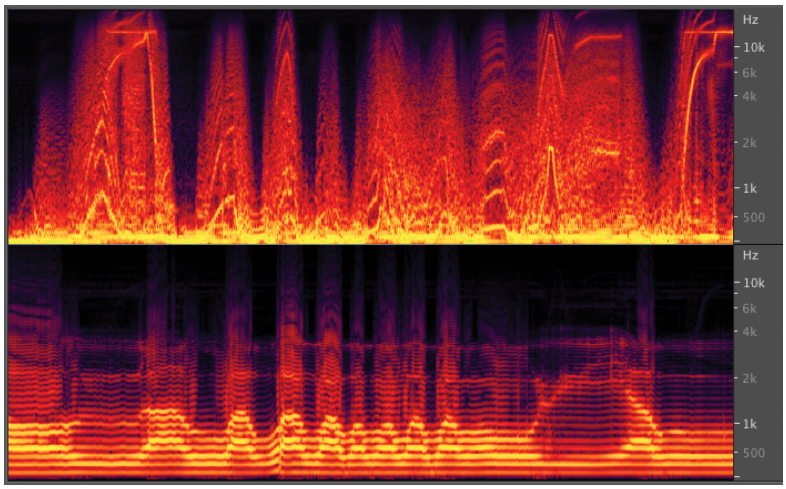

Figure 7.6: Example of DMI ID-4 vocal control, with the instrument sound spectrogram on top and the driving vocal-gesture spectrogram on bottom.

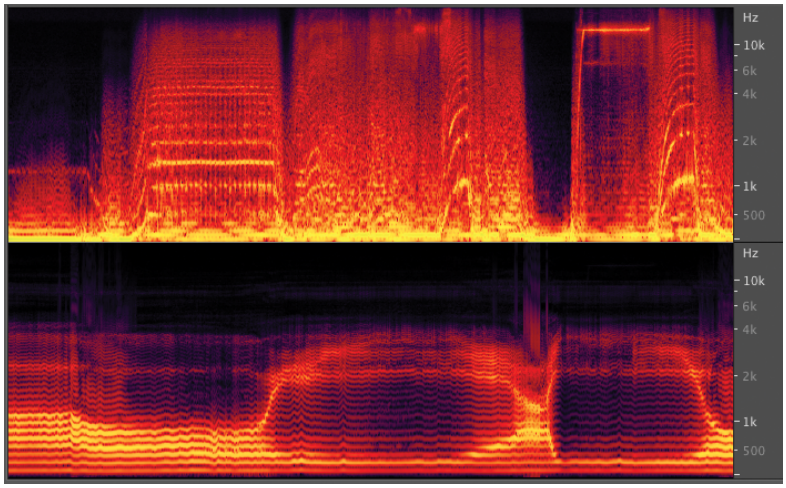

Figure 7.7: Example of DMI ID-5 vocal control, with the instrument sound spectrogram on top and the driving vocal-gesture spectrogram on bottom.

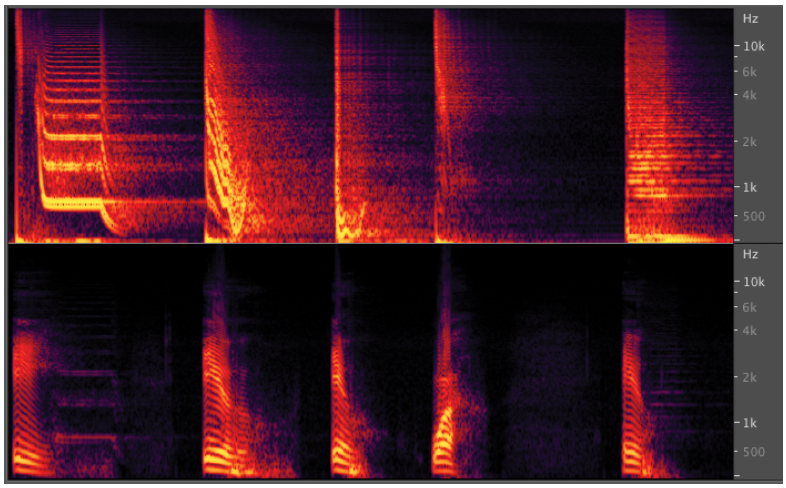

Figure 7.8: Example of DMI ID-5 vocal control plus note generator, with the instrument sound spectrogram on top and the driving vocal-gesture spectrogram on bottom.

7.3.3 Users free practicing and performing observations

Examining the recordings of the participants engaged in freely exploring and performing with the VCI4DMI we identified patterns or exceptions that were unexpected, in contrast with or undisclosed in the interview responses. The participants' experience, expertise, musical skills, understanding of VCI4DMI mapping principles, and familiarity with the DAW had little effect the approach to the interface. They started randomly exploring the response to then pass to a more systematic phase in which they tried to control the GC first and then the DMI sound. Most struggled in understanding the functionality of all available options, and limited the tuning to a few settings that turned out to be the most effective in adjusting the response. The radial distance limiting the sonic space search, the voice feature low pass filter cutoff, and the SOG lattice scale factor were recurrent choices for adjustments.

The DMI mappings presenting a higher number of entries in the sonic space were generally more difficult to use, despite the good ANN fitting identified by low mean squared error. We recognize that with a lower number of entries, between 1k and 3k, we set the analysis option for spending more time per state for the perceptual timbre analysis, resulting in a more accurate sonic mapping. Furthermore, the IDW interpolation provides a smoother control in browsing the sonic space with fewer entries, and this in turn minimizes any residual detrimental effect of the non-bijective sound-to-parameters relationship. The number DMI of variable parameters was totally uncorrelated with the ease or difficulty of use of a specific DMI mapping,

suggesting the parameter space is totally transparent to the user with the proposed control strategy implemented in the sonic space.

We noticed that participants tuned the voice pitch to the instrument sound when engaged in controlling sustained sound generators. In some cases, this resulted in a large gap in voice pitch between training and control phases, and in a few cases this limited access to certain sub-regions of the GC output space. This issue can be addressed within the interface, but the user should also guarantee consistency with the training vocal-gestures while using the system. Hearing one's own voice was mentioned as a disturbing feedback for some users and essential for others, so that they adjusted the DMI volume accordingly, or requested for headphones.

Although most subjects suggested the use of the VCI4DMI as extensions to traditional controllers to enable the concurrent use of more interfaces, only three participants effectively tested such a configuration during the free exploration. Most explained that they knew they could have played a keyboard at the same time, but preferred to focus on understanding and tuning the VCI4DMI system. The voice note generator was poorly exploited as well. Few participants explored the DMI sonic spaces directly controlling the parameters explicitly mapped to an external hardware controller, to compare and understand the possible sonic range with a traditional control strategy. Finally we noticed that the two subjects who reported most issues in reliable control were those that provided the poorest vocal training data set, in which the vocal-gestures included little timbre variation and long vocal-posture-like intervals. This suggests the implementation of an intuitive rating feedback on training data and vocal GC training outcome.

7.3.4 Use-cases quantitative evaluation

Step 12 of the protocol describes three use-cases that each participant performed after selecting the preferred pair of vocal GC and DMI mapping with the related operational modes. The results are presented in results Table 7.5, in which the first four columns describe each participant's VCI4DMI configuration. In particular it includes the selected GC dimensionality, the reduction mode, the selected instrument from those available in Table 7.1, the DMI mapping and search modes. For the use-case described in 12.a, we evaluate the stability over vocal-postures, measuring and comparing the average standard deviation for the voice low-level features, GC output and DMI generated parameters. Moreover for the different instances of identical vocal-postures we evaluate the resulting average distance in the voice feature, sonic, and parameter spaces. This evaluates the use repeatability that the interface offers,

also addressed in relation to vocal-gestures in use-case 12.b, in which we measured the average difference in the same three spaces. However prior to comparison, the vocal-gesture derived data has been equalized in length aligning the multidimensional streams using a technique based on dynamic time warping (Turetsky and Ellis, 2003). Finally coverage of the GC, sonic, and parameter spaces are measured on the data collected from use-case 12.c. In Table 7.5 the results are averaged per participant over the multiple instances and test repetitions, as defined in the use-case protocol. Participants had no feedback on accuracy, precision or completion of each task.

					use-case (a)						*use-case (b)*			*use-case (c)*	
participant ID	*GC dimensionality & reduction technique*	*selected DMI ID*	*DMI mapping mode*	*DMI Search mode*	*voice feat. avg deviation*	*GC out avg deviation*	*params. avg deviation*	*voice feat. avg distance*	*sonic avg distance*	*params. avg distance*	*voice feat. avg difference*	*sonic avg difference*	*params. avg difference*	*% GC and sonic space coverage*	*% params space coverage*
1	2-Isomap	3	du*=gc	Neigh.	0.06	0.03	0.07	0.26	0.21	0.24	0.56	0.48	0.55	69.9	15.7
2	2-M.LDA	4	d*=m(gc)	irad 0.1	0.06	0.05	0.08	0.20	0.17	0.19	0.63	0.52	0.92	71.1	15.0
3	2-Isomap	8	d*=m(gc)	irad .2	0.14	0.16	0.12	0.54	0.39	0.29	1.73	1.02	1.17	84.8	13.4
4	2-Isomap	7	d*=m(gc)	Neigh	0.07	0.08	0.11	0.34	0.24	0.35	0.72	0.71	2.50	75.0	28.3
5	2-M.LDA	1	du*=gc	Neigh.	0.04	0.05	0.09	0.20	0.15	0.30	0.41	0.32	1.17	73.5	22.4
6	2-M.LDA	4	d*=m(gc)	irad 0.5	0.11	0.06	0.13	0.12	0.13	0.11	0.47	0.38	1.05	76.4	25.3
7	2-Isomap	4	d*=m(gc)	irad 0.7	0.06	0.03	0.05	0.10	0.10	0.08	0.25	0.17	0.44	91.1	51.7
8	2-Isomap	4	d*=m(gc)	Neigh	0.19	0.06	0.13	0.19	0.22	0.31	0.27	0.22	2.68	74.8	56.7
9	2-Isomap	6	d*=m(gc)	Neigh	0.03	0.08	0.08	0.29	0.18	0.20	0.56	0.43	0.97	84.4	69.4
10	2-Isomap	2	d*=m(gc)	irad 0.4	0.04	0.02	0.03	0.06	0.06	0.07	0.21	0.19	0.17	94.3	53.3

Table 7.5: Participants' preferred VCI4DMI configuration and measurements related to three different use-cases.

Analyzing the participants' preferred configuration, shown in Table 7.5, we observe that the 2D gestural controllers were always selected. The 3D mappings were more difficult to use and we suppose that the limited expressivity improvement they provide in many cases, as illustrated in the Chapter 5 evaluation results, were not worth the higher cognitive complexity given by the extra dimension, at least for first time users. The multiclass LDA variant was preferred by a small group of subjects that had difficulties in finding vocal-postures close to the GC vertices with the Isomap mode. The most widely preferred DMI mapping mode method was the ANN-derived function, while choices on the search settings are very different. The search mode, within a user-defined parameter radius or within the parameter's Moore neighborhood, was identified by most participants as the most effective setting in changing the interface-to-DMI behavior, often addressed as "sensitivity", and

therefore we observe different choices that actually determine diverse vocal interaction. The span of DMI selection is wide as well but it also involves sonic aesthetic factors. However the DMI 4, which is the generator with the lowest number of entries in the sonic space is the most recurrent choice that users associated with improved usability and response. The results in Table 7.5 were obtained with the IDW in the vocal GC and in the DMI mapping enabled, as well as the input voice energy gate. The GC lattice search mode was always set to the Moore neighborhood.

The results in the first three columns of use-case (a) shows an overall small standard deviation for the voice features, GC output and DMI parameters determined by vocal-postures. The proposed method with noisy feature filtering already results in a small deviation in the feature space. This is further reduced in the GC output space, while it can slightly increase in the parameter space depending on case-specific settings. However in half the cases it is satisfactorily below 0.1. The diversity in these results is also due to the vocal control skills that participants developed to different levels within the free practicing time. These are more evident in the three following measurements that describe the average distance determined by instances of identical postures. The large distance in the parameters is in some cases due to poor vocal control skills, which is reflected in the considerable distances in the voice features space. In others cases, the non-bijective parameters-to-sound relationship may have contributed. For this reason we also display the average distances in the sonic space, which is generally lower and suggesting that the repeatability in the control of steady timbre is more precise than the parameters. This is verified also in the use-case (b) measurements, in which the repeatability of the vocal-gesture driven timbre dynamic variation is more critical, although this is not directly comparable because in this case we compute the average difference instead of the distance. For this measurement, the user-defined interface settings are responsible for the eventual difference between the averaged results in the sonic and parameter space. If we exclude participants 2, 3, and 4, the subjects managed to obtain similar paths at least in the sonic space. These are far from being identical, but demonstrate the repeatability of vocal GC trends. Finally, the coverage of the sonic space, within 60 seconds of use, demonstrates that most users were able to cover approximately 3/4 of the GC output space, while for the resulting parameter space the low percentages are often due to tight conditions set by the participants on the DMI search modes, and also due to the high number of total combinations. However the space coverage measured across the whole free practicing sessions are higher, and close to 100% especially for the GC output. These are not included in the results because they are not derived from a systematic test and they may include unintentional input data.

The quantitative results of this evaluation, although dependent on interface configuration and participant vocal skills, demonstrate that the interface is in broad compliance with the design principles defined in this dissertation. While the interface use is challenging, especially compared to hand-based controllers, it provides sufficient reliable and repeatable sonic control. The data obtained in the use-cases measurements supports the user comments that practicing to master the skills is essential and it can improve the control precision in all the tasks considered above.

7.4 Summary

In this chapter we presented user studies for the evaluation of the interface developed within the framework of this dissertation, which integrates the main contribution of this thesis. We presented the major issues in evaluation and validation of musical instrument and interfaces, and we described our specific strategy and systematic protocol. We recruited ten experienced musicians for using and evaluating our novel interface who discussed their experience with the VCI4DMI in open-ended interviews. We derived qualitative evaluation from the analysis of the audio-video recorded interviews, while additional quantitative results were based on specific use-cases measurements. We drew conclusions from the study of the resulting data identifying trends, patterns, and exceptions, which generally verified and confirmed the interface motivation, aim and principles. However participants also identified implementation improvements or suggested system modification, to address their perceptions of existing drawbacks. Finally we also detailed and discussed unexpected exceptions and conflicts in the findings, useful for future interface design.

Chapter 8

Conclusion

This thesis has presented a generative and adaptive mapping method to implement ad hoc vocally driven control for DMIs, which integrates the study of performer vocal characteristics and relationship between parameters and sound of a specific instrument. The expressivity of the interface is granted by customized machine learning algorithms that maximize the breadth of the explorable sonic space over a set of instrumental parameters and a given set of user vocal-gestures. The interface is generic and unsupervised since it does not set limitations on voice and DMI characteristics from which the system derives the mapping. Moreover the user interaction for training data preparation and system setup has been minimized. The results of this research work are integrated into an open-source functional prototype used to demonstrate the usability and advantages of the VCI4DMI for live performances. To conclude, we summarize the original contributions presented in this dissertation and we discuss their impact in a broader context. Then we discuss possible improvements of the system and future research directions in vocal control of musical instruments, advanced generative mapping techniques, and tools supporting users-centered interface implementation. The closing remarks include the author perspective and reflections on the system.

8.1 Summary of contributions

The key contributions made by this research include:

- An unsupervised learning technique that extracts robust, independent, and continuous control signals, representative of the control intention expressed by voice timbre variation. The method is compatible with any phonation mode and optimizes the computation towards noise minimization and accentuation of the spatial spread of the gesture.
- A novel method to implement ad-hoc multidimensional gestural controllers based on the output lattice of a self-organizing map, including a modified training procedure which ensures topology coherence between input and output spaces, and discontinuity-free output for continuous input trajectories.

Associating gestural extrema with vertices of the lattice we prevented application specific SOM pathologies.

- A generic framework to analytically model the relationship between control parameters and perceptual sonic response of the instrument, based on a taxonomy that organizes DMIs for their input-output relationship, and specific analysis technique for each class.
- A DMI few-to-many control strategy constructing an inverse map from the reduced-dimensionality perceptual sonic space to the parameter space, minimizing the losses in retrievable unique combinations, and addressing the non-bijective parameter-to-sound relationship with a trade off between parameters continuity and sonic space explorability.
- A novel voice-to-instrument dual-layer generative mapping strategy, which converts instantaneous vocal timbre variations into instrumental sonic space trajectories for the control and modulation of an arbitrary number of real-valued parameters. Neighborhood coherent reorganizations of the two spaces ensure the maximum overlap and explorability between vocal-gesture and instrument, while it linearizes the sonic response of the instrument.
- An open-source proof-of-concept functional prototype of the VCI4DMI for user studies and live performances, as well as for exploration, modification, and further development of the proposed voice to instrument mapping methods.

The resulting VCI4DMI interface is compliant with the design principles and requirements defined in section 2.3.1, and is a significant advancement of the state-of-the-art in automated gesture mapping and vocal control of musical instruments, which include: interfaces integrability, indirect gestural acquisition, error-safe multi-parametric control, consistent perceptual instrument response, low cognitive complexity, unsupervised and adaptive mapping generation, modular reconfigurable design, and runtime tuning features.

8.2 Impact

The work reported in this dissertation and the individual contributions have the potential for impact in several related areas. In the context of musical instruments, interface developers could adopt the Self Organizing Gesture method to implement gestural controllers regardless of the input modality and instrument mapping strategy

of their specific controller. This could provide their interfaces with a simultaneous and smooth control of continuous and independent signals that are learned, and therefore adapted, to the dynamics of specific gestural examples. This frees the designer from making prior assumptions about performer's gestures, which may restrict the final interface use. Similarly interface designs can benefit from the proposed DMI control strategy based on a low dimensional representation of the sonic space, which can be abstracted from the vocal control context and integrated with any input modality. This allows designers to implement an adaptive few-to-many mapping with minimal loss in the parameter space while maximizing the expressivity of the limited control dimensions of their specific interface. The open-source prototype allows researchers, developers, and students to explore the novel techniques developed in this work, to hack the system for different application scenarios, and to further improve and develop the contribution of this dissertation.

Musicians and performers searching for additional musical control dimensions will benefit from the adoption of the VCI4DMI, and as an alternative, they can integrate individual components of our system with their instrumental setup for performing with interfaces adapted towards the characteristics of the gesture or of the instrument's sound. The default unsupervised mapping generation allows basic but effective use of the VCI4DMI regardless of the level of user expertise. Moreover the high configurability, the low cognitive complexity, and the sonic expressivity of the interface provide a low entry barrier for novice users, but a high ceiling for performers' creative use and virtuoso skills development.

The extraction of continuous, robust and independent control signals from the voice timbre can be applied to generic HCI interfaces to provide an alternative to the common speech recognition, which limits the interaction at the level of discrete verbal voice commands and which presents high latency. The proposed voice-controlled interface provides low-latency and multidimensional control in real-time, and has the potential to extend the application domain of hands-free voice driven systems, especially when it requires the control of continuous quantities.

8.3 Future work

This work has raised a few questions, and identified areas of work that remain to be done before the ideal unsupervised framework for generating ad-hoc voice-controlled musical interfaces would be realized. From the study of these experiences we identified possible system improvements in terms of usability and functionality.

Moreover we discuss further research directions motivated by the outcomes and findings of this dissertation.

8.3.1 System improvement

The porting of the modules of the open-source prototype into a single programming environment would be a key development step to broaden VCI4DMI accessibility, especially for those users without programming expertise. Priority was given to algorithms research, sacrificing implementation elegance and further optimizations. Improvements in implementation could drastically reduce the system training time, and it may include a single multithreaded application for a more efficient VCI4DMI running on a single machine, plus a set of executable modules communicating via OSC for the network distribution of the interface components. Moreover a framework to reduce the analysis time for software plugin DMIs, generating output samples more frequently than that required by the real-time audio sampling rate could further improve the user experience in setting up the desired interface.

We discussed the central role of the visual feedback, especially for the process of understanding, practicing, and eventually tuning the unsupervised mapping. We believe an overall improvement of the training outcome visualization of the data loaded in the VCI4DMI, including graphic enhancements and dynamic representations would be beneficial to the users, and might contribute especially to helping users understand 3D mappings and encouraging them to use it more. Further, providing interface feedback through vibro-tactile stimulators and optical head mounted displays, such as Google Glass, could further reduce the visual engagement required with screens or with the minimalistic wrist controller.

The VCI4DMI timed energy gate and the use of a head-worn dynamic microphone with hyper-cardioid polar pattern were effective in reducing the detrimental effects that background noise can cause to degrade the voice to DMI parameters mapping, while the MIDI generator onset detector still presents some issues. Voice and noise separation were beyond the scope of this thesis and this problem could be addressed integrating existing real-time noise reduction or voice signal isolation techniques. In the typical use scenario the loudspeaker feedback is the dominant component of the background noise captured by the microphone. Thus the input of the interface system includes voice plus feedback, which is nothing but replicas of the mixing board output signal, filtered by the loudspeakers and room acoustic response. If we suppose that the mixing board output is known to the interface system, echo cancellation techniques (Gritton and Lin, 2003; Tandon,

Ahmad, and Swamy, 2004) could isolate the voice signal and reject the background noise. The energy gate could be replaced with voice-over-music detection (Lukashevich, Gruhne, and Dittmar, 2007; Rocamora and Herrera, 2007) or separation (Sofianos, Ariyaeeinia, and Polfreman, 2010; Usher, 2006) techniques. Furthermore, noise could be reduced using multiple microphones for source separation (Favrot, Erne, and Faller, 2006) and other highly directional microphone arrays based on receiver beamforming (Brandstein and Ward, 2001).

Finally the prototype GUI refinement should include basic and advanced user modes, exposing a few simple options in the first case and granting full detailed access to the interface internal settings in the second. Moreover the development of an additional GUI module to manage vocal GC and DMI mapping data structures, and to organizing these in banks would facilitate performances preparation and maintenance of the VCI4DMI mappings user library.

8.3.2 Research directions

The challenges addressed in this thesis and the proposed solutions have in turn stimulated other questions and possible ways to overcome limitations of this research in the direction of vocal control of musical instruments, advanced generative mapping techniques, and tools supporting users-centered implementation of interfaces.

The survey on related works showed that the gesture acquisition system for vocal interface could be something other than traditional microphones. In this context the high fidelity of the captured sound is not a primary required characteristic. Other solutions such as neck microphones, in mouth microphones, electroglottography, and mouth image tracking, are alternatives that provide intrinsic acoustic noise robustness but present their own drawbacks and are affected by other kinds of noise. We believe that composite gestural acquisition systems with an appropriate pre-processing stage should be explored and have the potential to mutually overcome the individual acquisition system shortcomings, in addition to controlling noise in the interface. However open challenges of this approach are in developing a strategy to merge the raw gestural data stream coming from different sensors, at different rates, with different latency, and in isolating the non-redundant information.

We discarded the temporal information when analyzing the vocal-gestures, and we discussed how different vocal timbres could still trigger the interface. We also discussed the strong limitation in this context of traditional HMM for providing temporal modeling of the training data. A more complex HMM topology than the traditional left-to-right scheme could be effective, but this would require a much

larger amount of training data for accurate modeling. We tested embedding transition probabilities on the links of the GC SOM lattice for a minimal temporal modeling, and we observed limitations in free gestural space exploration determined by transitions with zero probability. This approach could be further explored, developing a separate model trained on temporal information of the gesture, which identifies the coherence of the live input with the training data. Problems to be addressed here are the high latency and the limited space explorability that this approach may introduce.

The DMI parameter-to-sound analysis we introduced presents two limitations. Firstly, the user chooses the appropriate analysis mode and settings manually, and this affects the generated sonic space and mapping function. For a fully unsupervised mapping generation these choices should be automatic, and based on a quick preliminary DMI analysis, emulating the user decisional workflow. The open challenges here are the sound analysis strategy to classify the DMI into the proper category, and the development of a smart strategy to perform the right classification analyzing only a very small set of parameter combinations. Secondly, for sound processors the output sound perceptual analysis technique proposed in this dissertation is still basic and the topic is essentially poorly explored in existing works. The timbre modification and the related perceptual sonic difference determined by complex sound processors requires deeper user studies to develop a computational model first, and this topic presents plenty of opportunities for future research.

Further user studies are needed to explore how unsupervised mappings fulfill user expectations implicitly expressed in the training data set, and how tuning the mapping algorithms might bring the results more in line with those expectations. For unsupervised ML, mapping accuracy measurement is not possible and actual verification needs to be user-centered. In the specific case of vocal control of DMIs, voice and instruments present respectively narrow and broad sound generation capability, theoretically unlimited for the latters. Ensuring that the interface has associated timbres between the two domains as the user intends may simply not be possible. Further research is necessary to investigate and explore this path, providing a more meaningful overlapping of these spaces without sacrificing the modularity and independency of the two interface components.

Finally we believe that research in musical interfaces, regardless of the input modality and control purpose, would benefit from greater adaptability and personalization. This would be in line with a trend that is emerging in consumer programmable devices and applications such as mobile phones and web services that personalize the interface layout or the displayed information based on a large database of usage patterns and history. This process could be gradual and transparent

to the user, continuously collecting use data for offline reconfiguration of the interface response without disrupting the musical control skills already mastered.

8.4 Closing remarks

The VCI4DMI addresses issues and limitations of the current state-of-the-art in adaptive frameworks for musical interface design. The VCI4DMI empowers performers with novel musical control strategies that are easily accessible. The variety of types of musical control that the system can provide using different training settings and operational modes is designed to support performer creativity. The space of possibilities for voice-to-sound control is large and this in turn enables engaging performances with no hardware except for a microphone. However the VCI4DMI is a voice-to-instrument mapping toolkit that does not present any "out of the box" usable mapping, because it includes no prior knowledge about user's voice and DMI. The resulting analysis and system procedure, although already minimized and central to generate customized mappings, may represent a barrier for performers more oriented to plug and play systems. From extensive use of the VCI4DMI we recognize that the effort required for memorizing the mappings and master vocal control skills is similar to the traditional acoustic instruments. Performances based exclusively on the VCI4DMI were particularly challenging because the vocal interface was continuously reconfigured while performing, requiring familiarization and memorization with multiple mappings. Although the audience feedback on the performances was very positive, we believe that the greatest advantage of the vocal musical control is within the context of concurrent use of multiple multimodal interfaces. We do not think that in the future more advanced and refined vocal interfaces can simply substitute for the traditional ones in either professional, amateur, or gaming use. The addition of the extra channel of control derived from the vocal-gesture enhances the existing virtuosity and creativity of performers.

Musical ideas are realized into concrete sound through gestures compatible with specific instrument interfaces. For humans, especially musically untrained subjects, the vocal apparatus is the most natural and direct means to express sonic ideas. Although the timbre of the voice could be very different from any instrument's timbre, the temporal progressions of energy, pitch, and timbre can be nuanced, intuitive and perceptually accurate. Voice-controlled musical interfaces shorten the brain-to-sound path, and this thesis represents a step towards systems that realize sonic intentions without the need for explicit gestural expressions.

Bibliography

ABRIL, C. R. 2007. I have a voice but I just can't sing: a narrative investigation of singing and social anxiety. *Music Education Research* 9, 1–15.

ADRIEN, J. M. 1991. Physical model synthesis: the missing link. In *Representations of Musical Signals*, De Poli, G., Piccialli, A. and Roads, C. (Eds.), pp. 269–297. MIT Press.

ARFIB, D., COUTURIER, J. M., KESSOUS, L. AND VERFAILLE, V. 2002. Strategies of mapping between gesture data and synthesis model parameters using perceptual spaces. *Organized Sound* 7, 127–144.

BELLMAN, R. 1972. *Dynamic Programming*. Princeton University Press.

BENADE, A. H. 1990. *Fundamentals of Musical Acoustics*. Dover Publications.

BENCINA, R. 2005. The metasurface: applying natural neighbour interpolation to two-to-many mapping. In *Proceedings of the 5th international conference on New Interfaces for Musical Expression*, pp. 101–104. Vancouver, Canada.

BERNARDINI, N., SERRA, X., LEMAN, M., WIDMER, G. AND DE POLI, G. 2007. *A Roadmap for Sound and Music Computing*. The S2S Consortium http://smcnetwork.org/roadmap (Accessed February 17, 2014).

BEVILACQUA, F., MÜLLER, R. AND SCHNELL, N. 2005. MnM: a Max/MSP mapping toolbox. In *Proceedings of the 5th international conference on New Interfaces for Musical Expression*, pp. 85–88. Vancouver, Canada.

BEVILACQUA, F., SCHNELL, N., RASAMIMANANA, N., ZAMBORLIN, B. AND GUÉDY, F. 2011. Online gesture analysis and control of audio processing. In *Musical Robots and Interactive Multimodal Systems*, *Springer Tracts in Advanced Robotics*, Solis, J. and Ng, K. (Eds.), pp. 127–142. Springer Berlin Heidelberg.

BEVILACQUA, F., ZAMBORLIN, B., SYPNIEWSKI, A., SCHNELL, N., GUÉDY, F. AND RASAMIMANANA, N. 2010. Continuous realtime gesture following and recognition. In *Gesture in Embodied Communication and Human-Computer Interaction*, pp. 73–84. Springer.

BILMES, J. A., LI, X., MALKIN, J., KILANSKI, K., WRIGHT, R., KIRCHHOFF, K., SUBRAMANYA, A., HARADA, S., A, J., DOWDEN, P. AND CHIZECK, H. 2005. The vocal joystick: a voice-based human-computer interface for individuals with motor impairments. In *Human Language Technology conference and conference on Empirical Methods in Natural Language Processing*, pp. 995–1002. Vancouver, Canada.

BLAINE, T. AND FELS, S. 2003. Contexts of collaborative musical experiences. In *Proceedings of the 3rd international conference on New Interfaces for Musical Expression*, pp. 129–134. Montreal, Canada.

BONGERS, B. 2000. Physical interfaces in the electronic arts. Interaction theory and interfacing techniques for real-time performance. In *Trends in Gestural Control of Music*, Wanderley, M. M. and Battier, M. (Eds.), pp. 41–70. Paris, France: IRCAM Centre Pompidou.

BOYCE, C. AND NEALE, P. 2006. *Conducting In-Depth Interviews: A Guide for Designing and Conducting In-Depth Interviews for Evaluation Input*. Pathfinder International.

BRANDSTEIN, M. AND WARD, D. 2001. *Microphone Arrays: Signal Processing Techniques and Applications*. New York: Springer.

BRANDTSEGG, U., SAUE, S. AND JOHANSEN, T. 2011. A modulation matrix for complex parameter sets. In *Proceedings of the 11th international conference on New Interfaces for Musical Expression*. Oslo, Norway.

BRÜGGEN, M. 2001. Coloration and binaural decoloration in natural environments. *Acta Acustica united with Acustica* 87, 400–406.

BRYMAN, A. 2012. *Social Research Methods*. Oxford University Press.

BURGOYNE, J. A. AND MCADAMS, S. 2007. Non-linear scaling techniques for uncovering the perceptual dimensions of timbre. In *Proceedings of the 2007 International Computer Music Conference*vol. 1, pp. 73–76. Copenhagen, Denmark.

BURGOYNE, J. A. AND MCADAMS, S. 2008. A meta-analysis of timbre perception using nonlinear extensions to CLASCAL. In *Computer Music Modeling and Retrieval. Sense of Sounds*, Kronland-Martinet, R., Ystad, S. and Jensen, K. (Eds.), pp. 181–202. Berlin, Heidelberg: Springer-Verlag.

CADOZ, C., LUCIANI, A. AND FLORENS, J. L. 1993. CORDIS-ANIMA: A modeling and simulation system for sound and image synthesis. *Computer Music Journal* 17, 19–29.

CADOZ, C. AND WANDERLEY, M. M. 2000. Gesture-music. In *Trends in Gestural Control of Music*vol. 12, Wanderley, M. M. and Battier, M. (Eds.), p. 101. Paris, France: IRCAM Centre Pompidou.

CAGE, J. 1937. The future of music: Credo. In *Silence: Lectures and Writings*, Cage, J. (Ed.), pp. 3–6. Wesleyan University Press, Published in 1961.

CARAMIAUX, B. AND TANAKA, A. 2013. Machine learning of musical gestures. In *Proceedings of the 13th international conference on New Interfaces for Musical Expression*. Daejeon, Korea.

CARAMIAUX, B., WANDERLEY, M. M. AND BEVILACQUA, F. 2012. Segmenting and parsing instrumentalists' gestures. *Journal of New Music Research* 41, 13–29.

CARIOU, B. 1992. Design of an alternative controller from an industrial design perspective. In *Proceedings of the 1992 International Computer Music Conference*, pp. 366–367. San Francisco, US.

CHADABE, J. 2002. The limitations of mapping as a structural descriptive in electronic instruments. In *Proceedings of the 2nd international conference on New Interfaces for Musical Expression*, pp. 1–5. Singapore.

DE CHEVEIGNÉ, A. AND KAWAHARA, H. 2002. YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America* 111, 1917.

CHOI, I., BARGAR, R. AND GOUDESEUNE, C. 1995. A manifold interface for a high dimensional control space. In *Proceedings of the 2005 International Computer Music Conference*, pp. 385–392. San Francisco, US.

CHOWNING, J. 1971. The simulation of moving sound sources. *Journal of the Audio Engineering Society* 2–6.

CHOWNING, J. 1973. The synthesis of complex audio spectra by means of frequency modulation. *Journal of the Audio Engineering Society* 21.

CLARKE, E. F. 1988. Generative principles in music performance. In *Generative processes in music*, pp. 1–26. Clarendon Press.

CLARK, J. AND YALLOP, C. 1995. *An Introduction to Phonetics and Phonology*. Wiley.

COLLINS, N. 2009. Electronica. In *The Oxford Handbook of Computer Music*, Dean, R. T. (Ed.). Oxford University Press.

COMON, P. 1994. Independent component analysis, a new concept? *Signal Processing - Special issue on higher order statistics* 36, 287–314.

COOK, P. R. 1991. 'Identification of control parameters in articulatory vocal tract model, with application to the synthesis of singing'. PhD Thesis, Stamford University.

COOK, P. R. 2001. Principles for designing computer music controllers. In *Proceedings of ACM Computer-Human Interaction Workshop on New Interfaces for Musical Expression*, pp. 1–4. Seattle, USA.

COOK, P. R. 2004. Remutualizing the musical instrument: co-design of synthesis algorithms and controllers. *Journal of New Music Research* 33, 315–320.

COOK, P. R. 2009. Re-designing principles for computer music controllers: a case study of SqueezeVox Maggie. In *Proceedings of the 9th international conference on New Interfaces for Musical Expression*. Pittsburgh, US.

DAHLSTEDT, P. 2001. Creating and exploring huge parameter spaces: interactive evolution as a tool for sound generation. In *Proceedings of the 2001 International Computer Music Conference*, pp. 235–242.

DAVIS, S. AND MERMELSTEIN, P. 1980. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech and Signal Processing* 28, 357–366.

DEACON, J. 2014. 'The development of a software tool that employs vocals for the control of musical elements in a live performance'. BSc Thesis, University of Limerick.

DEKEL, O., KESHET, J. AND SINGER, Y. 2005. An online algorithm for hierarchical phoneme classification. In *Machine Learning for Multimodal Interaction*, pp. 146–158. Springer.

VAN DEN BERG, J. 1958. Myoelastic-aerodynamic theory of voice production. *Journal of Speech and Hearing Research* 1, 227–44.

VAN DER MAATEN, L. J. P., POSTMA, E. O. AND VAN DEN HERIK, H. J. 2009. *Dimensionality reduction: a comparative review*. Tilburg University Technical Report.

DICTIONARY.COM n.d. electronic music. *Dictionary.com Unabridged*. http://dictionary.reference.com/browse/electronic music (Accessed December 18, 2013).

DIJKSTRA, E. W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1, 269–271.

DOBRIAN, C. AND KOPPELMAN, D. 2006. The 'E' in NIME: musical expression with new computer interfaces. In *Proceedings of the 6th international conference on New Interfaces for Musical Expression*, p. 277.

ELLINGSON, T. J. 1979. 'The mandala of sound: concepts and sound structures in tibetan ritual music'. Ph.D. Thesis, University of Wisconsin at Madison.

EMMERSON, S. 1994. 'Live' versus 'real-time'. *Contemporary Music Review* 10, 95–101.

EMMERSON, S. 1998. Aural landscape: musical space. *Organised Sound* 3, 135–140.

EPPS, J., SMITH, J. R. AND WOLFE, J. 1997. A novel instrument to measure acoustic resonances of the vocal tract during phonation. *Measurement Science and Technology* 8, 1112.

ERICKSON, R. 1975. *Sound Structure in Music*. University of California Press.

FANT, G. 1960. *Acoustic Theory of Speech Production*. The Hague: Mouton.

FASCIANI, S. 2012. Voice features for control: a vocalist dependent method for noise measurement and independent signals computation. In *Proceedings of the 15th international conference on Digital Audio Effects*. York, UK.

FASCIANI, S. AND WYSE, L. 2012a. A voice interface for sound generators: adaptive and automatic mapping of gestures to sound. In *Proceedings of the 12th international conference on New Interfaces for Musical Expression*. Ann Arbor, US.

FASCIANI, S. AND WYSE, L. 2012b. Adapting general purpose interfaces to synthesis engines using unsupervised dimensionality reduction techniques and inverse mapping from features to parameters. In *Proceedings of the 2012 International Computer Music Conference*. Ljubljana, Slovenia.

FASCIANI, S. AND WYSE, L. 2013a. A self-organizing gesture map for a voice-controlled instrument interface. In *Proceedings of the 13th international conference on New Interfaces for Musical Expression*. Daejeon, Korea.

FASCIANI, S. AND WYSE, L. 2013b. One at a time by voice: performing with the voice--controlled interface for digital musical instruments. In *Proceedings of the NTU/ADM symposium on Sound and Interactivity 2013, extended for the Journal of Canadian Electroacoustic Community CEC, 16.2, 2014*. Singapore.

FAVREAU, E., FINGERHUT, M., KOECHLIN, O., POTACSEK, P., PUCKETTE, M. AND ROWE, R. 1986. Software developments for the 4X real-time system. In *Proceedings of the 1986 International Computer Music Conference*, pp. 43–46. San Francisco, US.

FAVROT, A., ERNE, M. AND FALLER, C. 2006. Improved cocktail-party processing. In *Proceedings of the 9th International Conference on Digital Audio Effects*. Montreal, Canada.

FELS, S. 2000. Intimacy and embodiment: implications for art and technology. In *Proceedings of the 2000 ACM workshops on Multimedia*, pp. 13–16.

FELS, S. 2004. Designing for intimacy: creating new interfaces for musical expression. *Proceedings of the IEEE* 92, 672–685.

FIEBRINK, R. 2011. 'Real-time human interaction with supervised learning algorithms for music composition and performance'. Ph.D. Thesis, Princeton University.

FIEBRINK, R., COOK, P. R. AND TRUEMAN, D. 2009. Play-along mapping of musical controllers. In *Proceedings of the 2009 International Computer Music Conference*. Montreal, Canada.

FOREMAN, D. 2013. *The polyphonic me*. http://www.ted.com/talks/beardyman_the_polyphonic_me.html (Accessed January 17, 2014).

FRANCOISE, J., CARAMIAUX, B. AND BEVILACQUA, F. 2012. A hierarchical approach for the design of gesture-to-sound mappings. In *Proceedings of the 9th Sound and Music Computing international conference*. Copenhagen, Denmark.

FYANS, A. C. AND GUREVICH, M. 2011. Perceptions of skill in performances with acoustic and electronic instruments. In *Proceedings of the 11th international conference on New Interfaces for Musical Expression*. Oslo, Norway.

GAFFNEY, B. B. AND SMYTH, T. 2013. Acoustics-like dynamics in signal-based synthesis through parameter mapping. In *Proceedings of the 10th Sound and Music Computing international conference*. Stockholm, Sweden.

GARNETT, G. E. AND GOUDESEUNE, C. 1999. Performance factors in control of high-dimensional spaces. In *Proceedings of the 1999 International Computer Music Conference*, pp. 268–71.

GELINECK, S. AND SERAFIN, S. 2009. A quantitative evaluation of the differences between knobs and sliders. In *Proceedings of the 9th international conference on New Interfaces for Musical Expression*.

GILLIAN, N., KNAPP, R. B. AND O'MODHRAIN, S. 2011. A machine learning toolbox for musician computer interaction. In *Proceedings of the 11th international conference on New Interfaces for Musical Expression*. Oslo, Norway.

GOLDSTEIN, J. L. 1973. An optimum processor theory for the central formation of the pitch of complex tones. *The Journal of the Acoustical Society of America* 54, 1496–1516.

GOODMAN, L. A. 1961. Snowball Sampling. *The Annals of Mathematical Statistics* 32, 148–170.

GRASSBERGER, P. AND PROCACCIA, I. 1983. Measuring the strangeness of strange attractors. *Physica D: Nonlinear Phenomena* 9, 189–208.

GRETTON, A., FUKUMIZU, K. AND SRIPERUMBUDUR, B. K. 2009. Discussion of Brownian distance covariance. *The Annals of Applied Statistics* 3, 1285–1294.

GREY, J. M. 1977. Multidimensional perceptual scaling of musical timbres. *Journal of the Acoustical Society of America* 61, 1270–1277.

GRILL, T. 2012. Constructing high-level perceptual audio descriptors for textural sounds. In *Proceedings of the 9th Sound and Music Computing international conference*. Copenhagen, Denmark.

GRITTON, C. AND LIN, D. 2003. Echo cancellation algorithms. *ASSP Magazine, IEEE* 1, 30–38.

GROSSBERG, S. 1987. Competitive learning: from interactive activation to adaptive resonance. *Cognitive Science* 23–63.

GUAN, H., FERIS, R. S. AND TURK, M. 2006. The isometric self-organizing map for 3D hand pose estimation. In *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition*, pp. 263 –268. Southampton, UK.

GUBRIUM, J. F. 2012. *The SAGE Handbook of Interview Research: The Complexity of the Craft*. SAGE Publications.

GUEST, G., BUNCE, A. AND JOHNSON, L. 2006. How many interviews are enough? An experiment with data saturation and variability. *Field Methods* 18, 59–82.

HARADA, S., WOBBROCK, J. O. AND LANDAY, J. A. 2007. Voicedraw: a hands-free voice-driven drawing application for people with motor impairments. In *Proceedings of the 9th international ACM SIGACCESS conference on Computers and Accessibility*, pp. 27–34. Orlando, US.

HARADA, S., WOBBROCK, J. O., MALKIN, J., BILMES, J. A. AND LANDAY, J. A. 2009. Longitudinal study of people learning to use continuous voice-based cursor control. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 347–356. Boston, US.

HARRISON, S., TATAR, D. AND SENGERS, P. 2007. The three paradigms of HCI. In *Proceedings of Alt. Chi. session at the SIGCHI conference on Human Factors in Computing Systems*. San Jose, US.

HAZAN, A. 2005. Performing expressive rhythms with billaboop voice-driven drum generator. In *Proceedings of the 8th international conference on Digital Audio Effects*. Madrid, Spain.

VON HELMHOLTZ, H. L. F. 1954. *On the Sensations of Tone*. New York: Dover Publications (original work published in 1877).

HENRICH, N. 2006. Mirroring the voice from garcia to the present day: some insights into singing voice registers. *Logopedics, Phoniatrics, Vocology* 31, 3–14.

HERMANSKY, H. 1990. Perceptual linear predictive (PLP) analysis of speech. *The Journal of the Acoustical Society of America* 87, 1738–1752.

HERMANSKY, H. AND MORGAN, N. 1994. RASTA processing of speech. *IEEE Transactions on Speech and Audio Processing* 2, 578–589.

HERMANSKY, H., MORGAN, N., BAYYA, A. AND KOHN, P. 1991. Compensation for the effect of the communication channel in auditory-like analysis of speech (RASTA-PLP). In *Proceedings of the 2nd European Conference on Speech Communication and Technology*. Genova, Italy.

HIGHHOUSE, S. AND GILLESPIE, J. Z. 2009. Do samples really matter that much? In *Statistical and Methodological Myths and Urban Legends: Doctrine, Verity and Fable in Organizational and Social Sciences*, Lance, C. E. and Vandenberg, R. J. (Eds.), pp. 249–268. New York, NY, USA: Routledge.

HILLER, L. AND RUIZ, P. 1971. Synthesizing musical sounds by solving the wave equation for vibrating objects: part I. *Journal of the Audio Engineering Society* 19, 462—470.

VON HORNBOSTEL, E. M. AND SACHS, C. 1914. Hornbostel–Sachs. In *Zeitschrift für Ethnologie*vol. 46, pp. 553–90. Braunschweig, A. Limbach [etc.].

HOUSE, B., MALKIN, J. AND BILMES, J. 2009. The VoiceBot: a voice controlled robot arm. In *Proceedings of the 27th international conference on Human Factors in Computing Systems*, pp. 183–192. Boston, US.

HUNT, A. AND KIRK, R. 2000. Mapping strategies for musical performance. In *Trends in Gestural Control of Music*vol. 21, Wanderley, M. M. and Battier, M. (Eds.), pp. 231–258. Paris, France: IRCAM Centre Pompidou.

HUNT, A. AND WANDERLEY, M. M. 2003. Mapping performer parameters to synthesis engines. *Organised Sound* 7, 97–108.

HUNT, A., WANDERLEY, M. M. AND KIRK, R. 2000. Towards a model for instrumental mapping in expert musical interaction. In *Proceedings of the 2000 International Computer Music Conference*, pp. 209–212. Berlin, Germany.

HYVARINEN, A. 1999. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks* 10, 626–634.

IGARASHI, T. AND HUGHES, J. F. 2001. Voice as sound: using non-verbal voice input for interactive control. In *Proceedings of the 14th annual ACM symposium on User Interface Software and Technology*, pp. 155–156. Orlando, US.

ARP INSTRUMENTS, I. 1976. For keyboard players who need an extra hand. The ARP Sequencer. *Contemporary Keyboard*.

ITAKURA, F. AND SAITO, S. 1967. Analysis-synthesis transmission system based on maximum likelihood spectrum estimation. In *Proceedings of the Convention of the Acoustic Society of Japan*vol. 2-3-1, pp. 231–232 (in Japanese).

JACOB, R. J. K., SIBERT, L. E., MCFARLANE, D. C. AND MULLEN,JR., M. P. 1994. Integrality and separability of input devices. *ACM Transactions on Computer-Human Interaction* 1, 3–26.

JANER, J. 2005a. Voice-controlled plucked bass guitar through two synthesis techniques. In *Proceedings of the 5th international conference on New Interfaces for Musical Expression*, pp. 132–135. Vancouver, Canada.

JANER, J. 2005b. Feature extraction for voice-driven synthesis. In *Proceedings of the 118th Audio Engineering Society convention*. Barcelona, Spain.

JANER, J. 2008. 'Singing-driven interfaces for sound synthesizers'. PhD Thesis, Universitat Pompeu Fabra, Barcelona.

JANER, J. AND DE BOER, M. 2008. Extending voice-driven synthesis to audio mosaicing. In *Proceedings of the 5th Sound and Music Computing international conference*vol. 4. Berlin, Germany.

JEHAN, T. 2001. 'Perceptual synthesis engine: an audio-driven timbre generator'. Master Thesis, Massachusetts Institute of Technology.

JEHAN, T. AND SCHONER, B. 2001. An audio-driven perceptually meaningful timbre synthesizer. In *Proceedings of the 2001 International Computer Music Conference*. Havana, Cuba.

JOHNSON, K. 2008. Speaker Normalization in Speech Perception. In *The Handbook of Speech Perception*, Pisoni, D. and Remez, R. (Eds.), p. 363. Wiley-Blackwell.

JORDÀ, S. 2004a. Digital instruments and players: part I - efficiency and apprenticeship. In *Proceedings of the 4th international conference on New Interfaces for Musical Expression*, pp. 59–63. Hamamatsu, Japan.

JORDÀ, S. 2004b. Digital instruments and players: part II – diversity, freedom and control. In *Proceedings of the 2004 International Computer Music Conference*, pp. 706–710. Miami, US.

JORDÀ, S., KALTENBRUNNER, M., GEIGER, G. AND BENCINA, R. 2005. The reacTable*. In *Proceedings of the 2005 International Computer Music Conference*. Barcelona, Spain.

KAPUR, A., BENNING, M. AND TZANETAKIS, G. 2004. Query-by-beat-boxing: music retrieval for the dj. In *Proceedings of the International Conference on Music Information Retrieval*, pp. 170–177. Barcelona, Spain.

KARTOMI, M. J. 1990. *On Concepts and Classifications of Musical Instruments*. Chicago: University of Chicago Press, Chicago studies in ethnomusicology.

KEISLAR, D. 2009. A Historical View of Computer Music Technology. In *The Oxford Handbook of Computer Music*, Dean, R. T. (Ed.). Oxford University Press.

KESTIAN, A. P. AND SMYTH, T. 2010. Real-time estimation of the vocal tract shape for musical control. In *Proceedings of the 7th Sound and Music Computing international conference*. Barcelona, Spain.

KIEFER, C., COLLINS, N. AND FITZPATRICK, G. 2008. HCI methodology for evaluating musical controllers: a case study. In *Proceedings of the 8th international conference on New Interfaces for Musical Expression*.

KIVILUOTO, K. 1996. Topology preservation in self-organizing maps. In *Proceedings of the IEEE International Conference on Neural Networks*vol. 1, pp. 294 – 299. Piscataway, US.

KOHONEN, T. 1982. Self-organized formation of topologically correct feature maps. *Biological Cybernetics* 43, 59–69.

KÖNIG, S. 2006. scrambled?HaCkZ! Paris, France.

KVALE, S. 1996. *InterViews: An Introduction to Qualitative Research Interviewing*. SAGE Publications.

KVALE, S. AND BRINKMANN, S. 2008. *InterViews: Learning the Craft of Qualitative Research Interviewing*. Second Edition. SAGE Publications, Inc.

KVIFTE, T. AND JENSENIUS, A. R. 2006. Towards a coherent terminology and model of instrument description and design. In *Proceedings of the 6th international conference on New Interfaces for Musical Expression*, pp. 220–225. Paris, France.

LALLEMAND, I. AND SCHWARZ, D. 2011. Interaction-optimized sound database representation. In *Proceedings of 14th International Conference on Digital Audio Effects*. Paris, France.

LAVER, J. 1980. *The Phonetic Description of Voice Quality*. Cambridge University Press.

LAZIER, A. AND COOK, P. R. 2003. Mosievius: feature driven interactive audio mosaicing. In *Proceedings of the 7th international conference on Digital Audio Effects*. Napoli, Italy.

LAZZARINI, V. AND TIMONEY, J. 2009. New methods of formant analysis-synthesis for musical applications. In *Proceedings of the 2009 International Computer Music Conference*. Montreal, Canada.

LECLUSE, F. L. E., BROCAAR, M. P. AND VERSCHURRE, J. 1975. The electoglottography and its relation to glottal activity. *Fol Phoniatr* 27, 215–24.

LEE, M. AND WESSEL, D. 1992. Connectionist models for real-time control of synthesis and compositional algorithms. In *Proceedings of the 1992 International Computer Music Conference*. San Jose, CA.

LEVENBERG, K. 1944. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathmatics* II, 164–168.

LEVINA, E. AND BICKEL, P. J. 2004. Maximum likelihood estimation of intrinsic dimension. *Advances in Neural Information Processing Systems* 17, 777–784.

LEVITIN, D. J., MCADAMS, S. AND ADAMS, R. L. 2002. Control parameters for musical instruments: a foundation for new mappings of gesture to sound. *Organized Sound* 7, 171–189.

LIKERT, R. 1932. A technique for the measurement of attitudes. *Archives of Psychology* 22 140, 55.

LINDSTRØM, H. P. 2007. Lindstrøm: Beyond space disco. http://www.residentadvisor.net/feature.aspx?805 (Accessed January 7, 2014).

LOSCOS, A. AND AUSSENAC, T. 2005. The Wahwactor: a voice controlled wah-wah pedal. In *Proceedings of the 5th international conference on New Interfaces for Musical Expression*, pp. 172–175. Vancouver, Canada.

LOSCOS, A., CANO, P. AND BONADA, J. 1999. Low-delay singing voice alignment to text. In *Proceesings of the 1999 International Computer Music Conference*. Beijing, China.

LOSCOS, A., CANO, P. AND BONADA, J. 2005. Larynxophone: using voice as a wind controller. In *Proceedings of the 2005 International Computer Music Conference*. Barcelona, Spain.

LUKASHEVICH, H., GRUHNE, M. AND DITTMAR, C. 2007. Effective singing voice detection in popular music using arma filtering. In *Workshop on Digital Audio Effects*. Bordeaux, France.

LYONS, M. J. AND FELS, S. 2012. Advances in new interfaces for musical expression. In *Proceedings of SIGGRAPH Asia 2012 Courses*. Singapore.

LYONS, M. J., HAEHNEL, M. AND TETSUTANI, N. 2003. Designing, playing, and performing with a vision-based mouth interface. In *Proceedings of the 3rd international conference on New Interfaces for Musical Expression*, pp. 116–121. Montreal, Canada.

VAN DER MAATEN, L. AND HINTON, G. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9.

MACHOVER, T. 2002. Instruments, interactivity, and inevitability. In *Proceedings of the 2nd international conference on New Interfaces for Musical Expression*. Dublin, Ireland.

MALLOCH, J., BIRNBAUM, D., SINYOR, E. AND WANDERLEY, M. M. 2006. Towards a new conceptual framework for digital musical instruments. In *Proceedings of the 9th international conference on Digital Audio Effects*, pp. 49–52. Montreal, Canada.

MARKEL, J. 1972. Digital inverse filtering - a new tool for formant trajectory estimation. *IEEE Transactions on Audio and Electroacoustics* 20, 129–137.

MARQUARDT, D. W. 1963. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics* 11, 431–441.

MARQUEZ-BORBON, A., GUREVICH, M., FYANS, A. C. AND STAPLETON, P. 2011. Designing digital musical interactions in experimental contexts. In *Proceedings of the 11th international conference on New Interfaces for Musical Expression*vol. 16, p. 21. Oslo, Norway.

MARTIN, P. Y. AND TURNER, B. A. 1986. Grounded theory and organizational research. *The Journal of Applied Behavioral Science* 22, 141–157.

MATHEWS, M. V. 1963. The digital computer as a musical instrument. *Science* 142, 553–557.

MATHEWS, M. V. AND GUTTMAN, N. 1959. Generation of music by a digital computer. In *Proceedings of the 3rd International Congress on Acoustics*. Amsterdam, Netherlands.

MAY, E. 1983. *Musics of Many Cultures: An Introduction*. University of California Press.

MCADAMS, S. AND BERGMAN, A. 1979. Hearing musical streams. *Computer Music Journal* 3, 26–43, 60.

MCADAMS, S. AND CUNIBLE, J. C. 1992. Perception of timbral analogies. *Royal Society of London Philosophical Transactions* 336, 383–389.

MCCANDLESS, S. 1974. An algorithm for automatic formant extraction using linear prediction spectra. *IEEE Transactions on Acoustics, Speech and Signal Processing* 22, 135–141.

MCGURK, H. AND MACDONALD, J. 1976. Hearing lips and seeing voices. *Nature* 264, 746–748.

MCLEAN, A., SHIN, E. AND NG, K. C. 2013. Paralinguistic microphone. In *Proceedings of the 13th international conference on New Interfaces for Musical Expression*. Daejeon, Korea.

MCPHERSON, A. 2012. TouchKeys: capacitive multi-touch sensing on a physical keyboard. In *Proceedings of the 12th international conference on New Interfaces for Musical Expression*. Ann Arbor, US.

MCPHERSON, A. 2013. Portable measurement and mapping of continuous piano gesture. In *Proceedings of 13th international conference on New Interfaces for Musical Expression*. Daejeon, Korea.

MEHRABANI, M. AND HANSEN, J. 2013. A study of speaker dependent formant space variations in karaoke singing. In *Proceedings of the 2013 Stockholm Music Acoustics Conference*. Stockholm, Sweden.

MEHTA, D. D., RUDOY, D. AND WOLFE, P. J. 2012. Kalman-based autoregressive moving average modeling and inference for formant and antiformant tracking. *The Journal of the Acoustical Society of America* 132, 1732.

MENZIES, D. 2002. Composing instrument control dynamics. *Organised Sound* 7, 255–266.

MEYER, L. B. 1956. *Emotion and Meaning in Music*. University of Chicago Press.

MIRANDA, E. R. AND WANDERLEY, M. M. 2006. *New digital musical instruments: control and interaction beyond the keyboard*. A-R Editions, Inc.

MOMENI, A. AND WESSEL, D. 2003. Characterizing and controlling musical material intuitively with geometric models. In *Proceedings of the 3rd international conference on New interfaces for Musical Expression*, pp. 54–62. Montreal, Canada.

MONSON, B. B. 2011. 'High-frequency energy in singing and speech'. Ph.D. Thesis, University of Arizona.

MOORE, B. C. J. AND GLASBERG, B. R. 1983. Suggested formulae for calculating auditory-filter bandwidths and excitation patterns. *The Journal of the Acoustical Society of America* 74, 750–753.

MOORE, F. R. 1988. The dysfunctions of MIDI. *Computer Music Journal* 12, 19–28.

MORRISON, D. AND ADRIEN, J. M. 1993. MOSAIC: A framework for modal synthesis. *Computer Music Journal* 17, 45–56.

MULDER, A. 2000. Towards a choice of gestural constraints for instrumental performers. In *Trends in Gestural Control of Music*, Wanderley, M. M. and Battier, M. (Eds.), pp. 315–335. Paris, France: IRCAM Centre Pompidou.

MUSTAFA, K. AND BRUCE, I. C. 2006. Robust formant tracking for continuous speech with speaker variability. *IEEE Transactions on Audio, Speech, and Language Processing* 14, 435 – 444.

NARMOUR, E. 1992. *The Analysis and Cognition of Melodic Complexity: The Implication-Realization Model*. University of Chicago Press.

NATH, A. R. AND BEAUCHAMP, M. S. 2012. A neural basis for interindividual differences in the McGurk effect, a multisensory speech illusion. *NeuroImage* 59, 781–787.

NESS, S. R. AND TZANETAKIS, G. 2009. SOMBA: Multiuser Music Creation Using Self-Organized Maps and Motion Tracking. In *Proceedings of the 2009 International Computer Music Conference*.

NGUYEN, H., BURKARDT, J., GUNZBURGER, M., JU, L. AND SAKA, Y. 2009. Constrained CVT meshes and a comparison of triangular mesh generators. *Computational Geometry* 42, 1–19.

NIELSEN, J. 1993. Iterative user-interface design. *Computer* 26, 32–41.

NORMAN, D. A. 1988. *The Psychology Of Everyday Things*. Basic Books.

NORMAN, D. A. AND DRAPER, S. W. 1986. *User Centered System Design: New Perspectives on Human-computer Interaction*. L. Erlbaum Associates Inc.

VAN NORT, D. 2009. 'Modular and adaptive control of sound processing'. Ph.D. Thesis.

VAN NORT, D. AND WANDERLEY, M. M. 2007. Control strategies for navigation of complex sonic spaces. In *Proceedings of the 7th international conference on New Interfaces for Musical Expression*, pp. 379–382. New York, US: ACM.

VAN NORT, D., WANDERLEY, M. M. AND DEPALLE, P. 2004. On the choice of mappings based on geometric properties. In *Proceedings of the 4th international conference on New Interfaces for Musical Expression*, pp. 87–91. Hamamatsu, Japan.

ODOWICHUK, G. AND TZANETAKIS, G. 2012. Browsing music and sound using gestures in a self-organized 3D space. In *Proceedings of the 2012 International Computer Music Conference*. Ljubljana, Slovenia.

OGDEN, J. AND LO, J. 2012. How meaningful are data from Likert scales? An evaluation of how ratings are made and the role of the response shift in the socially disadvantaged. *Journal of Health Psychology* 17, 350–361.

O'HALLORAN, K., PODLASOV, A., CHUA, A. AND E, M. K. L. 2012. Interactive software for multimodal analysis. *Visual Communication* 11, 363–381.

O'HALLORAN, K., TAN, S. AND E, M. K. L. 2014. Multimodal analytics: software and visualization techniques for analyzing and interpreting multimodal data. In *The Routledge Handbook of Multimodal Analysis*, Jewitt, C. (Ed.). Routledge.

OHM, G. 1843. *Annalen der Physik* 513.

OLIVER, W., YU, J. AND METOIS, E. 1997. The singing tree: design of an interactive musical interface. In *Proceedings of the 2nd conference on Designing interactive systems: processes, practices, methods, and techniques*, p. 264. Amsterdam, Netherlands.

O'MODHRAIN, S. 2011. A framework for the evaluation of digital musical instruments. *Computer Music Journal* 35, 28–42.

ORIO, N. 1997. A gesture interface controlled by the oral cavity. In *Proceedings of the 1997 International Computer Music Conference*. Thessaloniki, Greece.

ORIO, N. 2006. *Music Retrieval: A Tutorial and Review*. Now Publishers Inc.

ORIO, N., SCHNELL, N. AND WANDERLEY, M. M. 2001. Input devices for musical expression: borrowing tools from HCI. In *Proceedings of ACM Computer-Human Interaction Workshop on New Interfaces for Musical Expression*, pp. 1–4. Seattle, USA.

O'SHAUGHNESSY, D. 2003. Interacting with computers by voice: automatic speech recognition and synthesis. *Proceedings of the IEEE* 91, 1272–1305.

PARADISO, J. A. 2002. Electronic music: new ways to play. *Spectrum, IEEE* 34, 18–30.

PARADISO, J. A. AND O'MODHRAIN, S. 2003. Current trends in electronic music interfaces. Guest editors' introduction. *Journal of New Music Research* 32, 345–349.

PERSSON, P.-O. AND STRANG, G. 2004. A simple mesh generator in MATLAB. *SIAM Review* 46, 329–345.

PINCH, T. J. AND TROCCO, F. 2002. *Analog Days: The Invention and Impact of the Moog Synthesizer*. Harvard University Press.

PLOMP, R. 1976. *Aspects of Tone Sensation: A Psychophysical Study*. Academic Press.

PLOMP, R. AND STEENEKEN, J. M. 1971. Pitch versus timbre. In *Proceedings of the 7th International Congress of Acoustical Society*vol. 3, pp. 377–380. Budapest, Hungary.

POSCIC, A. AND KREKOVIC, C. 2013. Controlling a sound synthesizer using timbral attributes. In *Proceedings of the 10th Sound and Music Computing international conference*. Stockholm, Sweden.

POUPYREV, I., LYONS, M. J., FELS, S. AND BLAINE, T. 2001. New interfaces for musical expression. In *Proceedings of CHI 2001, Extended Abstracts*, pp. pp. 491–492. New York, US: ACM.

PRESSING, J. 1990. Cybernetic issues in interactive performance systems. *Computer Music Journal* 14, 12.

PUCKETTE, M. 1986. Interprocess communication and timing in real-time computer music performance. In *Proceedings of the 1986 International Computer Music Conference*, pp. 43–46. Den Haag, Netherlands.

PUCKETTE, M. 1988. The Patcher. In *Proceedings of the 1988 International Computer Music Conference*, pp. 420–429. Kologne, Germany.

PUCKETTE, M. 1991. Something digital. *Computer Music Journal* 15, 65–69.

PUCKETTE, M. 1996. Pure Data. In *Proceedings of the 1996 International Computer Music Conference*, pp. 224–227. Hong Kong, China.

PUCKETTE, M. 2004. Low-dimensional parameter mapping using spectral envelopes. In *Proceedings of the 2004 International Computer Music Conference*. Miami, US.

PUCKETTE, M. AND APEL, T. 1998. Real-time audio analysis tools for Pd and MSP. In *Proceedings of the 1998 International Computer Music Conference*. Ann Arbor, US.

PUCKETTE, M. AND LIPPE, C. 1994. Getting the acoustic parameters from a live performance. In *3rd International Conference for Music Perception and Cognition*, pp. 328–333. Liège, Belgium.

QUATIERI, T. F. 2008. *Discrete-Time Speech Signal Processing: Principles and Practice*. Pearson Education.

RABINER, L. R. 1978. *Digital Processing of Speech Signals*. Prentice Hall.

RABINER, L. R. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77, 257–286.

RABINER, L. R. AND JUANG, B.-H. 1993. *Fundamentals of Speech Recognition*. Pearson Education.

RAMAKRISHNAN, C., FREEMAN, J. AND VARNIK, K. 2004. The architecture of auracle: a real-time, distributed, collaborative instrument. In *Proceedings of the 4th conference on New interfaces for Musical Expression*, pp. 100–103. Hamamatsu, Japan.

RAO, C. R. 1948. The utilization of multiple measurements in problems of biological classification. *Journal of the Royal Statistical Society* 10, 159–203.

RASMUSSEN, J. 1986. *Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering*. Elsevier Science Inc.

RESNICK, M., MYERS, B., NAKAKOJI, K., SHNEIDERMAN, B., PAUSCH, R., SELKER, T. AND EISENBERG, M. 2005. Design principles for tools to support creative thinking. In *Proceedings of the NSF Workshop on Creativity Support Tools*, pp. 25–36. Washington, US.

RISSET, J. C. 1978a. *Paradoxes De Hauteur*. IRCAM.

RISSET, J. C. 1978b. *Hauteur et Timbre des Sons*. IRCAM.

RISSET, J. C. AND WESSEL, D. 1999. Exploration of timbre by analysis and synthesis. *The psychology of music* 113–169.

ROCAMORA, M. AND HERRERA, P. 2007. Comparing audio descriptors for singing voice detection in music audio files. In *Proceedings of the 11th Brazilian Symposium on Computer Music*vol. 26, p. 27. Sao Paulo, Brazil.

ROSENBLUM, L. D. AND SALDAÑA, H. M. 1996. An audiovisual test of kinematic primitives for visual speech perception. *Journal of Experimental Psychology. Human Perception and Performance* 22, 318–331.

ROTHBAUER, P. 2008. Triangulation. In *The SAGE Encyclopedia of Qualitative Research Methods*, Given, L. (Ed.), pp. 892–894. Sage Publications.

ROUSSEEUW, P. J. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* 20, 53–65.

ROVAN, J. B., WANDERLEY, M. M., DUBNOV, S. AND DEPALLE, P. 1997. Instrumental gestural mapping strategies as expressivity determinants in computer music performance. In *Proceedings of Kansei - The Technology of Emotions Workshop*. Genova, Italy.

ROWEIS, S. T. AND SAUL, L. K. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 2323–2326.

ROWE, R. 1993. *Interactive Music Systems: Machine Listening and Composing*. MIT Press.

ROWE, R. 1995. Incrementally improving interactive music systems. *Contemporary Music Review* 13, 47–62.

RYAN, J. 1992. Effort and Expression: Some Notes on Instrument Design at STEIM. In *Proceedings of the 1992 International Computer Music Conference*, Strange, A. (Ed.).

SACHS, C. 1940. *The history of musical instruments*. Dover Publications.

SANDERSON, C. AND PALIWAL, K. K. 1997. Effect of different sampling rates and feature vector sizes on speech recognition performance. In *TENCON'97. IEEE Region 10 Annual Conference. Speech and Image Technologies for Computing and Telecommunications., Proceedings of IEEE*vol. 1, pp. 161–164.

SCHAFER, R. W. AND RABINER, L. R. 1970. System for automatic formant analysis of voiced speech. *The Journal of the Acoustical Society of America* 47, 634–48.

SCHLOSS, W. A. 2003. Using contemporary technology in live performance: The dilemma of the performer. *Journal of New Music Research* 32, 239–242.

SCHNELL, N., BORGHESI, R., SCHWARZ, D., BEVILACQUA, F. AND MULLER, R. 2005. FTM - Complex Data Structure for Max. In *Proceedings of the 2005 International Computer Music Conference*. Barcelona, Spain.

SCHNELL, N., CIFUENTES, M. A. S. AND LAMBERT, J. P. 2010. First steps in relaxed real-time typo-morphological audio analysis/synthesis. In *Proceeding of the 7th Sound and Music Computing international conference*. Barcelona, Spain.

SCHNELL, N. AND SCHWARZ, D. 2005. Gabor, multi-representation real-time analysis/synthesis. In *Proceedings of the 8th international conference on Digital Audio Effects*. Madrid, Spain.

SCHOENBERG, A. 1922. *Harmonielehre*. Universal-edition.

SCHOUTEN, J. F. 1968. The perception of timbre. In *Reports of the 6th International Congress on Acoustics*, pp. 35–44, 90. Tokyo, Japan.

SCHWARZ, D. 2000. A system for data-driven concatenative sound synthesis. In *Proceedings of the 3rd international conference on Digital Audio Effects*. Verona, Italy.

SCHWARZ, D. 2006. Concatenative sound synthesis: the early years. *Journal of New Music Research* 35, 3–22.

SCHWARZ, D. 2012. The sound space as musical instrument: playing corpus-based concatenative synthesis. In *Proceedings of the 12th international conference on New Interfaces for Musical Expression*. Ann Arbor, US.

SCHWARZ, D., BELLER, G., VERBRUGGHE, B. AND BRITTON, S. 2006. Real-time corpus-based concatenative synthesis with catart. In *Proceedings of the 9th international conference on Digital Audio Effects*, pp. 279–282. Montreal, Canada.

SEBER, G. A. F. 1984. *Multivariate Observations*. John Wiley & Sons, Inc.

SELFRIDGE, O. G. 1959. Pandemonium: a paradigm for learning. In *Proceedings of the Symposium on Mechanisation of Thought Processes*, pp. 511–529. London, UK.

SENEFF, S. 1988. A joint synchrony/mean-rate model of auditory speech processing. *Journal of Phonetics* 16, 55–76.

SHAWE-TAYLOR, J. AND CRISTIANINI, N. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

SHNEIDERMAN, B. AND PLAISANT, C. 2010. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley.

SMITH, B. D. AND GARNETT, G. E. 2011. The self-supervising machine. In *Proceedings of the 11th international conference on New Interfaces for Musical Expression*. Oslo, Norway.

SMITH, B. D. AND GARNETT, G. E. 2012. Unsupervised play: machine learning toolkit for max. In *Proceedings of the 12th international conference on New Interfaces for Musical Expression*. Ann Arbor, US.

SMITH, J. O. 1991. Viewpoints on the history of digital synthesis. In *Proceedings of the 1991 International Computer Music Conference*. Montreal, Canada.

SMITH, J. O. 1992. Physical modeling using digital waveguides. *Computer Music Journal* 16, 74–91.

SNYDER, J. AND MCPHERSON, A. 2012. The JD-1: an Implementation of a hybrid keyboard/sequencer controller for analog synthesizers. In *Proceedings of the 12th international conference on New Interfaces for Musical Expression*. Ann Arbor, US.

SOFIANOS, S., ARIYAEEINIA, A. AND POLFREMAN, R. 2010. Singing voice separation based on non-vocal independent component subtraction and amplitude aiscrimination. In *Proceedings of the 13th international conference on Digital Audio Effects*. Graz, Austria.

STEVENS, K. 1971. Sources of inter- and intra-speaker variability in the acoustic properties of speech sounds. In *Proceedings of the 7th International Congress on Phonetic Sciences*, pp. 206–232. Montreal, Canada.

STEVENS, S. S. 1936. A scale for the measurement of a psychological magnitude: loudness. *Psychological Review* 43, 405–416.

STOWELL, D. 2008. *Characteristics of the beatboxing vocal style*. Queen Mary University of London.

STOWELL, D. 2010. 'Making music through real-time voice timbre analysis: machine learning and timbral control'. Ph.D. Thesis, Queen Mary University of London.

STOWELL, D. AND PLUMBLEY, M. D. 2008. Robustness and independence of voice timbre features under live performance acoustic degradations. In *Proceedings of the 11th international conference on Digital Audio Effects*. Espoo, Finland.

STOWELL, D. AND PLUMBLEY, M. D. 2010. Timbre remapping through a regression-tree technique. In *Proceedings of the 7th Sound and Music Computing international conference*. Barcelona, Spain.

SUMMERFIELD, Q. AND ASSMANN, P. F. 1989. Auditory enhancement and the perception of concurrent vowels. *Perception & Psychophysics* 45, 529–536.

SUNDBERG, J. 1987. *The Science of the Singing Voice*. Northern Illinois University Press.

SUNDBERG, J. 2001. Level and center frequency of the singer´s formant. *Journal of Voice* 15, 176–186.

SZÉKELY, G. J., RIZZO, M. L. AND BAKIROV, N. K. 2007. Measuring and testing dependence by correlation of distances. *The Annals of Statistics* 35, 2769–2794.

TANAKA, A. 2000. Musical performance practice on sensor-based instruments. In *Trends in Gestural Control of Music*, pp. 389–405. Paris, France: IRCAM Centre Pompidou.

TANAKA, A. AND KNAPP, R. B. 2002. Multimodal interaction in music using the electromyogram and relative position sensing. In *Proceedings of the 2nd international conference on New Interfaces for Musical Expression*, pp. 1–6. Dublin, Ireland.

TANDON, A., AHMAD, M. O. AND SWAMY, M. N. S. 2004. An efficient, low-complexity, normalized LMS algorithm for echo cancellation. In *Circuits and Systems, 2004. NEWCAS 2004. The 2nd Annual IEEE Northeast Workshop on*, pp. 161–164.

TENENBAUM, J. B., SILVA, V. AND LANGFORD, J. C. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 2319.

THOMAS, S. P. AND POLLIO, H. R. 2004. *Listening to Patients: A Phenomenological Approach to Nursing Research and Practice*. 1st Edition. Springer Publishing Company.

TOGNERI, R., ALDER, M. D. AND ATTIKIOUZEL, Y. 1992. Dimension and structure of the speech space. *IEEE Proceedings I (Communications, Speech and Vision)* 139, 123–127.

TOMPKINS, D. 2011. *How to Wreck a Nice Beach: The Vocoder from World War II to Hip-Hop, The Machine Speaks*. Stop Smiling Books.

TRAUNMÜLLER, H. 1990. Analytical expressions for the tonotopic sensory scale. *The Journal of the Acoustical Society of America* 88, 97–100.

TRUEMAN, D., COOK, P. R., SMALLWOOD, S. AND WANG, G. 2006. PLOrk: the Princeton laptop orchestra, year 1. In *Proceedings of the 2006 International Computer Music Conference*, pp. 443–450. New Orleans, US.

TURETSKY, R. J. AND ELLIS, D. P. W. 2003. Ground-truth transcriptions of real music from force-aligned MIDI syntheses. In *Proceedings of the 4th international conference on Music Information Retrieval*. Baltimore, US.

USHER, J. 2006. Extraction and removal of percussive sounds from musical recordings. In *Proceedings of the 9th international conference on Digital Audio Effects*. Montreal, Canada.

VERTEGAAL, R. AND UNGVARY, T. 1995. The sentograph: input devices and the communication of bodily expression. In *Proceedings of the 1995 International Computer Music Conference*, pp. 253–256. Banff, Canada.

VINET, H., HERRERA, P. AND PACHET, F. 2002. The CUIDADO Project. In *Proceedings of 3rd international conference on Music Information Retrieval*. Paris, France.

VOGT, F., MCCAIG, G., ALI, M. A. AND FELS, S. 2002. Tongue 'n' groove: an ultrasound based music controller. In *Proceedings of the 2nd international conference on New Interfaces for Musical Expression*, pp. 1–5. Dublin, Ireland.

WAIBEL, A. AND LEE, K.-F. 1990. *Readings in Speech Recognition*. Morgan Kaufmann.

WAISVISZ, M. 1985. The hands: a set of remote MIDI-controllers. In *Proceedings of the 1985 International Computer Music Conference*. Burnaby, Canada.

WANDERLEY, M. M. 2001. 'Performer–instrument interaction: applications to gestural control of sound synthesis'. Ph.D. Thesis, University Paris.

WANDERLEY, M. M. AND DEPALLE, P. 2001. Gesturally-controlled digital audio effects. In *Proceedings of the 4th international conference on Digital Audio Effects*.

WANDERLEY, M. M. AND DEPALLE, P. 2004. Gestural control of sound synthesis. *Proceedings of the IEEE* 92, 632–644.

WANDERLEY, M. M. AND ORIO, N. 2002. Evaluation of input devices for musical expression: borrowing tools from HCI. *Computer Music Journal* 26, 62–76.

WANDERLEY, M. M., SCHNELL, N. AND ROVAN, J. B. 1998. Escher - modeling and performing composed instruments in real-time. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*vol. 2, pp. 1080 –1084.

WANG, G., ESSL, G. AND PENTTINEN, H. 2008. Do mobile phones dream of electric orchestras? In *Proceedings of the 2008 International Computer Music Conference*. Belfast, North Ireland.

WESSEL, D. 1979. Timbre space as a musical control structure. *Computer Music Journal* 3, 45–52.

WESSEL, D. 1991. Instruments that learn, refined controllers, and source model loudspeakers. *Computer Music Journal* 15, 82.

WESSEL, D. AND WRIGHT, M. 2001. Problems and prospects for intimate musical control of computers. In *Proceedings of ACM Computer-Human Interaction Workshop on New Interfaces for Musical Expression*. Seattle, USA.

WIERENGA, R. 2012. A new keyboard-based, sensor-augmented instrument for live performance. In *Proceedings of the 12th international conference on New Interfaces for Musical Expression*. Ann Arbor, USA.

WIFFEN, P. 1988. Keyboards are for wimps. *Keyboard Magazine*, 117.

WILSON, S., GUREVICH, M., VERPLANK, B. AND STANG, P. 2003. Microcontrollers in music HCI instruction: reflections on our switch to the Atmel AVR platform. In *Proceedings of the 3rd international conference on New Interfaces for Musical Expression*, pp. 24–29.

WISHART, T. 1996. *On Sonic Art*. Harwood Academic Publishers.

WRIGHT, M. AND FREED, A. 1997. Open sound control: a new protocol for communicating with sound synthesizers. In *Proceedings of the 1997 International Computer Music Conference*. Thessaloniki, Greece.

WYSE, L. 2003. Free music and the discipline of sound. *Organised Sound* 8, 237–247.

WYSE, L. 2005. Generative sound models. In *Multimedia Modeling*, pp. 370–377.

YANG, X., MILLAR, J. B. AND MACLEOD, I. 1996. On the sources of inter- and intra-speaker variability in the acoustic dynamics of speech. In *PROCEEDINGS OF THE 4TH International Conference on Spoken Language*vol. 3, pp. 1792–1795. Philadelphia, US.

YAN, Q. AND VASEGHI, S. 2003. Analysis, modelling and synthesis of formants of British, American and Australian accents. In *Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing*vol. 1, pp. 712–715. Hong Kong.

ZADEL, M. AND SCAVONE, G. 2006. Different strokes: a prototype software system for laptop performance and improvisation. In *Proceedings of the 6th international conference on New Interfaces for Musical Expression*, pp. 168–171. Paris, France.

ZAHORIK, P., BRUNGART, D. S. AND BRONKHORST, A. W. 2005. Auditory distance perception in humans: a summary of past and present research. *Acta Acustica united with Acustica* 91, 409–420.

ZBYSZYŃSKI, M., ZICARELLI, D. AND COLLECCHIA, R. 2013. fzero~: fundamental estimation for Max 6. In *Proceedings of the 2013 International Computer Music Conference*. Perth, AU.

ZUMBO, B. D. AND ZIMMERMAN, D. W. 1993. Is the selection of statistical methods governed by level of measurement? *Canadian Psychology/Psychologie Canadienne* 34, 390–400.

ZWICKER, E. 1961. Subdivision of the audible frequency range into critical bands (frequenzgruppen). *The Journal of the Acoustical Society of America* 33, 248.

Appendix A

DMI Evaluation Detailed Results

DMI ID	**D*** Isomap % tot. variance		% param **D*** sonic control		**d** NN pairs average normaliz. dist. in **D**			**d** NN pairs average normaliz. dist. in **I**			IDW in **D** and **D*** difference	
	2D	3D	2D	3D	2D	3D	Full	2D	3D	Full	2D	3D
1	81.30	88.12	66.20	99.95	0.004	0.008	0.042	0.127	0.097	0.066	0.199	0.148
2	97.34	98.19	70.68	99.00	0.004	0.005	0.016	0.285	0.253	0.178	0.387	0.322
3	94.36	95.74	50.31	82.81	0.003	0.003	0.022	0.222	0.205	0.179	0.328	0.299
4	99.87	99.93	38.80	89.48	0.000	0.000	0.000	0.195	0.181	0.136	0.189	0.141
5	79.53	85.72	57.79	92.85	0.004	0.008	0.039	0.275	0.236	0.175	0.293	0.241
6	94.16	95.94	95.08	99.81	0.014	0.015	0.024	0.165	0.147	0.107	0.164	0.155
7	69.69	83.44	87.81	99.90	0.012	0.021	0.031	0.068	0.055	0.045	0.130	0.115
8	96.02	97.30	67.11	94.93	0.003	0.004	0.015	0.197	0.176	0.136	0.351	0.318
9	96.89	98.49	80.09	99.85	0.005	0.006	0.012	0.194	0.186	0.156	0.384	0.280
10	97.57	98.31	46.88	93.27	0.002	0.003	0.027	0.169	0.153	0.126	0.162	0.150
11	96.74	97.62	70.62	96.75	0.004	0.004	0.025	0.222	0.218	0.177	0.256	0.249
12	78.75	89.17	100	100	0.030	0.035	0.053	0.117	0.109	0.094	0.176	0.222
13	70.60	78.82	48.23	98.37	0.004	0.010	0.054	0.234	0.189	0.121	0.360	0.302
14	60.97	66.70	68.17	99.95	0.004	0.009	0.099	0.116	0.090	0.083	0.198	0.221
15	94.99	97.13	25.26	89.58	0.002	0.002	0.031	0.227	0.227	0.097	0.924	0.909
16	90.45	93.05	97.14	100	0.016	0.019	0.055	0.106	0.087	0.064	0.138	0.081
17	79.76	83.28	79.16	100	0.006	0.016	0.057	0.135	0.076	0.068	0.244	0.235
18	80.83	84.34	94.97	100	0.011	0.014	0.084	0.154	0.139	0.140	0.270	0.151
19	48.36	56.57	56.03	86.01	0.004	0.010	0.174	0.149	0.128	0.118	0.543	0.084
20	81.71	84.26	94.22	100	0.009	0.015	0.055	0.184	0.183	0.169	0.376	0.229
21	92.93	94.71	28.06	56.56	0.001	0.003	0.014	0.246	0.237	0.221	0.331	0.184
22	98.65	99.00	34.07	77.58	0.002	0.002	0.001	0.181	0.169	0.136	0.068	0.057
23	84.55	91.16	70.50	96.70	0.008	0.008	0.016	0.154	0.146	0.124	0.272	0.240
24	87.65	92.39	100	100	0.038	0.052	0.054	0.172	0.110	0.094	0.306	0.258
25	97.38	98.14	79.30	99.42	0.006	0.006	0.003	0.165	0.161	0.149	0.096	0.096
26	90.32	93.91	87.04	99.63	0.007	0.006	0.019	0.096	0.094	0.076	0.236	0.249
27	67.28	83.02	70.68	99.54	0.004	0.005	0.037	0.174	0.149	0.114	0.385	0.201
28	99.69	99.84	59.79	84.66	0.004	0.003	0.001	0.066	0.067	0.068	0.050	0.027
29	80.97	92.84	58.94	100	0.004	0.011	0.023	0.165	0.134	0.082	0.281	0.219
30	99.69	99.84	59.79	84.66	0.004	0.003	0.001	0.066	0.067	0.068	0.050	0.027
31	42.59	60.40	74.31	83.33	0.008	0.013	0.015	0.170	0.135	0.114	0.328	0.399
32	13.70	20.27	54.29	70.48	0.009	0.010	0.004	0.123	0.094	0.063	0.325	0.333
33	61.79	62.60	43.18	75.45	0.000	0.000	3E-05	0.166	0.165	0.162	0.205	0.206
34	100	100	93.91	100	0.001	0.000	0.000	0.299	0.292	0.292	0.045	0.026

Table A.1: Detailed numerical results of the Section 4.5 validation including percentage of total Isomap components variance, sonic control obtainable parameter combinations %, timbre descriptor entries nearest neighbors average distance in D and I spaces, output parameters IDW interpolation difference with weights computed in the original and reduced sonic spaces, for the 2D and 3D Isomap lower dimensional spaces.

DMI ID	uniformity covariance λ					redistrib. % full errors		ANN hidden layers neurons and MSE				% param. m(**Du***)		% param. **d***=m(**gc**) sonic control		linear descriptror response residual	
	D Full	**D*** 2D	**D*** 3D	**Du*** 2D	**Du*** 3D	2D	3D	cfg 2D	MSE 2D	cfg 3D	MSE 3D	2D	3D	2D	3D	2D	3D
1	0.244	0.511	0.685	0.053	0.048	0.003	0.003	20-7	0.4	18-5	0.1	58.3	67.1	94.7	94.8	0.037	0.058
2	0.758	0.925	1.095	0.053	0.048	0.060	0.055	18-4	9.6	18-5	2.4	33.8	52.1	75.3	93.6	0.188	0.257
3	0.499	0.853	1.097	0.057	0.054	0.047	0.055	17-8	3.5	13-3	0.9	38.3	50.3	82.4	86.0	0.153	0.203
4	0.740	1.232	1.465	0.050	0.058	0.034	0.017	20-3	2k	8-4	394	30.3	47.9	57.1	82.7	3.072	6.125
5	0.825	0.979	1.264	0.054	0.046	0.013	0.014	18-6	0.8	14-7	0.2	47.4	55.8	88.8	94.3	0.050	0.067
6	0.746	0.892	1.030	0.056	0.052	0.028	0.022	12-7	0.8	9-6	0.1	49.6	64.2	91.3	97.9	0.053	0.055
7	0.833	0.955	1.092	0.055	0.047	0.019	0.024	19-6	0.9	18-6	0.2	55.2	63.7	93.8	94.4	0.055	0.084
8	0.652	1.204	1.332	0.058	0.048	0.064	0.056	9-6	11	13-6	2.6	38.8	49.0	80.7	92.8	0.181	0.240
9	0.508	0.837	0.855	0.057	0.047	0.067	0.048	19-7	6.3	18-5	0.9	47.3	58.0	91.2	96.2	0.124	0.162
10	0.709	1.160	1.515	0.055	0.050	0.045	0.073	17-4	4.3	11-7	1.4	20.5	47.2	47.9	93.9	0.193	0.336
11	0.553	1.159	1.350	0.060	0.051	0.100	0.064	15-6	25	11-7	4.7	23.9	43.0	65.6	88.0	0.391	0.434
12	0.385	1.192	1.576	0.067	0.061	0.039	0.000	10-6	270	15-4	127	62.4	69.6	96.0	91.2	1.163	1.299
13	0.234	0.612	0.739	0.051	0.072	0.003	0.002	20-7	140	15-6	18	56.3	68.3	92.2	89.2	1.106	1.156
14	0.316	0.653	0.835	0.053	0.049	0.008	0.008	19-7	49	17-6	6.9	53.3	64.9	89.9	96.0	0.515	0.630
15	0.470	1.384	1.402	0.062	0.048	0.131	0.072	14-6	274	18-6	74	31.3	41.5	87.3	78.5	1.241	2.121
16	0.315	0.524	0.633	0.061	0.052	0.028	0.028	18-6	326	11-5	69	57.7	69.1	91.4	99.2	1.135	1.628
17	0.386	0.574	1.095	0.061	0.049	0.017	0.052	9-8	56	18-3	17	62.9	59.9	90.9	95.9	0.595	1.034
18	0.473	0.712	0.754	0.060	0.054	0.054	0.034	10-6	211	14-7	22	49.8	64.1	89.1	96.7	0.701	0.666
19	0.617	1.357	1.168	0.053	0.056	0.013	0.011	20-7	86	16-5	11	49.3	57.5	91.9	94.7	0.828	0.871
20	0.199	0.596	0.741	0.055	0.052	0.013	0.024	17-6	92	18-4	25	54.9	59.0	94.3	94.4	0.513	0.708
21	1.668	2.177	2.285	0.054	0.048	0.061	0.112	20-6	138	14-7	26	31.4	33.0	86.2	60.2	1.209	1.946
22	1.038	1.559	1.755	0.052	0.067	0.032	0.014	20-5	4k	17-5	994	40.2	54.8	77.1	87.5	4.098	10.41
23	0.995	1.452	1.564	0.058	0.048	0.076	0.023	11-7	28	18-6	6.2	38.1	53.6	82.0	93.2	0.351	0.343
24	0.340	0.426	0.538	0.066	0.054	0.006	0.019	16-3	4.8	17-3	1.5	67.9	77.8	98.0	96.4	0.117	0.126
25	1.327	1.647	1.865	0.060	0.055	0.041	0.051	13-5	19k	16-5	5k	52.5	66.8	93.3	96.8	11.75	19.09
26	0.318	1.100	1.174	0.080	0.056	0.209	0.085	17-7	13	13-7	4.9	33.7	47.8	91.5	94.4	0.157	0.327
27	0.335	1.035	1.165	0.063	0.052	0.083	0.092	18-5	194	16-4	51	42.0	49.5	87.7	88.3	0.760	0.704
28	1.470	1.482	1.483	0.060	0.055	0.056	0.027	11-8	∞	16-4	∞	50.0	50.3	100	81.0	52.43	104.5
29	0.297	0.490	0.849	0.050	0.048	0.003	0.002	19-7	0.2	15-0	0	64.3	68.9	93.2	94.2	0.031	0.053
30	1.470	1.482	1.483	0.060	0.055	0.056	0.027	11-8	∞	16-4	∞	50.0	50.3	100	81.0	52.43	104.5
31	0.612	1.865	2.189	0.064	0.050	0.206	0.140	17-6	0	5-0	0	38.2	36.8	71.5	71.5	0.008	0.017
32	1.028	1.803	1.982	0.083	0.050	0.696	0.712	7-0	0.01	5-0	0.02	30.5	17.5	74.3	54.3	0.001	0.004
33	1.448	9.108	9.479	0.068	0.058	0.095	0.050	6-4	∞	18-4	∞	16.8	7.7	70.5	17.3	930.2	1273
34	1.715	1.722	1.838	0.072	0.048	0.117	0.034	19-6	58k	18-3	8k	53.4	55.3	98.8	91.8	11.81	21.01

Table A.2: Detailed numerical results of the Section 5.3 validation including covariance λ, redistribution algorithm error %, ANN configuration and MSE, obtainable parameter combinations %, and estimation sonic response linearity for the 2D and 3D sonic spaces cases.

DMI ID	% parameters $d^*=m(gc)$						parameters continuity $d^*=m(gc)$						spread deviation $d^*=m(gc)$					
	2D			3D			2D			3D			2D			3D		
	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4
1	70	80	77	73	77	76	0.09	0.21	0.29	0.09	0.2	0.28	3.34	2.29	2.59	3.15	3.05	2.54
2	<1	94	89	<1	96	95	0.05	0.19	0.28	0.05	0.18	0.28	1222	9.94	67	1454	9.18	115
3	56	77	74	62	81	79	0.08	0.2	0.3	0.07	0.2	0.29	31.6	9.65	19.2	25.3	8.43	15.6
4	61	63	59	60	70	71	0.08	0.2	0.3	0.08	0.2	0.29	9.84	7.01	7.96	9.9	6.19	7.38
5	73	78	79	72	82	83	0.06	0.14	0.21	0.06	0.13	0.21	11.7	8.33	11	16	10.4	14.2
6	96	97	97	99	100	100	0.09	0.19	0.25	0.09	0.18	0.23	29.6	20.8	28.6	27	16.2	26.3
7	97	95	94	98	96	95	0.08	0.15	0.18	0.07	0.14	0.16	13.1	11.1	12.9	12	14.4	11.6
8	4	92	87	4	95	94	0.07	0.19	0.27	0.07	0.19	0.27	420	13	86.2	405	15.2	78
9	79	90	92	87	95	97	0.09	0.2	0.28	0.09	0.2	0.27	15.2	10.9	14.1	12.9	9.23	11.6
10	81	61	54	93	95	94	0.07	0.18	0.24	0.07	0.17	0.24	22	37.2	24.5	19.7	14.1	18.1
11	87	78	71	97	95	95	0.06	0.16	0.23	0.05	0.16	0.23	211	53.4	232	43.7	36	43.3
12	100	98	97	100	98	97	0.07	0.15	0.21	0.07	0.15	0.21	102	75.4	101	77	70	76.7
13	5	55	55	5	55	55	0.08	0.21	0.31	0.08	0.21	0.3	35	2.22	8.75	38.8	2.43	10.3
14	73	78	78	76	80	82	0.08	0.18	0.25	0.08	0.18	0.25	8.74	5.97	7.45	8.09	7.83	7.1
15	66	65	58	61	65	63	0.09	0.2	0.3	0.09	0.2	0.29	3.8	4.28	3.36	4.4	4.81	3.69
16	99	97	96	100	100	100	0.08	0.16	0.19	0.08	0.15	0.18	28.8	24	27.7	18.2	19.1	17.2
17	96	98	97	98	98	98	0.07	0.16	0.22	0.07	0.15	0.17	22.2	13.7	21.2	22.3	18.4	22.1
18	8	97	97	8	100	99	0.08	0.2	0.27	0.08	0.2	0.27	195	20.2	85.1	206	16.6	86.5
19	74	82	79	77	84	83	0.08	0.18	0.27	0.08	0.19	0.27	6.9	5.69	5.72	7.21	5.35	6.4
20	1	98	96	1	98	98	0.02	0.19	0.28	0.02	0.19	0.27	4064	14.6	255	3642	13.7	314
21	<1	4	72	<1	4	67	0	0.12	0.27	0	0.12	0.28	0	165	0	0	290	0
22	56	66	66	60	68	69	0.08	0.19	0.29	0.08	0.19	0.29	7.87	6.24	6.02	6.72	8.8	5.34
23	5	92	84	2	97	94	0.02	0.18	0.28	0.02	0.17	0.27	904	19.8	125	2087	15.4	262
24	100	100	100	100	99	100	0.09	0.17	0.21	0.08	0.16	0.19	58.1	35.8	57.7	43.4	25	42.4
25	99	99	96	99	99	99	0.08	0.16	0.21	0.07	0.16	0.21	32	33.8	31.6	28.1	24.3	27.5
26	100	94	93	100	100	99	0.09	0.18	0.23	0.08	0.16	0.19	43.1	79	42.8	33.1	55.3	32.6
27	<1	94	97	<1	100	99	0	0.17	0.29	0	0.17	0.28	0	36.2	371	0	24.8	0
28	94	97	97	91	88	88	0.09	0.17	0.22	0.08	0.17	0.23	45.7	40.4	45.2	44.3	46.1	43.6
29	62	73	76	62	70	73	0.09	0.2	0.29	0.09	0.2	0.28	3.13	2.17	2.37	3.24	3.62	2.54
30	94	97	97	91	88	88	0.09	0.17	0.22	0.08	0.17	0.23	45.7	40.4	45.2	44.3	46.1	43.6
31	96	83	80	96	85	81	0.08	0.17	0.23	0.08	0.17	0.22	61	70.5	62.3	61.3	64.9	63.2
32	85	83	73	87	67	54	0.04	0.1	0.18	0.05	0.11	0.17	193	152	205	142	189	113
33	90	66	60	70	44	30	0.08	0.18	0.25	0.08	0.18	0.25	181	252	181	181	431	253
34	92	97	98	95	98	98	0.1	0.2	0.28	0.1	0.2	0.28	9.2	8.48	8.69	9.82	8.51	9.51

Table A.3: Detailed numerical results of the Section 5.3 validation including percentage of obtained parameters, parameters continuity, and sonic space coverage spread for different values of i_{rad}. Results are derived from the emulation of the GC output fed to the ANN-based mapping function, for the 2D and 3D cases.

DMI ID	% parameters $du^*=gc$						parameters continuity $du^*=gc$						spread deviation $du^*=gc$					
	2D			3D			2D			3D			2D			3D		
	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4	irad 0.11	irad 0.25	irad 0.4
1	66	82	84	75	79	80	0.09	0.2	0.29	0.08	0.19	0.28	4.61	2.61	2.52	2.8	2.54	2.28
2	0.5	96	98	0.5	98	99	0.04	0.16	0.26	0.03	0.14	0.23	992	11.1	106	1183	11.4	123
3	61	84	91	55	85	89	0.06	0.19	0.29	0.05	0.17	0.28	7.21	5.25	5.65	10.7	7.15	7.76
4	55	76	82	62	75	80	0.08	0.19	0.28	0.06	0.18	0.27	6.31	4.25	5	7.87	6.63	5.72
5	84	95	97	80	91	95	0.06	0.14	0.22	0.05	0.13	0.21	7.28	4.71	6.48	10.7	6.41	9.35
6	95	100	100	99	100	100	0.09	0.18	0.24	0.06	0.16	0.22	33.9	17.7	32.7	26.7	13.9	24.8
7	98	100	100	99	100	100	0.07	0.15	0.19	0.06	0.14	0.17	15.5	5.93	14.2	11.5	9.97	11.1
8	4	98	100	4	99	99	0.05	0.17	0.26	0.04	0.15	0.24	330	10.5	78.9	312	11.3	77.3
9	73	94	99	86	98	99	0.08	0.19	0.26	0.05	0.16	0.24	18.3	12.1	16	13.1	10.6	12.3
10	90	99	100	95	99	99	0.06	0.17	0.24	0.05	0.15	0.23	12.1	8.33	11.3	13.7	9.19	12.5
11	95	99	100	96	99	100	0.05	0.16	0.24	0.05	0.14	0.22	29.9	25.1	28.7	28.1	21.5	27.8
12	100	100	100	100	100	100	0.06	0.15	0.2	0.05	0.13	0.18	108	58.5	108	119	35	119
13	5.3	55	57	5.5	55	58	0.08	0.21	0.31	0.05	0.2	0.29	43.7	2.36	5.81	28.4	2.68	7.06
14	78	89	94	78	85	91	0.07	0.18	0.25	0.07	0.17	0.25	5.3	3.93	4.31	8.36	6.79	6.53
15	57	82	85	61	76	78	0.07	0.19	0.29	0.05	0.18	0.27	4.95	2.67	3.81	5.38	3.37	4.28
16	99	100	100	100	100	100	0.07	0.15	0.19	0.05	0.14	0.18	33.6	15.1	33	20.2	15.1	19.4
17	95	100	100	99	100	100	0.06	0.16	0.22	0.06	0.14	0.17	22.7	8.02	21.5	19.3	11.5	18.3
18	8.3	100	100	8.3	100	100	0.05	0.18	0.26	0.05	0.16	0.24	241	19.8	99.5	284	18.7	108
19	60	92	95	74	90	93	0.07	0.19	0.27	0.06	0.18	0.26	8.4	4.18	6.69	7.94	4.74	6.14
20	0.7	96	100	0.7	100	100	0.02	0.18	0.28	0.01	0.15	0.25	2415	15.4	225	2740	13.8	123
21	<1	4	82	<1	4	82	0	0.1	0.28	0	0.1	0.26	0	89.6	106	0	111	0
22	57	75	83	56	72	79	0.07	0.19	0.29	0.06	0.17	0.28	6.12	4.51	4.57	9.07	8.13	6.01
23	5	98	100	5	98	100	0.03	0.17	0.27	0.04	0.14	0.25	375	16.2	87.9	787	18.5	173
24	100	100	100	100	100	100	0.08	0.17	0.21	0.06	0.15	0.19	69.2	29.4	68.1	38.9	20.3	38.2
25	96	100	100	99	100	100	0.07	0.15	0.2	0.05	0.14	0.19	42.3	20.6	41.1	35.3	25.8	34.5
26	100	100	100	100	100	100	0.07	0.16	0.22	0.05	0.15	0.21	43.4	26.7	42.6	60.1	28.8	58.7
27	0.2	97	100	0.2	99	100	0	0.14	0.26	0	0.12	0.23	0	21.6	63.8	0	19.5	495
28	83	100	100	100	100	100	0.08	0.17	0.22	0.07	0.17	0.22	66.3	29.1	62	49	19	48.1
29	57	76	80	61	71	76	0.09	0.2	0.3	0.09	0.2	0.28	3.95	2.04	2.79	3.9	2.86	3.03
30	83	100	100	100	100	100	0.08	0.17	0.22	0.07	0.17	0.22	66.3	29.1	62	49	19	48.1
31	99	100	100	98	100	100	0.06	0.16	0.23	0.05	0.14	0.21	52.2	28.2	51.1	50.7	28.7	44.5
32	99	100	100	100	100	100	0.05	0.13	0.21	0.04	0.11	0.19	44.2	39.8	43.8	42	35.1	41.6
33	98	100	100	100	100	100	0.07	0.16	0.21	0.05	0.14	0.19	65.1	35.3	64.4	63.6	36.5	62.7
34	73	94	99	90	98	100	0.08	0.18	0.27	0.05	0.16	0.25	17.5	10.6	15.7	16.2	19.4	15.3

Table A.4: Detailed numerical results of the Section 5.3 validation including percentage of obtained parameters, parameters continuity, and sonic space coverage spread for different values of i_{rad}. Results are derived from the emulation of the GC output directly projected in $\mathbf{D_u}^*$, for the 2D and 3D cases.