ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA DE TELECOMUNICACIÓN
UNIVERSIDAD DE MÁLAGA

TESIS DOCTORAL

INTERFACES AVANZADOS APLICADOS A LA INTERACCIÓN MUSICAL

ALEJANDRO ROSA PUJAZÓN

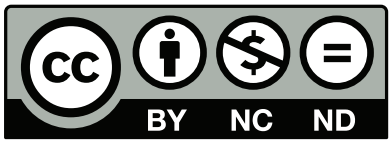
MÁLAGA 2014

UNIVERSIDAD DE MÁLAGA

Publicaciones y Divulgación Científica

AUTOR: Alejandro Rosa Pujazón

EDITA: Publicaciones y Divulgación Científica. Universidad de Málaga



Abstract

The latest advances in human-computer interaction technologies have brought forth changes in the way we interact with computing devices of any kind, from the standard desktop computer to the more recent smartphones. The development of these technologies has thus introduced new interaction metaphors that provide more enriching experiences for a wide range of different applications.

Music is one of most ancient forms of art and entertainment that can be found in humanity's legacy, and conforms a strong interactive experience on itself. The application of new technologies to enhance music computer-based interaction paradigms can potentially provide all sorts of improvements: providing low-cost access to music rehearsal, lowering knowledge barriers in regard to music learning, virtual instrument simulation, etc. Yet, surprisingly, there has been rather limited research on the application of new interaction models and technologies to the specific field of music interaction in regard to other areas.

This thesis aims to address the aforementioned need by presenting a set of studies which cover the use of innovative interaction models for music-based applications, from interaction paradigms for music learning to more entertainment-oriented interaction interfaces, such as virtual musical instruments, ensemble conductor simulation, etc. The main contributions of this thesis are:

- It is shown that the use of signal processing techniques on the music signal and music information retrieval techniques can create enticing interfaces for music learning. Concretely, the research conducted includes the implementation and experimental evaluation of a set of different learning-oriented applications which make use of these techniques to implement inexpensive, easy-to-use human-computer interfaces, which serve as support tools in music learning processes.
- This thesis explores the use of tracking systems and machine learning techniques to achieve more sophisticated interfaces for innovative music interaction paradigms. Concretely, the studies conducted have shown

that it is indeed feasible to emulate the functionally of musical instruments such as the drumkit or the theremin. In a similar way, it is shown that more complex musical roles can also be recreated through the use of new interaction models, such as the case of the ensemble conductor or a step-aerobics application.

- The benefits in using advanced human-computer interfaces in musical experiences are review and assessed through experimental evaluation. It is shown that the addition of these interfaces contributes positively to user perception, providing more satisfying and enriching experiences overall.
- The thesis also illustrates that the use of machine learning algoriths and signal processing along with new interaction devices provides an effective framework for human gesture recognition and prediction, and even mood estimation.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER

1

INTRODUCTION

Before the emergence of personal computing, human-computer interaction paradigms were mostly associated with a very specific typology of users, mainly information technology professionals or enthusiasts. However, around the 70s, the rise of personal software applications (text editors, interactive games, etc.) as well as the spread of the first personal computer systems expanded considerably the variability in the standard computer user's profile. This in turn highlighted the usability deficiencies in the human-computer interface at the time, thus indicating a necessity to improve the interaction models to provide a much more satisfying use and experience according to the particular purpose of each application.

Nowadays, the amount of different computer applications and fields is vast, and the constant evolution of technology has allowed for the development of new interactive hardware which provides interaction paradigms that potentially surpass the capabilities of the more conventional keyboard-mouse interface. However, no perfect interface model has been designed yet; instead, user-computer interface design is commonly found to be strongly linked to the nature of the application for which it was incepted. Furthermore, in most of cases, the de-facto interaction paradigm considered revolves around the use of standard mouse and keyboard inputs, and it is not further studied whether the use of a different, more specific design in the interface could actually bring an improvement over

the usability capabilities and the overall user experience.

Music is in itself a highly interactive activity, yet it demands a strong background in terms of both theory and practice for a user to be able to play a given instrument. The complexity in the act of playing an instrument is a strong handicap for a nave user, and the abstract concepts behind music theory also hindered the access to them for many potential students. However, every person can easily enjoy music, and indeed, music is an integral part of our everyday lives.

The application of new technologies for the development of new human-computer interaction paradigms for music interaction can help to drop these barriers, providing a much easier and accessible experience to both nave users and musicians alike. In addition, the development of intelligent applications for the analysis of the musical signal can also provide additional tools to support the learning processes of music theory and practice. Surprisingly though, little research has been conducted in this regard, therefore leaving many holes to fill in the study of human-computer interfaces for music interaction.

1.1 Objectives

This document aims to provide additional steps towards the research of interaction paradigms that help improve user experience in interactive applications for music, both from the perspective of offering new ways of interaction with music through the use of novel technologies as well as with the purpose of enhancing the learning of music theory and practice by means of active interaction. Concretely, it will advance on the works and research performed towards the following goals:

- The study of the different technologies and processing techniques for the development of interactive applications with music, with an emphasis in offering a real-time, seamless user experience. This involves the design of novel interaction metaphors that minimize motion hampering and intrusion levels for the user, and processing techniques that allow for the extraction of high level information from both user input and the musical structure of the audio signal

- The study and implementation of interactive applications from an explorative, entertainment-oriented perspective, in order to improve user experience and overall usability of the interaction paradigm

- The study and implementation of interactive applications geared towards the improvement of the learning process in music theory and practice, providing the means for both amateurs and novices to further improve their musical skills.

- The research and development of new interaction paradigms that surpass conventional human-computer interfaces, allowing for the implementation of new ways of musical expression and musical instrument simulation through the use of augmented reality applications.
- The evaluation of the different interaction paradigms studied, through exhaustive user studies and experimental methodology in order to corroborate whether the use of such paradigms entails an improvement towards the purpose of each application.

1.2 Document overlay and organization

This section depicts how the different chapters in this document have been structured according to the contents covered. Specifically:

- The second chapter is devoted to presenting the current state of the art in music interaction research, illustrating the most relevant studies in this field, as well as including examples of interactive music-based applications as well as the use of innovative interaction paradigms for new ways of musical performance.
- The third chapter is entirely dedicated to describing the works performed in the study of interactive applications for the purpose of improving music learning processes. This chapter will thus present different techniques that can be used to aid both naïve users and musicians in order to improve their performances, both in the form of support tools that guide the user during his or her performance and tools for automatic correction of the mistakes made in a given performance.
- The fourth chapter encompasses the research performed in the development of innovative human-computer interfaces and how the use of this new interaction paradigms can enhance user experience in interactive applications with music. Concretely, the chapter pays special attention to the application of motion capture and machine learning techniques for the implementation of advanced interaction models that surpass the capabilities of the standard human-computer interaction paradigm.
- The firth chapter addresses a less conventional interaction model: the study of emotional interfaces through the use of brain-computer interaction. In particular, the chapter delves into the study of EEG signals to classify user mood.
- Finally, the last chapter portrays the main conclusions extracted from the research performed, as well as proposing potential future lines to further study the applications and benefits of advanced interaction techniques in music applications.

CHAPTER

2

STATE OF THE ART

2.1 Interactive applications with music

As previously stated, Music is fundamentally a learning and entertainment activity, however, the amount of different applications that revolve around music interaction is surprisingly low.

In recent years, the genre of the so-called musical videogames illustrates the best known examples of interactive musical applications, ranging from singing karaoke applications, like Singstar, to very basic instrument simulators, like GuitarHero. The emergence of off-the-shelf technologies for motion tracking, such as Nintendo's Wiimote or Microsoft's Kinect has also allowed for the implementation of more sophistic gaming experiences mainly geared towards simple dance simulation (e.g. Let's Dance). Not surprisingly, there is some concern about whether these applications are too strictly game-oriented or not. This concern comes from the fact that, in many cases, these applications focus only on inputting basic commands (such as button pressing) according to a simple rhythmic sequence, which hardly respects the creative component of actually playing music (Grollmisch et al. [2009]), as this task is mostly reliant on dexterity and swift reflexes, whereas the capability to actively participate in music creation is only marginal. On the other hand, the economic and social impact of these games is quite high, they foster the

interest of children and adolescents in music, and recent studies corroborate that this kind of applications can contribute to developing some form of musical knowledge in children (Gower and McDowall [2012], Wang and Lai [2011]).

In order to achieve a more natural and novel interface that provides for higher levels of immersion and creativity, previous studies on music interaction have proposed the use of embodied metaphors (Antle et al. [2008]), connecting body motion to sound generation, and also showing that such interaction metaphors can indeed improve learning processes. Other studies (Khoo et al. [2008], Castellano et al. [2007]) have considered the use of such embodied metaphors for the implementation of explorative musical experiences, regulating acoustic parameters such as pitch, amplitude, tempo or time signature according to the detection of previously defined motion.

Some researchers have even investigated the possibility of modifying visual patterns according to speech or sung voice, effectively making users' voice visible (Levin and Lieberman [2004]), or even allowing motor-impaired users to performs actions such as drawing (Harada et al. [2007]).

Further research has been performed regarding the use of tangible interaction in order to manipulate the pitch, volume and tempo of ongoing tones (Bakker et al. [2011]), or haptic devices for aid in rhythmic patterns learning (Holland et al. [2010]). In this regard, there has been a remarkable focus in the use of advance technologies for motion tracking and their use for musical expression and musical instruments simulation and creation. Examples can range from specific implementations of new and innovatibe instruments, such as the Reactable (Jordà [2010]), to the use of off-the-shelf solutions for the implementation of new interaction paradigms for music performance and exploration, such as the previously mentioned Wiimote (Qin) and Kinect (Mandanici and Sapir [2012], Todoroff et al. [2011], Odowichuk et al. [2011], Yoo et al. [2011], Stierman [2012]) devices, and even regular mobile phones and smartphones (Essl and Rohs [2009], Halpern et al. [2011]).

Regarding musical instrument simulation, perhaps the most remarkable example can be found in drum-hitting simulation (Höofer et al. [2009], Ng [2004], Trail et al. [2012], Odowichuk et al. [2011]), yet some examples can be found of research performed to emulate other instruments, such as the piano (Zhao et al. [2010]) or the guitar (Hwang and Yang [2012]).

One example of a musical interaction metaphor that is inherently linked to human body motion is that of the orchestra conductor, yet surprisingly there are only a handful of studies that address conducting simulation through the use of advanced human-computer interfaces. In order to adequately coordinate and synchronize the performance of an ensemble, the conductor must gesturally guide his fellow musicians, giving indications

regarding how the nuances in dynamics and tempo change as the piece performed is being played. To this end, the conductor relies on hand and baton gestures to issue such indications. Several studies have focused on capturing the conductor's hand or baton motion through the use of infrared sensors (Morita et al. [1991], Borchers et al. [2004], Peng and Gerhard [2009], Lee et al. [2004]), while others have preferred the use of inertial trackers (Bakanas et al. [2012]) or the Wiimote (Bradshaw and Ng [2008], Nakra et al. [2009]). For the most part, previous research has focused solely on modifying the conducting tempo nuances according to the commands given by the user, yet there have been a few studies which have also considered some form of dynamics control through gesture detection (Borchers et al. [2004]) or heuristics (Baba et al. [2010]), providing a more complete experience.

In general terms, prior research delves in using advanced interfaces to provide more satisfying experiences to the user, offering a more natural interaction model and higher level of immersion, or in improving learning processes through the development of additional support tools. Thus, these two aspects are the main focus of the research reported in this document.

2.2 Gesture recognition

Advanced human-computer interfaces are usually designed as a very specific solution to a very concrete problem. This usually implies the design of specific hardware suited to this purpose, however it also implies that the devices used are either expensive, intrusive and/or bulky, or difficult to use outside their intended original application. On the other hand, more generic sensing devices can overcome these issues, yet they need to rely on a more profound analysis of user motion and the use of advanced motion recognition techniques in order to adequately recognize the different gestures a given user may signal.

One of the most typical solutions found in the bibliography for human gesture detection is that of Hidden Markov Models (HMM) (Lee and Kim [1999], Yoon et al. [2001], Kim et al. [2007], Mannini and Sabatini [2010], Jacob and Wachs [2013]). Through the use of HMMs, it is possible to extract commands from a given continuous stream of data, according to the gestures and behaviours encoded in that stream. This is achieved through the identification of the sequence of states associated to the motion represented in the observable data. The use of HMMs conveys some relevant issues; concretely, HMMs require large training databases, as well as reliable means to segment the different subsets of motion in the data stream that correspond to each command, thus becoming especially sensitive to segmentation errors (Calinon [2007], Bandera et al. [2012]). Examples of the

application of HMMs to human motion recognition can be found across a wide array of fields: gait identification (Cheng et al. [2008], Chen et al. [2009]), tennis stroke analysis (Yamato et al. [1992]), robot interaction/imitation (Calinon and Billard [2004], Asfour et al. [2008]), medical tools (Jacob and Wachs [2013]), etc.

Other approaches follow the use of Dynamic Programming techniques to match tracked trajectories to the ones in the training database (Chen et al. [2005], Croitoru et al. [2005]). The most commonly used technique (Bandera et al. [2012]) is Dynamic Time Warping (DTW) (Muhlig et al. [2009], Li and Greenspan [2011], Celebi et al. [2013], Bandera et al. [2009]), which allows for the matching of sequences of different lengths. Machine learning approaches have also been considered in previous works, such as Support Vector Machines (SVM) (Schuldt et al. [2004], Ardizzone et al. [2000], Cao et al. [2009]), Neural Networks (Stanton et al. [2012], El-Baz and Tolba [2013]), Logistic Regression (Itauma et al. [2012]), Principal Components Analysis or Linear Discriminant Analysis (Bandera et al. [2012]).

A common problem identified within most of the works previously mentioned is that they do not allow for a fast enough gesture recognition model, thus the response of the system is too slow for a response in real-time, severely hindering its potential use for the implementation of real-time musical interactive applications.

Also, as previously commented, the size of the database is determinant in defining the capacity for different gesture discrimination, specially in the case of HMMs, and more so for gestures that are very similar in execution among themselves. Some researchers have found Dynamic Programming techniques to allow for faster recognition speeds as well as a lower dependence on the size of the database (Bandera et al. [2009]), yet the delay introduced is still too high for time-critical interaction application, such as musical instrument simulation.

CHAPTER

3

INTERACTION WITH MUSIC AND LEARNING

3.1 Introduction

Music learning involves the use of abstract reasoning and concepts that are not usually known to most lay people. All of this can conform a strong handicap or deterrent for new students, specially at the early stages. On top of that, learning music requires not only regular study of music theory but constant practice and rehearsal as well, yet the traditional classroom scope is insufficient to address this issues completely, as the assessment of a given musical performance requires specialized and specific attention from one knowledgable reviewer.

The application of innovative interaction paradigms to music learning-oriented applications helps overcome part of these constraints. Particularly, interaction on its own can act as a strong motivator, as well as offer a more involving and personalized experience. Furthermore, through an interactive human-computer paradigm, it is possible to follow a learning-through-action model, lowering the barriers of the inherent abstract nature in music theory and offering an explorative and more satisfying experience.

The use of simple interaction models also allows for easy implementation of many

practice-oriented applications that can be used by any student at home without the need of a big investment in terms of hardware or materials. Concretely, by using a simple microphone and mathematically analysing the sampled musical signal, it is possible to define learning applications and support tools which are autonomous and do not require any kind of specialized attention from an expert reviewer or conductor.

This chapter delves in the use of these advanced signal processing techniques and their application to the development of interactive systems for music learning. In particular, this chapter presents the research performed in the development of this kind of application. First, the chapter addresses the use of these techniques to provide users assessment on their musical performance. After this, the focus is turned to consider the use of these techniques to provide students with a guide during their performance.

3.2 Support tools for the correction of user performance

When conceiving an interactive learning application, perhaps the most immediate implementation that comes to mind is that of a support tool for the automatic correction of a given performance. In extent, the purpose of this kind of application is to provide an assessment on the piece played by the user, and identify the points where mistakes were made or where further improvement is needed. This type of feedback is especially important in the early stages of musical education, as new students do not have yet developed the necessary listening skills to properly assess their performance on their own.

We have developed two systems aimed to the implementation of automatic correction of a given performance, and studied their application as support tools for learning. The first system focuses on the correction of polyphonic piano pieces, while the second is oriented to the early stages of musical education. Both systems, their implementations an the tests performed are described below.

3.2.1 Automatic correction of polyphonic piano performances

As previously indicated, the purpose of this system is to offer a support tool for piano students, allowing them to correct their mistakes when rehearsing without the need of having another musician acting as an external reviewer. Concretely, the system analyses a given musical piano polyphonic recording, and assesses its correctness. In order to do so, both the notes and figures played are segmented, identified and compared with the ones in the original score (previously known by the system), helping the student to find

the mistakes performed and the aspects that need improvement.

The system uses a MIDI score as a reference for the piece analysed, which is extracted from the data extracted from a WAV file sampled at 44100 Hz corresponding to the piece played; the tempo of the piece played must be input as an additional parameter. Then, the system divides the signal into temporal slots or "partitions", each of these corresponding to the time at which a given note is being played. Each of these partitions is analysed to extract the duration of the note(s) played, as well as the pitch values covered, taking into account that more than one note can occur at the same time interval.

After this analysis is completed, the resulting score from gathering this data is compared with the original score, and potential mistakes are detected and indicated. Both the analysis and correction stages will be depicted below, and the description of the system will end with a presentation of the results obtained from the tests performed.

3.2.1.1 Temporal segmentation and partition analysis

In order to segment the input signal into the aforementioned partitions, an onset detector (Bello et al. [2005], Boogaart and Lienhart [2009], Benetos and Dixon [2011]) was implemented. This onset detector analyses the energy signature of the input signal, using a sliding window procedure similar to the implementation in Barbancho et al. [2004] in order to detect energy peaks corresponding to note attack times. For each sliding window, the energy E_i is calculated as follows:

$$E_i = \sum_{j=x_i}^{x_i+L-1} \Big(y(j)\Big)^2, \tag{3.1}$$

where x_i is the index for the initial sample of window i, L is the length of the window in samples and $y(j)$ represents the jth-sample of the piano piece. The input energy signal was previously normalized to a maximum value of 1.

The onset detection algorithm implemented follows the scheme in Fig. 3.1. Essentially, a set of threshold values are defined to identify energy peaks that potentially represent an onset; μ_0 is used to filter energy peaks from noise, and the parameters μ_1 to μ_3 define conditions to address energy overlapping between consecutive note events, comparing the energy of a given window with the energy for the preceding and next windows.

The different parameters were empirically set to L=3000 samples, $\mu_0 = 0.7$ μ_1=6.05, μ_2=1.9 and μ_3=3. To account for the fact that a strong enough onset can mask neighbouring ones, a second search for masked onsets is subsequently performed, this time with $L = 2000$ samples and μ_1=5.5.

The piano signal is windowed without overlapping if no attacks are detected. However,

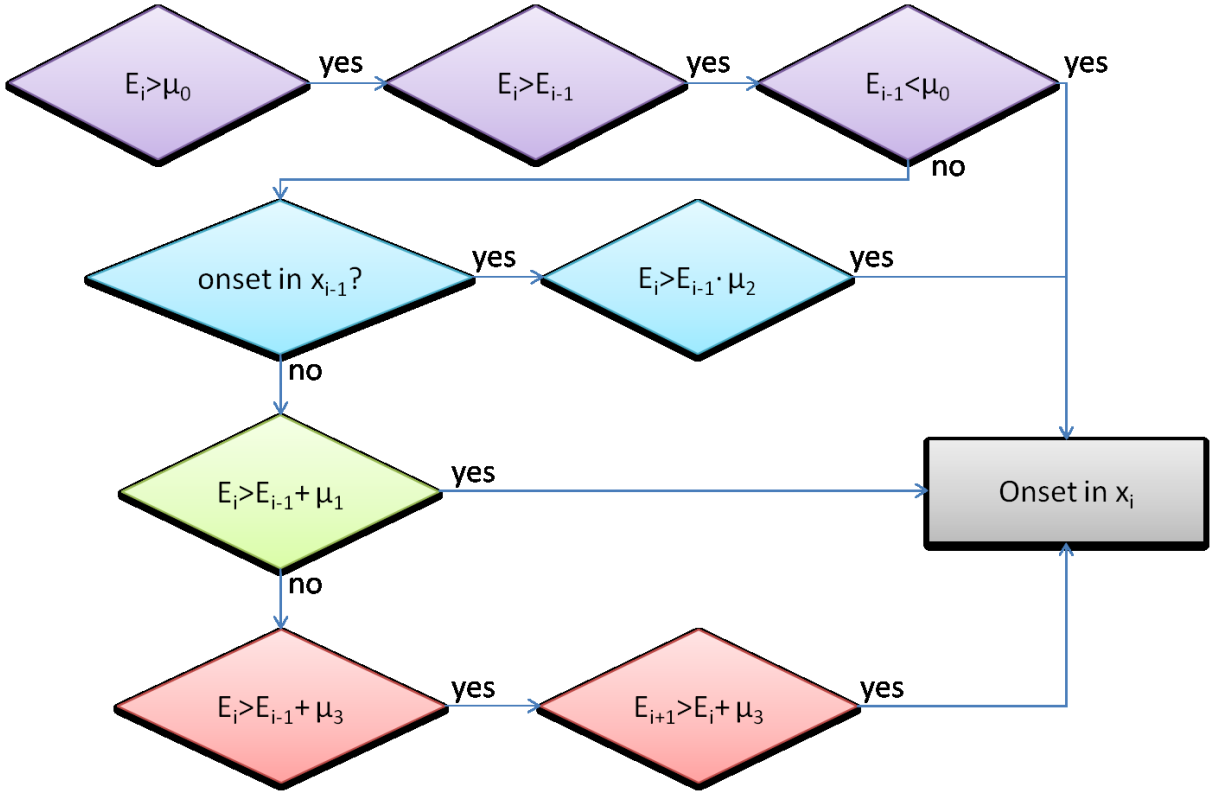


Figure 3.1: Onset detection process. E_i represents the energy for the i-th window, while the different μ_j values represent the thresholds used to find whether an onset is present or not.

if an attack is found in the ith-window, the next window is set to start 20% windows samples before the location of the maximum amplitude sample found in the ith-window.

Finally, in order to prevent a single note from generating more than one onset peak, a minimum distance between onsets is set, following the *ADSR* (attack-decay-sustain-release) model (Jensen [1999]), to two thirds of the shortest figure duration in the score. An example of the output of the onset detector implemented can be seen in Fig. 3.2

Once the onset times have been found, the signal is partitioned according to these onsets and the *ADSR* model. Each partition is set to start 1000 samples before a given onset time, and to end 3200 samples before the next onset takes place.

At a sampling frequency of 44100 Hz, these numbers guarantee isolation between the partitions found for the range of frequencies covered by a piano.

Partition analysis

After segmenting the signal into partitions, each partition is then subsequently analysed to find the pitch and the duration of its associated note.

The duration of the note is estimated by dividing the duration of each partition in

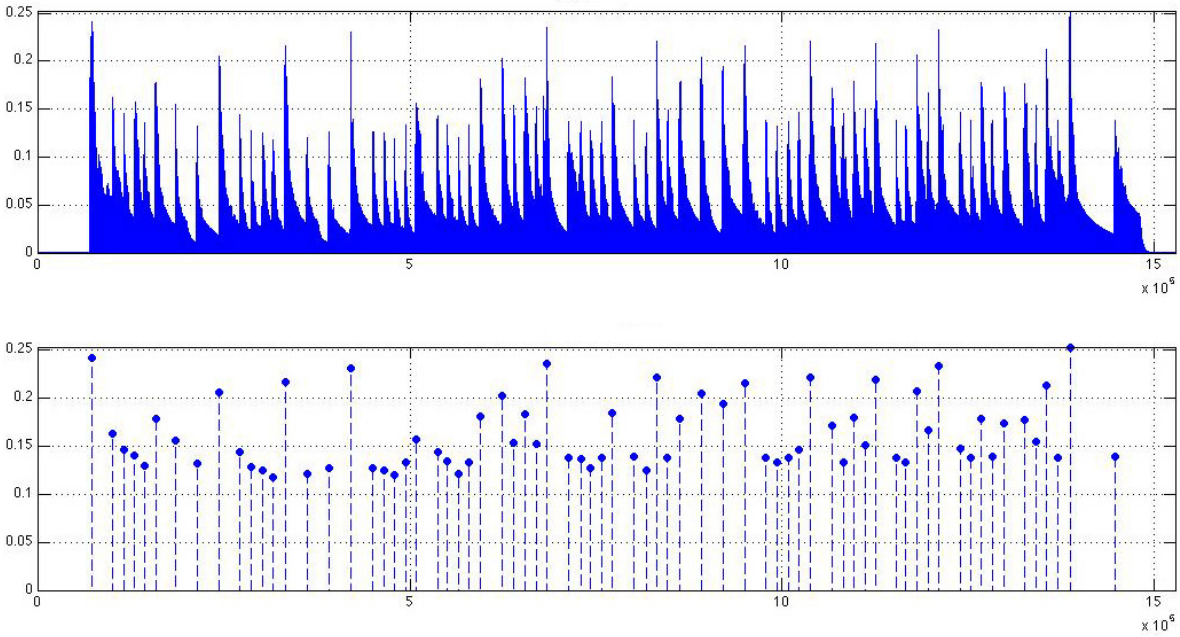


Figure 3.2: Input signal (up) and its corresponding detected onsets (down).

samples by the sampling frequency and the black figure's duration (inverse of the tempo). The resulting normalized duration is then associated to have the duration of the nearest figure (whole half, black, half-time, quarter-time, etc.); the presence of dotted notes is also considered in the duration detection analysis.

The pitch of the note in the partition is calculated by using a *DFT* (Discrete Fourier Transform), and finding the energy peaks in the subsequent spectrum. The pitch is then characterised by a pattern of normalized peaks with magnitude 1, placed in the frequencies corresponding to the real peaks detected, and a value of 0 for the rest of frequency values.

3.2.1.2 Assessment of correction

Using the reference MIDI score, the system knows beforehand the notes that are going to be played at each instant, as well as the duration of these notes. The fundamental and partial peaks from the MIDI spectrum are calculated and used to build a mask of narrow filters of width 1 (in the MIDI scale) centered at the frequencies were the fundamental and partial peaks should lie in theory. This mask is used to check whenever a given partition's peak pattern fits the corresponding MIDI spectrum in the reference score, and also provides the system with some resilience against inharmonicity errors (Barbancho et al. [2011]).

If the number of partitions detected and note times in the reference score are the same,

the duration of each partition is compared with the duration of the notes found for the corresponding note time. Similarly, the pitch peak patterns are compared by filtering the partition's pattern by the corresponding reference pattern mask (letting pass only the frequency values for which the mask is 0), and the resulting spectrum is summed along all frequencies. If both patterns match, the expected sum will be zero, otherwise the notes played were not correct (there are peaks in the partition spectrum that do not have their match in the reference pattern).

If the number of partitions and note times are not coincident, there are 3 possible cases in the assessment of a given partition:

- Partition's pitch pattern fits the expected one: in this case, the system proceeds as per the normal case.
- Pitch patterns do not match and there are more notes in the recording than in the score: this means that new notes have been played or one note has been played more than once. The pitch pattern of the current partition is compared with both the corresponding pattern mask and the next one. If the current pattern matches the next expected notes in the reference score, it is then assumed that the user played a new non-existent note in the current partition. Otherwise, it is assumed that the user simply played a wrong note.
- Pitch patterns do not match and there are less notes in the recording than in the score: this covers the case in which the user skipped some notes in his performance. Again, the current partition's pattern is compared with the corresponding and next pattern masks. If the partition fits the next expected pattern, it is assumed that a note was skipped, otherwise it is again assumed that the user played the note wrongly.

3.2.1.3 Tests and results

The onset detector implemented was tested using a reference score of ten quavers or quarter-notes played at different tempos, ranging from 40 to 230 beats-per-minute. The performance indicators considered for the tests conducted were: the rate of detected notes over the total number of notes played (denoted by N), the rate of false notes detected (false negatives, FN), and a score indicator combining the previous indicators with the false positives ratio (FP), defined as follows:

$$Score = \frac{N}{N + FP + FN} \times 100 \tag{3.2}$$

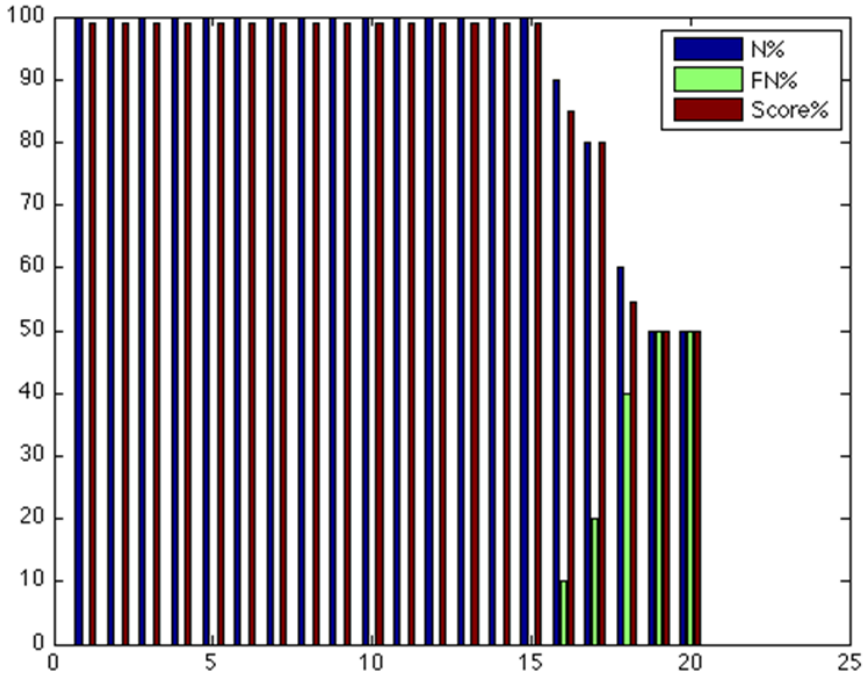


Figure 3.3: Results for the evaluation of the onset detector. N = the rate of detected notes over the total number of notes, FN = the rate of false negatives), $Score = \frac{N}{N+FP+FN} \times 100$ with FP being the rate of false positives.

The results yielded are summarized in Fig. 3.3. In general terms, the onset detector implemented is very effective for tempos lower than 180 bpm, the quality of the detection worsening gradually otherwise.

To assess the usefulness of the system implemented as a support tool for aided rehearsal, we conducted an experiment in which the song 'menuet 114' was played and recorded at 21 different speeds. The feedback on the user's performance was presented visually, using a coloured-coded image which indicates the types of errors detected using different colours. A sample of results can be found in Fig. 3.4. Most of the mistakes the users performed came from not keeping the duration of the figures as indicated in the score, and in general users performed better at lower tempos. This is a logical result to had, as it just illustrates that, at higher tempos, it becomes harder to keep a perfect performance, specially with regards to keeping the exact duration of th notes as intended in the original score.

The results found show that the system implemented does indeed allow to interact with user's performances, as well as to provide them with a detailed evaluation of the mistakes performed. While there are some limitations to keep in mind (namely the fact that the

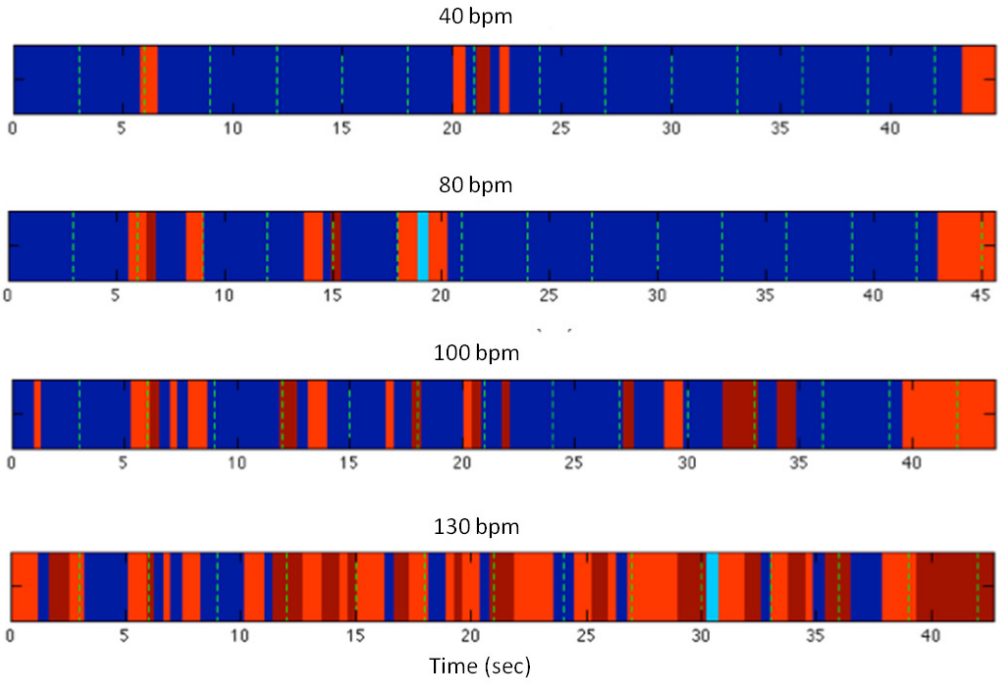


Figure 3.4: Evaluation of different performances: each bar color is associated to one type of error, i.e. dark blue - no error, orange - wrong duration of note, red - wrong note, cyan - skipped note.

onset detector loses accuracy for very high tempo values over 180 beats-per-minute), the research performed shows that it is possible to interact with piano performances following the approach proposed, thus offering users a powerful tool to aid in the correction and improvement of their performances and musical skills.

3.2.2 Robot-based support tool for solfeo learning

One of the basic subjects in all the stages of musical education and specially in the early stages is that of solfeo: being able to read and listen a score both in terms of rhythm and notes is a fundamental skill that requires constant practice to develop. Unfortunately, assessing whenever a student is properly reading a given score requires the assistance of an external musician. To address this need, we developed a system for Solfeo learning, SolfaBOT®, which allows the user to perform three types of training tasks:

- Rhythm reading: the user repeats a rhythmic pattern read in a score, and the system evaluates his performance
- Melody reading: the user reads a sequence of notes in a score, and the system assesses whether the notes were properly intoned or not.

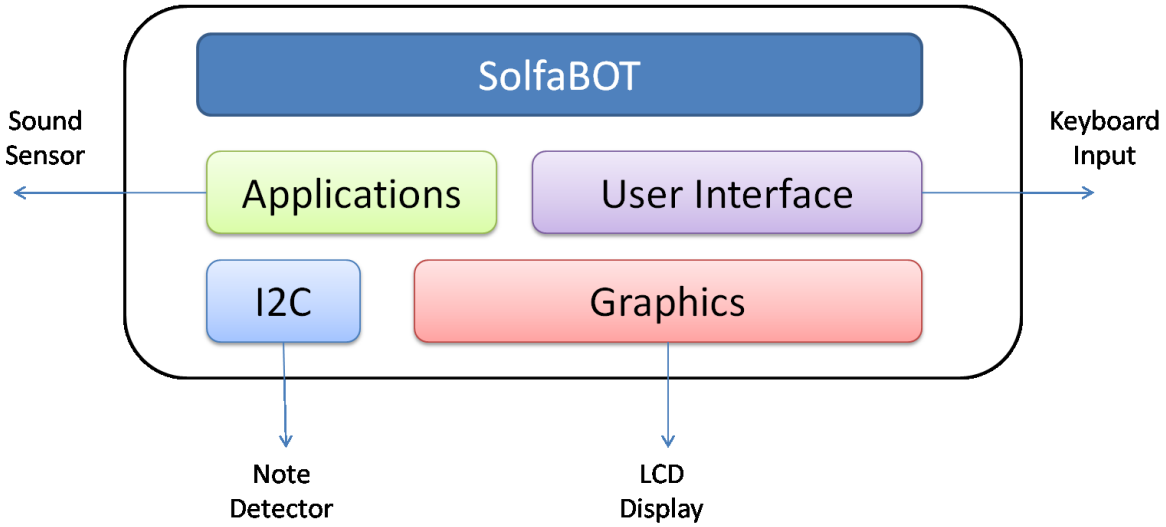


Figure 3.5: Overview of the system implemented.

- Music dictation: the system plays a given score for the user to listen to, so that he can write down the corresponding score.

In order to implement these functionalities, a Lego Mindstorms NXT 2.0 kit was selected as the hardware basis of the SolfaBOT® system. The application was coded using NXC, as this language is generates a much more efficient code memory-wise than other alternatives (Hansen [2007]). The system uses a 100×64 display to provide graphical information about the score. The system implemented followed the schema represented in Fig. 3.5

The Applications module implemented the three training exercises considered in the cases of use, the Graphics module allows for the dynamic rendering of the score in each exercise, and the User Interface module provides the other modules with access to the keyboard-input commands.

Rhythm detection and musical detection can be directly implemented at the application level using the NXT kit. However, the available commercial options for pitch detection proved to be insufficiently accurate for the task at hand. Thus, we developed an intelligent low-cost note detection sensor that allows for pitch detection and note discrimination in real time; this sensor communicates with the rest of the system though an I2C interface. An image of the robot alongside the sensor can be found in Fig. 3.6

3.2.2.1 Intelligent note-detection sensor

Fig. 3.7 presents an overview of the block structure of the note detector implemented, grouped into two main stages: a sampling stage and a processing stage.

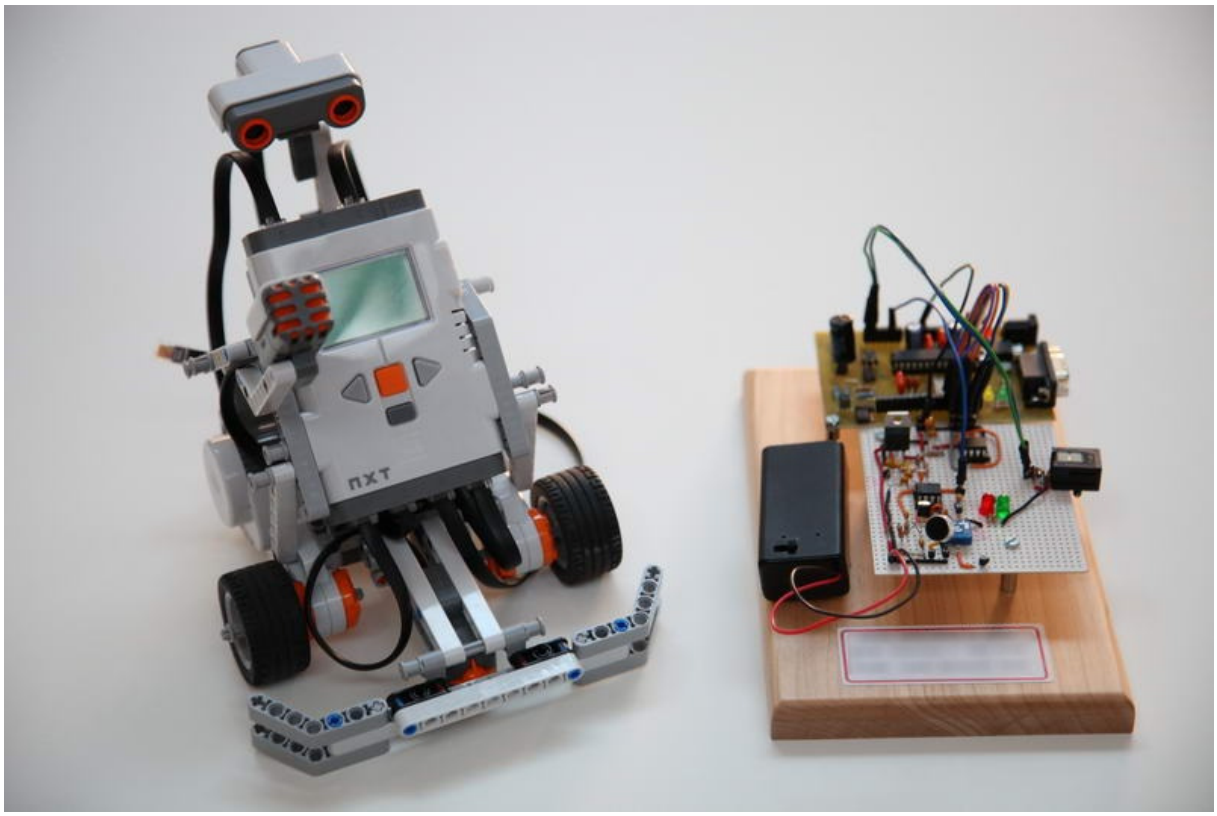


Figure 3.6: Capture of the intelligent sensor and the Lego Mindstorms NXT station.

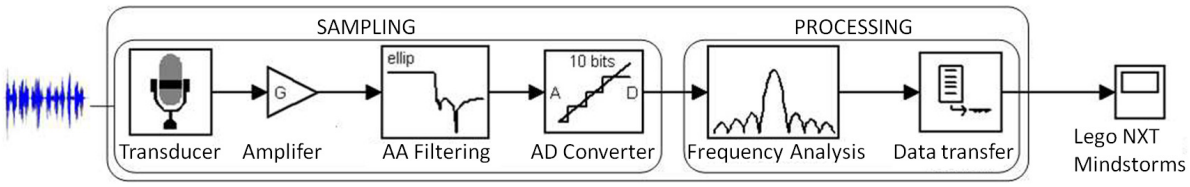


Figure 3.7: Intelligent note-detection sensor components.

The sampling stage consists of a set of components which capture the analog audio signal and output a digital equivalent signal. The processing stage analyses this digital signal to extract the frequency components and stores them in order to send them to the next processing stage.

The overall system is intended for use in a solfeo learning scenario, and thus, the range of notes detected must fit the frequency range of the human voice and discriminate the different notes in that range (this requirements are illustrated in Table 3.1). In addition, the delay in the system response should not exceed 2 seconds, in order to not stymie the overall user experience.

The AD converted chosen had thus a frequency bandwidth of 100Hz to 16 KHz, and the amplification stage was implemented with a common-emitter configuration. A filter

Table 3.1: Frequency requirements for the intelligent sensor.

Frequency range	[261.63 - 1975.50]Hz
Frequency resolution	10 Hz

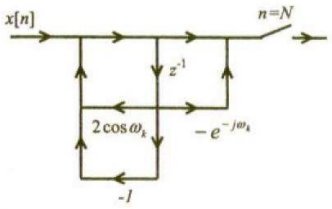


Figure 3.8: Block diagram for Goertzel algorithm.

with a cut-off frequency of 2 KHz was selected for the anti-aliasing component.

The processing stage of the intelligent sensor was implemented using an Arduino platform, concretely the Arduino Severino version 3 with the ATmega168 microcontroller, which also provides an I2C interface.

In order to analyse the input signal and extract its spectral components, it was determined that the FFT (Fast Fourier Transform) algorithm was not a good solution, as the limited processing capabilities of the Arduino micro-controller (8 bits at 16 MHz) did not allow to fulfill the requirement of system response time below 2 seconds, given that the standard FFT algorithm has a complexity of $\Theta(NlogN)$ (Oppenheim et al. [1999]). Instead, Goertzel algorithm is used (Goertzel [1958]). This algorithm calculates a localized part of the frequency spectrum of the signal, with a complexity of $\Theta(N)$ operations (concretely, $2N+2$ products and $4N-2$ sums), therefore offering a much faster response. A graphical layout of the Goertzel algorithm block diagram can be found in Fig. 3.8.

The transfer function for the Goertzel algorithm is as follows (Oppenheim et al. [1999]):

$$H_k = \frac{1 - e^{-jw_k} z^{-1}}{1 - 2cos\left(\frac{2\pi k}{N}\right) z^{-1} + z^{-2}} \tag{3.3}$$

where $w_k = 2\pi\frac{k}{n}$, $x[n]$ is the digitalized input signal, N is the number of samples of signal $x[n]$, and k is an integer that indexes the corresponding spectral coefficient, calculated as follows:

$$k = \left\lfloor 0.5 + \frac{N f_{target}}{f_s} \right\rfloor \tag{3.4}$$

where f_{target} is the frequency component to analyse, f_s represents the sampling frequency, and the $\lfloor\rfloor$ operator outputs the integer part of the equation specified.

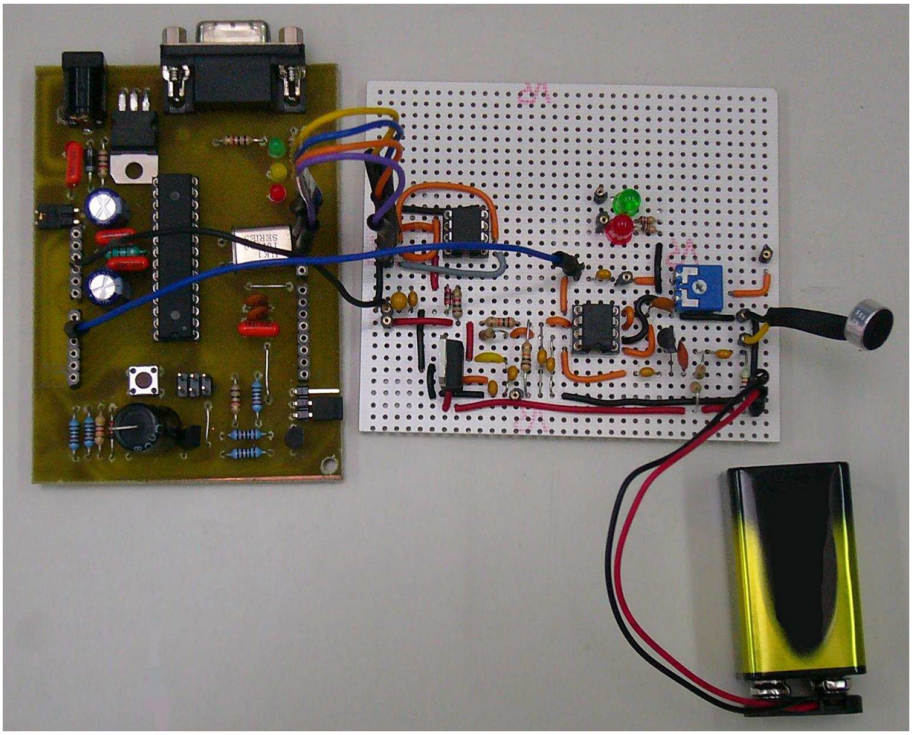


Figure 3.9: Intelligent note-detection sensor prototype.

In the case of the SolfaBOT® system, the sampling frequency was set at $f_s = 4$KHz, as our anti-aliasing filter has been defined with a bandwidth of 2KHz, and since the required frequency resolution as per Table 3.1 is 10Hz, a total of $N = 400$ audio samples are needed. Given the frequency range considered in Table 3.1, a total of 36 spectral coefficients are calculated (corresponding to 3 musical scales of 12 notes each, concretely the 4th, 5th and 6th scales). A picture of the final prototype can be found in Fig. 3.9

3.2.2.2 Integration and validation tests

Each component of the intelligent sensor implemented was tested separately to empirically verify that they meet the specifications. The components pertaining to the sampling stage (amplifier, anti-aliasing filter, AD converter) were tested using a tone generator and an oscilloscope, and all of them worked as expected.

A set of samples of flute recorder sounds extracted from the RWC database (Goto et al. [2004]) corresponding to the 4th, 5th and 6th musical scales were used to test the frequency analysis stage. The same analysis was performed in MATLAB using both the Goertzel and FFT algorithms. An example of this comparison can be found in Fig. 3.10 for the note D5. It can be checked that the output of the frequency analyser implemented is similar to the one given by the MATLAB simulations.

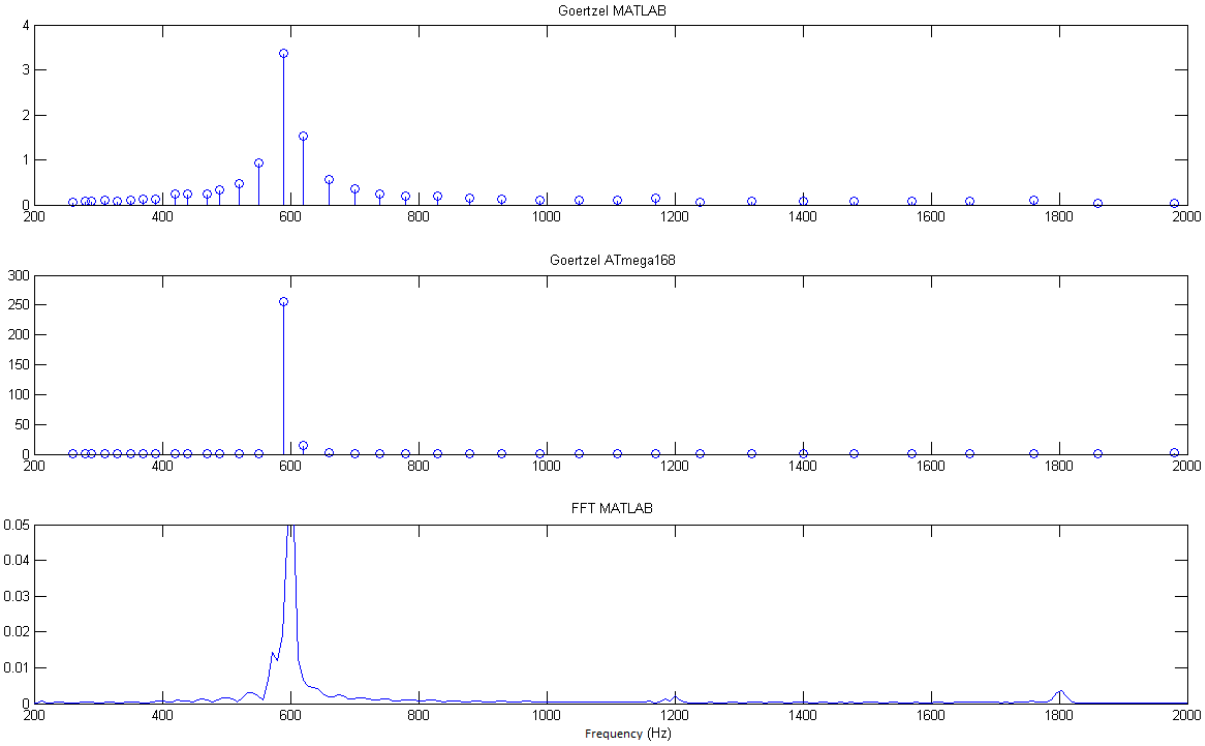


Figure 3.10: Frequency analysis comparison for note D5. The middle graph shows the output of the component implemented. The upper graph corresponds to the Goertzel algorithm output from MATLAB, and the lower graph represents de FFT spectrum of the note.

Table 3.2: Note detection for notes played with a piano.

Note	frequency(Hz)	detected freq (Hz)	Error
C5	523,25	493	5,78%
C5#	554,37	587	-5,89%
D5	587,33	587	0,06%
D5#	622,25	622	0,04%
E5	659,26	659	0,04%
F5	698,46	698	0,07%
F5#	739,99	739	0,13%
G5	783,99	830	-5,87%
G5#	830,61	830	0,07%
A5	880,00	880	0,00%
A5#	932,33	932	0,04%
B5	987,77	987	0,08%

The total processing time required for the intelligent sensor was lower than half a second (roughly 375 milliseconds in the worse case for the frequency analyser, and around 100 milliseconds for the sampling stage), thus fulfilling the specified response time condition of 2 seconds maximum.

The note-detector sensor as a whole was tested again using a sample set from the RWC database, as well as audio samples recorded from two different instruments: a Kawai CA91 piano, and a Hohner flute recorder. The results from these tests are summarized in Tables 3.2, 3.3 and 3.4.

From the results yielded, it can be inferred that the note is correctly detected most of the time. The worse detection rate is found for the highest frequency in the notes considered. This effect is caused due to the lower gain introduced by the anti-aliasing filter, as these frequency values lie near the cut-off frequency of 2KHz. The results for the Hohner recorder consistently yielded a 5% error rate. The tests for this instrument were repeated using the MATLAB framework with both the Goertzel and FFT algorithms, and the results yielded were similar. This indicates that this overall 5% error divergence is introduced because the instrument itself was not properly tuned.

The study conducted shows that the use and application of the aforementioned technologies and methods allows for real-time interaction with music. As shown by the research performed, this interaction can be used to implement an useful support tool to aid in the learning process for the early stages of music education, helping students identify the mistakes in their performances and to improve them accordingly.

Table 3.3: Note detection for notes extracted from the RWC database.

Note	frequency(Hz)	detected freq (Hz)	Error
E4	329,63	329	0,19%
F4	349,23	369	-5,66%
F4#	369,99	392	-5,95%
G4	392,00	415	-5,87%
G4#	415,30	415	0,07%
A4	440,00	440	0,00%
A4#	466,16	466	0,03%
B4	493,88	493	0,18%
C6	1046,50	1046	0,05%
C6#	1108,70	1108	0,06%
D6	1174,70	1174	0,06%
D6#	1244,50	1318	-5,91%
E6	1318,50	1318	0,04%

Table 3.4: Note detection for notes played with a flute recorder.

Note	frequency(Hz)	detected freq (Hz)	Error
C5	523,25	554	-5,88%
C5#	554,37	587	-5,89%
D5	587,33	587	0,06%
D5#	622,25	659	-5,91%
E5	659,26	698	-5,88%
F5	698,46	739	-5,80%
F5#	739,99	783	-5,81%
G5	783,99	830	-5,87%
G5#	830,61	880	-5,95%
A5	880,00	932	-5,91%
A5#	932,33	987	-5,86%
B5	987,77	1046	-5,90%
C6	1046,50	1108	-5,88%
C6#	1108,70	1174	-5,89%
D6	1174,70	1244	-5,90%
D6#	1244,50	1244	0,04%
E6	1318,50	1396	-5,88%
F6	1396,90	1480	-5,95%
F6#	1480,00	1568	-5,95%
G6	1568,00	1661	-5,93%
G6#	1661,20	1108	33,30%
A7	1760,00	1864	-5,91%
A7#	1864,70	1975	-5,92%
B7	1975,50	1046	47,05%

3.3 A tool to guide musicians: the virtual conductor

Previous sections delved into the works we have conducted towards the development of automatic correction applications that aid users in detecting the aspects in their performances that need improvement. This is perhaps the most direct type of interaction metaphor for learning purposes. A different type of interaction metaphor from a learning perspective would be that of guidance, i.e. a system that acts as a meaningful support tool to guide the musician during the execution of his performance.

In a music ensemble, this guiding role rests on the shoulders of the conductor, as he is responsible for coordinating all of the musicians in the ensemble in unison to play a given piece in harmony. We have conducted research towards the implementation of a virtual conductor, that is, a support tool that simulates the role of the conductor, thus helping a group of musician to coordinate their perfomances when rehearsing. In particular, this application implements a virtual orchestra conductor simulator for string quartet practice. Concretely, the system allows a musician to practise his/her performance either individually or in a group, and also provides the means to assess and evaluate the practitioner's performance.

In this section, we will cover the details of the research performed to this aim. A brief overview of the application is presented in the next subsection, while the core elements of the system (the tuner, the melody evaluator and the virtual conductor emulation model itself) will be explained further in separate subsections.

3.3.1 Application overview

The application assumes a string quartet ensemble of four different instruments: violin, viola, cello and contrabass; however, the system can be configured to a different set of instruments, such as the more typical distribution of 2 violins, 1 viola and 1 cello. The system gives indications to the user regarding the beat times, changes in tempo, etc. The system also offers feedback to the user regarding his/her performance, evaluating the accuracy of the student when playing the corresponding piece. The application stores the information of each of melodies considered in MIDI format, and also allows for the playback of the MIDI data, so that the user can hear the piece as a whole for a better reference.

A block diagram of the system can be found in Fig. 3.11. The application starts with a presentation screen and then prompts to a configuration menu to setup the session parameters. The system allows for different study and playback modes, depending on whether the user wishes to perform solo or group practice. The application includes

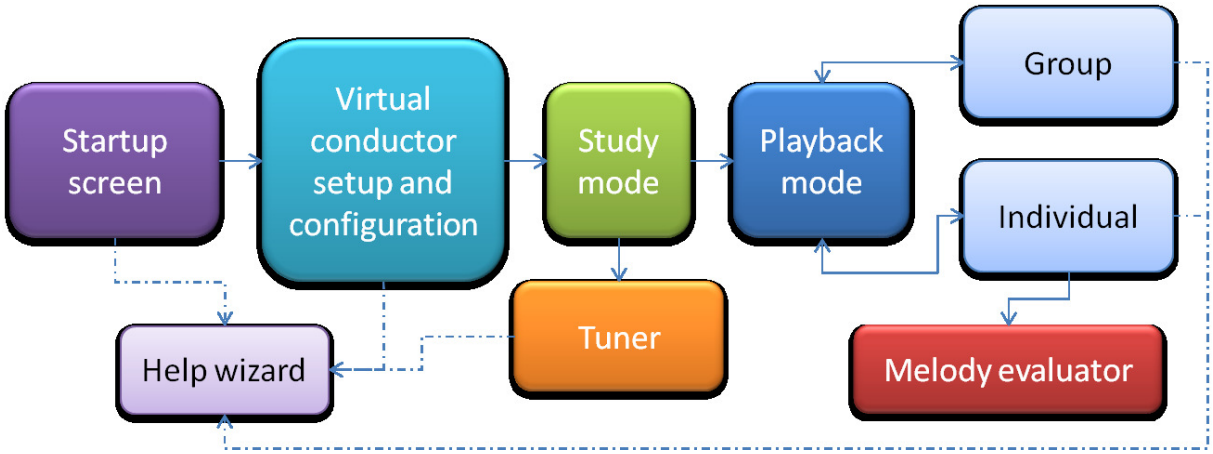


Figure 3.11: Modules of the Virtual Director application.

also a tuner module to ensure that the instruments are properly tuned before starting the practice session, as well as a melody evaluator that gives the user feedback on how accurate is his/her performance. A set of help menus are also provided to assist musicians in the use of the application. The tuner and melody evaluator are explained in detail in the following subsections.

3.3.2 Tuner

The tuner module accesses the input signal recorded from the microphone, calculates its spectrum at a sample frequency of 88200 Hz (achieving a frequency resolution of 0.5 Hz for an input signal of 44100 Hz) and subsequently extracts the fundamental and partial frequencies of the note played. The peaks in the spectrum are detected using a sliding window of 11 samples centred at each potential peak candidate. Peaks with a magnitude below 20% of the maximum value found are pruned, and further pruning is performed iteratively: first, the sample with highest magnitude is found, then the 4 adjacent samples are erased, and a new iteration begins.

In the resulting simplified spectrum, the distances between each successive peak, denoted as $d(n_i, n_{i+1})$, are calculated and stored. In the frequency domain, the fundamentals and partials of a note would be found equally spaced along the spectrum, in frequencies f_{pitch}, $2f_{pitch}$, $3f_{pitch}$, $4f_{pitch}$ etc. Following this schema, it is possible to detect the fundamental and partial frequencies, thus effectively extracting the pitch of the note played. The system then evaluates each peak detected to determine whether it belongs to a fundamental-partials set or not. If the system finds either the fundamental frequency and at least 2 partials, or, alternatively, 3 partials or more, then it proceeds to assess whether the string is tuned or not. If the fundamental frequency of the note detected is

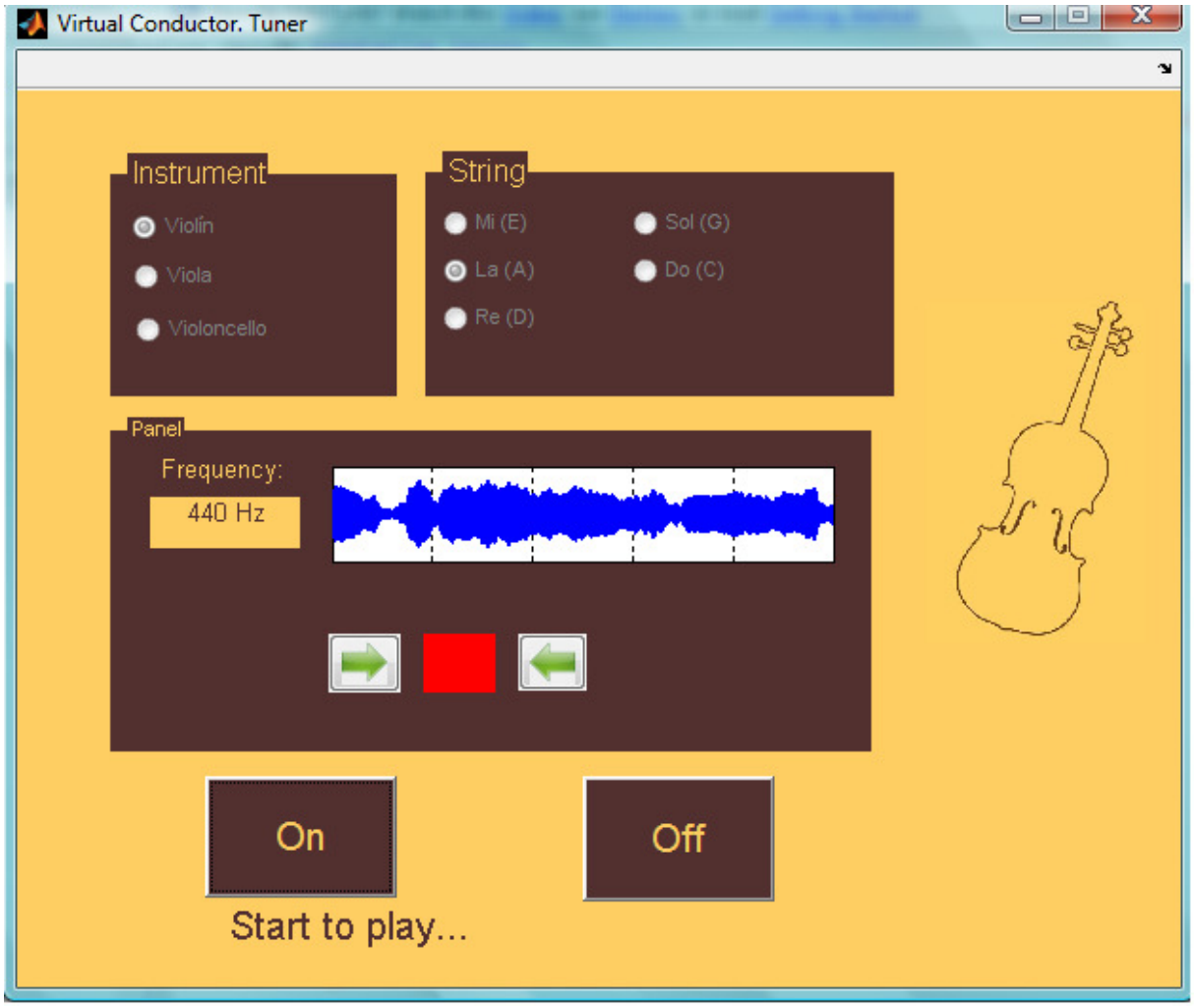


Figure 3.12: Tuner module interface.

within 2.2 Hz of the expected value, it is assumed that the corresponding string is tuned. Otherwise, the string must be tuned accordingly. This is indicated to the user through a simple interface, turning a red button into green if the string is tuned (Fig. 3.12).

3.3.3 Melody Evaluator

This block analyses the input signal to determine if the notes played are correct. In order to do so, the audio signal is windowed into slots so that each slots stores the samples corresponding to one beat, using the MIDI sequence as reference. For each of these windowed slots, the application uses the same detection method describe for the tuner to find the fundamental frequencies, and compares them with the ones that should be had according to the notes assigned to that beat in the MIDI file.

To prevent detecting false errors because of temporal misalignment due to a desynchro-

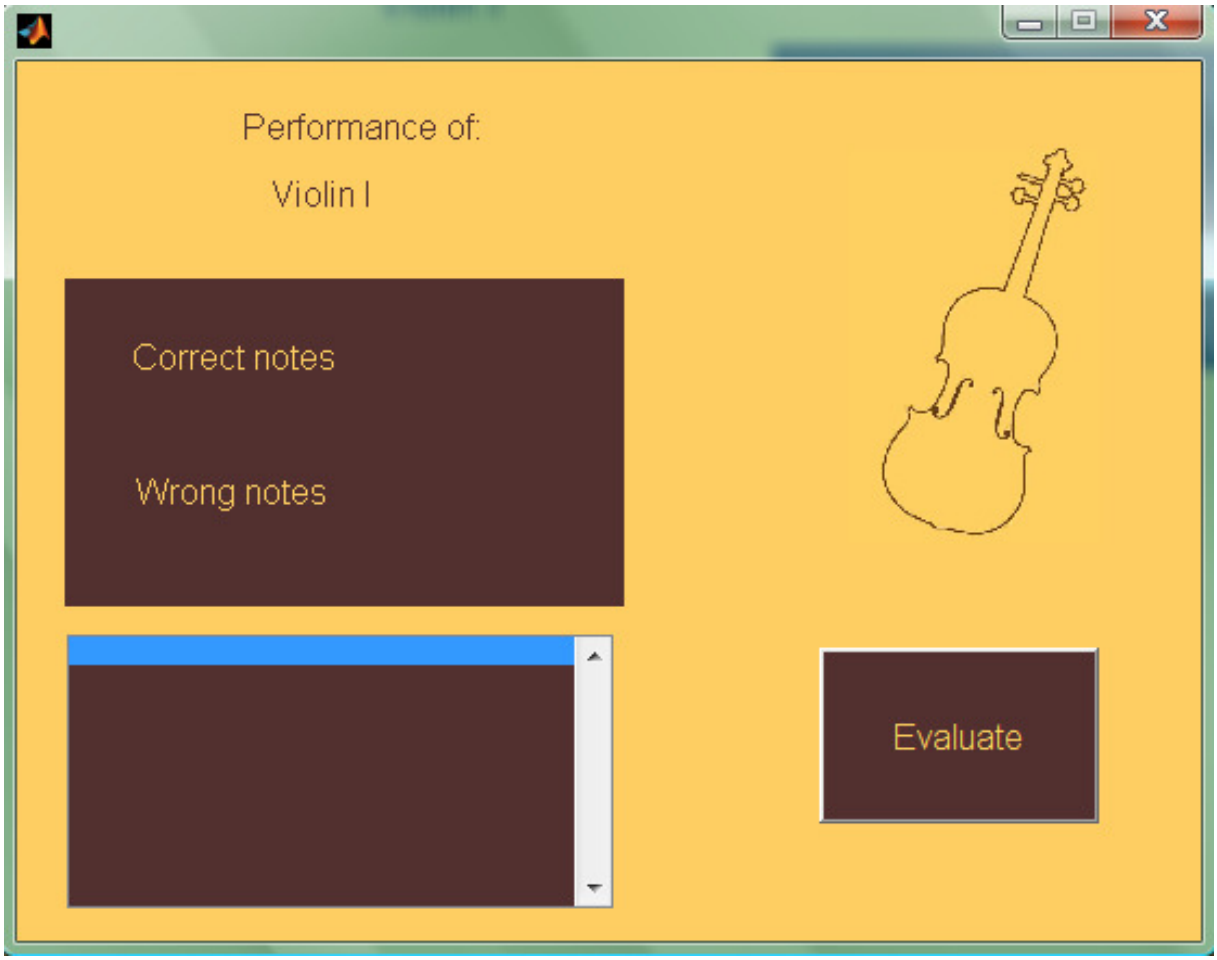


Figure 3.13: Melody evaluator dialog.

nization between the MIDI sequence and the input audio, each note detected is compared with the current, previous and next window in the MIDI sequence. A note is assumed to be correctly played if the difference between its fundamental frequency and the expected one is lower than a given minimum, which is different for each instrument and it's calculated as the difference between the lowest note possible for that instrument and its sharp version. For example, in the case of the violin, the lowest note available is G3, and the difference between G3 and G#3 is 11 Hz. Thus, if a given detected note is within 11 Hz of the note expected for the time beat evaluated, the system labels it as a correct note, or as a mistake otherwise.

For each melody evaluated, the system indicates the user the amount of correctly played notes, as well as the number of notes which the user played wrong, and the corresponding beat times in the score. The dialog in the final application can be seen at Fig. 3.13).

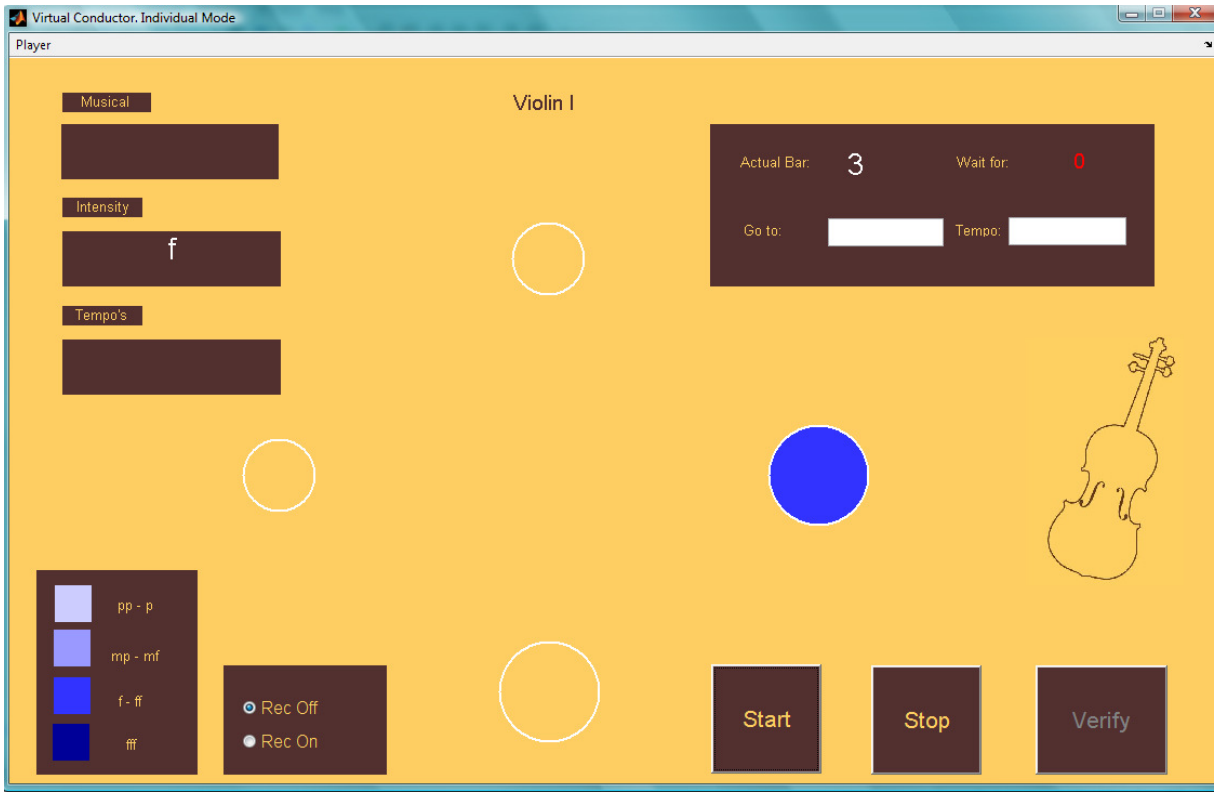


Figure 3.14: Virtual conductor for solo practice.

3.3.4 Virtual Conductor

The main functionality of the system implemented is that of emulating the indications that an ensemble conductor gives to his/her fellow musicians when practising and playing a given piece.

The application uses a virtual baton to give such indications, represented by a set of four circles displayed on the computer screen, in the four positions typically used to signal beat times. The shape and colour of these circles change according to the beat and dynamics of the piece being played. As the piece is played and beats are played, the circle corresponding to the current beat time is coloured or lighted accordingly. The circles are coloured as if seen from the point of view of the musician, i.e. a 3/4 time signature would be signalled in the order down-right-up.

Dynamics nuances are indicated in written form but also through the exact colour and size of each circle. Thus, for a *piano* or *pianissimo* nuance, there is a small light-coloured circle, while for a *mezzoforte-forte* intensity, the circle becomes larger and darker (see Fig. 3.14). Additional dynamics indications are indicated only in text form, as well as the bar number. If the conductor needs to indicate a *fermata*, this is signalled to the user by painting a red circle in the center of the baton (as per Fig. 3.15).

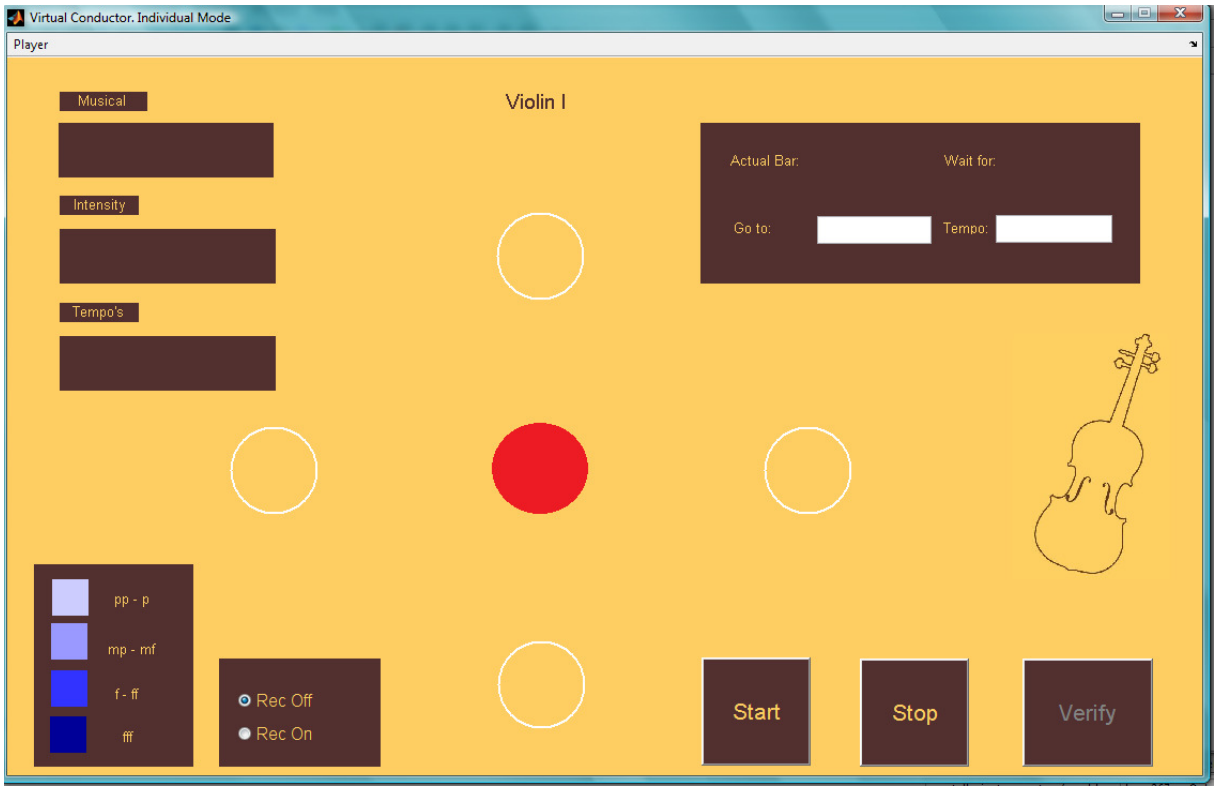


Figure 3.15: Indication of a *fermata*.

The user can also stop the performance at any time, restart at any given bar number, and manually change the tempo on the fly. This last option has been provided to specifically account for the fact that the tempo in rehearsals is usually initially lower to the actual tempo of the piece, and it is slowly increased as the musicians practise further.

The virtual conductor can be used for solo practice or group practice. In the case of the latter, the information provided by the system differs slightly from the previously commented features. Concretely, the space devoted to the virtual baton on the screen is more confined, and the indications given refer to the general indications that affect every single instrument globally. For specific indications for each of the instruments taking part in the performance, a set of panels are provided (first and second violin, viola, cello and contrabass) as seen in Fig. 3.16

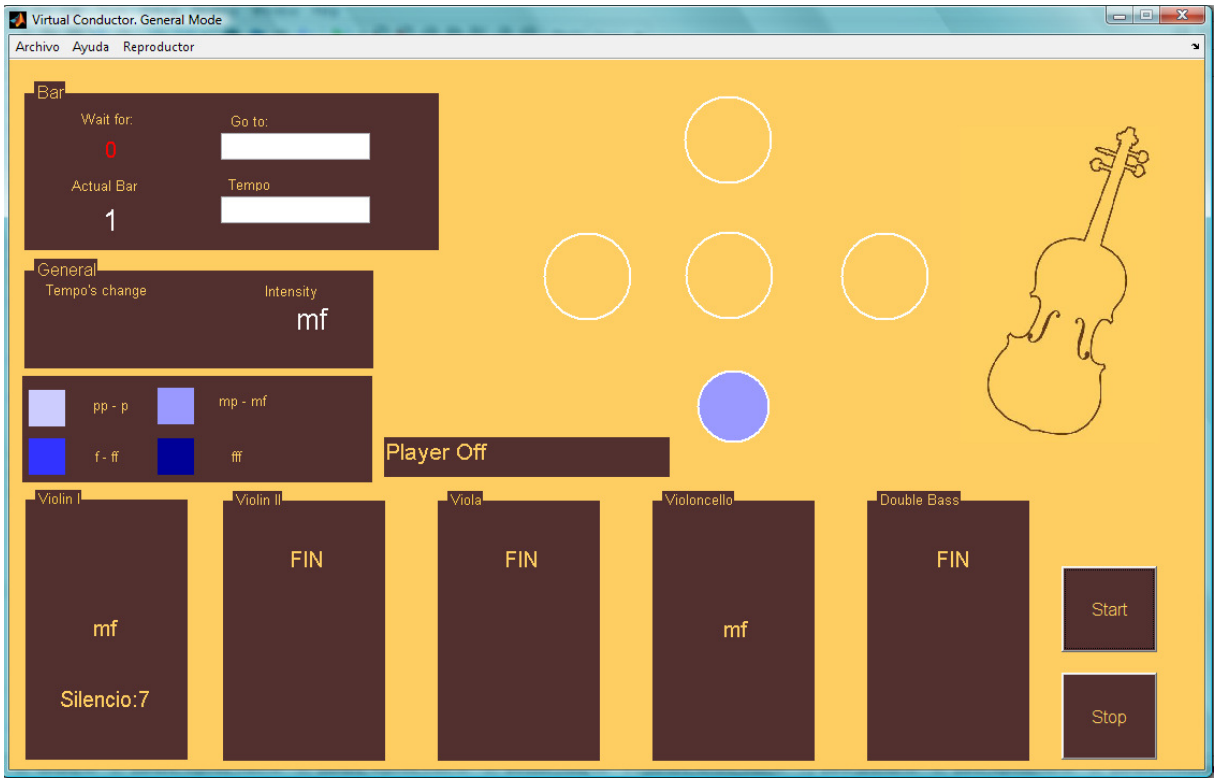


Figure 3.16: Virtual conductor for group practice.

3.3.5 Results

The final application was tested by a set of musicians from the Chamber Orchestra of Málaga, including an ensemble conductor. Each participant learned how to use the application and was asked to fill in a questionnaire of 6 items. They were asked to assign each item a value ranging from 0 to 10. In particular, the items each participant was asked to evaluate were:

1. Satisfaction and overall utility
2. Ease of use
3. Value for personal use
4. Perceived usefulness as a learning tool
5. Clearness in the virtual conductor indications.
6. Personal opinion

The answers collected were overwhelmingly positive, with average scores between 9 and 10 for all the items in the questionnaire (see Fig. 4.20). Participants also indicated

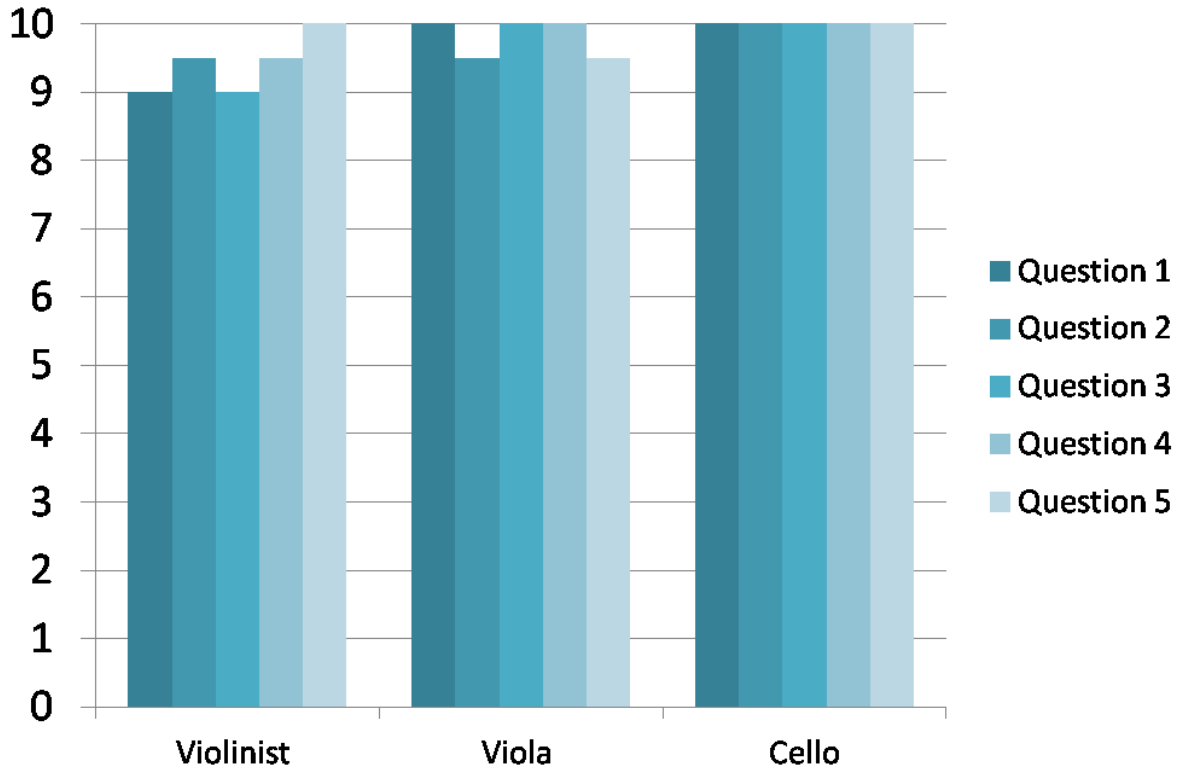


Figure 3.17: Virtual Conductor assessment from users' questionnaire.

that they found the tool designed covered an important need for music practice that is not currently addressed by the more commonly used learning tools.

The system was also tested by a professional ensemble conductor, who found the virtual conductor proposed to be an excellent pedagogical tool for musicians at any learning stage, as it addresses one important handicap in the learning process, which is solo rehearsing of chamber pieces. Furthermore, she found particularly enticing the group practice possibilities of the application, for it makes the learning process less lonely as well as it gives the student a much better context for his/her performance.

The research conducted has shown that it is possible to simulate part of the ensemble's conductor role through the application of information technologies and music information retrieval techniques to the musical signal and MIDI data. Furthermore, the user tests conducted show that this is a useful addition to the array of tools available for music rehearsal, as it helps overcome some of the limitations in the conventional methods for string quartet practice.

CHAPTER

4

ADVANCED HUMAN-COMPUTER INTERFACES FOR MUSIC INTERACTION

4.1 Introduction

In the previous chapter, we discussed the possibilities of applying human-computer interaction techniques to reinforce the use of computer-based applications for music learning. This chapter will instead revolve around the potential uses of incipient technologies for the development of new and innovative interaction paradigms for music interaction.

Concretely, this chapter will give a special focus to motion-tracking based human-computer interfaces and their application towards the achievement of more immersive and satisfying musical experiences. The use of these advanced interfaces also allows for the implementation of virtual instruments as well as ways to modify the performance of previously recorded musical pieces in real-time.

4.2 Virtual musical instruments

This section presents the research conducted towards the simulation of virtual instruments through the use of advanced human-computer interfaces. In particular, we have especially focused our research on the implementation of a drumkit simulator, since the detection of motion in this kind of musical instrument is easier to accomplish with the current state of the technology. Alongside the works on the implementation of a motion-based virtual drumkit, this section will also portray the research performed in the development of other virtual instruments, such as the virtual theremin or a sound-processing based virtual drumkit.

4.2.1 Virtual drumkit

The most direct implementation of a virtual drumkit would be to simply define a virtual volume in space around the user, so that whenever the user *hits* this volume, the system behaves as a drumkit, playing the corresponding sound. With this in mind, in order to track user movements, we opted for a Kinect camera sensor. This type of sensor provides a mostly non-intrusive experience, without hindering or hampering user motion in almost any way, it is mostly inexpensive and provides multi-point skeleton tracking capabilities. Concretely, it allows for simultaneous tracking of both hands, which is sufficient for the implementation of a volume-based virtual drumkit. For simplification purposes, we have assumed that the hands are the actual drumsticks.

As part of the implementation, the application rendered a virtual environment using C/C++, OpenGL graphics library (Shreiner [1999]), and the OGRE graphics engine (Junker [2006]), so that the user had a visual reference in order to "hit" the virtual drum (see Fig. 4.1). The sounds were read from wav files, using the OpenAL audio library. The human-computer interaction interface was implemented with the OpenNI library and the NITE plugin, using full-body skeleton tracking to track the motion of both hands at the same time. The skeleton node corresponding to the head was used as a reference to place the virtual objects in the environment (i.e. the virtual drum). An overview of the application block structure can be found in Fig. 4.2.

This interface has two important shortcomings. First of all, since it relies on hitting a specific physical volume around the user, it becomes increasingly harder for the user to hit a given concrete volume as more drums are added. It also requires the user to keep fixating his point of view on the screen to find the exact location of those volumes, which can be counter-intuitive as well as problematic if the drums do not lie all at the same depth level (and, for a real drumkit, that is clearly not the case); this is not a problem

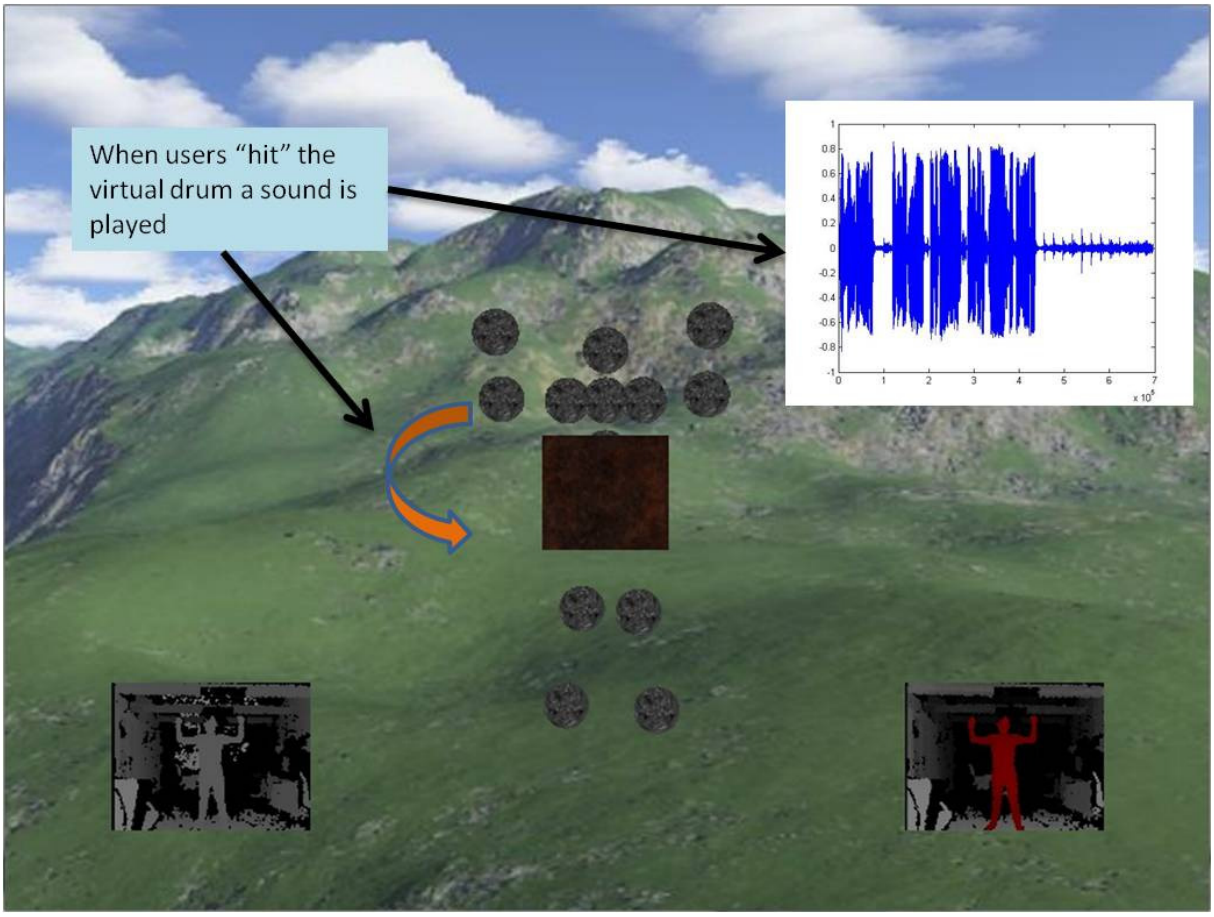


Figure 4.1: Virtual Environment.

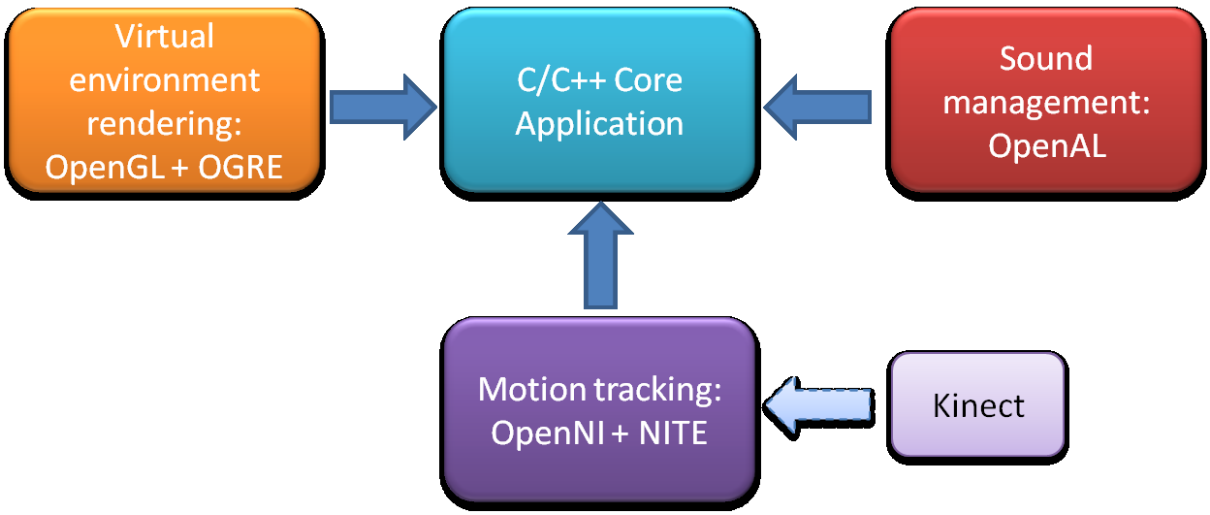


Figure 4.2: Application block structure.

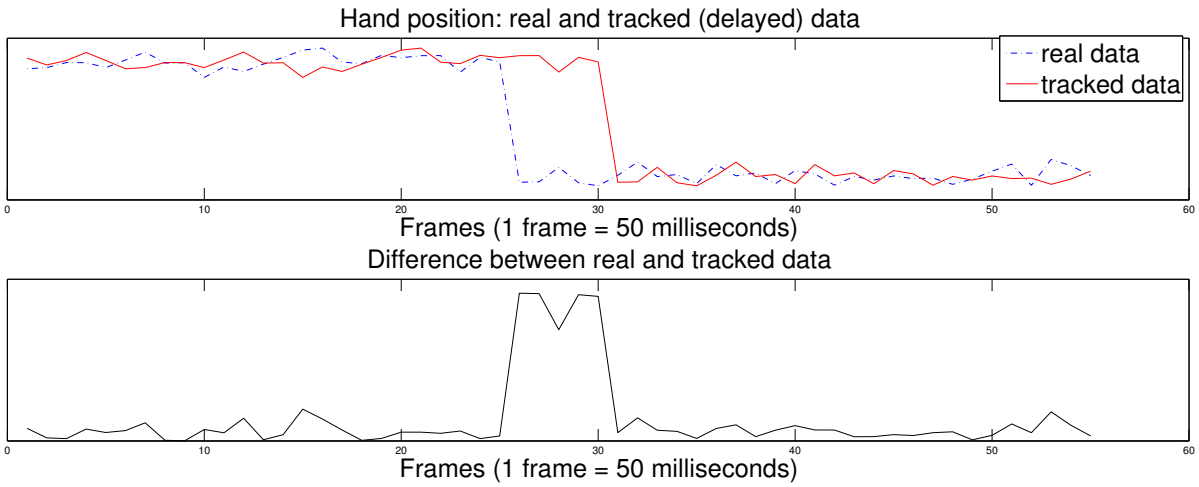


Figure 4.3: Real data and tracked data: effects of the delay

when there is a single drum in the simulation, but becomes an important issue when the number of drums is 3 or higher.

The second drawback of this approach relies on the limitations of the tracking system itself, and, in particular, the latency in the response of the system. Some user performances using this interface were recorded with an EOS 5D Mark II camera. By combining the analysis of the recordings performed and the actual times when the drum-hitting event was triggered, it was found that the overall lag in our application was approximately of 160 ms (the application ran at approximately 28 fps, so the lag was roughly 5 rendering frames, fig. 4.3). These results match with prior studies (Odowichuk et al. [2011], Livingston et al. [2012]), in which applications with Kinect-based tracking have been shown to introduce a latency of almost 0.3 seconds (Livingston et al. [2012]). According to previous studies on audio perception (Lago and Kon [2004]), this lag introduced is very noticeable and will have an impact on user experience. Specially, the lag induced becomes particularly relevant in the case of swift movements, as is the case of a drum-hitting gesture (see fig. 4.3).

In light of these setbacks, we resorted to a different interaction model: instead of detecting collisions with a previously defined volume, the system would instead rely on recognizing gestures and playing sounds according to the gesture detected. The gesture detection model will also include a prediction model to compensate for the effects of the lag introduced by the system. This new approach is further described in the next section.

4.2.1.1 Prediction of a fast-hitting gesture

As previously indicated, the latency introduced by the system was found to be too high for an adequate simulation. Thus, in order to minimize its effects, a different model is

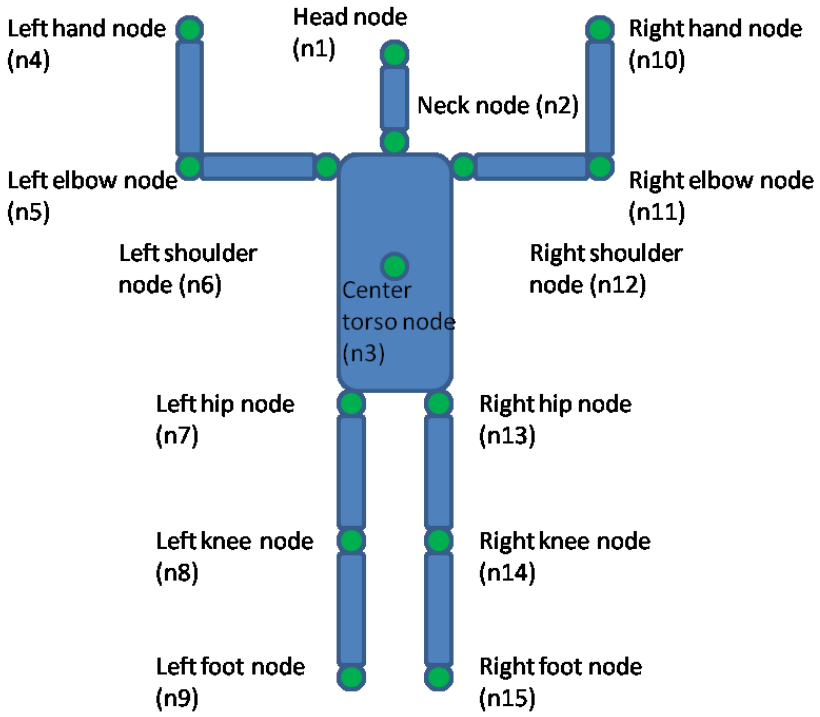


Figure 4.4: 15-nodes configuration for skeleton tracking.

used to detect user hitting gestures. In particular, the main aim of the system is to detect fast moving motion along a given axis using a prediction model to eliminate the lag, and then classify each gesture accordingly to play the corresponding sound.

The skeleton model considered in our system used a 15-node configuration (Ope [2010], Pri [2010]) as illustrated in Fig. 4.4, taking the cartesian X,Y,Z coordinates for each node. These data, however, needs to be processed further, as it is strongly influenced by user position and height.

To define a common reference point, the center of mass $\overrightarrow{n_0} = (c_x, c_y, c_z)$ of the user silhouette is calculated as:

$$c_i = \frac{1}{N} \sum up_{ji} \tag{4.1}$$

where up_{ji} stands for the value of coordinate $i = (x, y, z)$ for the j-th user pixel. The normalized cartesian coordinates for each node nn_i were calculated as:

$$\overrightarrow{nn_i} = \frac{1}{S}(\overrightarrow{n_i} - \overrightarrow{n_0}) \tag{4.2}$$

where $S = |n_{1y} - n_{3y}|$ is a reference scale value set at the difference between the height values of the head and the torso nodes. By using this rescaling factor, we make the normalized coordinates more resilient to potential variability in users' size and their relative

position to the camera.

In order to compensate for the effects of lag, a linear predictor based on Wiener filtering was used Wiener [1964]. The stream of tracked hand motion is used as an input signal to a Wiener filtering process . The Wiener l-samples predictor filter $h[n]$ or N-order solves the equation (4.3) for an input signal $x[n]$ with autocorrelation $R_{xx}[n]$, implementing a least minimum mean square error estimator (LMMSE).

$$\begin{bmatrix} R_{xx}[0] & R_{xx}[1] & \dots & R_{xx}[N-1] \\ R_{xx}[1] & R_{xx}[0] & \dots & R_{xx}[N-2] \\ \vdots & \vdots & & \vdots \\ R_{xx}[N-1] & R_{xx}[N-2] & \dots & R_{xx}[0] \end{bmatrix} \begin{bmatrix} h[1] \\ h[2] \\ \vdots \\ h[N] \end{bmatrix} = \begin{bmatrix} R_{xx}[l] \\ R_{xx}[l+1] \\ \vdots \\ R_{xx}[l+N-1] \end{bmatrix} \tag{4.3}$$

Instead of using the raw data corresponding to the hand positions, the signal $x[n]$ was further processed to store only the motion performed in the directional axis of interest. Let $\overrightarrow{d}$ represent the unit vector in the directional axis of interest and let $\overrightarrow{nn_i[n]}$ be the streamed position data for the node of interest i, and $\overrightarrow{nv_i[n]}$ its corresponding normalized velocity signal. The motion over the axis of interest, $\overrightarrow{d} \cdot \overrightarrow{nn_i[n]}$, can be decomposed into:

$$\overrightarrow{d} \cdot \overrightarrow{nn_i[n]} = \sum_j D_{j,T_j,s_j}[n - s_j] + \sum_k R_{k,M_k,s_k}[n - s_k] \tag{4.4}$$

where each D_{j,T_j,s_j} is a chunk of size T_j of the original $\overrightarrow{d} \cdot \overrightarrow{nn_i[n]}$ signal that fulfils:

$$D_{j,T_j,s_j}[n] = \begin{cases} \overrightarrow{d}\,\overrightarrow{nn_i[n+s_j]}, \text{ with } \overrightarrow{d}\,\overrightarrow{nv_i[n+s_j]} > \sigma & n \in [0, T_j - 1] \\ 0 & n \notin [0, T_j - 1] \end{cases} \tag{4.5}$$

Similarly, R_{k,M_k,s_k} is analogously defined as:

$$R_{k,M_k,s_k}[n] = \begin{cases} \overrightarrow{d}\,\overrightarrow{nn_i[n+s_j]}, \text{ with } \overrightarrow{d}\,\overrightarrow{nv_i[n+s_j]} \leq \sigma & n \in [0, M_k - 1] \\ 0 & n \notin [0, M_k - 1] \end{cases} \tag{4.6}$$

The input signal $x[n]$ is then defined as:

$$x[n] = \sum_j D_{j,T_j,s_j}[n - jT_j] \tag{4.7}$$

The value of σ was empirically adjusted $\sigma = 0.15$ normalized units, as it was found that this value minimized the influence of noise in the measures while preserving a high enough sensibility to motion.

As previously indicated, the latency in our system could be as high as a 5-samples delay. Therefore, a bank of 5 Wiener filters were used to predict the $l = 1$ to 5-th future samples, which were in turn used as features for the fast-gesture detector.

4.2.1.2 Features for gesture discrimination

So far, we have isolated a set of features to detect a hitting motion while compensating latency at the same time. In order to determine which drum the user wishes to hit, though, we need to further define an additional set of features to discriminate among different hitting-gestures.

In regard to this aim, we considered two different sets: the first one relies on finding parameters to describe the trajectory followed by the hand before detecting the hitting motion, while the other focuses on features extracted from the current pose of the user's arm at the aforementioned instant.

The trajectory feature set took a total of 15 features. Concretely, the features considered were the normalized cartesian coordinates of the hand vector at the time a fast gesture is detected and at four previously defined times, as shown in figure 4.5. These 4 previous times were calculated by keeping a history of samples and searching this history for the sample at which the the motion in the direction of interest started. Once the starting point is found, the three remaining points are determined through linear interpolation, so that the 5 final position values are pairwise equidistant in time. This feature set has been labelled as T_{ix}, T_{iy}, T_{iz}, with i ranging from 1 to 5, the former representing the coordinates at the time the gesture is detected, and the latter corresponding to the start of the motion.

Arm pose feature set totalled 8 features: the 3 coordinates of the unit vector for the elbow-to-hand limb (EH_x, EH_y, EH_z), the 3 coordinates of the unit vector for the shoulder-to-elbow limb (SE_x, SE_y, SE_z), and the 2 coordinates of the unit vector of the projection of the normalized hand vector onto the XZ plane (H_x, H_z).

4.2.1.3 Gesture classification

Once we have defined a proper feature set, the next step is to evaluate the best combination of set of features and classifier in order to discriminate drum-hitting gestures. In this case, the fast-gesture should be performed along the direction of interest $\overrightarrow{d} = (0, 0, -1)$, that

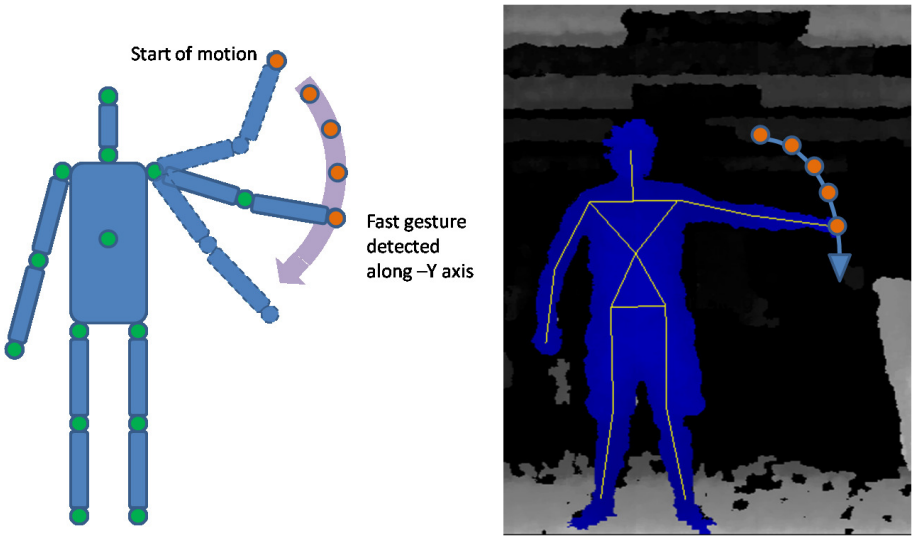


Figure 4.5: Features defining the trajectory of the gesture.

is, the negative Y axis. The gesture detection system follows the scheme presented in Fig. 4.6.

First, the system determines whether the user is performing a fast gesture in the direction of interest, using the features extracted from Wiener prediction stage. After that, if a fast gesture is detected, the features considered for trajectory and arm pose indicate which piece of the drumkit must be played.

We defined the system so that a total of 6 gestures could be detected for each arm/hand, each of these gestures corresponding to one of 7 possible "drums" in the drumkit. Concretely, the 6 gestures considered correspond to hitting either the Snare, the left and right frontal Toms, left and right Cymbals, or the Lateral drum (low Tom or Hi-Hat, depending on which hand triggered the sound). These gestures were labelled respectively as *Snare*, *T_FrLeft*, *T_FrRight*, *C_Left*, *C_Right* and *T_Lateral*. Fig. 4.7 shows some examples of the distribution of the previously defined classes for one hand in feature space.

A database of 1108 gestures was gathered, performed by 3 different individuals, with a total of gestures per-classe of 145, 206, 224, 151, 155 and 227 respectively. Alongside these data, the database also included a total of 977 segments for tracked motion that did not correspond to a fast hitting gesture (for the purpose of training the fast gesture detector based on the Wiener filter prediction).

Since the detection of fast gestures over the Y axis is a fairly simple problem of binary classification, we deemed that a Logistic Regression model (Hosmer Jr et al. [2013]) would suffice for the classification task. After training the model with a tenfold cross-validation along the whole database, the maximum success rate found was of 99.3 % for

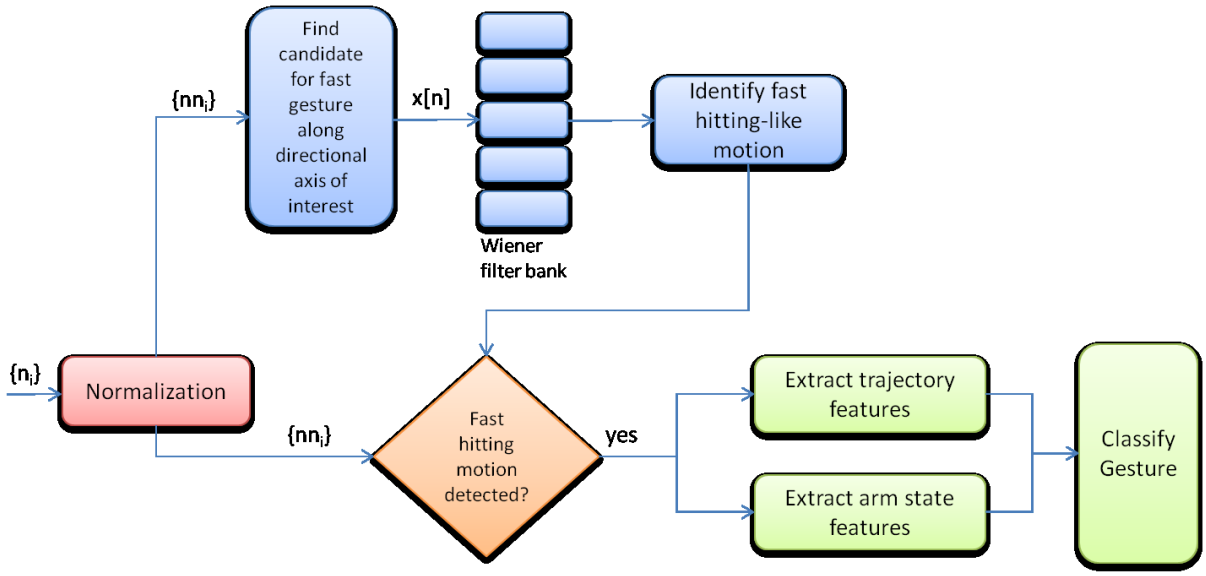


Figure 4.6: Gesture detection system.

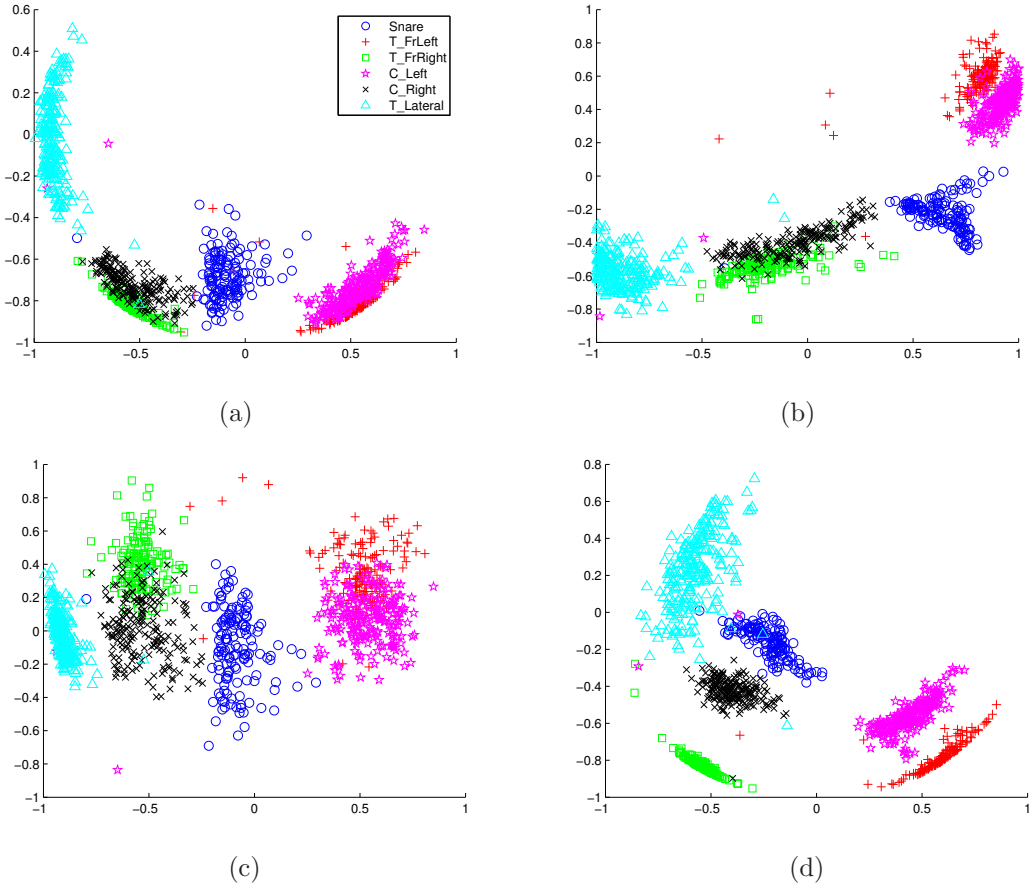


Figure 4.7: Distribution of the 6 types of gestures for some feature combinations: H_x vs H_z (a), EH_x vs SH_z (b), H_x vs EH_y (c) and SH_x vs SH_z (d).

a regularization ridge value of 0.7.

The task of discriminating which drum is being played demanded a more detailed analysis, and thus, we conducted separate analyses for the trajectory and arm pose feature sets, as well as a third analysis with both sets combined. In all the studies performed, we used a total of seven classifiers: Naïve Bayes (John and Langley [1995], Rish [2001]), Support Vector Machines (Cortes and Vapnik [1995], Steinwart and Christmann [2008]) with two different kernels (polynomial and gaussian), K-Nearest Neighbours classifier or k-NN (Aha et al. [1991], Altman [1992]), decision tree classification based on the C4.5 algorithm (Quinlan [1993]), Logistic Regression (Hosmer Jr et al. [2013], Le Cessie and Van Houwelingen [1992]) and Multilayer Perceptron (Haykin [1994], Haykin [2007]). The results found for each classifier are illustrated in the following paragraphs.

Naïve Bayes

A Naïve Bayes classifier is a simple probabilistic model which assumes that the different features are independent from each other (John and Langley [1995], Rish [2001]). The success rates found after training the model with the data gathered are summarized in Table 4.1, using tenfold cross-validation. The corresponding confusion matrices can be found in Fig. 4.8

Support Vector Machine

Support Vector Machine (SVM) classifiers make use of n-dimensional boundaries or hyperplanes in order to discriminate the training samples corresponding to each class from the rest of samples in the learning set (Cortes and Vapnik [1995], Steinwart and Christmann [2008]). Previously to the training stage, the different data samples are typically transformed using a specific function or kernel, $K(x_i, x_j)$, where x_i and x_j represent input feature vectors. This transformation helps to ensure that a n-dimensional hyperplane can be found for optimal separation of the different classes considered.

We have trained two different SVM classifiers, attending to the type of kernel used: polynomial and gaussian radial basis. A polynomial kernel of e-degree uses an equation in the form of Eq (4.8) to expand feature space through lineal polynomial combinations

Feature set	Success rate
Trajectory features	83.75%
Arm pose features	98.65%
All features combined	99.37%

Table 4.1: Classification success rate for Naïve Bayes classifier.

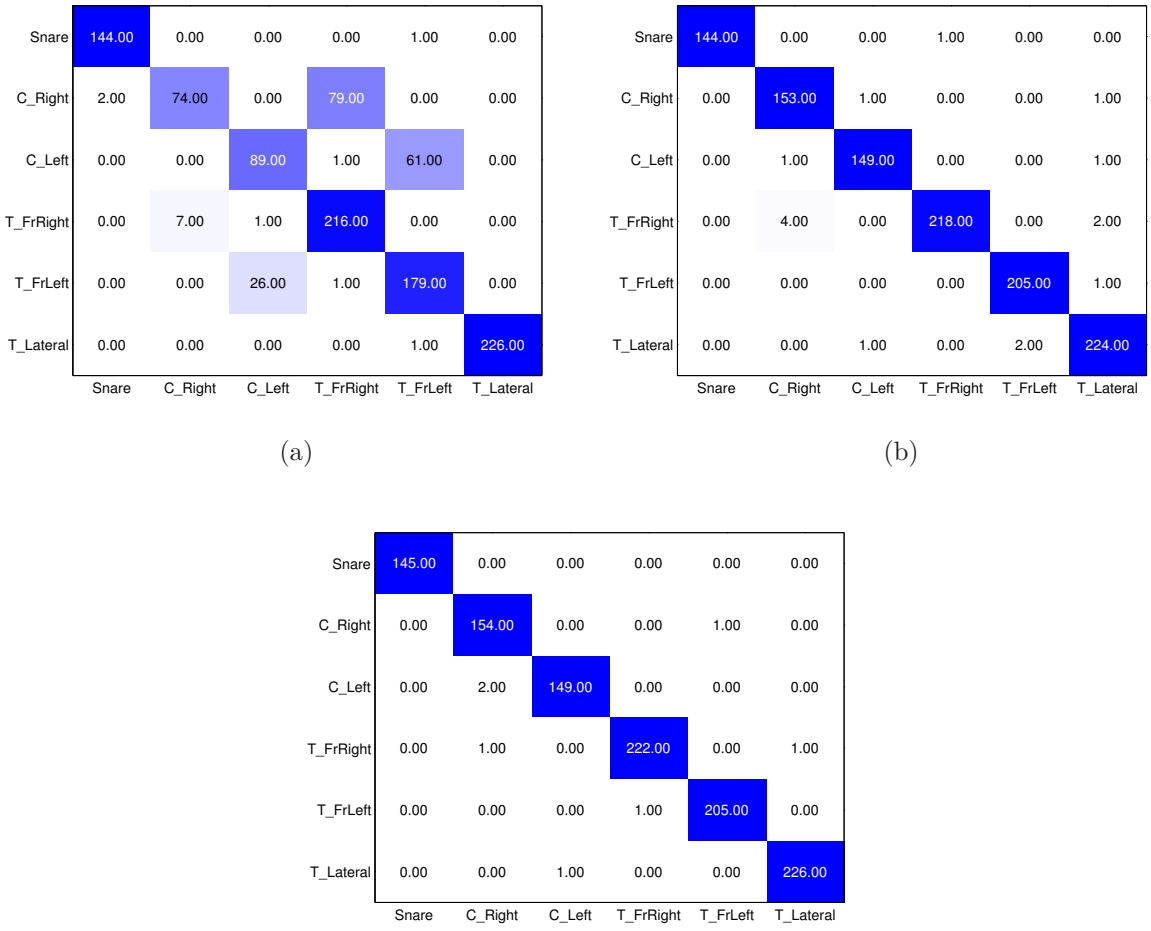


Figure 4.8: Confusion matrix for Naïve Bayes classifier using the Trajectory features (a), Arm Pose features (b), and all features combined (c).

of the source training set.

$$K(x_i, x_j) = (x_i \cdot x_j)^e \tag{4.8}$$

The value of e was chosen so that the overall success rate after tenfold cross-validation was optimal. The range of degree values tested went from $e = 1$ to 10. The best success rates found along with their corresponding e value are summarize in Table 4.2. The corresponding confusion matrices for these cases can be found in Fig. 4.9

The Gaussian radial based kernel transforms the original samples into an expanded feature space following Eq (4.9).

$$K(x_i, x_j) = e^{\gamma \|x_i - x_j\|^2} \tag{4.9}$$

Feature set	Success rate	Parameter (e)
Trajectory features	92.33%	3
Arm pose features	99.01%	1
All features combined	99.28%	1

Table 4.2: Classification success rate for SVM classifier with polynomial kernel.

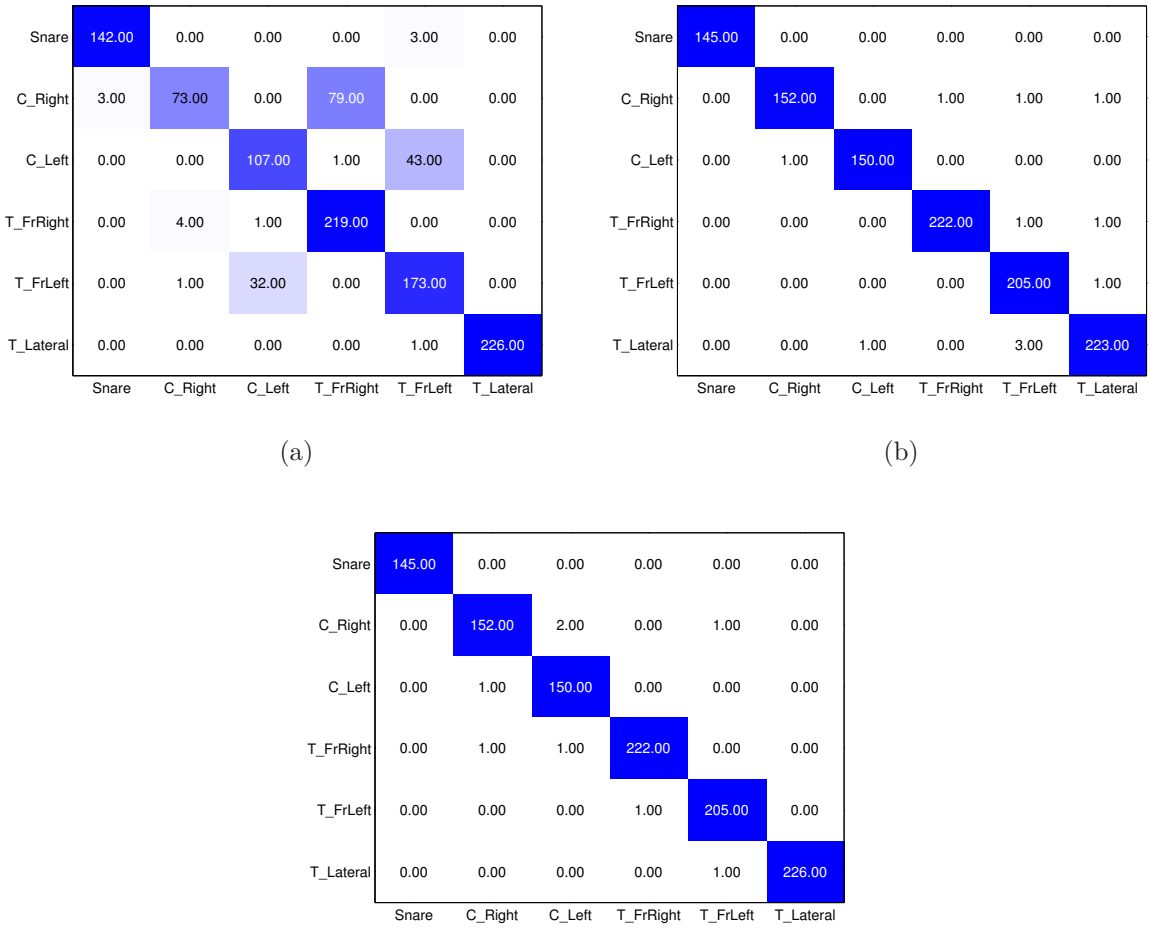


Figure 4.9: Confusion matrix for SVM classifier with polynomial kernel using the Trajectory features (a), Arm Pose features (b), and all features combined (c).

Parameter γ oscillated between 0.001 and 1000, in order to find the optimal value for which success rate in the classification was maximum. Table 4.3 illustrates the best results found after performing a tenfold cross-validation. Fig. 4.10 presents the associated confusion matrices.

Feature set	Success rate	γ
Trajectory features	84.84%	170
Arm pose features	99.10%	5
All features combined	99.37%	5.5

Table 4.3: Classification success rate for SVM classifier with gaussian kernel.

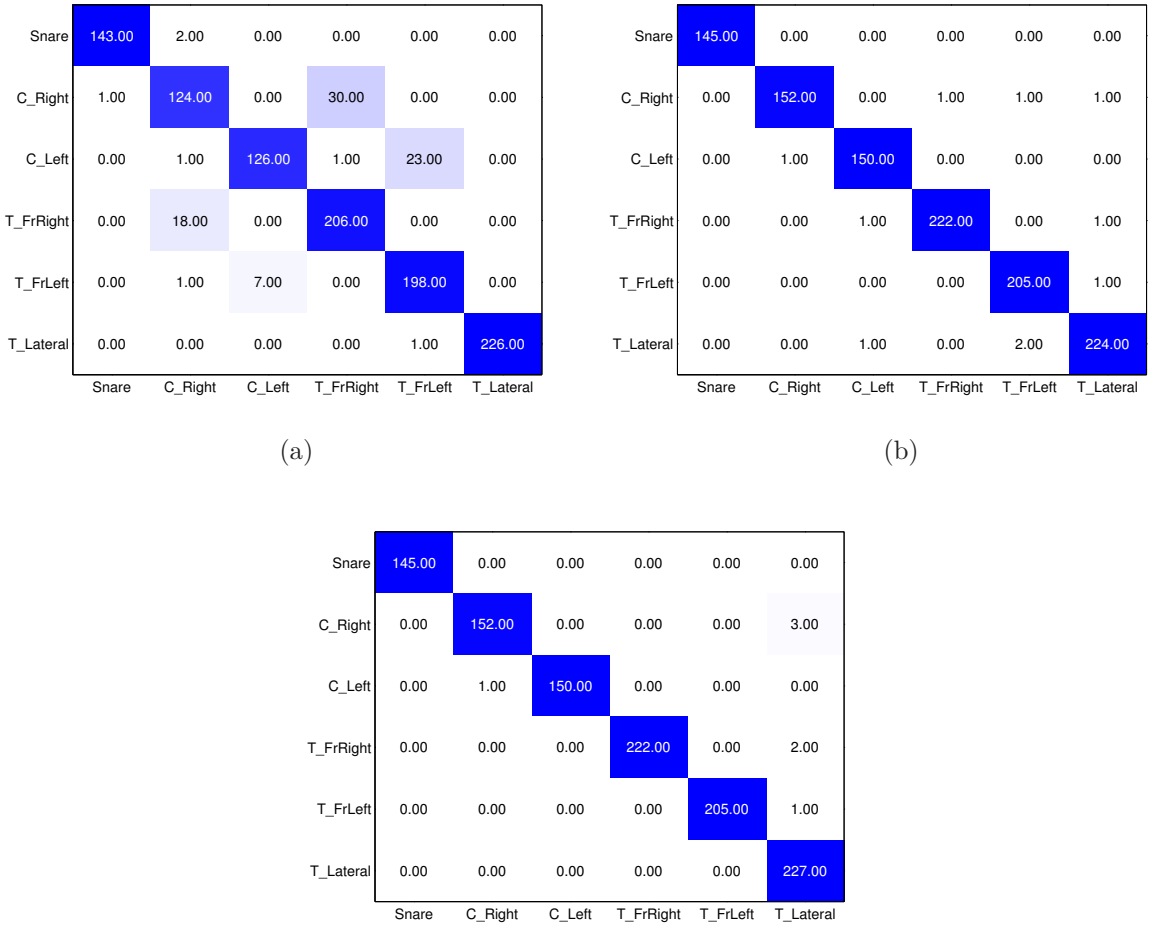


Figure 4.10: Confusion matrix for SVM classifier with gaussian kernel using the Trajectory features (a), Arm Pose features (b), and all features combined (c).

K-Nearest Neighbours

K-Nearest Neighbours (k-NN) follows a lazy algorithm based on a majority voting process to classify a given sample according to the classes of its k nearest neighbours (Aha et al. [1991], Altman [1992]). Once more, the value of k was taken as a parameter in the training process, chosen to maximize success rate after tenfold cross-validation; k values

Feature set	Success rate	Parameter (k)
Trajectory features	90.43%	8
Arm pose features	99.10%	3
All features combined	99.37%	2

Table 4.4: Classification success rate for k-NN classifier.

tested ranged from 1 to 15. The best classification results can be found in Table 4.4 and Fig 4.11.

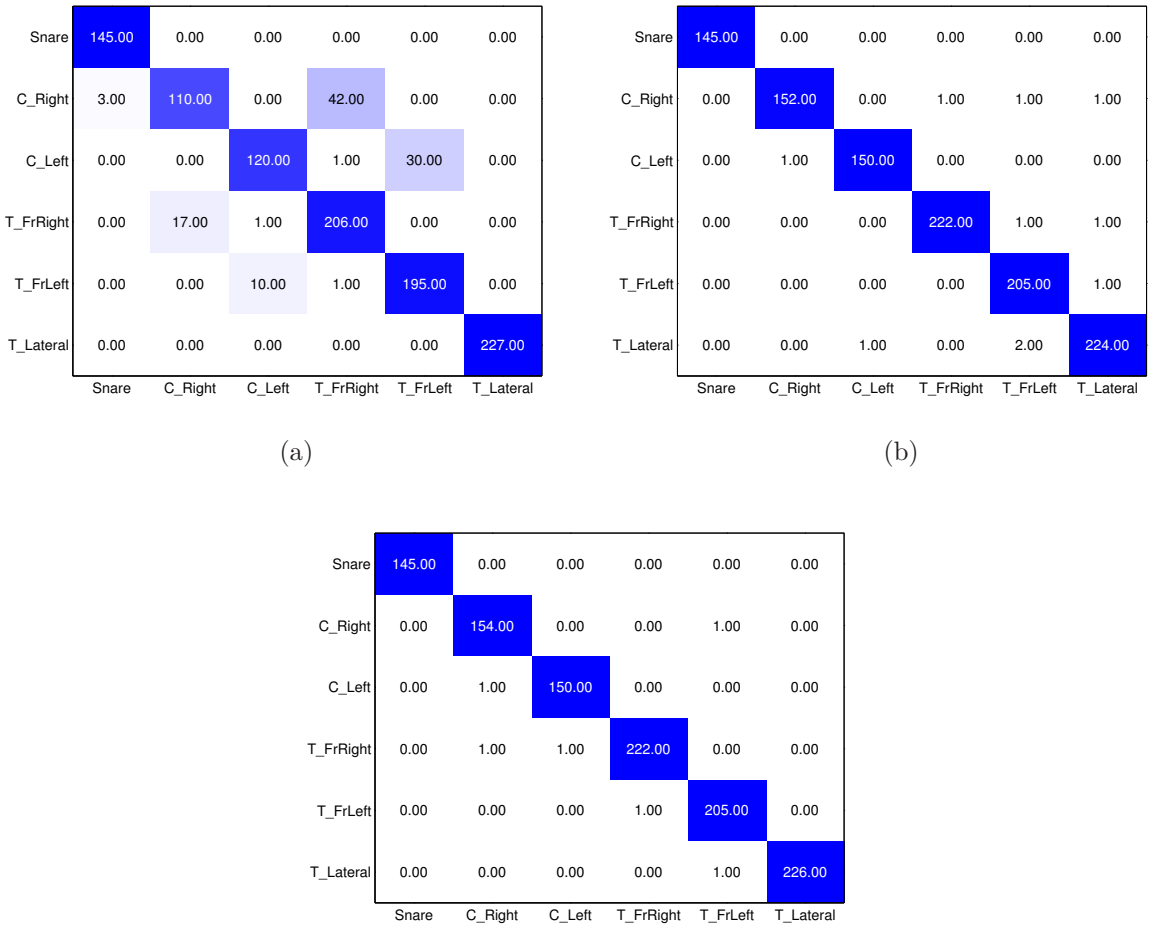


Figure 4.11: Confusion matrix for kNN classifier using the Trajectory features (a), Arm Pose features (b), and all features combined (c).

C4.5 Decision tree classifier

This algorithm uses the training samples as the basis for training a logical tree (Quinlan

Feature set	Success rate
Trajectory features	86.91%
Arm pose features	99.01%
All features combined	99.10%

Table 4.5: Classification success rate for C4.5 classifier.

[1993]). The leaves in the tree represent classes, while each other node represents a set of conditions on a given input sample to decide which child node to follow. Thus, given an input feature vector, the condition for the root node of the tree is evaluated, next child node to follow is determined, and the process is repeated, until a leaf is reached. Results after conducting tenfold cross-validation are summarized in Table 4.5 and Fig 4.12.

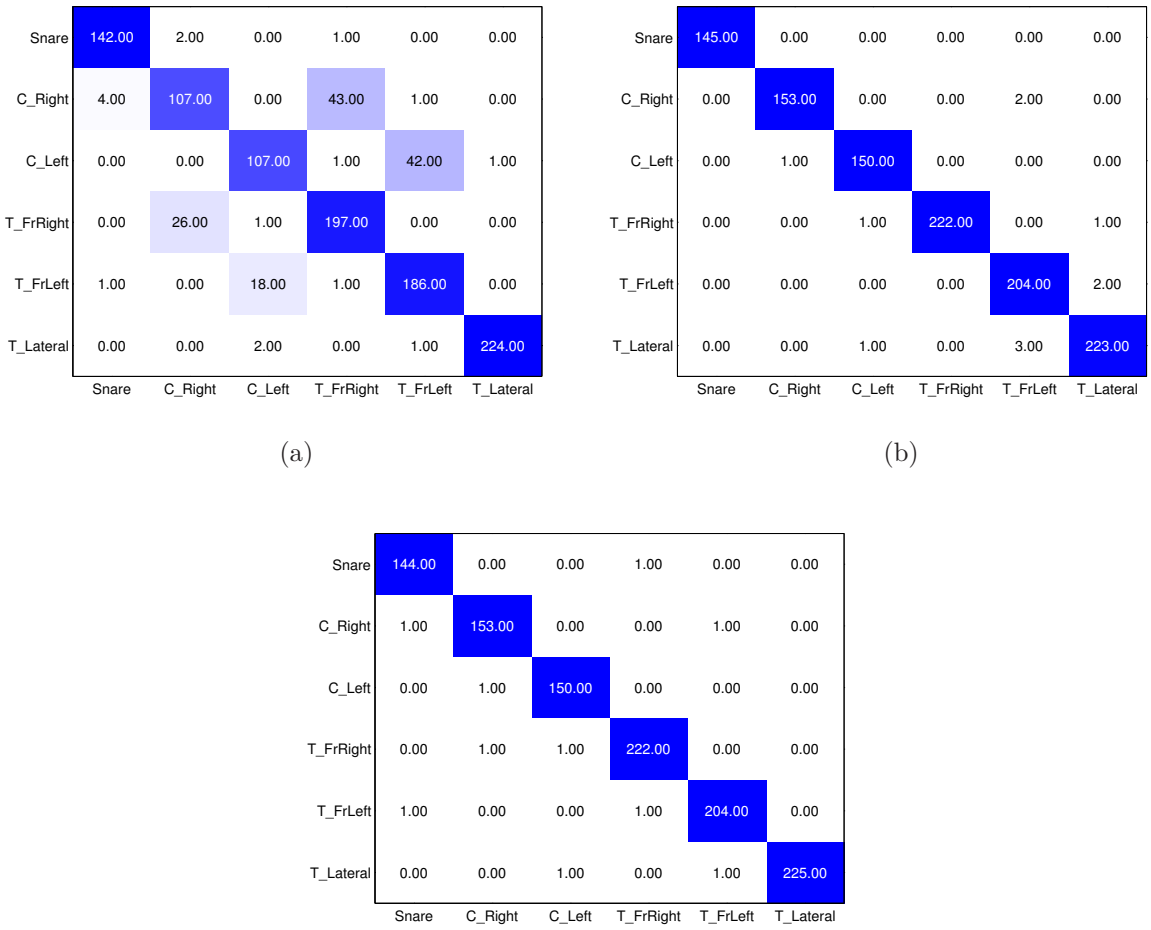


Figure 4.12: Confusion matrix for C4.5 classifier using the Trajectory features (a), Arm Pose features (b), and all features combined (c).

Feature set	Success rate	Parameter (λ)
Trajectory features	83.66%	$3 \cdot 10^{-4}$
Arm pose features	99.01%	0.01
All features combined	99.37%	10^{-4}

Table 4.6: Classification success rate for logistic regression classifier.

Logistic regression

Logistic regression classifiers estimate the relationship between a given feature vector $x = x_{i1}^{N}$ and the categorical class variable (Hosmer Jr et al. [2013], Le Cessie and Van Houwelingen [1992]) by calculating a probability score as per Eq. (4.10)

$$H_\beta = \frac{e^{\beta_0 + \sum \beta_i x_i}}{1 + \beta_0 + \sum \beta_i x_i} \tag{4.10}$$

A given sample x is classified as belonging to class y according to the value of its corresponding hypothesis probability H_β. The parameters of the model are the different β_i values, which are set to minimize the cost of errors in the classification of the training set. The cost function minimized, $J(\beta)$, is illustrated in Eq (4.11).

$$J(\beta) = \sum (H_\beta - y)^2 + \lambda \sum \beta_i^2 \tag{4.11}$$

where λ is a regularization or ridge parameter that is used to prevent the model from overfitting the specific training set used. Once again the best success rates were calculated, with λ ranging between 10^{-8} and 100. Table 4.6 presents the best classification rates and their corresponding λ parameters. Fig. 4.13 illustrate the confusion matrices.

Multilayer Perceptron

The Multilayer Perceptron classifier is based on a feedforward neural network, conforming a directed graph of successive layers fully interconnected (Haykin [1994], Haykin [2007]). Each node or neuron in the network implements a sigmoid function, and the edges connecting each pair of nodes between successive layers have a weight value assigned. When used for classification problems, the last layer usually has as many nodes as classes are. The weights of the edges are adjusted in the training stage to maximize the success rate of the classification process. The sigmoid function of each neuron is defined by the the sigmoid equation as per Eq (4.12), with z representing the weighted sum of inputs to that neuron.

$$\Phi(z) = \frac{1}{1 + e^{-z}} \tag{4.12}$$

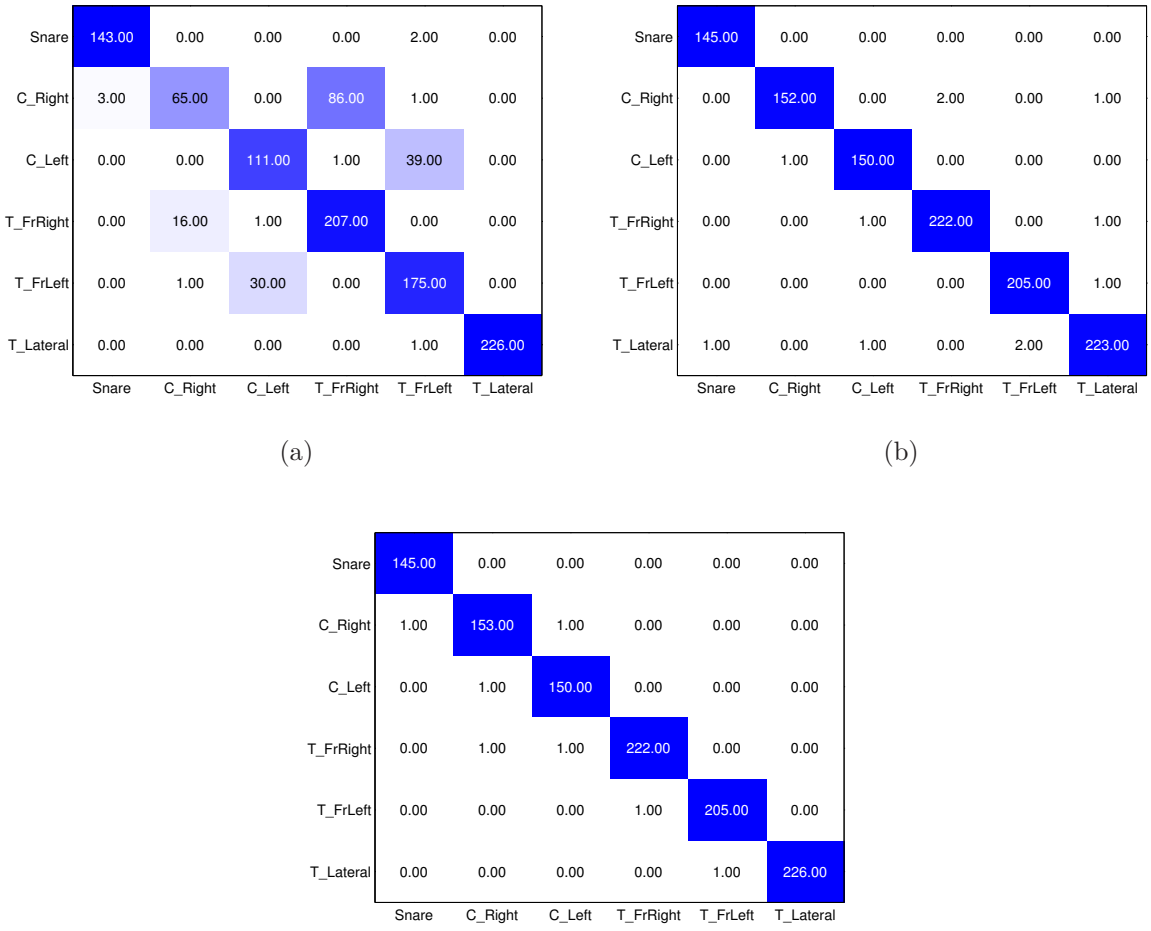


Figure 4.13: Confusion matrix for Logistic Regression classifier using the Trajectory features (a), Arm Pose features (b), and all features combined (c).

The number of neurons per layer and number of hidden layers in the neural network were adjusted to achieve a topology that offered optimal success rate for each configuration set. The number of hidden layers ranged from 1 to 3, but it was found that adding more than one layer did not yield better results in any of the cases considered; the number of neurons per hidden layer oscillated between 1 and 25. The optimal results found are present in Table 4.7 and Fig. 4.14.

Discussion

The three feature sets considered show high accurate success rate in the discrimination of gestures, with an average success rate across all classifiers of 87.54% for Trajectory features, 99.00% for Arm Pose features and 99.36% for all features combined.

Feature set	Success rate	Parameter (γ)
Trajectory features	90.88%	14
Arm pose features	99.10%	7
All features combined	99.55%	7

Table 4.7: Classification success rate for Multilayer Perceptron classifier.

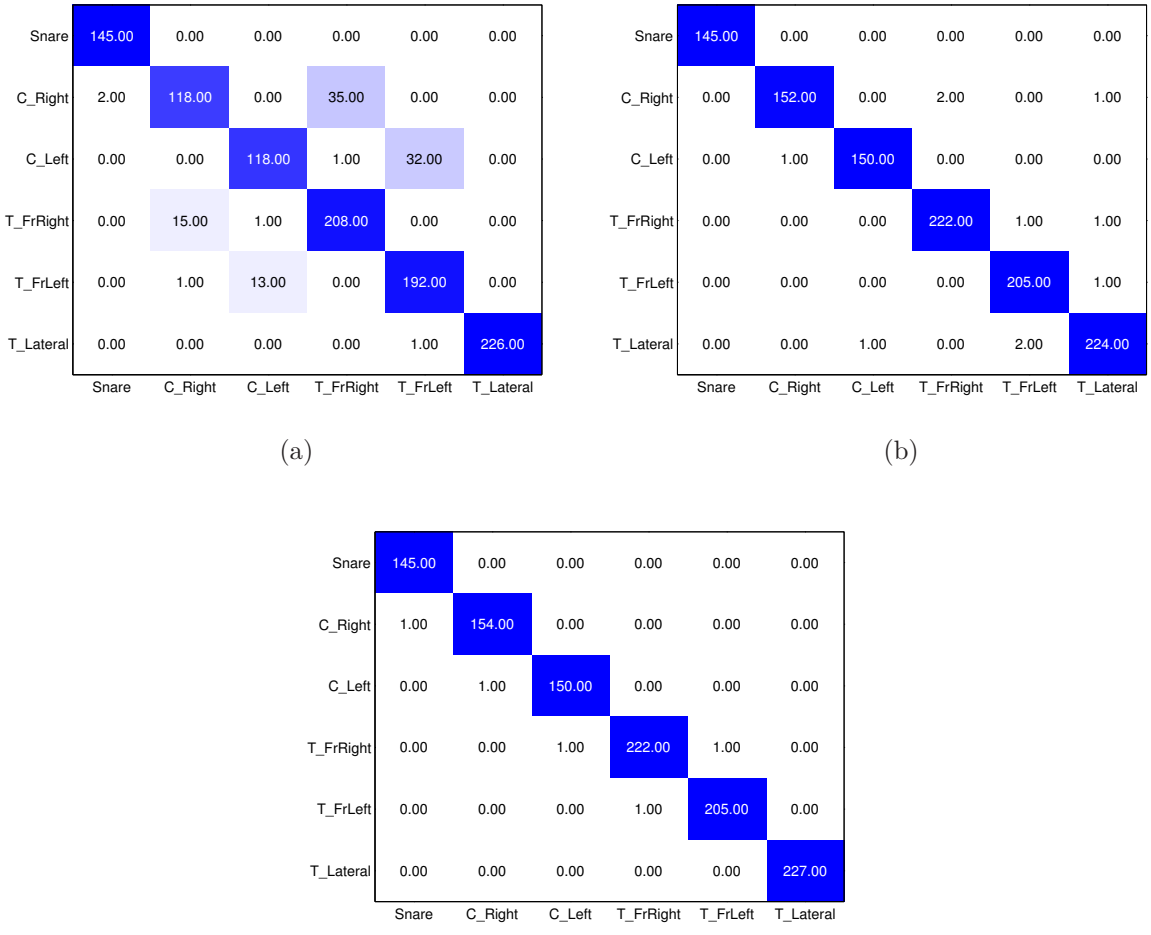


Figure 4.14: Confusion matrix for Multilayer Perceptron classifier using the Trajectory features (a), Arm Pose features (b), and all features combined (c).

The multilayer perceptron classifier is the one that gets the highest success rates: up to 99.55% when using all features, and 99.10% when using only Arm Pose features. However, when using the Trajectory feature set, *SVM* classifier is the one that has the best results (92.33%). In the case of Arm Pose features and all features combined, the classifier used does not have much of an impact on the success rates yielded. In the case of

the Trajectory feature set, though, there are meaningful fluctuations in the classification rate, ranging from 83% to 92%.

In general terms, the results yielded show that the feature models considered describe adequately the motion modelled, especially in the case of Arm Pose features.

It seems obvious, according to these results, that Arm Pose features are much more informative and a better alternative to model fast-moving gesture discrimination than Trajectory features. Furthermore, success rates for Arm Pose features alone are nearly as high as the ones had when using all features combined, yet the number of features used in each case (8 versus 23) is vastly different. This fact makes Arm Pose features alone more adequate for systems that require updating their classification models with a certain frequency.

The success rate of the system implemented is similar to the ones had in previous studies. For example, prior research has yielded success rates of 93.14% (Lee and Kim [1999]), 93.25% (Yoon et al. [2001]), 95.42% (Kim et al. [2007]) or 99.1% (Mannini and Sabatini [2010]) when using Hidden Markov Models, 96.7% (Muhlig et al. [2009]) and 96% (Li and Greenspan [2011]) for Dynamic Time Warping, 94% for SVM (Cao et al. [2009]) and 94.45% for artificial neural networks (Stanton et al. [2012]). In particular, previous studies on gesture recognition using a Kinect device have yielded success rates of 92.26% (Jacob and Wachs [2013]) and 96% (Itauma et al. [2012]).

In our study, taking into account the success rate of detecting a fast gesture, overall successful classification rates have been shown to be as high as 98.85% when using multilayer perceptron classification and all features, or even 98.41% when using Arm Pose features alone. While admittedly the set of gestures detected are not as varied as the one had in other systems (e.g. Bandera et al. [2009]), our system discriminates similar gestures with a high success rate, and more importantly, does so in a very short amount of time while compensating for the lag in the tracking system, allowing for instantaneous real-time gesture classification in interactive applications.

4.2.1.4 Experimental framework

We conducted an experiment to further assess the usefulness of the feature model defined for the implementation of an augmented reality drumkit application. The details of the experiment conducted as well as the methods followed and the results yielded are presented below.

Participants

A total of 12 participants took part in the experiment conducted, 1 female and 11 male,

with ages ranging from 26 to 34 years (average 30,67 years, variance 10,55). There were 6 graduates and 6 postgraduates. From the 12 participants, 2 had a strong formation in music, and 1 of these along with another participant were actual professional musicians. Of the remaining 9 participants, 3 had previously played a musical instrument regularly. The rest of the participants (6) were naïve musical users, with no previous formation or experience in music practice or music theory knowledge.

Materials

As the Arm Pose features proved to have high enough detection rates and the Trajectory features did not add much additional information, only the Arm Pose feature set was considered for the experiment. As previously discussed, the best success rates in gesture discrimination for the Arm Pose set were of 99.01% or 99.10%, depending on the classifier used.

The experiment was conducted at the ATIC research lab in the E.T.S de Telecomunicaciones of Málaga. The application showed a virtual representation of the user's tracked skeletal nodes for visual reference. A set of 7 sound *WAV* files were used to play the corresponding drum sounds whenever the user performed a certain drum-hitting gesture.

Procedure

Two independent experimental sessions were considered for each participant. The first session aimed to further corroborate the success rate in gesture classification (both detection and discrimination). The second experimental session focused on evaluating the capabilities of the system to prevent users from noticing lag. Each participant performed the trials assisted by a researcher, who explained him/her the details of the tests as well as observed and took notes regarding the participants behaviour during the experiment. At the end of the first experimental session, the participants were asked to fill in a questionnaire concerning their opinion on the experience; additionally, the researcher also had a casual interview with the participants regarding their overall experience and their perception of the strengths and weak points of the system.

In the first experimental session, each user was asked to perform six successive trials. An experimental factor *gesture* or 6 levels was defined to assess if the type of gesture had an effect on the success rate, each level corresponding to each of the six types of gestures which the system can detect. Each trial consisted in performing one of the six gestures considered 8 times in succession, taking the error rate of unsuccessfully detected gestures as a dependent variable. The type of error (gesture not detected, or incorrectly classified) was also observed. At the end of the first experimental session, the participants were asked to fill in a questionnaire. As part of this questionnaire, participants indicated the

level of precision, delay in the response and realism that they perceived, and they did it so for each of the six gestures considered; each item in the questionnaire was assessed with a score between 0 (least satisfactory) and 10 (most satisfactory).

The second experimental session consisted of a total of 10 successive trials. In each trial, users were asked to perform a drum-hitting gesture, and to indicate if the perceived lag in the sound that was subsequently played. Users were asked to assess the latency with a score between 0 (no lag) and 10 (high lag). In 5 of the trials, the system played the sounds without taking advantage of the forecasting capabilities of the Wiener predictor, and thus, the corresponding sound was played with a delay of roughly 160 milliseconds; in the other 5 trials, the predictor was used to nullify this delay. An experimental two-level *delay* factor registered whether there was delay or not in the system response. The average score given for each of the two cases was used as the dependent variable or indicator. In this case, no gesture discrimination was performed, and the sound played was thus always the same one to ensure minimal interference in the perception of the delay.

A repeated measures approach was followed (Howell [2011], von Ende [2001]), so that every participant was subject to all the experimental conditions considered (6 and 2 respectively). The order in which the participants performed their trials in both experimental sessions was partially counter-balanced to prevent order effects. Prior to the experiment, participants were given a short time of no more than 5 minutes to get familiar with the environment and the interface. In this preliminary session, users were allowed to use both hands to play the drums. However, for the rest of the experiment, users were asked to use only one hand when performing the drum-hitting gestures.

As part of the questionnaire presented at the end of the first trial, each participant was asked to express their opinion on the following aspects of their overall experience, and to evaluate them with a score from 0 (least satisfactory) to 10 (most satisfactory):

- Overall satisfaction with the application
- How intuitive was the interaction
- Ease of use of the application
- Level of realism perceived

4.2.1.5 Results and discussion

A repeated measures one-factor ANOVA (Neter et al. [1990], Brown [1997]), was performed on the factor *gesture* to analyse its effect on the rate of unsuccessfully classified gestures. The principal effects analysis showed a significant effect on the error rate

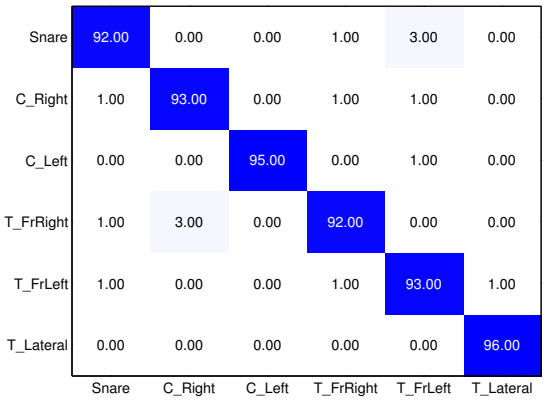


Figure 4.15: Confusion matrix for gesture discrimination in the experiment conducted

($F_{5,7} = 10.97$, $p < 0.003$). This result can be explained in the fact that the error rate for the sixth gesture (*T_Lateral*) was 0. This case was excluded in a second repeated measures ANOVA, yielding this time no significant effects whatsoever ($F_{4,8} = 0.764$, $p < 0.554$). Another repeated measures one-factor ANOVA was performed on factor *gesture*, this time to assess the effects on user perception on precision, delay and realism; again, no significant effects were found for any of the variables considered ($F_{5,7} = 0.795$, $p < 0.558$; $F_{5,7} = 1.017$, $p < 0.416$; $F_{5,7} = 0.429$, $p < 0.826$)

User estimations on the delay perceived during the second session were analysed with a paired-samples T-test on the factor *delay*. From the results yielded, it was found that the presence of the forecasting capabilities of the system did have a significant impact on the perception of lag ($T_{1,11} = 14.53$, $p < 0.000$). When there was a delay in the sound, in 85% of the cases users reported a score higher than 5, indicated that the delay was noticeable. When the predictor was in use, only 5% of the cases had a score higher than 5.

Attending to the total of 576 gestures (12 participants, with 48 gestures per participant) performed during the first session, the rate of successfully detected fast gestures was of 98.09%, and the rate of successfully discriminated gestures was of 97.74% (making a total of 95.87% successful detection rate). The corresponding confusion matrix can be found in Fig. 4.15

The averaged scores of the items in the questionnaire pertaining to the general experience can be found in Table 4.8. The overall response of the participants was quite positive, and most of them reported to have found the application interesting and enjoyable

Satisfaction	7.36
Intuitiveness	8.72
Ease of use	8.64
Realism	7.50

Table 4.8: Participants' scores attending their general experience with the application
.

Discussion

The detection rate found was of 95.87%, which is slightly lowered than the one found for the tenfold cross-validation tests previously presented for the Logistic Regression classifier with Arm Pose detection (98.41%), yet this detection rate is still pretty high, which further corroborates the effectiveness of the system developed for fast gesture recognition.

The experimental analysis on the effects of the factor *gesture* showed that the type of gesture performed has a strong effect on the successful detection rate for that gesture. In fact, further analysis showed that this effect comes from the classification rates for the *T_Lateral* gesture. From simple inspection of the previously presented confusion matrices, this result matches the behaviour found when analysing the different classifiers, and it is a reasonable finding, as the *T_Lateral* gesture is executed by performing a lateral hitting-motion, while the rest of gestures are fairly more similar in execution.

The results found showed that the presence of the forecasting capabilities of the predictor system had a significant effect on user perception of delay, thus showing that the predictor implemented does indeed compensate for the delay introduced by the tracking system, allowing for almost instantaneous interaction with sound.

Overall, participants' assessment of the user experience was quite positive. However, some of the participants (in particular, two of them who were or have had previous experience as drummers) remarked to have find some issues with the system implementation. Concretely, while they found the system to perform adequately well in general terms, they regarded they reported their concern about the capabilities of the system to recognize typical fast drum beat patterns, which usually requires shorter gestures executed in quick succession. They also reckoned the interaction metaphor to be too demanding for prolonged use of the system, potentially leading to drummer exhaustion.

As a conclusion, the research performed shows that by combining signal processing techniques with machine learning algorithms and advanced tracking system, it is possible to define novel interaction paradigms that emulate real-life instruments, thus allowing for musical exploration without the need of investing or using their real (and usually expensive) counterparts. The study conducted has shown that the use of these techniques

can also overcome the limitations of the sensing hardware (namely the delay and latency in the input) and that user experiences have been quite positive.

4.2.2 Augmented tabletop drumkit

The previously presented virtual drumkit allows users to map their movements into the execution of concrete drum-hitting gestures and their corresponding sounds, therefore providing an innovative way for musical performance.

Now, we propose an alternative way of emulating a drumkit's functionality: instead of mapping motion to sound, sounds are mapped into different sounds. Concretely, the idea is to implement a drumkit simulator without using any specific hardware aside from a microphone, providing a system capable of classifying the different types of sounds recorded (for example, tapping on the table, hitting a glass with a pen, etc.) to synthesise the selected drum sounds. This synthesis is performed through the use of MIDI sounds. Thus, for example, whenever the user stroke a glass with a pen, a cymbal MIDI file would play; if the user hits a box, a tom MIDI sound is player, while if the user instead strikes two pens together, the sound played corresponds to the one had with two drumsticks, and so on.

The input signal captured by the microphone is processed in order to extract a set of features that characterize the original sound. These features are used to classify the sound captured into one of the MIDI sounds used by the system, according to a previously trained machine-learning model stored in a database. Subsequently, the corresponding MIDI file is played.

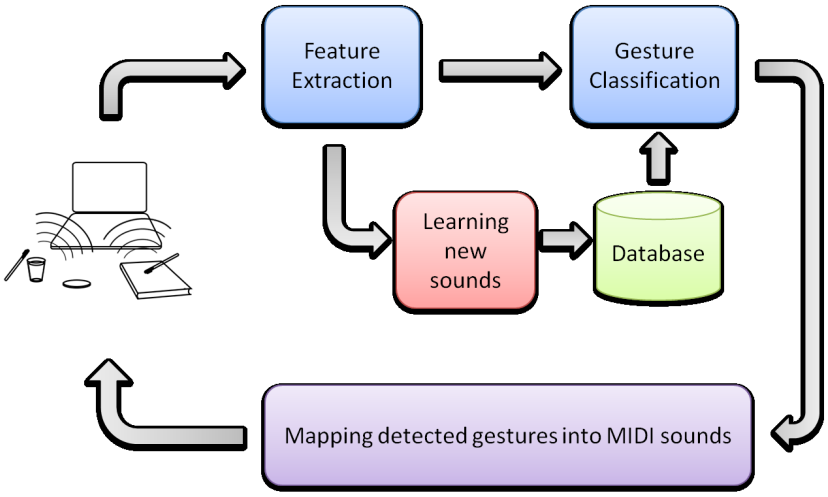


Figure 4.16: Block model of the application.

Fig. 4.16 illustrates the basic block diagram of the system's main functionalities. In

addition to the aforementioned functionality, the system also allows for on-site learning of new mappings of user-generated sounds to drumkit sounds, thus allowing the user to re-train the machine-learning model with a different training set of sample sounds.

4.2.2.1 Input processing and feature set definition

Previously to the feature extraction process, a Butterworth filter with a cutting frequency of 25 Hz is used to segment the input signal into individual beats or "strokes". Each segment is then windowed into frames using a Hamming window, and for each frame, the different descriptors or features are calculated, and the whole segment is then represented by the mean and standard deviation of each of these descriptors. This process is illustrated in the block diagram in Fig. 4.17.

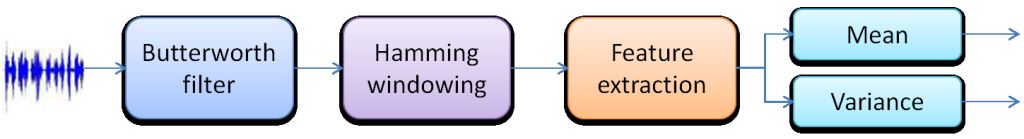


Figure 4.17: Preprocessing stage.

A total of 17 features were used in the analysis, 2 time-domain features and the rest in the frequency domain. The different descriptors are presented in lines below:

Energy

The energy of a frame i, given the input frame x_i, is estimated as

$$E(i) = \frac{1}{N}\sum_{n=0}^{N-1}|x_i(n)|^2. \tag{4.13}$$

This is a descriptor typically used in speech-music discrimination studies, as speed signals have usually low energy frames, while music signals have a more homogeneous distribution (Tardón et al. [2010]).

Zero-Crossing Rate

The zero-crossing rate describes the rate of sign changes along the length of a signal, and can be described as

$$Z(i) = \frac{1}{2N}\sum_{n=0}^{N-1}|\mathrm{sgn}[x_i(n)] - \mathrm{sgn}[x_i(n-1)]|, \tag{4.14}$$

$$\text{sgn}[x_i(n)] = \begin{cases} 1 & x_i(n) \geq 0 \\ -1 & x_i(n) < 0 \end{cases}. \tag{4.15}$$

This feature has been widely used in music and sound classification tasks as it is a good indicator of the dominant frequency (Theodoridis and Koutroumbas [2008]) and, moreover, it performs very well in percussive sounds classification (Gouyon et al. [2000]).

Spectral Centroid

The spectral centroid indicates how sparse are the different spectral components along the frequency range considered. It is defined as

$$C(i) = \frac{\sum_{m=0}^{N-1} m|X_i(m)|}{\sum_{m=0}^{N-1} |X_i(m)|}, \tag{4.16}$$

where $X_i(m)$ are the m DFT coefficients of the frame i. Perceptually, it is related to the brightness of the sound (Padmavathi et al. [2010]).

Spectral Roll-Off

The spectral roll-off point indicates the frequency value for which c% of the DFT coefficients are located below this frequency. It indicates the skewness of the spectrum (Scheirer and Slaney [1997]). Given a frame $X_i(m)$, the roll-off point $m_c^R(i)$ is determined as

$$\sum_{m=0}^{m_c^R(i)} |X_i(m)| = \frac{c}{100} \sum_{m=0}^{N-1} |X_i(m)|, \tag{4.17}$$

where N is the frame length and c is normally situated around 85-99%.

Spectral Flux

The spectral flux, defined as

$$F(i) = \sum_{m=0}^{N-1} (N_i(m) - N_{i-1}(m))^2, \tag{4.18}$$

It's an indicator of energy flux changes between successive frames. It is typically used to detect onset attack times in beat detection algorithms (Bello et al. [2005]).

Mel-Frequency Cepstral Coefficients (MFCCs)

The Mel scale consists in a rearrangement of the natural frequency scale meant to adapt

it to the logarithmic response of the human ear. One typically used transformation to Mel frequencies is as follows (O'shaughnessy [1987]):

$$f_{mel} = 2592 \log_{10} 10(1 + f/700). \tag{4.19}$$

The coefficients are then obtained by calculating a Discrete Cosine Transform of the resulting signal in the Mel scale:

$$MFCC_k = \sum x_n cos \left[\frac{\pi}{N} \left(n + \frac{1}{2} k \right) \right]. \tag{4.20}$$

These coefficients have been widely employed in instrument recognition and classification tasks (Eronen [2001]) as well as for classification of percussive sections (Gillet and Richard [2004]) and other aspects of music modelling (Logan et al. [2000]). In our study, we have consider the first twelve Mel coefficients as additional features of a given beat.

4.2.2.2 Sound classification

Prior studies have made used of machine learning classification techniques for instrument recognition (e.g. Herrera-Boyer et al. [2003]) or, in the case or percussive sounds, to determine whether the sound being played comes from the snares, toms, cymbals, hihat, etc (Herrera et al. [2002]). Nevertheless, in this study we aim to translate sounds generated from everyday objects into drumkit sounds (i.e, tapping, clapping, hitting a glass with a pen, etc.). In this regard, no similar studies have been previously performed to address or analyze the kind of percussive sounds presented in this paper.

In order to achieve our objective, we have considered four types of classifiers: Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis, naïve Bayes and k-Nearest Neighbors (k-NN).

Linear Discriminant Analysis (LDA)

LDA aims to find a linear combination of features to separate two or more classes of objects or events (Stark and Woods [1986]), assuming that all classes have the same covariation matrix. It is assumed that the features considered follow a normal distribution (Tardón et al. [2010], Ihara et al. [2007]).

Naïve Bayes Classifier

As previously indicated, a Naïve Bayes classifier is a simple probabilistic model which assumes that the different features are independent from each other. This concept may seem unintuitive, since most of the features presented are in fact correlated, yet, as it can

be seen in previous related works (Brown [1999], Basili et al. [2004]), the results can be surprisingly accurate, even if the original assumption does not hold.

Quadratic Discriminant Analysis

A generalization of LDA, in which every class has an unique covariance matrix. (Stark and Woods [1986], Breebaart and Mckinney [2002], Agostini et al. [2001]).

k-Nearest Neighbors

As previously indicated, each sample is assigned a class label according to majority voting of the k nearest training samples (Deng et al. [2008], Herrera et al. [2002], Livshin and Rodet [2004]).

Classifying model tests

The previously presented classifiers were used with the aforementioned feature sets to train several classifying models. The purpose of this study is to find the combination of features and classifiers which offers the best results for sound classification. In order to do so, a database of 400 sample sounds was manually recorded and edited. 140 of the samples were devoted to the training of the classifier, while the remaining 260 were used for cross-validation. Four classes of sounds were present in the database, labelled as A to D according to the way they are generated: hitting on a glass with a pen (class A), pounding a table (class B), snapping fingers (class C) and tapping on a table with a pen (class D). For each class, there were 35 training samples and 65 cross-validation samples. Examples for the distribution in feature space of the different samples can be found in Fig 4.18.

Tables 4.9 and 4.10 portray the error rates yielded when training each classifier with the segment mean and variance of each feature separately. It is noticeable that the features that attained a lower error rate were in general Zero-Crossing Rate, Spectral Roll-Off, Spectral Flux, MFCC1 and MFCC3. Taking this data as reference, the classifiers were trained using several combinations of features. After exhaustively testing the potential combinations, the best feature-classifier combinations found were:

- All the features with the classifiers LDA, Naïve Bayes, QDA and k-NN. It can be observed in this case that the lowest error rate is found when using either LDA or k-NN with $k = 3$, with no errors whatsoever in most cases.

- The combination of the ZCR, SRO, SF and MFCC3 features with the LDA classifier which yields an overall error rate of 4.62% (Table 4.11).

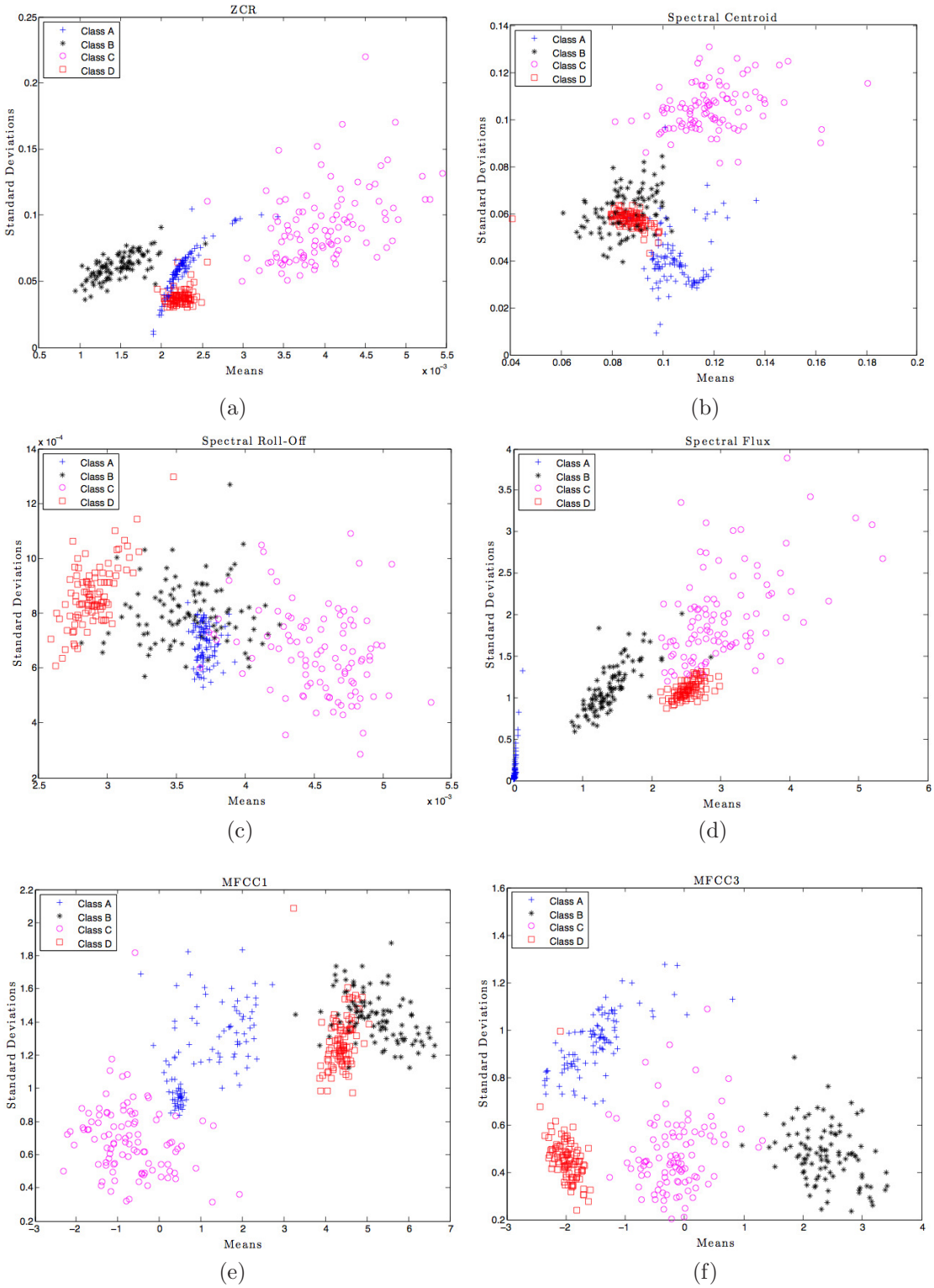


Figure 4.18: Mean vs standard deviation for each class for features ZCR (a), SC (b), SRO (c), SF (d), MFCC1 (e) and MFCC3 (f) respectively.

- QDA with the features SRO, SF and MFCC3 (Table 4.12).
- The combination of ZCR, SRO, SF, MFCC1 and MFCC3 with the k-NN algorithm, which returned similar results (Table 4.13).

These results effectively corroborate that it is indeed feasible to discriminate different types of sounds generated by hitting specific everyday objects. However, while it may be tempting to use all the features considered in the final application, it is important to consider the computational resources demanded in this case. For a more concrete example, given a sample set of 240 sounds, using all the features with the LDA classifier yielded a total processing time of 246 seconds, while using only the LDA classifier with the ZCR, SRO, SF and MFCC3 features yielded a processing time of 48 seconds, almost five times faster.

Attending to this fact, the combinations that were considered for the final application were the ones that required less computation time while keeping a low enough error rates (around 4%). Concretely, the two following combinations were chosen as the most adequate ones:

- Combination 1: QDA with SRO, SF and MFCC3.
- Combination 2: LDA with ZCR, SRO, SF and MFCC3.

LDA	EN	ZCR	SRO	SC	SF	M1	M2	M3
Class A	0.32	0.17	0.02	0.28	0.02	0.32	0.37	0.17
Class B	0.91	0.00	0.38	0.32	0.03	0.03	0.00	0.00
Class C	0.28	0.08	0.12	0.00	0.34	0.09	0.26	0.08
Class D	0.29	0.06	0.00	0.40	0.00	0.40	0.00	0.00
LDA Mahal								
Class A	0.40	0.29	0.98	0.52	0.12	0.89	0.86	0.51
Class B	0.15	0.00	0.09	0.23	0.06	0.02	0.00	0.00
Class C	0.43	0.00	0.06	0.00	0.00	0.00	0.09	0.00
Class D	0.60	0.05	0.00	0.58	0.26	0.57	0.05	0.00
Naive Bayes								
Class A	0.18	0.26	0.02	0.29	0.02	0.18	0.42	0.17
Class B	0.71	0.03	0.38	0.35	0.02	0.02	0.00	0.00
Class C	0.34	0.08	0.12	0.02	0.34	0.09	0.25	0.08
Class D	0.26	0.03	0.00	0.37	0.00	0.37	0.00	0.00
QDA								
Class A	0.35	0.28	0.40	0.43	0.06	0.86	0.80	0.51
Class B	0.37	0.00	0.22	0.42	0.05	0.02	0.00	0.00
Class C	0.43	0.00	0.09	0.02	0.18	0.02	0.14	0.00
Class D	0.20	0.05	0.00	0.05	0.00	0.43	0.00	0.00

Table 4.9: Error rate per feature (Energy, ZCR, Spectral Roll-Off, Spectral Centroid, Spectral Flux and MFCC1-MFCC3): this table represents the error rate in the classification attending to the value of the feature illustrated for the classifiers LDA, LDA with Mahalanobis distance, Naïve Bayes and QDA.

k=1	EN	ZCR	SRO	SC	SF	M1	M2	M3
Class A	0.42	0.51	0.31	0.29	0.00	0.51	0.75	0.58
Class B	0.43	0.15	0.26	0.26	0.03	0.03	0.02	0.00
Class C	0.49	0.15	0.12	0.03	0.18	0.03	0.22	0.06
Class D	0.37	0.14	0.09	0.29	0.00	0.58	0.06	0.00
k=2								
Class A	0.42	0.51	0.31	0.29	0.00	0.51	0.75	0.58
Class B	0.43	0.15	0.26	0.26	0.03	0.03	0.02	0.00
Class C	0.49	0.15	0.12	0.03	0.18	0.03	0.22	0.06
Class D	0.37	0.14	0.09	0.29	0.00	0.58	0.06	0.00
k=3								
Class A	0.38	0.58	0.12	0.29	0.00	0.66	0.42	0.46
Class B	0.45	0.32	0.25	0.25	0.03	0.02	0.02	0.00
Class C	0.55	0.25	0.11	0.02	0.20	0.08	0.25	0.05
Class D	0.40	0.11	0.00	0.28	0.00	0.52	0.02	0.00
k=4								
Class A	0.42	0.54	0.11	0.29	0.00	0.57	0.40	0.48
Class B	0.48	0.32	0.26	0.23	0.03	0.02	0.02	0.00
Class C	0.51	0.22	0.11	0.02	0.22	0.08	0.22	0.05
Class D	0.35	0.08	0.00	0.31	0.00	0.52	0.02	0.00
k=5								
Class A	0.34	0.60	0.11	0.28	0.02	0.55	0.42	0.46
Class B	0.49	0.38	0.31	0.28	0.03	0.03	0.00	0.00
Class C	0.58	0.29	0.09	0.02	0.20	0.08	0.22	0.05
Class D	0.28	0.06	0.00	0.17	0.00	0.45	0.02	0.00

Table 4.10: Error rate per feature (Energy, ZCR, Spectral Roll-Off, Spectral Centroid, Spectral Flux and MFCC1-MFCC3) when considering a k-NN classifier with k ranging from 1 to 5.

	Class A	Class B	Class C	Class D
LDA	0.00	0.00	4.62	0.00
NB	0.00	1.54	4.62	0.00
QDA	10.77	0.00	0.00	0.00
LDA Mahal	23.08	0.00	0.00	0.00

Table 4.11: Total error rate for ZCR SRO SF and MFCC3 features: this table illustrates the error rates found for each class when the classification task is performed with said features combined for the classifiers LDA, LDA with Mahalanobis distance, Naïve Bayes and QDA.

	Class A	Class B	Class C	Class D
LDA	0.00	0.00	9.23	0.00
NB	0.00	0.00	7.69	0.00
QDA	1.54	0.00	3.08	0.00
LDA Mahal	10.77	0.00	0.00	0.00

Table 4.12: Total error rate for Spectral Roll-Off , Spectral Flux and MFCC3 features: this table portrays the error rates in the classification task for the classifiers specified (LDA, LDA with Mahalanobis distance, Naïve Bayes and QDA) when the feature set considered consist of the Spectral Roll-off, Spectral Flux and MFCC3 features only.

k	Class A	Class B	Class C	Class D
1	7.69	0.00	1.54	3.08
2	7.69	0.00	1.54	3.08
3	0.00	0.00	4.62	0.00
4	0.00	0.00	6.15	0.00
5	0.00	0.00	6.15	0.00

Table 4.13: Total error rate for k-NN with the features ZCR, SRO, SF, MFCC1 and MFCC3: this table presents the error rates given when using the k-NN classifier with all the features considered in this paper. It can be observed that the error rate does not meaningfully diminish for values of $k > 3$.

4.2.2.3 Tests and results

After identifying the best combinations of classifier and feature sets, the system was further tested to evaluate its behaviour in a real scenario. A pattern of 50 consecutive strikes for each of the classes considered were recorded for the purposes of this evaluation.

Additionally, the number of classes considered was expanded to 6, introducing two new categories: class A+C and class B+C, which correspond precisely to having both classes A and C or B and C present in the same beat time. The reasoning behind this change is that, when playing a drumkit, it is usual to execute double beats, being the most common combinations hi-hat with bass drum and hi-hat with snare drum, which are represented by these two classes.

The two combinations of feature sets and classifiers previously presented were tested separately. In addition, the system was tested using 3 different sets of user generated sounds, defined as follows:

Set 1:

- **Class A**: Tapping with the thumb on a table.
- **Class B**: Tapping with the body of a pen on a table.
- **Class C**: Hitting with the point of a pen on a table
- **Class D**: Hitting with the body of a pen on a glass.

Set 2:

- **Class A**: Striking a pen on a shoe box.
- **Class B**: Tapping with a pen on the edge of a table.
- **Class C**: Tapping with a pen (point) on a table.
- **Class D**: Hitting with a pen (point) on a glass.

Set 3:

- **Class A**: Tapping with a plastic piece on a shoe box.
- **Class B**: Tapping with a plastic piece on a table.
- **Class C**: Tapping with a pen (point) on a table.
- **Class D**: Tapping with a pen (point) on a notebook.

As previously commented, we are going to consider a total of 6 classes for each set by adding Class A+B and Class B+C. Class A+B and B+C are therefore simultaneous combinations of the A and B, and B and C sounds respectively.

After re-training the classification models, the resulting confusion matrices are portrayed in Fig 4.19. It can be observed that under the new, more restricting conditions considered, combination 1 is clearly outperformed by the combination 2. In general, it can be observed that classes A+B and B+C have the highest error rates across the different sets considered.

User opinion survey

In addition to these objective tests, we also conducted a survey with a set of 9 participants, who were asked to fill a questionnaire concerning their opinion on the application developed.

The different items considered in the survey that the participants were asked to assess were:

- Perceived usefulness of the application
- Overall satisfaction
- Novelty
- Ease of use
- Interest as a cellphone application

The graph in Fig. 4.20 illustrates the results of the questionnaire. In general terms, users found the application to be amusing and interesting. The biggest issues reported concerned the fact that system does not work in real time, but instead requires some time to process a given beat pattern, classify it and play the associated MIDI files.

As conclusion from the research conducted, it can be stated that the proposed approach provides a novel and satisfying form of interaction with music, that can be easily implemented with simple materials (it only requires a processing device and a microphone). Providing that the processing times are reduced further, there is a big potential for the use of this system as a smartphone application.

	A	B	C	D	A+C	B+C
A	50	0	0	0	0	0
B	0	50	0	0	0	0
C	0	0	38	0	12	0
D	0	0	0	50	0	0
A+C	5	3	0	2	40	0
B+C	0	0	0	1	12	37

(a)

	A	B	C	D	A+C	B+C
A	50	0	0	0	0	0
B	2	48	0	0	0	0
C	0	0	50	0	0	0
D	0	0	0	50	0	0
A+C	6	8	0	0	36	0
B+C	9	0	8	6	0	27

(b)

	A	B	C	D	A+C	B+C
A	50	0	0	0	0	0
B	2	47	0	1	0	0
C	0	0	50	0	0	0
D	0	0	0	50	0	0
A+C	0	0	11	4	35	0
B+C	13	0	0	2	1	34

(c)

	A	B	C	D	A+C	B+C
A	50	0	0	0	0	0
B	0	50	0	0	0	0
C	0	0	50	0	0	0
D	3	0	4	43	0	0
A+C	0	0	0	0	50	0
B+C	0	0	0	6	14	30

(d)

	A	B	C	D	A+C	B+C
A	50	0	0	0	0	0
B	0	50	0	0	0	0
C	0	0	49	0	0	1
D	0	0	0	40	0	0
A+C	0	0	0	0	49	1
B+C	0	0	0	0	12	38

(e)

	A	B	C	D	A+C	B+C
A	50	0	0	0	0	0
B	0	49	0	1	0	0
C	0	0	50	0	0	0
D	0	0	0	50	0	0
A+C	0	0	3	0	47	0
B+C	0	0	0	0	0	50

(f)

Figure 4.19: Confusion matrices for the different combinations of sounds sets and classifiers-features configurations considered. The resuls for Configuration 1 are presented in (a), (b) and (c) for the 3 sets of sounds respectively, and similarly, (d), (e) and (f) illustrate the results for Configuration 2

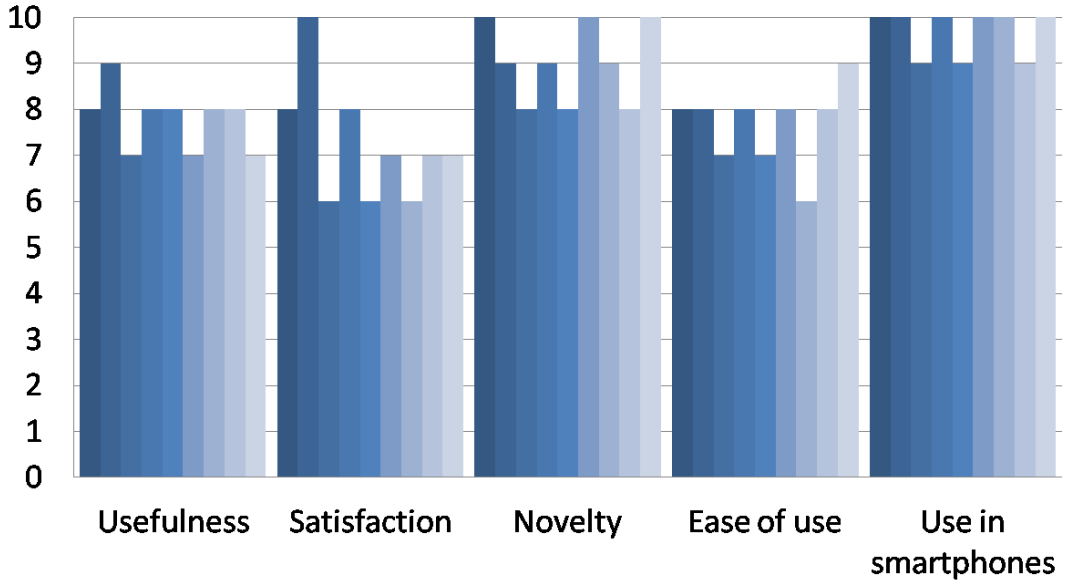


Figure 4.20: Results of survey on user opinion

4.2.3 Virtual theremin

A theremin or thereminophone is a particular type of electronic musical instrument that is played through the use of arm motion without actual physical contact between the performer and the instrument itself (Theremin [1928]). It is named after the westernized name of its Russian inventor, Léon Theremin, who patented the device in 1928. This instrument has a couple of metal antennas that sense the position of both user's hands, each of them controlling a certain aspect of the performance: one of the hands (the right hand usually) controls the pitch or frequency of the sound played, while the other hand changes its volume.

Given its nature, this is a played-through-motion instrument, without physical contact whatsoever. Thus, the implementation of a virtual simulator of a theremin is a problem that can be addressed with current motion-tracking technologies.

Concretely, the use of a camera-based sensor such as Kinect perfectly fits the implementation of this type of interaction metaphor. Control over volume is quite direct to achieve, as most of current audio APIs allow for direct modification of this parameter. Thus, the real interest in the implementation of a virtual theremin comes in handing how to map motion into pitch changes.

In signal processing, pitch shifting is the process of changing the pitch of a given piece or sample without affecting its duration or speed (Laroche and Dolson [1999]). Given a digital audio excerpt, it is possible to rebuild the original continuous waveform and resample it again at a different rate, hence effectively changing it pitch, but also its duration. In order to keep the duration unmodified, it is necessary to perform a time-scaling process by the same factor as the given resampling.

Another possibility is to use a phase vocoder (Zölzer et al. [2002]). The phase vocoder is a time-frequency processing technique that uses short-time Fourier analysis and synthesis to transform a given data signal, according to the equation of the short-time Fourier transform (STFT) with a window given by $h(n)$,

$$X(n,k) = \sum_{m=-\infty}^{\infty} x(m)h(n-m)W_N^{mk} \tag{4.21}$$

$$k = 0, 1, \ldots, N-1, \quad W_N = e^{-j2\pi/N} \tag{4.22}$$

Working with this equation, it is possible to implement the phase vocoder using two different models (Zölzer et al. [2002]): the filter bank summation model and the block-by-block analysis/synthesis model.

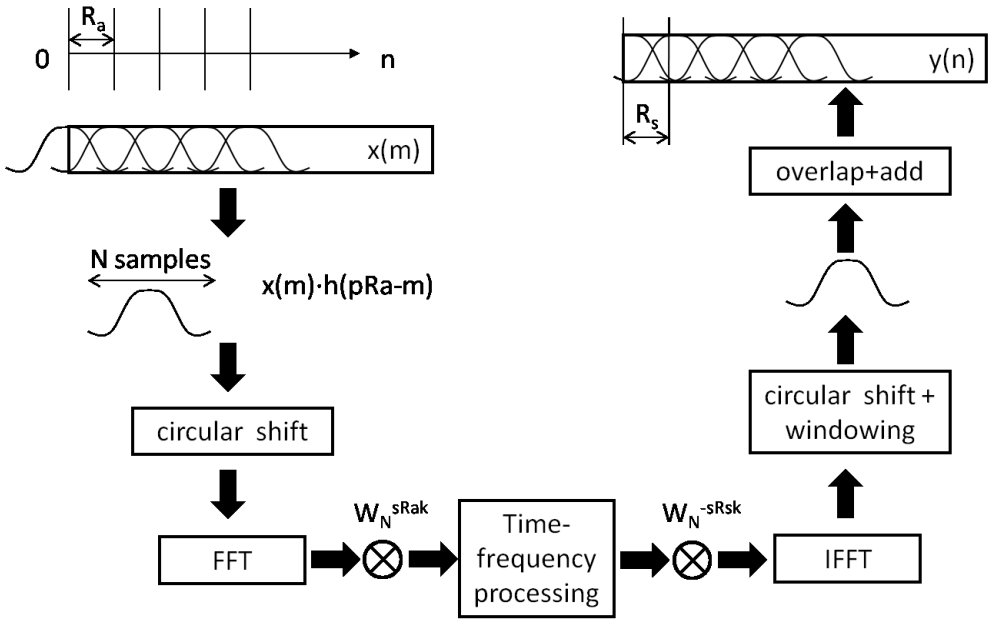


Figure 4.21: FFT/IFFT block implementation for phase vocoder.

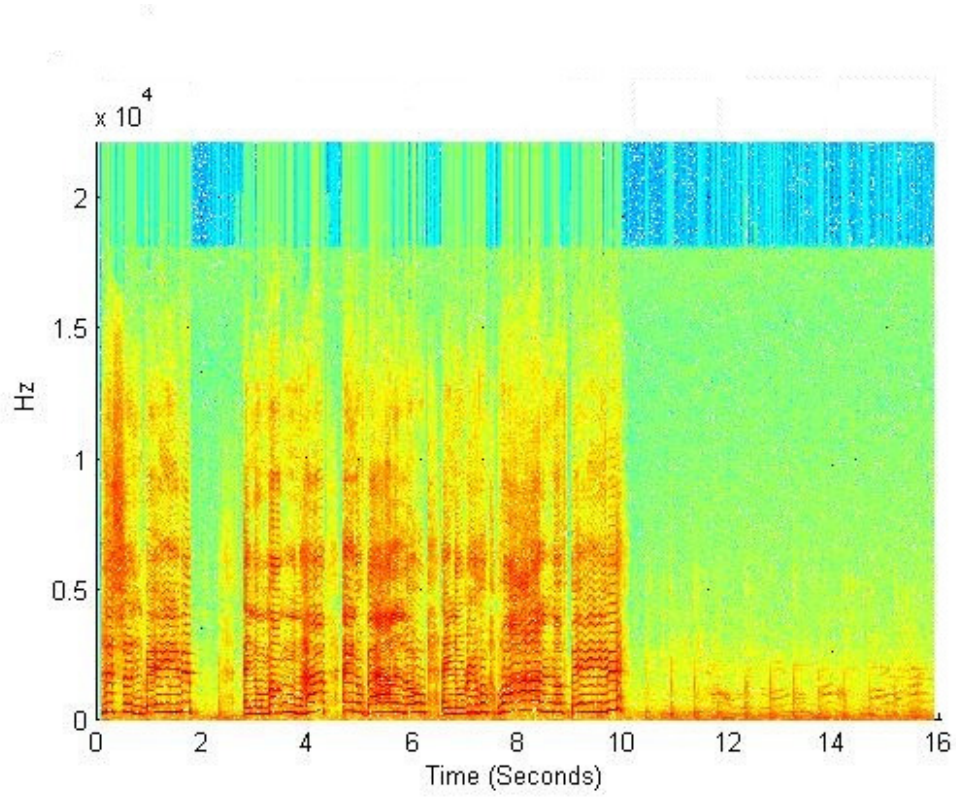


Figure 4.22: Spectrogram for a non-pitch-shifted vocal audio excerpt.

Concretely, we used a block-by-block analysis/synthesis model for its implementation. This model is based around the use of the fast fourier transform and its inverse (FFT/IFFT), dividing the input signal in overlapping segments, given a certain hop-size R_a. The spectrum of each segment is calculated, adding additional processing to ensure phase coherency among segments. The spectral components are transformed according to the processing desired, and the resulting output signal is re-synthesized in the time-domain, combining the successive processed segments by an overlap and add method with hop-size R_s. This process is portrayed in figure 4.21.

For each one of the short-time spectra calculated, it is possible to manipulate both the magnitude and the phase of each frequency bin directly before re-synthesizing the signal back to the time-domain. Thus, in order to perform a pitch-shifting operation, the spectral components of each frame have their frequency value scaled accordingly by the pitch-shifting factor desired. In the synthesis stage, the original magnitude values are kept, yet the phase values for each frequency bin are extracted from the transposed frequency values. An example of a vocal signal and its pitch-shifted version by 3 semitones using this algorithm can be found in Fig. 4.22 and 4.23 respectively.

Regarding the gesture control model, the user can modify the pitch of the piece being played in real time by simply raising or lowering his right hand, as illustrated in Fig. 4.24.

Concretely, the pitch-shift factor depending on the relative distance between the user head and his right hand along the Y axis. If the hand was placed raised over the head's level, the pitch-shift factor was greater than one, and viceversa. Pitch transposition is limited to a maximum of 12 semitones with respect to the original value (corresponding respectively to pitch-factors of value 2.0 and 0.5). The highest pitch shift is applied when the user's right hand Y coordinate is approximately 50cm higher than the head's one. Similarly, the lowest pitch shift possible is performed when the hand is about 50cm smaller than the user's head height. In intermediate positions, the pitch shifting introduced is continuously distributed along the interval of semitones considered (-12,12) proportionally to the relative height of the hand with respect to the head.

A real theremin would normally play a single tone and modify its pitch and magnitude. However, in our implementation, any audio excerpt played in a loop can be used as input for the system.

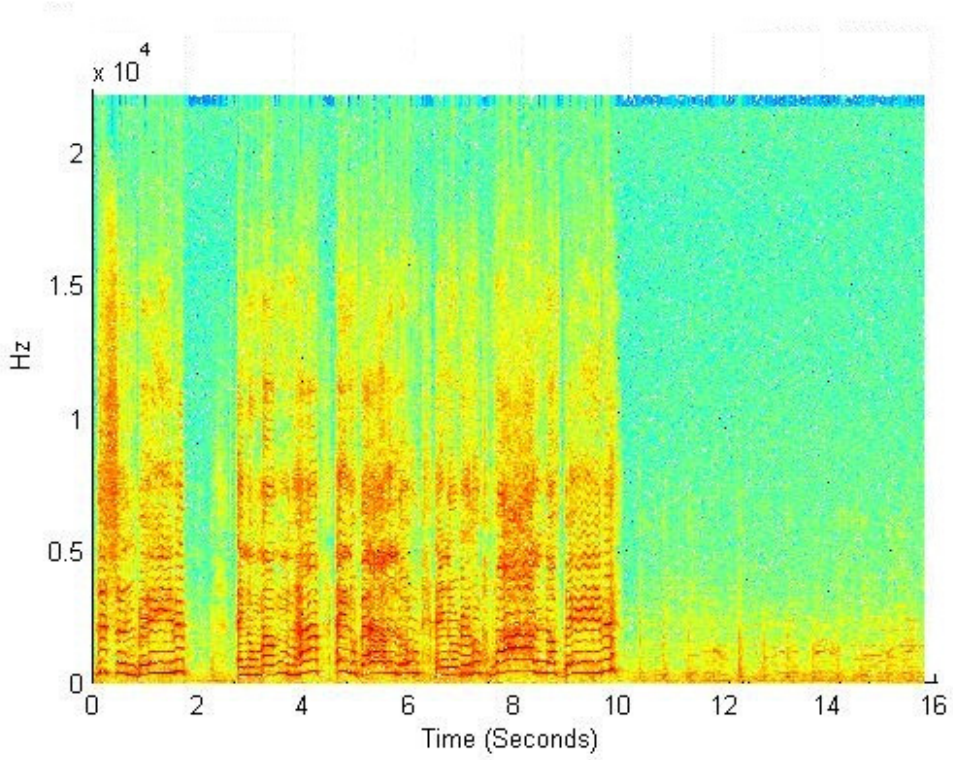


Figure 4.23: Spectrogram for a pitch-shifted vocal audio excerpt (3 semitones factor).

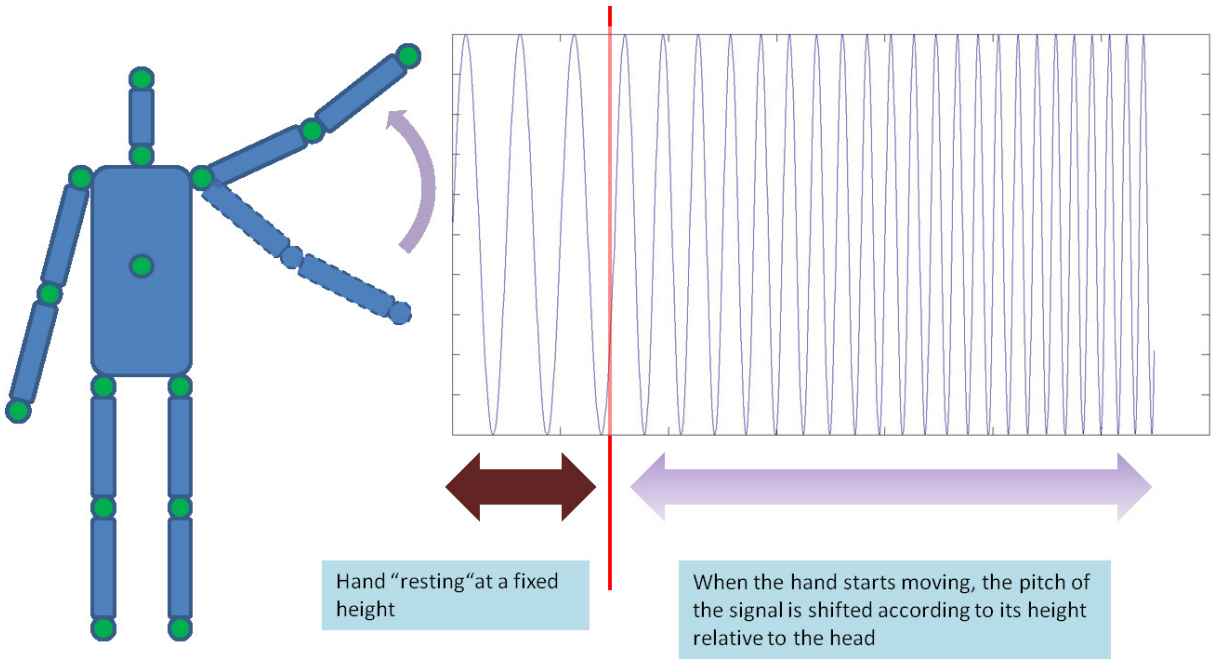


Figure 4.24: Example of pitch-shifting of a single tone. When the hand moves from the blurred position to the end position, the pitch shift factor changes according to the height of the hand at each instant.

4.3 Interactive musical experiences

Section 4.2 has been devoted to the application of advanced user-computer interaction metaphors for the specific task of instrument simulation. However, advanced machine-person interfaces are not limited to the simulation of virtual or augmented musical instruments, and there are many other types of applications in which the use of these interfaces allows for more enriching experiences.

In this section, we present the work performed towards this aim, to assess the effects of innovative interaction paradigms on user experience and satisfaction. Concretely, we have studied the use of these interfaces for the specific cases of two concrete application: the simulation of the role of the ensemble conductor, and the simulation of a steps-aerobics instructor.

4.3.1 Conducting a virtual ensemble

The aim of this study is to present a new interaction model for conducting gesture recognition, so that the user can effectively conduct a virtual orchestra, signalling through gestures the tempo and beat times of the piece performed, as well as the overall dynamics and the specific volume levels for each of the instrument sets taking part in the ensemble. The effects on user experience will be assessed through the use of an exhaustive user study by conducting a thorough experiment.

In order to properly capture the feel of conducting an ensemble, it becomes necessary to track the motion of both hands, and thus, once again, the use of a camera-based sensor offers the most effective and non-intrusive way to implement such functionality.

In this regard, we used a technological framework similar to the one had for the virtual drumkit, resorting to a Microsoft Kinect for XBOX device as the hardward basis in our human-computer interface design, again using the OpenNI and NITE APIs in order to track hand positions in 3D cartesian space. A virtual environment resembling a concert hall was coded in C/C++ using the OpenGL graphics library (Shreiner [1999]) and the OGRE graphics engine (Junker [2006]) 4.25). The purpose of this environment is to provide a visual reference of the different action the conductor is performing, and, specially, the instrument set that are currently selected for the conductor to issue commands. PortAudio library is used for sound management. WAV files were used to store and read the corresponding tracks for each of the sets of instruments in the virtual ensemble.

In order to adequately implement an ensemble conductor simulator, there are two main problems that need to be addressed: how to translate conducting gestures to changes in the performance, and how to smoothly change the tempo of the piece being played. These

Figure 4.25: Virtual environment for the application

issues will be discussed further in the sections below.

4.3.1.1 Dynamic modification of tempo in real-time

Real-time modification of tempo requires the system to be able to adjust the speed of the piece being played dynamically, according to the commands issued by the conductor. However, the duality of time and frequency implies that any modifications in the playback speed are invariantly linked to proportional changes in the performance's pitch: slower tempos will result in lower pitch levels, and viceversa.

In order to avoid these pitch artifacts, it is necessary to use a time-stretching algorithm. Time-stretching is a process that allows for smooth changes in the duration of a given input signal while keeping its frequency spectrum unmodified (Malah [1979]). The most typical implementation of time-stretching algorithms in time-domain responds to the so-called Synchronous Overlap-and-Add algorithm or SOLA (Zölzer et al. [2002]). SOLA divides the input signal into successive segments, and then adds these segments together with a certain overlap, as illustrated in Fig. 4.26. The overlapping is performed using

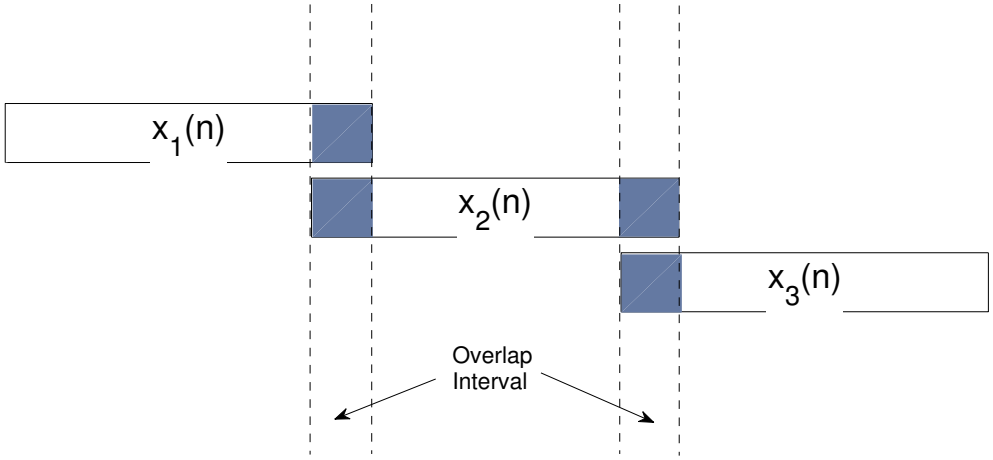


Figure 4.26: Time-stretching in time-domain

a "fade-in" and "fade-out" process, so that the last part of each segment progressively decreases its energy until disappearing, while the first part of the next segment increases its energy little by little, until reaching its nominal level. The mixing of both overlapping sections take into account the cross-correlation between themselves, thus maximizing the smoothness in the transition.

However, after testing the use of this algorithm for our purposes, we decided to discard it. SOLA is computationally fast, but works best with relatively simple signals (such as speech (Malah [1979])), and it does not work so well with polyphonic data, which is the case of an ensemble performance. Additionally, the time-stretching range is limited, and for a time-stretching factor of, for example, 1.5, several distortion artifacts become very noticable in the piece processed.

Alternatively, we resorted to the use of a frequency-domain time-stretching algorithm based on a phase vocoder (Zölzer et al. [2002]), in the same way as the one used for the virtual theremin 4.21, with hop-size R_a for the analysis stage and hop-size R_s for the synthesis stage.

A time-stretching process with a phase-vocoder requires that the spectral components are the same, yet the hop-sizes for analysis and synthesis (R_a and R_s) must selected accordingly to the time-stretching factor desired (R_s/R_a). The phase value of each frequency bin must also be adjusted accordingly (Zölzer et al. [2002]).

In our application, the time-stretching factor had a value ranging from 0.5 to 2.0. The tests performed using the MATLAB environment showed that the signal was effectively

transformed without the appearance of the distorted artifacts otherwise found with the SOLA algorithm.

4.3.1.2 Gesture recognition and interpretation

In a real ensemble, the conductor is responsible for providing expressiveness to the performance, managing the different parts of the orchestra, indicating tempo changes, beat times, entry points for the different instrument sets and orienting the dynamics of the piece.

The role of the conductor is critical and very difficult and daunting to emulate for a naïve user. In order to ease this burden, the gesture recognition model chosen has been kept simple enough so that it can be used by expert and lay users alike. In particular, right-hand waving gestures control the tempo of the performance as well as the time positions of the beats, while left-hand gesturing is used to select a given instrument set and lower or raise its volume.

Instead of using the standard signalling model for music beats, beat times are indicated by simply moving the right hand in an horizontal waving motion. The application keeps track of the start and stop times for each waving motion, and the time difference between them is used to dynamically modify the tempo of the piece played, by updating the time-stretching factor (T_s) according to the relation between the original tempo of the piece in play and the tempo signalled by the conductor.

$$T_s = \frac{beatsPerMinuteOriginal}{beatsPerMinuteIndicated} \tag{4.23}$$

In order to prevent false positives in case of noisy measures, an additional restriction was in place for a given detected waving motion to be taken into account for the indication of a new tempo. Concretely, the length of the overall gesture is calculated, as per the following equations:

$$u(t) = \begin{cases} 1 & \text{if } \mid \frac{\overrightarrow{dp(t)}}{dt} \mid \geq V \\ 0 & \text{otherwise} \end{cases} \tag{4.24}$$

$$d(t) = \sum_{n=0}^{N_{start}} |(\overrightarrow{p(t - nT_f)} - \overrightarrow{p(t - (n-1)T_f)})|u(t) \tag{4.25}$$

where $\overrightarrow{p(t)}$ is the 3D vector position of the right hand at instant t, T_f represents the time between frames (roughly 30 milliseconds), N_{start} is the last known sample for which $u(t)$

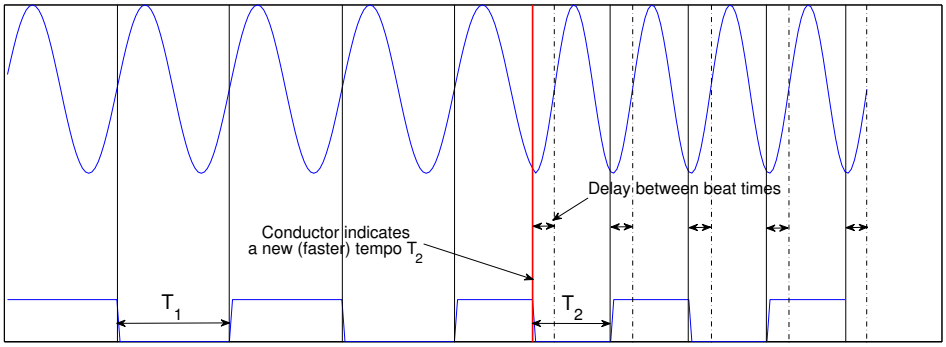


Figure 4.27: Delay effect when beat times are not properly synchronized.

changed to a value of 1 and V is a minimum velocity value (set at approximately 0.2 m/s).

Thus, given a waving gesture, if that gesture is performed horizontally and the accumulated distance moved $d(t)$ exceeds a certain minimum value L, it is assumed that the user has performed a conducted gesture. L was set to a reasonable value for such waving gestures, but long enough so that no arbitrary noise could trigger a false positive (approximately 400 mm).

In addition to noise, there is another consideration to make for the system to work as intended: it is important that the beat times of the piece being played are coincident with the beat times indicated by the conductor. The conductor does not normally keep track of the velocity of his hand in order to modify the tempo of the ensemble's performance. Instead, he listens to the ensemble's performance, and changes its tempo by signalling new beat times. If the ensemble simply changes its tempo accordingly but does so in a way that the beat times are not synchronized, it will result in a phase difference between the beat times indicated by the conductor, and the actual beat times of the piece played. Such a situation creates the feeling that the orchestra is too slow and cannot follow the conductor gestures appropriately.

The aforementioned problem is illustrated in Fig. 4.27, using a simple sinusoidal wave whose period is modified under the previously presented model. If the conducted tempo only changes the playback tempo without taking the beat times into consideration (represented in the figure by the instants where the sinusoidal wave has a phase value of 0 radians), a delay is introduced: the beat times of the piece played come at later time than the beat times indicated by the conductor.

In order to properly synchronize both beat patterns, it is necessary to take into account not only the velocity of the conductor's waving motion, but also the next expected beat

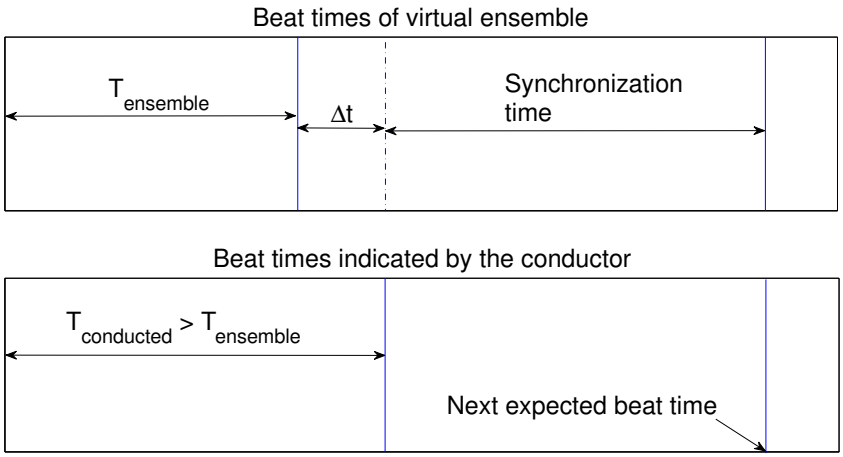


Figure 4.28: Beat time synchronization when the conductor indicates a slower tempo.

time to be indicated by the conductor. This assumption only makes sense if the tempo between beats is expected not to change too abruptly, but this is a reasonable assumption for the performance of an orchestra in real life.

In particular, if the user conducts the virtual ensemble towards a slower tempo, the ensemble must actually play at an even slower tempo than the one indicated in order to synchronize its beat times with the ones of the conductor, and vice versa. This situation is portrayed in Fig. 4.29.

In this figure, the user conducts the ensemble towards a slower pace, yet the system does not realize this until a time of Δt seconds has passed. In order to match the conductor's next expected beat time, the actual tempo of the piece played must be decreased further for the time period denoted as "synchronization time". Fig 4.28 illustrates the analogous situation when the conductor signals a faster tempo.

Therefore, the actual time-stretching factor T_s is updated accordingly to this time difference Δt, according to the following equations:

$$T_s = \begin{cases} \dfrac{T_{conducted}}{T_{conducted} - \Delta t} T_s & \text{if } T_{conducted} > T_{ensemble} \\ \dfrac{T_{conducted}}{T_{conducted} + \Delta t} T_s & \text{if } T_{conducted} < T_{ensemble} \end{cases} \tag{4.26}$$

However, using these equations to automatically update tempo times, while correct, gives a very robotic and unnatural response on behalf of the virtual ensemble. In fact, given a real orchestra, the musicians would not probably change the tempo in their performance instantly with the motion of the conductor, but would rather do it over a period of time. Thus, in order to offer a more natural response, instead of automatically updating

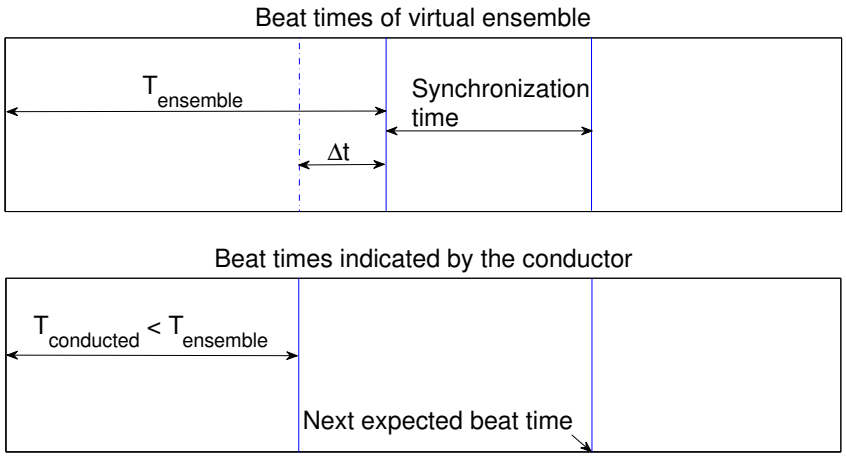


Figure 4.29: Beat time synchronization when the conductor indicates a faster tempo.

the tempo to the new value indicated by the conductor, the system dynamically updates the tempo of the piece played until both system and conductor beat times are sufficiently synchronized. Concretely, the tempo is slowed or accelerated by adding a factor of ± 0.025 to the timestretching factor at a rate of 4 times per second (thus, the timestretching value is updated in intervals of 250 ms).

Regarding dynamics control, the left hand was used to select a given instrument set in the ensemble and to raise and lower the volume of its corresponding track. An instrument set is selected by pointing towards it (using the coordinates of the left shoulder and left hand as reference to create the pointing vector), and raising or lowering the hand modifies subsequently the volume for that instrument set. The currently selected instrument is indicated by placing a red arrow over the image that represents that instrument set (see figure 4.30).

4.3.1.3 Experimental setup

An experiment was conducted to assess user experience when using the previously presented system.

Participants

A total of 24 participants took part in the experiment conducted, 3 female and 21 male, with ages ranging from 23 to 34 years (average 29,71 years, variance 10,30). There were 1 undergraduate, 15 graduates and 8 postgraduates. From the 24 participants, 2 had a strong formation in music, and 1 of these along with 2 more participants were actual professional musicians. Of the remaining 20 participants, 4 had played previously a musical

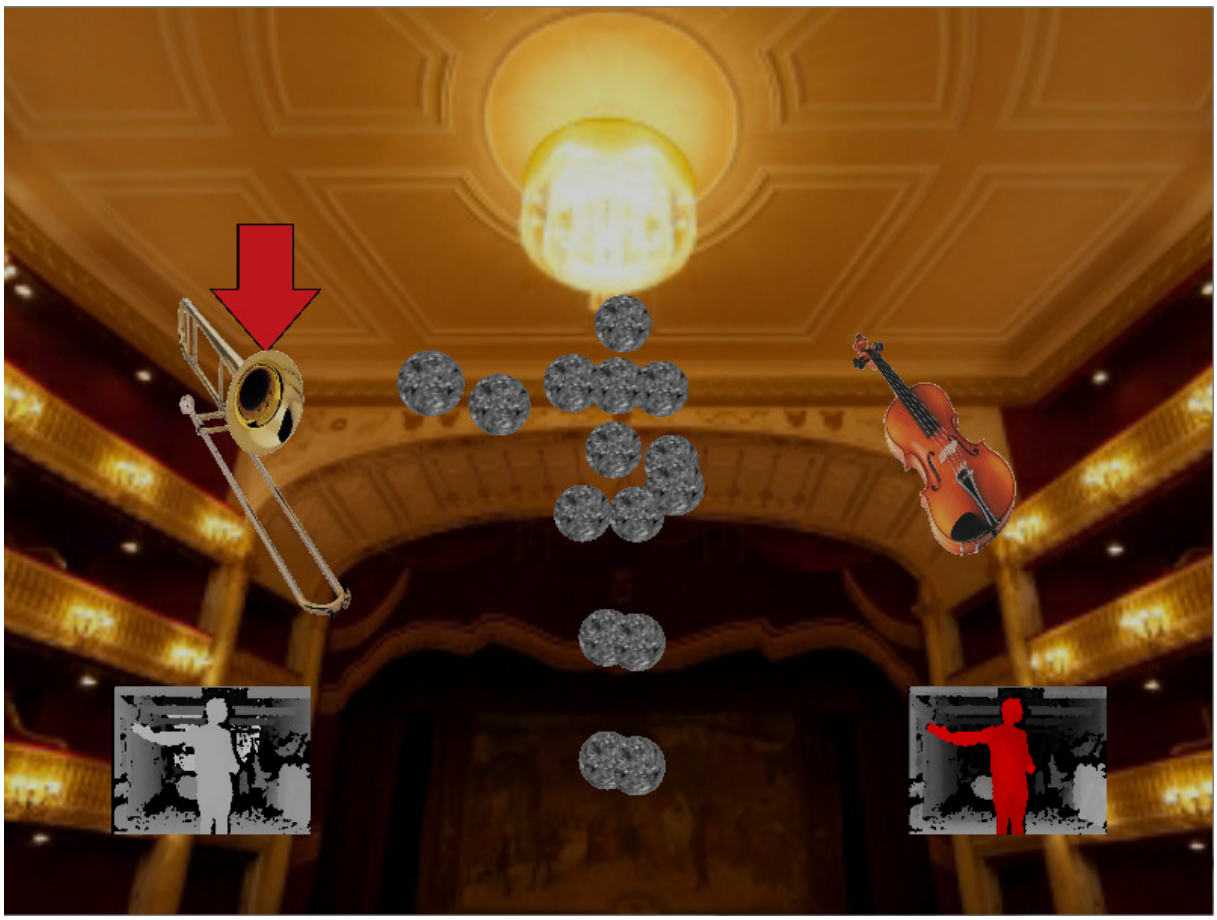


Figure 4.30: Instrument selection for dynamics control

instrument regularly. The rest of the participants (16) were naïve musical users, with no previous formation or experience in music practice or music theory knowledge.

Materials

The application was based on the previously presented interface and virtual environment. The virtual ensemble performance was represented by an excerpt of Peer Gynt's "In the hall of the mountain king", which was played constantly in a loop. Two separate tracks were used for the experiment, corresponding to the instrument sets for the violin and the trombone respectively.

Procedure

The experiment was performed in the ATIC research lab in the School of Telecommunications of Málaga. A researcher assisted the participant during the trials, giving indications about the tests as well as observing and taking notes on their behaviour. Participants were instructed to use their right hand waving motion to conduct the tempo in the en-

semble, and their left hand to indicate changes in dynamics (in the same way as described in the previous section). At the end of the experiment, the researcher interviewed the participants and asked them to fill in a questionnaire regarding their satisfaction and personal experience and their view on the application.

A previous pilot study with 4 participants without musical knowledge or experience had showed that users did not notice the effects of having the conducting beat times synchronized with the ensemble beat times, and seem to be content with the fact that by simply waving their hands, they could control the tempo in the performance. Also, the pilot study showed that users found the gesture recognition system for dynamics control to be cumbersome and detrimental to the experience.

In order to further assess these issues, we defined two experimental factors in our study: a *tempo* factor and a *dynamics* factor. Both factors had two levels. In the case of the *tempo* factor, these levels correspond to the presence or absence of the beat time synchronization algorithm. For the *dynamics* factor, the two levels refer to the presence or absence of the left hand gestures in the interface.

The combination of the two factors yields a total of $2 \times 2 = 4$ experimental conditions. A repeated measures approach was followed (Howell [2011], von Ende [2001]), so that there were 4 experimental sessions for each participant, each session corresponding to one of the aforementioned experimental conditions. To avoid order effects, the order in which the participants performed their sessions was fully counter-balanced. At each session, each participant was told to spend as much time as they deemed necessary trying out the application, with the constraint that each session was scheduled to last a minimum of 2 and a maximum of 7 minutes.

Data retrieval on user experience

The items in the questionnaire were evaluated by the participants for each of the experimental conditions separately. Thus, for each item in the questionnaire, a dependent variable was created and monitored. Users were asked to evaluate the following aspects of their experience with a score from 0 (least satisfactory) to 10 (most satisfactory):

- Overall satisfaction with the application (*Satisfaction*)
- Level of control over the parameters of the piece played (*OverallControl*)
- Level of control over the tempo of the piece played (*TempoControl*)
- Synchronization between motion and the changes in the piece played (*Synchronization*)

- How intuitive was the interaction (*Intuitiveness*)
- Ease of use of the application (*EaseOfUse*)
- Level of realism perceived (*Realism*)

The names of the corresponding dependent variables are indicated between brackets. Participants were also encouraged to state personal comments and impressions regarding their experience with the application.

4.3.1.4 Results and Discussion

A repeated measures two-factors ANOVA (Neter et al. [1990], Brown [1997]) 2×2 was performed on the factors *tempo* and *dynamics* previously defined. The principal effects analysis for the *tempo* factor had a significant effect on the variables *Satisfaction*($F_{1,23} = 25.09$, $p < 0.000$), *OverallControl*($F_{1,23} = 18.81$, $p < 0.000$), *TempoControl*($F_{1,23} = 21.49$, $p < 0.000$), *Synchronization*($F_{1,23} = 15.02$, $p < 0.001$) and *Realism*($F_{1,23} = 6.27$, $p < 0.020$). In the case of the *dynamics* factor, there was a significant effect on the variables *Satisfaction*($F_{1,23} = 9.75$, $p < 0.005$), *OverallControl*($F_{1,23} = 9.37$, $p < 0.006$), *Synchronization*($F_{1,23} = 15.02$,$p < 0.005$), *Intuitiveness*($F_{1,23} = 5.28$, $p < 0.031$) and *EaseOfUse*($F_{1,23} = 13.80$, $p < 0.001$). The estimated marginal means for the variables *Satisfaction*, *OverallControl*, *TempoControl* and *Synchronization* are presented in figure 4.31.

The quantitative effects that each of these factors had on the average values for each of the variables observed are summarized in table 4.14. It can be observed that the presence of beat time synchronization algorithm in all the cases were the *tempo* factor had a significant effect imply a higher score. In the case of the *dynamics* factor, the situation is similar, except for the cases of the *Intuitiveness* and *EaseOfUse* variables, which score worse when the left hand was allowed to control each instrument set's volume.

No significant second order interactions were found between the two experimental factors considered ($F_{1,23} <= 3.01$ for all the variables observed). Overall, the average scores reported across the different experimental conditions suggest that participants found the experience to be quite enjoyable and positive, with *Intuitiveness* being the item in the questionnaire that scored the highest values and *EaseOfUse* being the one that presented the highest variance, as can be seen in Fig. 4.32.

The results yielded from the experiment conducted showed a quite positive response from the participants that took part in the experiment. It is remarkable the fact that, according to the personal interviews had with each participant, most of them did not consciously notice that beat times in the ensemble were not properly synchronized with

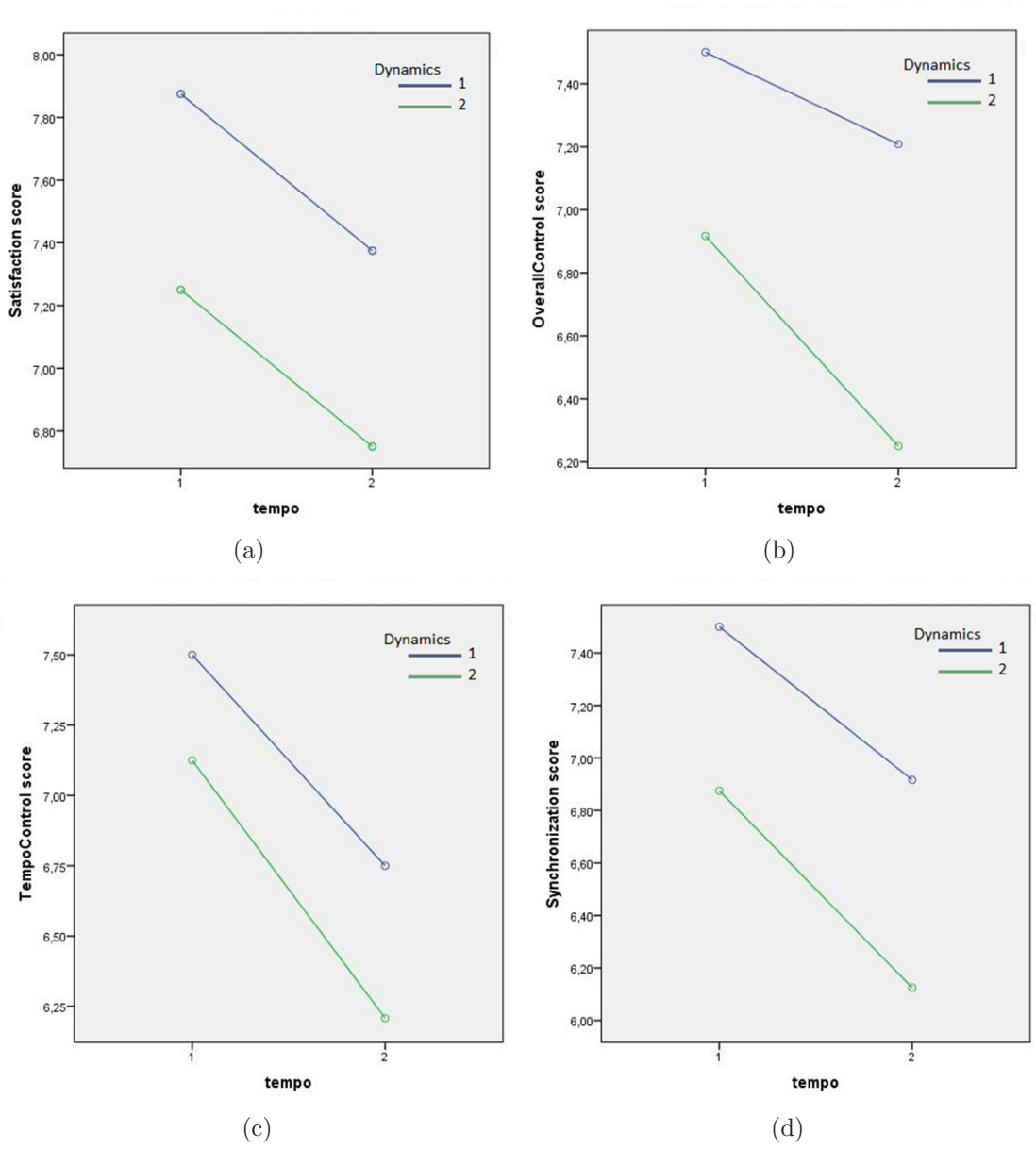


Figure 4.31: Estimated Marginal Means for the dependent variables Satisfaction (a), OverallControl (b), TempoControl (c) and Synchronization (d). The *tempo* factor takes values 1 (tempo synchronization present) or 2 (tempo synchronization not present). The *dynamics* factor takes values 1 (dynamics control present) or 2 (dynamics control not present).

the conducted beat times. However, the numerical results show that the presence of this synchronization feature was critical for the perceived experience, in particular in case of the items pertaining to satisfaction, control and realism. The 4 participants that had a stronger musical background did noticed this difference between the two conducting modes, yet the statistical analysis did not show any particular differences in this regard.

	Cond. 1	Cond. 2	Cond. 3	Cond. 4
Satisfaction	7.875	7.25	7.375	6.75
OverallControl	7.5	6.917	7.208	6.25
TempoControl	7.5	7.125	6.75	6.208
Synchronization	7.5	6.875	6.917	6.125
Intuitiveness	8.292	8.417	8.125	8.458
EaseOfUse	7.208	7.792	7	7.7917
Realism	7.25	7.083	7	6.833

Table 4.14: Average scores for each variable observed at each of the 4 experimental conditions: condition 1 (both tempo synchronization and dynamics control present), condition 2 (only tempo synchronization present), condition 3 (only dynamics control present) and condition 4 (none present)
.

Contrary to the findings of the pilot study, the presence of dynamics control also seems to denote a positive effect in user experience, concretely regarding the perceived satisfaction, control and synchronization. However, it was also found that the added complexity of the interface was counter-productive in terms of ease of use as well as making the interface less intuitive.

Furthermore, from observation of participants' behaviour during the experiments, it was noted that some participants had problems to conduct the beat times and simultaneously indicate changes in dynamics and volume, specially when trying to control the volume of the violin, which was located towards the right side of the participant, and thus, their left hand sometimes hampered their right hand motion. This is a flaw that can be explained in the camera-based nature of the system, as it may be possible that one hand obscured the line of sight of the 3D sensor to the other one. In the particular case of the right hand, the system was highly sensitive to this kind of occlusion, as it could prompt the system to perform unintended tempo changes.

Most of the previous work has focused mainly on capturing the conductor's gestures and modifying the tempo of the piece played accordingly to a time-stretching algorithm. A few studies, however, have also implemented the possibility of controlling the dynamics of the piece played, e.g. as seen in Borchers et al. (Borchers et al. [2004]), offering a much more complete experience to the users. While the study performed does indeed confirm that this additional degree of control brings a more compelling experience, the camera-based tracking technology considered shows that additional steps must be taken to ensure that users can actively used both hands without interfering the commands given by each other because of occlusion.

Most of the previous research has favored the use of infrarred or inertial batons

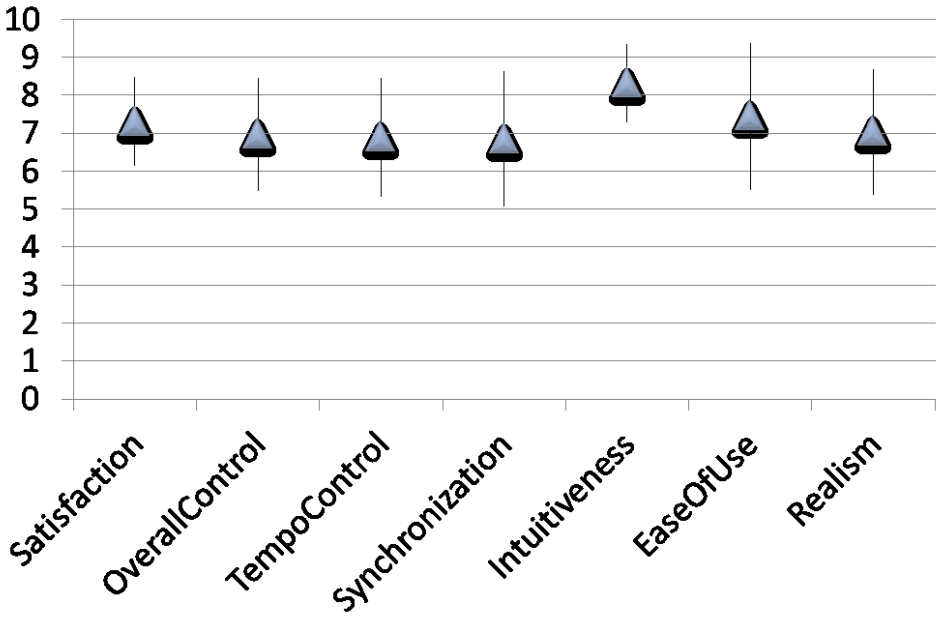


Figure 4.32: Average values for the variables considered, along with ± their standard deviations σ

(Borchers et al. [2004], Peng and Gerhard [2009], Lee et al. [2004], Bakanas et al. [2012]), or, more recently, the use of Nintendo's Wiimote (Bradshaw and Ng [2008], Nakra et al. [2009]). However, this kind of devices is usually very expensive (Lee et al. [2004]) or have ergonomic and usability issues (in the case of the Wii Remote, its shape is not that adequate for baton emulation, and its additional weight when compared with an infrared baton (Baba et al. [2010]) might rise some issues in long sessions). We have addressed this issue by using a Kinect device, thus providing an inexpensive and non-intrusive interaction paradigm.

Yet, one interesting fact that came up in the experiment conducted is that of participant exhaustion. In particular, some of the participants explicitly indicated to find the experience physically demanding, and 2 of them even reported arm pain in their right arm due to the repetitiveness of the conducting gesture. This, however, can be explained in the unusual length of the experimental sessions. Some participants also noticed a latency between their motion and the actual system response; this is caused because of a delay introduced by the sensing device, and it is an issue that should be improved in its next iteration.

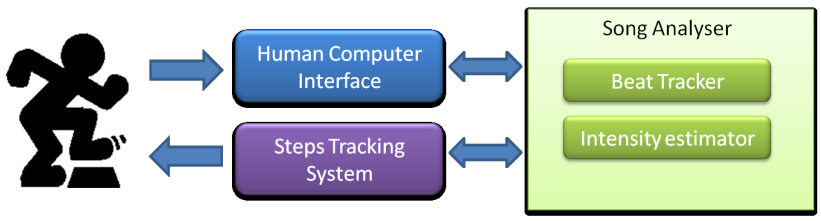


Figure 4.33: Application module structure.

4.3.2 Virtual steps-aerobics instructor

In this section, we present a novel design aimed towards the implementation of a steps-aerobics instructor emulator. Concretely, the purpose of this application is to act as a guide of sorts to potential users, issuing aerobics-like commands according to the rhythm of the song played. In order to do so, the system analyses a given audio excerpt, extracts its rhythmic pattern structure and the level of rhythmic intensity for each segment of the song. After that, this information is used to issue the corresponding commands to the user, in synchrony with the song being played.

Fig. 4.33 illustrates the overall block structure of the system designed, which has been developed for use in an Android smartphone device using the Android SDK toolset. The human-computer interface module aims to provide an adequate interaction model with the application, including the preferred way to present the different commands to the user. The step tracking system is in charge of detecting the different steps performed by the user, and processing the data into a suitable format, in order to indicate the user whether a given exercise has been performed correctly or not. In order to do this, the system uses a Wii Balance Board device to detect the position of the center of gravity of the user and the position of the feet.

The song analyser module implements a more complex logic unit to process a given audio excerpt and detect its rhythmic patterns. These data is then used to implement the emulation of the virtual instructor. Fig. 4.34 portrays the functional elements which this module is comprised of:

- An *onset detector*, which is in turn comprised of two elements: a spectral analysis module that implements a Short-Time Fourier Transform (STFT), and an element that subsequently determines the spectral energy flux of the PCM data.
- A *tempo estimation* component that determines the most likely tempo values for the song processed.
- An *agent-based beat tracker* which calculates the position of the rhythm beats according to the tempo hypothesis provided.

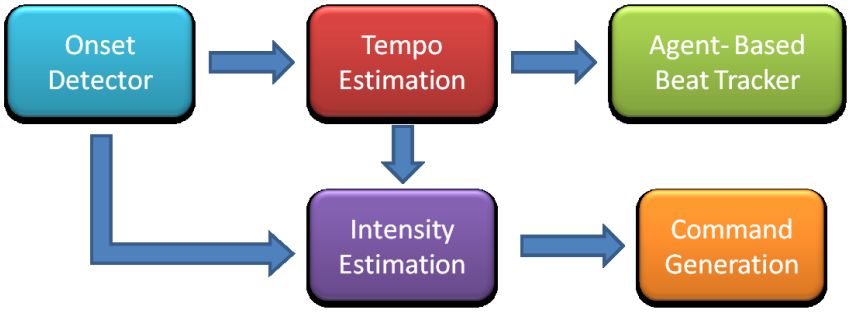


Figure 4.34: Beat-Tracker block structure.

- A *rhythmic intensity estimator*, to determine which segments in a given clip are played with higher or lower rhythmic patterns.
- An steps *commands generator* to issue the corresponding commands according to the beat patterns detected and the overall rhythmic intensity of the song at each segment.

The basic functionalities offered by these five elements are combined to provide two higher-level capabilities: finding and tracking the rhythmic beats in the song analysed, and the emulation of the virtual steps instructor role by issuing the corresponding step-exercise commands. Both functionalities are described further in the following subsections.

4.3.2.1 Beat tracking function

The beat-tracking algorithm considered in this work extracts the onset times from the input data signal and analyses its structure to locate the most likely positions for the beat times. As previously mentioned, the onset detector relies on a STFT to find the time-frequency spectrum of the input signal, $X(t, f)$, and then extracts the corresponding spectral flux, $SF(t)$, as per the equation 4.27. Spectral flux was chosen as the onset detection function because it is quite a proficient method with regards to factors such as onset detection accuracy, simplicity of programming, and execution speed (Dixon [2006]).

$$SF(t) = \sum_{f}(X(t, f) - X(t-1, f)) \tag{4.27}$$

Concretely, the following algorithm (illustrated in 4.35) was used for the onset detection process: the system uses a Hamming windowing procedure with window size of 2048 samples for the STFT process, and then the spectral flux of the resulting spectrum is calculated by dividing the spectrum into eight different frequency bands, as per Table 4.15. Each of these eight spectral fluxes is rectified and pruned, so that for each spectral flux

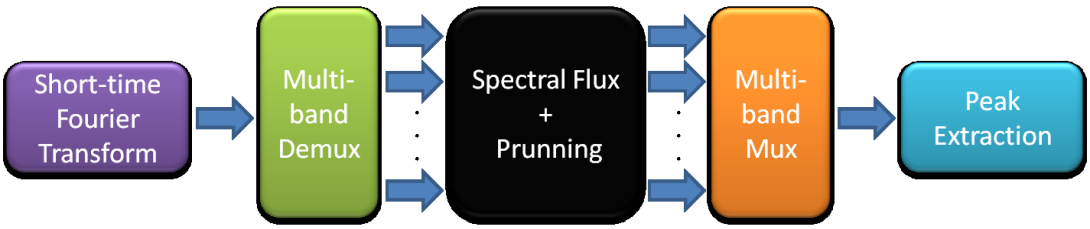


Figure 4.35: Onset detector block diagram.

Frequency range
0Hz-250Hz
250Hz-500Hz
500Hz-1KHz
1KHz-2KHz
2KHz-4KHz
4KHz-8KHz
8KHz-16KHz
16KHz-max

Table 4.15: Frequency bands for onset detection.

sample, that sample is set to 0 if it lies below a given threshold value, and kept otherwise. The different threshold values for each sample are dynamically set by finding the average of the spectral flux value in a window of 1 second centred on the i-th sample processed, setting the i-th threshold value to 1.5 times this average.

The resulting eight pruned spectral flux functions are recombined into a single signal. A peak detection process in then applied to this combined signal and, finally, the onset times are extracted by filtering this peak signal so that every pair of consecutive peaks are spaced by at least 50 milliseconds (as rhythmic information is typically confined in the 50ms-2s inter-beat time range (Dixon [2001])); if a pair of peaks does not fulfil this criteria, the peak with lower amplitude is deleted.

The onset signal is then used as input for the tempo estimation and agent-based beat tracker components, thus delivering the position of the beat times in the audio excerpt.

The implementation of both algorithms follows the model described in (Dixon [2001]). Tempo induction is performed by studying inter-onset time intervals in the input following a clustering approach. Different tempo hypotheses are offered according to the score given by the algorithm, and are set as further input data for the agent-based beat tracker. This module defines a set of agents, each one associated with one of the tempo values proposed and an initial start position. Each agent then runs through the onset signal, starting from their start position, and using its tempo value to predict the position of the next beat. If

the position of the actual onset does not match the position predicted, the agent updates its tempo value dynamically, according to the actual positions of the beats found. If a given agent finds that the changes to its tempo to keep up with the beat times are too excessive, then a second agent is instead created ad-hoc, starting from that beat time to cover the new tempo hypothesis. After all the possible agents have been run through the onset signal, the one scoring better in terms of relative error of its predicted beat times positions is chosen to determine the actual beat times of the song.

4.3.2.2 Intensity Estimation: Virtual aerobics instructor

In order to issue the different step-based commands, the system first proceeds to classify the original piece in segments according to their corresponding rhythmic intensity.

In a similar process to the one followed for the beat tracking algorithm, the system first calculates the onset times signal previously described. This signal is then divided into chunks, using windows of size $2T_c$ seconds with an overlap of 50%, where T_c is the total of seconds covered by 24 beat times played at the original song's tempo. The total number of actual beats collected at each chunk is then compared with the total amount of beats in the overall song, thus defining a beat-density function that spans the whole song length. This density function is used to find different rhythmic intensity levels in the clip analysed. Concretely, a k-Nearest Neighbours unsupervised machine learning process is performed over the beat-density signal, therefore labelling each chunk into one of the k potential levels of intensity found.

After every chunk has been classified into one of the possible k categories, the total number of onset beats found for each category is calculated, and the different categories are ordered in increasing order of total number of onset beats. Additionally, the most commonly found category among the different chunks, hereafter category l, is also determined. This category is used as a reference to determine the intensity levels: the chunks in category l are assumed to have standard rhythmic intensity, while the rest of chunks are assigned a higher or lower level of rhythmic intensity depending on the place where their category lies in the previously computed order. An example for a fictitious case with $k = 3$ categories is portrayed in Fig. 4.36.

Finally, once the different chunks of the song have been assigned an equivalent intensity level, the system randomly generates a pattern of step exercises to output while playing the song as commands given by a virtual trainer. Each chunk is then associated with a set of exercises according to the level of intensity identified. Thus, while the song is playing, the commands corresponding to the particular chunk being played in that moment are presented to the user in synchronization with the rhythmic patterns identified previously

(example for *k*=3 categories)	A	B	C
Average number of Beats per category	15	23	47
Intensity label	Low	Standard	High

Figure 4.36: Category labelling for $k = 3$ classes: the chunks are labelled according to the k-NN classifier, and the intensity for each category is set according to the average number of beats per chunk, taking the most common category (B) as the standard rhythmic intensity

by the beat tracker module.

4.3.2.3 Experiments and Results

We conducted an experiment aimed at collecting data regarding user experience when using the designed virtual aerobics trainer. Concretely, a total of 10 participants took part in the experiment. All of the participants were male, with an average age of 29.4 and two of them had a strong musical background. Only one of them had attended previously a class of steps-aerobics.

Each participant was asked to "attend" one training session with the virtual instructor, which consisted in one song being played along with the corresponding commands. For the purpose of this training session, the application was run on a personal computer, and thus the different commands were presented to the users through a standard desktop display.

The songs considered for this test were all extracted from compilations of steps music, concretely they were: Feel this Moment (Pitbull), Gangnam Style, Kiss N Tell (Kesha), Can't Hold Us (Macklemore) and Don't Stop the Party. Each song had a corresponding BPM value of 136, 130, 144, 146 and 128 respectively. Each participant was randomly assigned one song, so that each song was used in two different experimental sessions (hence covering the 10 participants). For the purpose of this experiment, we considered $k = 3$ levels of rhythmic intensity, which, in the case of all the 5 songs considered, resulted in 3 levels of intensity: low intensity, average intensity and high intensity.

Prior to the virtual step aerobics session, each participant was presented with the set

of exercises from which the system would randomly select the different exercise patterns. Participants were given as much time as possible to get familiar with the different kinds of steps that could be issued. The different exercises considered were appropriately labelled as low, standard and high intensity exercises, attending to the complexity of the exercises considered. All of the standard and low intensity exercises were performed in 4 beat times, while the high intensity exercises required 8 beat times to be performed.

During each experimental session, for each beat time detected the system would issue a command according to the level of rhythmic intensity in the segment of the song currently in play (e.g. Basic Left Step), and the following commands would just keep the count of beat times associated to the exercise (e.g. 2, 3, 4) until its end. After that, a new command is issued in the next beat time, and the process is repeated until the song ends.

At the end of the experimental session, each participant was asked to fill in a questionnaire in which they assessed their experience with the virtual instructor. Concretely, the users were asked to evaluate the following items:

- Utility perceived.
- Satisfaction with the experience.
- Novelty of the application.
- Ease of use.
- Synchronization between commands and rhythmic intensity.

For the particular case of the "Ease of use" item, participants were explicitly asked to take out of consideration perceived difficulties because of having to memorize the different exercises previously.

In addition to the questionnaire, additional data was extracted from an informal interview between the participants and a researcher concerning their overall perception of the system. The results obtained are summarized in Fig. 4.37

Overall, the application had a warm welcome for the most part, with the most valued aspects being its novelty and the synchronization between rhythmic intensity and the commands issued. Satisfaction was the item that received the worst critics, and in fact several participants stated that they found that having to read the commands from the display detracted from the overall experience. This suggest that including pre-recorded voice commands would greatly improve the experience, but the overall response was still quite positive.

Some participants expressed their concerns regarding the complexity of the commands given, as they reckon some of the combinations of exercise patterns proved to be too

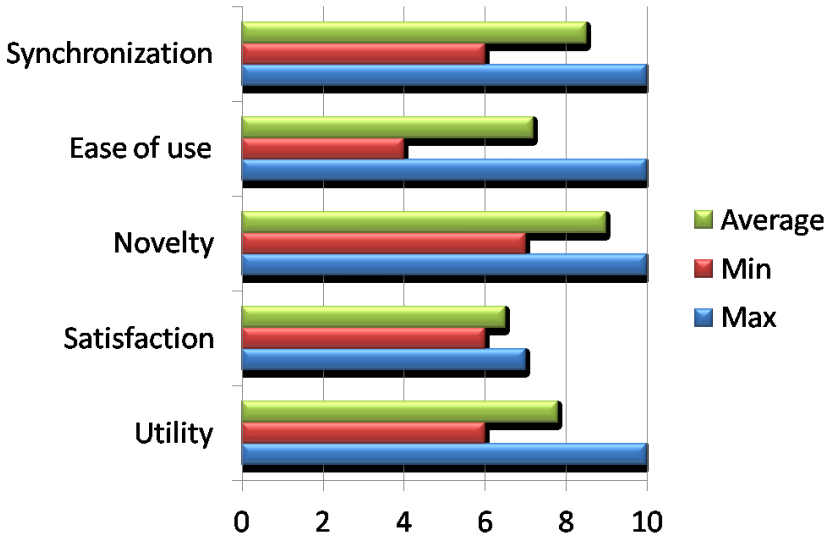


Figure 4.37: Results from participants' survey.

difficult to follow. In fact, the system randomly selects exercise patterns taking only their difficulty into consideration, but does not evaluate whether a given succession of exercises is more or less adequate to conform a global exercise. In order to solve this issue, a more detailed analysis must be performed regarding how the different exercises are selected, ideally with the assistance of a professional trainer. However, the research performed confirms that this approach allows for rich and interactive experiences with music through the simulation of an aerobics steps session.

CHAPTER

5

BRAIN-COMPUTER EMOTIONAL INTERFACES

Previous chapter proposed the application of novel interaction models to the development of new musical experiences. Most of the proposed interaction techniques considered were based around motion-tracking technologies; yet, this chapter will explore the use of more unconventional interaction metaphors, namely the use of brain-computer interfaces.

Concretely, this chapter collects the research performed towards the study of emotion-based interaction with music. The aim of this study is to advance on the research and design of a system that detects the current emotional state of an user, so that this information can be later used in order to automatically propose the user a selection of songs according to his/her mood. To implement this system, it is necessary to find an adequate way to extract information regarding the user's current mood state, and analyse this information to extract the most relevant features that allow for emotion discrimination.

5.1 Data acquisition: EEG brainwaves and signals

We have considered EEG (electro-encephalographic) signals as the source of emotional data for this system. EEG signals are both a measure of the frequency and amplitude of

the human brain's electrical activity, and provide a non-invasive and relatively inexpensive way to gather emotional data.

For an adult, EEG signals amplitude taken with a scalp electrode lies between 10 and 100 μV (Aurlien et al. [2004]). Depending on the frequency of their oscillation, the different EEG rhythms are classified into one type or another. According to (Sterman [1996]), there are 5 main groups of EEG rhythms:

- Delta waves: these waves oscillate at frequencies between 0.5 and 4 Hz, and are usually associated with the deepest stages of sleep (Teplan [2002]).

- Theta waves: their frequency range is roughly 4 to 7.5 Hz. They are usually present during the states of drowsiness and arousal, idling and inhibition of elicited responses (Kirmizi-Alsan et al. [2006]).

- Alpha waves: they normally cover the frequencies from 7.5 to 13 Hz, and are usually associated with different states of the wake-sleep cycle, as well as with stages of relaxation or lack of agitation. They are also associated with closing the eyes (Niedermeyer [1997], Pivik and Harman [1995]).

- Beta waves: covering the range of roughly 13 to 30 Hz, these brainwaves are typically associated with normal waking consciousness, and can be divided into three subbands, namely Low Beta or Beta 1 power (up to 16 Hz), normal Beta or Beta 2 power waves (16-20 Hz) and High Beta or Beta 3 power (more than 20 Hz) (Rangaswamy et al. [2002]). They are associated with different states of conscious thinking: relaxation but with focus, thinking, awareness of self and surroundings, alertness, agitation, etc.

- Gamma waves: the oscillate between 30 and 100 Hz (Hughes [2008]), and relate to high-level information processing and integration of thoughts, as well as sensorial multimodality integration. It is also believed that they are associated with creating the unity of conscious perception (Buzsaki [2009]), although there is some controversy in this regard (Vanderwolf [2000]).

In order to acquire the different EEG signals from a given user, an EMOTIV EPOC sensor device was used. This device includes a total of 14 electrodes to provide a corresponding set of 14 EEG signals for each measure sample, 2 electrodes to set a reference potential, and 2 gyroscopes to provide information on the position and tilt state of the user's head. The electrodes are distributed on the user's head as per the 10-20 system

(Sanei and Chambers [2008]), illustrated in Fig. 5.1. The data provided from the aforementioned 14 EEG data channels was taken as the input for the purported system. Fig. 5.2 shows a sample of EEG signals captured with this setup.

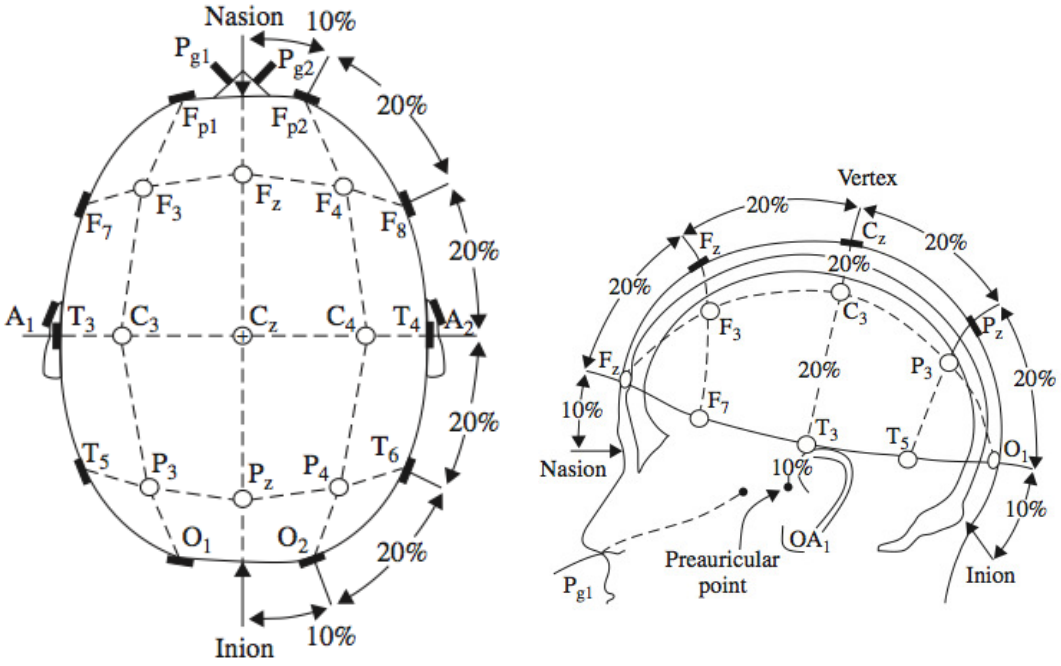


Figure 5.1: Electrode positions as per de 10-20 system: top view (a) and side view (b)

The system designed is comprised of the functional blocks described in Fig. 5.3. As illustrated in said figure, there are two different flows of information: a first flow of information uses previously stored EEG data to train a classifier model for emotion estimation, while the second flow of data takes an actual EEG measure from a given user and processes it according to the previously trained classifier model.

The different functional blocs are described in the subsequent sections in this chapter

5.2 EEG Database

The training data was expected to be extracted from an already available EEG database, so that each sample os the database would consist in a group of EEG signals associated to one particular emotion. However, it was not possible to find an already available emotion-labelled EEG database to be used as part of our system. In order to supply this deficiency, we resorted to building our own EEG database by conducting an experiment to gather EEG data and label it accordingly.

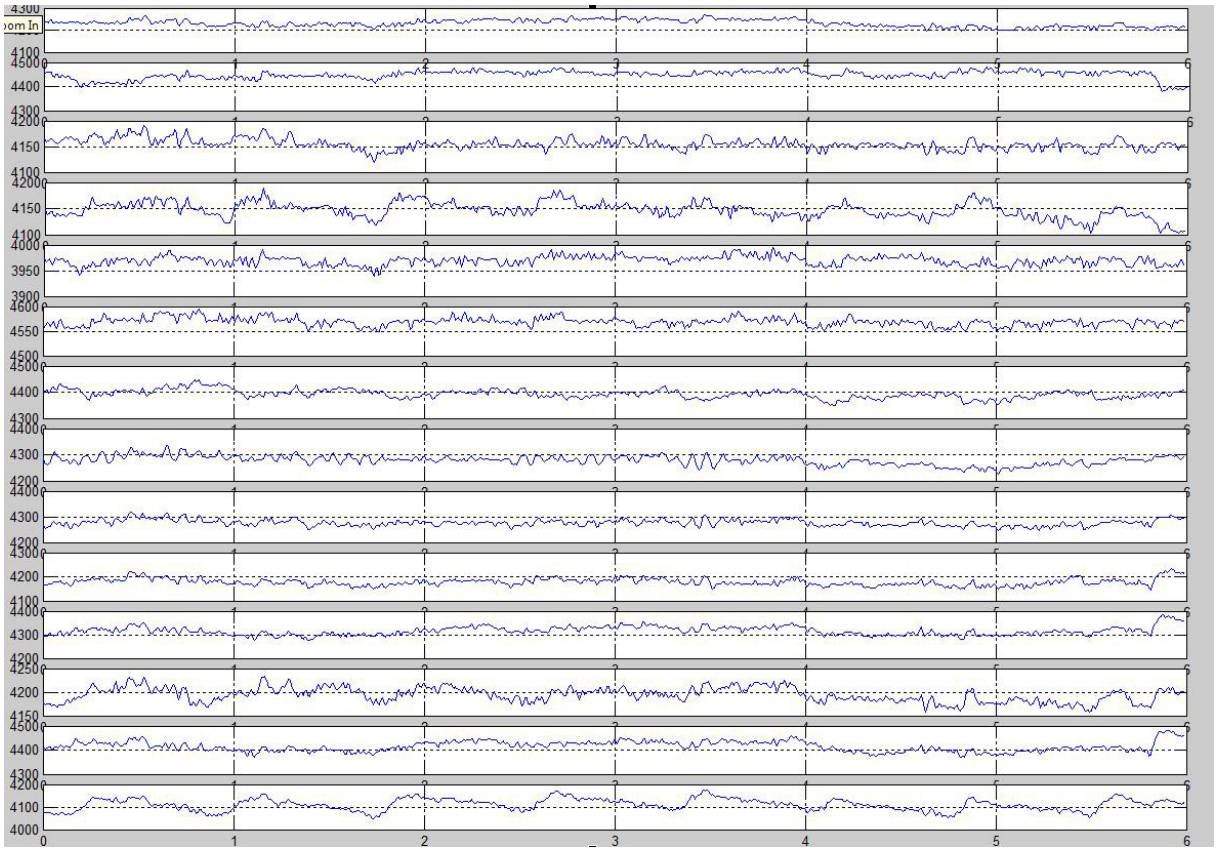


Figure 5.2: Unprocessed sample of EEG data.

5.2.1 Participants

A total of 20 participants took part in the experiment (11 males and 9 females), with ages ranging between 18 and 35 years old. None of the participants had any physical handicap regarding their visual or auditive capabilities.

5.2.2 Method and Materials

The main idea behind the experiment was to present users with a set of audiovisual stimuli and collect the corresponding EEG data generated, and then ask the user to classify the type of emotion that each stimulus had induced. For the purpose of the experiment, we considered two types of stimuli: visual stimuli in the form of static images, and auditive stimuli in the form of pre-recorded sounds. These stimuli were extracted respectively from the IAPS (International Affective Picture System) (Lang et al. [1999]) and the IADS (International Affective Digitalized Sounds) (Bradley and Lang [2007]) databases. Each database is comprised of 1182 and 182 samples respectively. Users' electroencephalographic cerebral activity was recorded using the EMOTIV sensor.

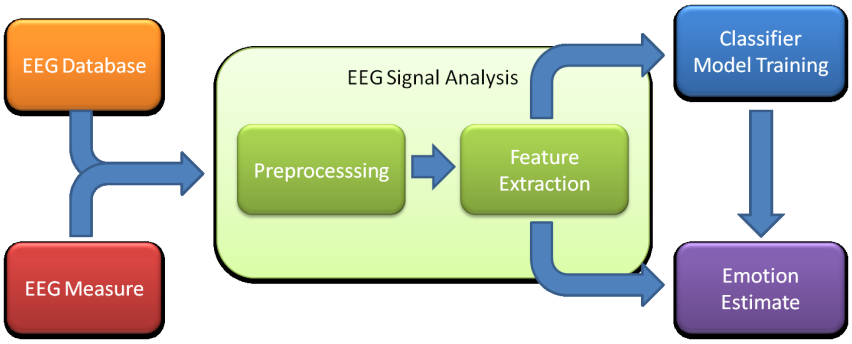


Figure 5.3: Block diagram for the emotion estimation system.

5.2.3 Procedure

At each session, each participant was presented with a set of 50 different pictures and 30 different sounds, picked at random from the sets available in the previously mentioned databases.

The experiment was conducted in the ATIC research lab of the University of Málaga, keeping the same lighting and environmental conditions for all the participants, and all of them wore headphones. At the beginning of each experimental session, participants were given precise instructions on their assigned tasks. Each session consisted in the following stages for each of the stimuli considered.

- Pre-stimuli stage: users were presented with a blank screen for a period of 6 seconds and no sound was played at this stage, to ensure an stimuli-free state
- Stimuli presentation: a given stimulus (audio clip or picture) is presented to the participant for a brief period of time (6 seconds for the audio clips, 2 seconds for the pictures).
- Stimuli labelling: the participant is presented with an image of an emotion wheel as per Fig. 5.4, based on Plutchik's Flower model (Plutchik [1980]), and is then asked to press the button corresponding to the emotion felt, thus labelling the feeling evoked by the last stimulus experienced.

At the end of the third step, the corresponding EEG data set of 14 signals is labelled according to the emotion that the user indicated to have felt. Given that the sampling frequency of the EMOTIV device is of 128 samples per second, the size of each labelled EEG sample was either 768 for EEG data sets derived from an auditive stimuli, or 256 for those derived from visual stimuli.

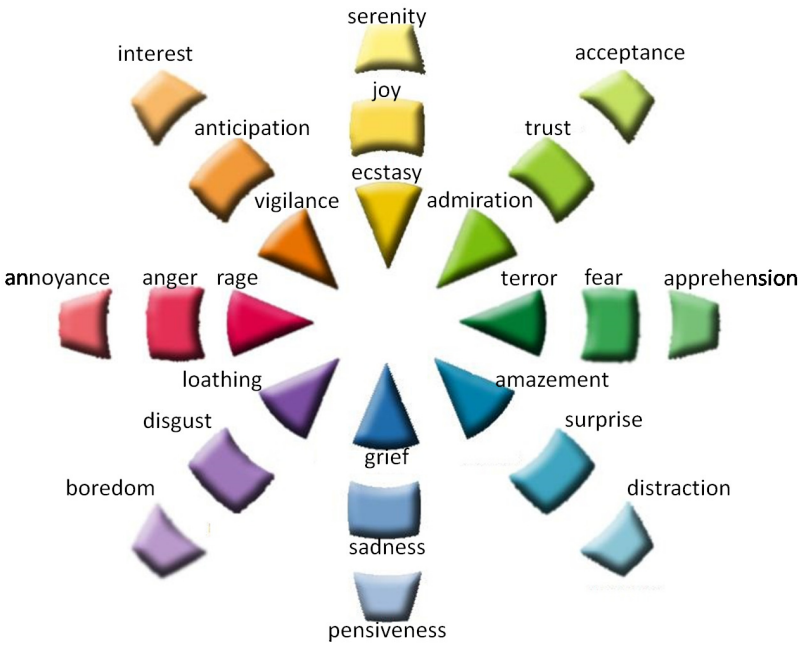


Figure 5.4: Feelings considered in the affective circle.

5.3 EEG Signal Analysis

The EEG signal analysis block has two main objectives: first, the EEG raw data is preprocessed to erase potential interferences from artifacts that might mask the cerebral activity signal, and then a set of features are extracted from the resulting signals to be used in the classifying stage.

Regarding the preprocessing stage, the first step taken is to eliminate potential artifacts in the signal caused by ocular movement. In order to do this, a blind source separation process is followed, according to the algorithm in (Gómez-Herrero et al. [2006]).

Additionally, we do not need to analyse the whole EEG signal bandwidth: frequencies below 3.5 Hz are mostly Delta waves, which are more related to deep sleep neural activity. On the other hand, higher frequency waves are normally associated with more hectic emotional states than the emotions that the proposed system aims to induce.

Therefore, taking into account other examples in the bibliography (Bos [2006], Sanei and Chambers [2008], Gold [1999]), we have considered a maximum frequency upper bound of 45 Hz. Subsequently, the EEG data was passed through a Butterworth bandwidth IIR filter covering this frequency range. An example of the resulting processed sample set of EEG signal had is portrayed in Fig. 5.5

The feature extraction stage aims to infer a set of descriptors for each set of labelled EEG signals so that differently labelled sets can be distinctly identified as such. However, EEG signals are strongly non-stationary and random, which makes the use of traditional

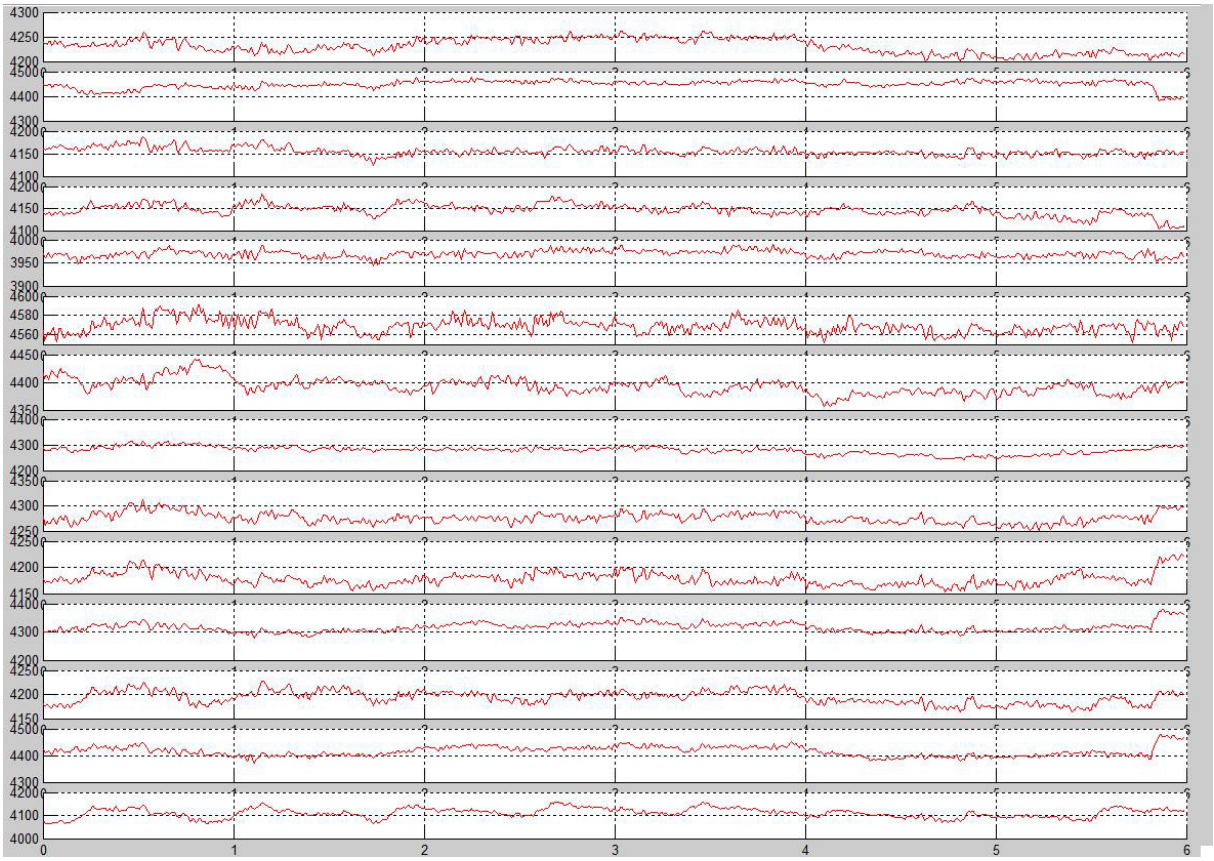


Figure 5.5: Processed sample of EEG data.

Fourier analysis rather ineffective.

Ideally, EEG signals are better analysed through a time-frequency approach such that it is possible to provide good resolution either in the time or the frequency domain, as needed. The use of Wavelet transform analysis is a suitable solution to this problem (Akin [2002]). Concretely, we applied a wavelet decomposition analysis of eight levels using a Daubechies wavelet of sixth order (Jensen and la Cour-Harbo [2001]), thus resulting in 9 different subsignals per EEG signal analysed. For each of this subsignals, we extracted 2 features: energy and entropy, defined as ENG_x and ENT_x for a given signal $x[n]$ with spectral density $S_x[k]$ and normalized spectral density $\overline{S_x[k]}$ as per the following equations:

$$ENG_x = \sum S_x[k] \tag{5.1}$$

$$ENT_x = -\sum \overline{S_x[k]} log(\overline{S_x[k]}) \tag{5.2}$$

In addition to these $2 \times 9 = 18$ spectral features per EEG signal, we also considered 6 additional features defined in the time-domain: the average (μ_x, Eq. 5.3), standard

deviation (σ_x, Eq. 5.4), average of the first order difference (δ_x, Eq. 5.5), average of the first order difference of the standard signal ($\overline{\delta_x}$, Eq. 5.6), average of the second order difference (γ_x, Eq. 5.7) and the average of the second order difference of the standard signal ($\overline{\gamma_x}$, Eq. 5.8), defined for a given signal $x[n]$ of size N, and its corresponding standardized equivalent $\overline{x[n]} = \frac{x[n] - \mu_x}{\sigma_x}$.

$$\mu_x = \frac{1}{N}\sum_{1}^{N} x[n] \tag{5.3}$$

$$\sigma_x = \sqrt{\frac{1}{N}\sum_{1}^{N}(x[n] - \mu_x)^2} \tag{5.4}$$

$$\delta_x = \frac{1}{N-1}\sum_{1}^{N-1}(x[n+1] - x[n]) \tag{5.5}$$

$$\overline{\delta_x} = \frac{1}{N-1}\sum_{1}^{N-1}(\overline{x[n+1]} - \overline{x[n]}) \tag{5.6}$$

$$\gamma_x = \frac{1}{N-2}\sum_{1}^{N-2}(x[n+2] - x[n]) \tag{5.7}$$

$$\overline{\gamma_x} = \frac{1}{N-2}\sum_{1}^{N-2}(\overline{x[n+2]} - \overline{x[n]}) \tag{5.8}$$

With all the descriptors previously defined, a total of 24 features are extracted per EEG signal. Given that each labelled sample is in turn comprised of 14 EEG signals, the total number of features per sample for the classifier stage is of $14 \times 24 = 336$.

5.4 Emotion Classification

Finally, we evaluated the possibility of performing automatic emotion classification task of the different EEG signals supplied. A set of different classifier models is first trained with the data provided from the EEG database, and then these classifiers are used to try to identify the emotions felt by users according to the EEG measures retrieved.

For the purpose of this study, we opted to consider the following classification techniques:

- Neural network classification based on a Multilayer Perceptron.
- Support Vector Machines using a polynomial kernel.

Classifier	SR_e	SR_c
Multilayer Perceptron	22.22%	38.88%
Support Vector Machines	11.11%	26.98%
Tree Classifier (C4.5)	12.70%	27.77%

Table 5.1: Classification results for EEG-labelled signals from audio stimuli.

Classifier	SR_e	SR_c
Multilayer Perceptron	24.66%	33.33%
Support Vector Machines	12.66%	28.00%
Tree Classifier (C4.5)	15.33%	29.33%

Table 5.2: Classification results for EEG-labelled signals from visual stimuli.

- Decision Trees classification, using the algorithm C4.5.

The EEG-labelled databases for audio and visual stimuli gathered previously 5.2 were split into two groups: a training group, which included 85% of the instances in each of the two databases (510 and 850 EEG sample sets for audio and image data respectively), and a second group formed by the remaining 15% (90 and 150 respectively), to be used for cross-validation. For each database, a classifier model was trained separately for each of the classifying techniques considered. The different classifiers were evaluated using the cross-validation sets, according to the success rate per emotion (SR_e) and per class (SR_c). Success rate per emotion was defined as the average of the percentages of correctly classified instances for each emotion considered (as per the emotion wheel in Fig. 5.4). Success rate per class was similarly defined for each emotion class, each class being defined as the combination of the three emotions of the same colour illustrated in the emotion wheel 5.4. The results yielded are summarized in Tables 5.1 and 5.2.

While the resulting classification rates might not seem to be very high, it is important to notice that the number of categories considered is actually pretty high: 24 different emotion labels, and the resulting 8 classes. A random classification scheme would offer success rates of 4.17% and 12.5% respectively for the SR_e and SR_c variables. Thus, it is clear that the system implemented is capable of much better discrimination than the naïve approach. Results from previous research have offered success rates of 41.7% for 5-emotion discrimination (Takahashi [2004]), or values of 58% (Chanel et al. [2006]), 63% (Chanel et al. [2009]) and 86% Li et al. [2009] for 3-emotion classification. While the results yielded in this study have a meaningfully lower success rate, this is compensated by the fact that the number of emotions considered is much higher.

The study conducted also shows that the application of signal processing and machine

learning techniques allows for brain-computer interaction metaphors using relatively affordable rigs, as the EMOTIV system, while expensive when compared with other more conventional interaction devices, is definitely much less expensive than other systems for EEG data sampling.

CHAPTER

6

CONCLUSIONS AND FUTURE WORKS

This last chapter will revolve around the conclusions extracted from the work performed, as well as discussing the potential future works that conform the most enticing prospect in order to continue the lines of research studied.

One of the main pillars in the research performed as part of this thesis is the application of interactive and intelligent support modules that can be used as aid tools in order to enhance learning processes in music education. In regard to this, we have studied the use of advanced music processing techniques for the development of intelligent software for automatic correction of musical performances; concretely, we have developed a set of different applications that allow practitioners to automatically assess the correctness of their performances, identifying the mistakes made and thus allowing for further improvement in their musical skills. In a similar way, we have studied the use of music information retrieval techniques for the implementation of an intelligent guidance system that aids user during their performances, simulating the role of an ensemble conductor.

Given the strong focus on interaction in this document, the development of new ways of interaction with music presents itself as the most relevant line of research. Consequently, a huge amount of effort has been devoted to the development of new innovative paradigms for computer-based music interaction and to the simulation of several musical roles and instruments. In particular, we have studied the capabilities of motion-sensing technologies along with machine learning and signal processing techniques for the implementation of several musical instruments, such as the drumkit and the theremin, as well

as the emulation of guiding roles, such as those of the ensemble conductor or the aerobics instructor. More unconventional interfaces have been also studied, as is the case of the use of brain-computer interaction to detect user emotions and mood. The different applications and modules implemented have also been further tested experimentally, through the use of questionnaires and more detailed experimental analysis techniques.

Overall, the research performed shows that it is indeed possible to achieve novel and innovative interaction paradigms for music-driven applications, and the use of these types of interaction models allows for more enriching and satisfactory experiences.

From the research conducted, we draw the following main conclusions:

- It is indeed technologically feasible to use music information retrieval techniques to create support tools for music learning. The applications developed and subsequent tests have showed that both rhythmic patterns and note pitch can be extracted and analysed in order to assess the correctness of a given performance, re-tune an instrument, etc.

- Twin-linked with the previous assertion, the implementation of support tools and aid applications for music learning can be achieved in an inexpensive way, without the need of contrived hardware components. Instead, the use of intelligent processing modules to analyse the musical signal proves to be sufficient for the implementation of such support tools.

- The use of IT technologies can reinforce learning processes, providing students and music enthusiasts with the means to objectively assess their own skill levels without the need of an external evaluator.

- While still no replacement for their real counterparts, efficient musical instrument simulation is a reality achievable by means of more generic and easily affordable devices or interaction models. In fact, in the research performed, only the Kinect device by itself allowed for the implementation of three different interaction paradigms (drumkit, theremin and ensemble conductor). It has been shown that the limitations in the input system can be overcome through the implementation of intelligent processing algorithms tailored for the application at hand.

- The use of advanced human-computer interfaces does allow for better overall experiences over the standard input-output paradigm. In fact, user response to and satisfaction with the different interaction models studied has been overwhelmingly positive, and shows that the use of new interaction paradigms can indeed offer a more immersive and gratifying experience in computer-based applications.

- Time and delay constraints are critical in the implementation of interactive musical applications. This is an issue that has been glaringly brought to light in the studies performed for drumkit simulation. Even when not consciously acknowledged, users have showed to notice small lags between their motion and the corresponding system delay, thus making the use of predictors a necessity if the input system involves some form of latency. This issue becomes more exacerbated if the user has a previous background knowledge and skill in music performance.

- The combination of machine learning techniques and signal preprocessing yields an effective framework for gesture recognition in the design of human-computer interfaces for music. Furthermore, it has been found that, by adequately selecting the features or descriptors for the gestures considered, it is even feasible to achieve recognition rates as high as 90% in real-time interaction in different interaction contexts.

- It is feasible to develop brain-computer interfaces for emotion detection using an affordable system while keeping reasonably high detection rates in relation to the cardinality of the emotion set considered.

Given that the scope of the research performed is so huge, the amount of potential lines of research that can be derived from here on is almost insurmountable. Therefore, we would like to remark the lines that we have considered to be more relevant with regard to the studies conducted:

- Corroborate the effectiveness of the applications developed as support tools in the learning process by means of conducting a controlled experiment within a musical course. While the applications designed have been shown to be effective to highlight the mistakes made and guide students during their performances, it has not yet been proven formally that the use of these advances within the frame of a music classroom does indeed imply an improvement in the competences developed and acquired as part of a given course. Thus, the next step would be to formally define an experiment to assess the validity of the tools developed in such an environment

- Improve on the current implementation of virtual instruments achieved. Particularly, while the study performed shows very satisfactory results for the current implementation of the virtual drumkit, it does not yet achieve a full emulation of a real drumkit. In particular, it is necessary to further study the features considered to allow for detection of fast rhythmic patterns, and it might also be interesting to

add additional degrees of control (such as tracking the motion of the feet to modulate the sound played for the hit-hat, or adding additional cymbals and toms). The implementation of the ensemble conductor interaction model could also be expanded to account for different time signature indications detection, as well as other dynamic nuances (*fermata*, etc.). On a related note, it would be also interesting to extend the studies and experiments conducted for the simulation of other musical instruments. A good example of such instruments would be the Xylophone, but it might be feasible to implement others, such as the Piano (using a camera-based systen along with markers for each hand) or the Violin (by combining a camera-based tracker with an inertial sensor).

- Further study the effects of the delay in the overall satisfaction of the user with the experience. As previously indicated, it has been found that even short lags can have a relevant impact on the immersive experience perceived by the users. It would be interesting to further study up to which levels the lag starts to become being noticeable. Furthermore, while not directly tested in our research, some cues where found that seem to suggest that user perception of lag may be modulated by visuals cues directly related with the motion captured. This finding suggests that it might be possible to modulate or even mitigate the effects of latency by appropriately configuring the multimodality aspect of the interface.

- Study the use of additional signal processing and gesture recognition techniques in order to determine whether their use might bring improvements over the currently selected set of techniques. Examples of such techniques might be dynamic programming techniques such as dynamic time warping and distance metrics to recognize different gestures according to their trajectories, the use of Kalman filters in the preprocessing stage for motion prediction, or the use of more advanced machine learnings paradigms, such as deep neural networks, among others.

- Recreate and expand the user-centred experiments conducted by collecting data from a bigger population of participants, in order to better generalize the results obtained and further validate them, paying special attention to the profile of the user. In particular, it would especially interesting to study the perception of different user populations depending on their musical skill level, or, in the case of the aerobics instructor simulator, according to their level of health and fitness state.

BREVE RESUMEN EN CASTELLANO

Vivimos en una sociedad en la que estamos en constante interacción con ordenadores y dispositivos con capacidad de cómputo, ya sean ordenadores de sobremesa, portatiles, smartphones, etc. La inmensa variedad de diferentes tipos de aplicaciones existentes unido al desarrollo de nuevos dispositivos y tecnologías para la interacción ha propiciado la aparición de nuevos modelos y paradimas de interacción, generalmente hechos a la medida de cada aplicación en particular.

La música, por otra parte, siempre se ha distinguido por ser una actividad eminentemente interactiva, si bien existen una serie de barreras que dificultan o limitan el grado de acceso del usuario medio al mundo musical, tales como la complejidad abstracta de los conceptos musicales o la dificultad inherente a tocar un instrumento dado.

La aplicación de las nuevas tecnologías para el desarrollo de nuevos paradigmas de interacción persona-máquina en el ámbito de la interacción musical no sólo puede ayudar a reducir la dificultad de algunas de las barreras anteriormente citadas, sino que, además, puede dar lugar al desarrollo de nuevas herramientas que mejoren los procesos de aprendizaje musical y hacer más accesible la interacción musical a los usuarios en general.

Sorprendentemente, existe poca investigación sobre el uso de paradigmas de interacción avanzada en el ámbito musical, de ahí la motivación detrás de la presente tésis doctoral, que busca ahondar en el desarrollo de nuevas formas de interacción con la música, tanto desde el punto de vista educativo como el meramente creativo.

Para ello, se proponen los siguientes objetivos:

- Estudiar diferentes tipos de tecnologias y técnicas de procesado de seal para el desarrollo de aplicaciones musicales interactivas en tiempo real, minimizando el

grado de intrusión que el interfaz imponen al usuario y haciendo uso de técnicas que permitan extraer información a alto nivel a partir de los comandos introducidos por el usuario y de la propia estructura musical de la seal de audio.

- Estudiar e implementar aplicaciones interactivas desde el prisma recreativo, de tal forma que se puede determinar cómo mejorar la experiencia del usuario y la usabilidad del paradigma de interacción propuesto.

- Estudiar e implementar aplicaciones interactivas que permitan potenciar los procesos de aprendizaje tanto a nivel de la teoría musical como de la práctica de las destrezas adquiridas, proporcionando nuevas vías para tanto usuarios noveles como experimentados de cara a mejorar su habilidades.

- Investigar y desarrollar nuevos paradigmas de interacción que vayan más allá de los modelos de interacción convencionales, creando nuevas formas de expresión musical a través de aplicaciones de realidad virtual y aumentada.

- Evaluar los diferentes paradigma de interacción propuestos a través de estudios de usuario exhaustivos y la aplicación de metodología experimental, con objeto de verificar si la innovación en el interfaz comporta mejoras a nivel de la experiencia de usuario.

Interacción con la música y el aprendizaje musical

El primer aspecto que se aborda en la tesis es la aplicación de técnicas avanzadas de interacción y procesado musical para la implementación y mejora de las aplicaciones orientadas a potenciar la enseanza musical. Para ello, se han diseado una serie de aplicaciones que permitan servir de apoyo al estudiante en la práctica y aprendizaje de los elementos musicales, tanto a efectos de corregir los fallos cometidos por éstos en sus propias interpretaciones, como para guiarlos a lo largo del ensayo de las mismas.

En lo referente al uso de herramientas correctoras para el aprendizaje musical, se han abordado dos implementaciones concretas:

- La primera de ellas corresponde al diseo de un corrector automático de grabaciones polifónicas a piano, que permite a un usuario dado grabar sus interpretaciones y procesarlas automáticamente para detectar posibles fallos en las mismas. Esta aplicación se basa en procesar la seal musical de la grabación polifónica para encontrar

los *onsets* de la misma, esto es, los tiempos de ataque de las diferentes notas presentes en la grabación. Para ello, se enventana la seal de entrada y se procesa mediante un algoritmo iterativo que compara la energía de varias ventanas consecutivas conforme a una serie de umbrales definidos heurísticamente. Atendiendo a las propiedades frecuenciales y temporales de las notas musicales emitidas por el piano, se procede a etiquetar y filtrar los onsets encontrados, hasta llegar a una secuencia de instantes temporales que corresponden cada uno de ellos al momento en que se detecta una nueva nota musical, y a partir de la cual se construye una secuencia de particiones. Posteriormente, se analiza cada particion en cuanto a duración y frecuenci para determinar qué nota está sonando en la partición en cuestión. Una vez determinadas la localización temporal y la duración y frecuencia de cada nota detectada, se realiza un proceso de evaluación de la interpretación realizada comparando con una referencia de la pieza tocada en formato MIDI. El sistema implementado fue puesto a prueba con una serie de experimentos para su validación. Además, el detector de onsets implementado se evaluó de forma separada, obteniéndose muy buenos resultados siempre que el tempo de la pieza interpretada no excediera de 180 *beats* por minuto

- El segundo diseo considerado consiste en una herramienta de bajo coste para el apoyo a la enseanza musical básica basado en un robot, el sistema SolfaBOT®. SolfaBOT® se implementó tomando de base un modelo Lego Mindstorms NXT 2.0, sobre el cual se programó una aplicación que permitiera al usuario practicar la lectura de ritmos, la lectura de melodías o entonación, y el dictado musical. El principal escollo en la implementación fue dotar al sistema de 'oído musical', es decir, de la capacidad de dstinguir diferentes notas musicales de acuerdo a su valor frecuencial o *pitch*. Para ello, se diseó un módulo inteligente de detección de notas que aportase dicha funcionalidad. Dicho módulo consta de un bloque de captura de datos, que se encarga de acondicionar la seal de audio captada por el transductor, amplificarla, filtrarla y convertirla al dominio digital. El segundo bloque se encarga de procesar la seal acondicionada y de clasificar qué nota está sonando en cada momento, utilizando un sistema Arduino Severino con microcontrolador ATmega168. Debido a las limitaciones en la capacidad de cómputo y almacenamiento del sistema de bajo coste escogido, no era posible utilizar la transformada de Fourier para analizar los valores frecuenciales de las notas detectadas, por lo que en su lugar se recurrió al uso del algoritmo de Goertzel, que permite realizar el análisis deseado con complejidad en tiempo lineal. Finalmente, se realizaron una serie de pruebas de integración y evaluación de la funcionalidad del sensor implementado. Cada componente fue

probado por separado, corroborando su correcto funcionamiento. El sensor en su conjunto se evaluó a su vez utilizando una base de datos de notas musicales extraidas tanto de la base de datos RWC como de un conjunto de grabaciones realizadas con piano y flauta dulce.

Además de estas investigaciones orientadas hacia la creación de herramientas para ayudar al estudiante a corregir los errores en sus ensayos, también se ha estudiado la aplicación de estas técnicas para la implementación de aplicaciones que ofrezcan al usuario una guía durante sus ensayos. En concreto, se abordó este estudio bajo el punto de vista de simular la función de un director de orquesta para cuarteto de cuerda: 2 violines, 1 violonchelo y 1 contrabajo. De forma similar al corrector automático de grabaciones a piano, la información de la pieza musical en cuestión que se fuera a reproducir en el ensayo se almacena en formato MIDI internamente, y el sistema analiza la seal grabada por micrófono para evaluar al estudiante. El sistema incluye tres funcionalidades: afinador, evaluador de melodías y director virtual.

El afinador es un módulo que permite al usuario comprobar si el instrumento que desea tocar se encuentra afinado adecuadamente o no. Para ello, se analiza la seal de entrada a una frecuencia de muestreo tal que ofrezca resolución frecuencial de 0.5 Hz por muestra, y le se aplica un proceso de detección de picos basado en ventana deslizante. Para cada pico detectado, el sistema comprueba si puede encontrar otros picos que conformen junto con el primero un conjunto de parciales con su frecuencia fundamental. Si el sistema encuentra la frecuencia fundamental más al menos 2 parciales, o, en su defecto, 3 o más parciales, se compara la frecuencia fundamental asociada al patrón detectado con la que se tendria que tener en el instrumento, y se indica al usuario si la afinación es correcta o no.

El evaluador de melodías divide la seal de entrada en bloques que corresponden cada uno a un tiempo musical, utilizando como referencia para ello la partitura MIDI. Para cada tiempo musical, utiliza un modelo similar al afinador para encontrar la frecuencia fundamental de la nota tocada por el músico, y la compara con la que se tiene en la partitura MIDI para ese instante temporal. Se considera que la nota ha sido tocada correctamente si la diferencia entre fundamentales es menor que la que se tiene para los dos semitonos más pequeos que se pueden tocar con el instrumento en cuestión que se esté utilizando. Además, para evitar errores de alineamiento temporal, se compara cada fundamental detectado con el que se tiene no solo en el instante temporal MIDI correspondiente, sino también en los instantes adyacentes: el inmediatamente anterior y el inmediatamente posterior.

Por último, la simulación del director virtual de orquesta se realiza mediante una serie

de indicaciones que emulan las que se tendrían por parte del director en el mundo real. La batuta virtual se representa mediante 4 círculos, distribuídos conforme al modelo del compás de cuatro tiempos musicales, y cuyo color y tamao se modifica de forma acorde a la dinámica y el tiempo musical de la obra ensayada en cada momento. Además de sealizar el ritmo y la intensidad de cada nota o tiempo musical, el sistema permite indicar un calderón en la pieza interpretada mediante un círculo rojo en el centro de la batuta.

El sistema final fué evaluado por un grupo de músicos de la Orquesta de Cámara de Málaga, incluído un director de orquesta, con resultados muy positivos. En general, se apreció mucho la utilidad de la herramienta en tanto cubre una necesidad concreta que no se cubre con las herramientas habituales, y además ofrece un contexto adecuado para el ensayo tanto individual como grupal.

Interfaces persona-máquina avanzados para interacción musical

El segundo bloque abarcado por la tesis corresponde al diseo y uso de interfaces avanzados persona-máquina para aplicaciones musicales interactivas. Fundamentalmente, el objetivo es estudiar si el uso de técnicas y tecnologías de interacción avanzadas, tales como la captura de movimiento, permiten alcanzar mejoras en la calidad de la experiencia percibida, así como ofrecer una mayor inmersión por parte del usuario. Este estudio se ha estructurado bajo dos puntos de vista: por un lado, el diseo de simuladores de intrumentos musicales virtuales, y por el otro, la simulación de otros roles en experiencias musicales interactivas.

Instrumentos virtuales

Fundamentalmente, se han tratado tres implementaciones de instrumentos virtuales: la batería virtual, una batería a través de elementos cotidianos de escritorio, y el theremin.

En lo referente a la simulación de instrumentos virtuales, se ha prestado especial interés a intentar recrear una presentación virtual de una batería, ya que se trata de un instrumento que exige realizar movimientos relativamente amplios para ser tocado. Una primera implementación de la bateria virtual se basó en definir un volúmen virtual en torno al usuario, que vendria a representar el tambor que se ha de tocar. Este sistema, no obstante, no resulta adecuado para implementar una batería virtual, ya que resulta complicado de utilizar cuando se definen varios tambores, y existe un marcado retardo

introducido por el sistema de captura de movimientos (alrededor de 160 milisegundos), que se hace muy notorio a la hora de proceder a la simulación de un instrumento.

Para solucionar este problema, se procedió a definir un modelo para detectar e identificar gestos por parte del usuario que pudieran entenderse corresponden a un intento de dar un golpe sobre la batería virtual. Concretamente, utilizando un sistema de captura de movimientos basado en cámara 3D, Kinect, se realiza una modelización de las coordenadas nodales normalizadas del esqueleto del usuario, y a partir de éstas, se estudian los gestos realizados con celeridad suficiente para indicar una intención de golpear. Una vez segmentada la información que corresponde a un gesto rápido para cada mano, se procesa la secuencia de datos de entrada como si fuera una seal, aplicándosele un filtrado lineal de Wiener de 5 muestras para predecir futuras muestras, con objeto de compensar de esta forma el retardo introducido por el sistema.

Una vez se ha detectado un gesto rápido, es necesario diferenciarlo de otros gestos similares para poder decidir si se ha golpeado un tambor u otro. Para ello, se estudiaron dos conjuntos de descriptores a partir de la información espacial derivada de las coordenadas nodales: por un lado, se tomaron unos descriptores de la trayectoria esbozada por las manos en las últimas cinco muestras capturadas, y por el otro, un conjunto de descriptores que describen la posición del brazo del usuario en el momento en que se detecta un gesto rápido. Se conformó además una base de datos de gestos extraídos, con 1108 gestos reales que corresponden a intentar golpear uno de 6 posibles tambores (caja, timbales izquierdo y derecho, platillos izquierdo y derecho, y timbal lateral o charlet), y 977 gestos que no se corresponden a movimientos de interés.

Esta base de datos se utilizó conjuntamente con una serie de algoritmos de aprendizaje-máquina para encontrar la mejor combinación de algoritmo, parámetros y descriptores que permitía una clasificación óptima. Los algoritmos considerados fueron el Clasificador Bayesiano Ingenuo, Support Vector Machines con núcleos polimonial y gaussiano, clasificador k-nn, algoritmo C4.5 para árboles de decisión, Regresión Logística y Perceptrón Multicapa. Los resultados obtenidos demuestran que, en general, los descriptores utilizados son adecuados para describir los gestos que se pretenden caracterizar, alcanzándose tasas de éxito en la clasificación superiores al 90%. Los descriptores basados en la posición del brazo resultan ser más efectivos y determinantes que los descriptores basados en la trayectoria recorrida, y el Perceptrón Multicapa se muestra como el mejor algoritmo en la clasificación.

El estudio de la batería virtual se finalizó con una evaluación experimental con un total de 12 participantes, en la que se pidió a los mismos que realizaran una serie de pruebas con la aplicación diseada, para verificar el corrector funcionamiento de la aplicación y

para ver, además, si la predicción introducida conseguía anular el retardo introducido de forma efectiva. Tras estudiar los datos resultantes, se corroboró efectivamente que el sistema funcionaba correctamente, e igualmente, se encontró evidencia significativa de que el predictor de movimiento introducido hace casi imperceptible el retardo introducido por el sistema de captura de movimientos.

Además de la bateria virtual, también se abordó la implementación de una batería a partir de elementos cotidianos de una mesa de escritorio, de forma que el sistema se limitaria a analizar el sonido producido por cada objeto para posteriormente reproducirlo como un sonido de batería. La ventaja de este sistema sería que no precisa de más hardware específico que un micrófono para conseguir implementar su modelo de interacción. Este sistema analiza la seal de entrada al micrófono, la procesa y enventana, y calcula para cada ventana una serie de descriptores tempo-frecuenciales, un total de 17: energia, tasa de cruces por cero, centroide espectral, factor de caída espectral, flujo espectral, y los 12 primeros coeficientes de la escala Mel. Conjuntamente a estos descriptores, se consideraron un total de 4 clasificadores: análisis lineal discriminante, bayes ingenuo, análisis cuadratico discriminante, y k-nn.

Utilizando los descriptores y clasificadores anteriormente expuesto, se conformó una base de datos de 400 sonidos, de 4 tipos distintos. Tras realizar varias pruebas de clasificación, se obtuvo que los mejores resultados se tenían para los descriptores de flujo y caída espectral y el tercer coeficiente de Mel para clasificación con análisis cuadrático discriminante, y para estos mismos descriptores más la tasa de cruces por cero para una clasificación basada en análisis lineal discriminante.

Posteriormente, estas dos combinaciones se probaron en una serie de contextos con diferentes tipos de sonidos generados con objetos cotidianos, y se aadió además la posibilidad de que algunos sonidos se diesen simultaneamente, incrementando el número de clases a 6. La segunda combinación obtuvo los mejores resultados, y se realizó una nueva tanda de pruebas con usuarios para que evaluasen la utilidad percibida por la aplicación. Los resultados obtenidos denotan una acogida positiva, si bien es de remarcar que el retardo introducido por el preprocesamiento de las muestras resulta molesto en la experiencia del usuario.

Por último, también se aplicó el uso de herramientas de captura de movimiento para implementar un Theremin, un tipo de instrumento electrónico que consiste en manipular la propiedades de un campo electromagnético al acercar o alejar las manos de un par de antenas, generando variaciones en la frecuencia e intensidad de los tonos que se emiten en consecuencia. Para ello, se tomaron los valores de la posición de las manos usando un dispositivo Kinect, de forma tal que la altura relativa de las manos respecto a la cabeza

del usuario le permitía controlar los parámetros de un proceso de pitch-shifting que se aplicaba a una determinada seal de entrada. Este proeso se basa en una implementación basada de phase-vocoder para analizar y resintetizar la seal con los nuevos valores de frecuencia, utilizando un análisis basado en la transformada corta de Fourier. La seal resultante es reproducida en bucle constantemente, y los cambios a los parámetros del proceso de pitch-shifting se realizan en tiempo real, por lo que el usuario percibe de inmediato los cambios que él mismo introduce en la reproducción.

Experiencias musicales interactivas

Además de la simulación de instrumentos musicales virtuales, también se ha abordado el estudio del uso de interfaces avanzados para el desarrollo de aplicaciones interactivas musicales en las que no existe una simulación explícita de un instrumento musical, sino que, en su lugar, se propone al usuario tomar parte en un rol distinto.

El ejemplo más claro lo tenemos en la simulación del rol de director de orquesta. La comunicación entre el director y la orquesta se sustenta fundamentalmente en una vía de comunicación gestual, lo cual hace que el modelo interactivo de este rol sea razonable de abordar mediante el uso de un dispositivo de captura de movimientos basado en cámara, como es el caso del dispositivo Kinect. Para ser posible de caracterizar el modelo de interacción del director de orquesta de forma adecuada, es necesario disponer de alguna técnica de procesado que permita acelerar o decelerar el tempo de la obra que está siendo reproducida sin modificar los contenidos frecuenciales de la misma, es decir, el procesado dual al pitch-shifting anteriormente propuesto para el Theremin virtual, y que se suele denotar por time-stretching. En este caso, se plantearon dos soluciones: el algoritmo SOLA (Synchronous Overlap and Add), y la re-adaptación del bloque de análisis/síntesis del phase vocoder para el algoritmo de time-stretching, siendo esta última la opción finalmente escogida, en tanto el algoritmo SOLA no está indicado para su uso con obras polifónicas.

Para sealizar los diferentes tiempos musicales, se utiliza un movimiento horizontal con la mano derecha, de forma que las posiciones de inicio y final de la mano en cada movimiento definen un nuevo tempo, que se relaciona con el tempo de la obra original para obtener el factor de time-stretching efectivo que ha de aplicarse sobre la pieza reproducida. Además, se impusieron una serie de restricciones para evitar que el sistema respondiera de forma antinatural o ante ruído espúreo, tales como limitar el rango del factor de time-stretching a valores comprendidos entre 0.5 y 2, y restricciones en cuanto a la velocidad y extensión mínima del movimiento realizado. Además, hay que asegurar que los tiempos

musicales de la reproducción coinciden con los tiempos musicales indicados por el director, de otra forma la sensación introducida desde el punto de vista del usuario es que la orquesta no es capaz de seguir las indicaciones dadas, y mantiene un retardo respecto a cada cambio de tempo, que además se va acumulando. La mano izquierda se utiliza para seleccionar un conjunto de instrumentos concretos, y modificar las dinámicas de volumen de los mismos, alzando o bajando la mano de forma acorde.

Se realizó una implementación concreta en una aplicación, en la que sonaba en bucle un fragmento de la pieza de Peer Gynt "En la gruta del rey de la montaa", permitiendo al usuario controlar indistintamente el volúmen del conjunto de violines por un lado, y del conjunto de trombones por el otro, así como el tempo de toda la obra. Se realizó una evaluación experimental con 24 participantes, tomándose los parámetros de control de la obra (dinámica y tempo) como factores experimentales, y observándose su efecto sobre aspectos relativos a la experiencia de usuario (satisfacción, sensación de control, retardo, etc.). Los resultados demuestran que el modelo de interacción propuesto tiene una contribución positiva en general en la experiencia del usuario, si bien existen ciertos aspecto en los que requiere de cierto refinamiento, principalmente por las limitaciones que introduce el sistema de captura de movimientos, que hacen que la oclusión introducida por los brazos del propio usuario afecte al grado de control del que éste dispone sobre el sistema.

Además de la simulación del rol de director de orquesta, también se diseo un sistema que emulaba a un entrenador de step-aerobics, utilizando una tabla de presión Wiiboard para que el usuario realizara sus ejercicios. El sistema es capaz de analizar una canción como seal de entrada, utilizando un modelo de detección de onsets basado en el cálculo del flujo espectral multibanda de dicha entrada para, a su vez, estimar el tempo de dicha canción y la intensidad rítmica de la misma. El tempo y la posición exacta de los tiempos musicales se estiman mediante un sistema de seguimiento basado en agentes. La intensidad rítmica se determina mediante un procesado de clustering, utilizando el algoritmo k-nn para categorizar el nivel rítmico de cada fragmento en la canción; posteriormente, al reproducir la pieza analizada, el sistema reproduce por pantalla una serie de comandos de steps, que aparecen en sincronía con los tiempos musicales previamente identificados, de forma que para fragmentos de mayor intensidad rítmica se presentan ejercicios más complejos. El sistema se probó por un conjunto de usuarios que realizaron una sesión de steps con el modelo propuesto, registrándose nuevamente una buena acogida, si bien quedó patente que el sistema necesita utilizar una vía de comunicación distinta de la visual para presentar los ejercicios de steps a realizar.

Interacción emocional cerebro-máquina

El último bloque abordado en esta tesis corresponde al estudio de interfaces avanzados más inusuales, como es el caso del interfaz cerebro-máquina. El propósito de este último bloque es investigar sobre la viabilidad de crear interfaces emocionales que permitan estimar qué siente un usuario dado al ser expuesto a un cierto contenido audiovisual.

Para ello, se ha utilizado un sistema de electrodos de bajo coste basado en el sitema EMOTIV EPOC, y se utiliza el mismo para capturar seales electro-encefalo-gráficas (EEG). Se llevó a cabo un experimento enfocado a capturar diferentes conjuntos de muestras EEG en situaciones de exposición a elementos audiovisuales asociados a una determinada emoción de entre 24 posibles, lo cual permite en última instancia obtener una base de datos EEG etiquetados conforme a la emoción a la que se asocian.

Posteriormente, una vez obtenida la pertinente base de datos, se aplica un análisis en tiempo y frecuencia a las seales EEG extraídas para obtener unos descriptores de las mismas. Dado que las seales EEG son fuertemente no estacionarias, el análisis de Fourier no es apto para extraer sus características frecuenciales, por lo que se utilizó en su lugar una descomposición Wavelet en 8 niveles, usando una wavelet de Daubechies de sexto orden, lo cual genera un total de 9 subseales por cada seal EEG analizada. Para cada una de estas subseales se definieron dos descriptores: energía y entropía, generando un total de 18 descriptores frecuenciales por seal EEG. Además, también se definen un conjunto de 6 descriptores temporales por seal EEG. Estos descriptores se utilizaron conjuntamente con una serie de algoritmos de aprendizaje-máquina (Perceptrón Multicapa, Support Vector Machines con núcleo polinómico y árboles de decisión) para obtener finalmente unas tasas de clasificación por emoción. El resultado final demuestra que el sistema es capaz de estimar emociones a partir de un interfaz cerebral con una tasa de aciertos en relación a la cardinalidad del espacio emocional elevada.

Conclusiones

El trabajo realizado ha permitido la implementación y diseo en una amalgama de modelos y técnicas de interacción que, acompaados de su correspondiente aplicación interactiva, han sido evaluados experimentalmente, tanto en cuanto a la funcionalidad de sus componentes como en lo referente al impacto que suponen a nivel de la experiencia de usuario. Las conclusiones más relevantes que se han extraído de este estudio son:

- Se ha verificado que efectivamente el uso de técnicas de análisis y procesado de la seal musical posibilita extraer información de alto nivel de ésta, y que esta información

puede utilizarse para disear herramientas de apoyo al aprendizaje musical.

- Estas mismas técnicas permiten el diseo de interfaces inteligentes a bajo coste, sin necesidad de utilizar soluciones hardware específicas, ya que todo el procesamiento puede realizarse a bajo coste, como certifica el caso de la herramienta SolfaBOT®.

- La aplicación de estas técnicas en el diseo de herramientas de apoyo al aprendizaje cubre parcialmente la necesidad de contar con el apoyo de un experto a la hora de ensayarl

- La implementación o simulación de instrumentos musicales virtuales a partir de elementos de interacción más genéricos y asequibles económicamente es viable, y el uso conjunto de estos elementos y dispositivos con técnicas de procesado avanzadas permite superar las limitaciones propias de los primeros y conseguir metáforas de interacción musical más fidedignas.

- El uso de interfaces persona-máquina avanzados ofrece mejoras potenciales en la experiencia a nivel de usuario en aplicaciones interactivas, como atestigua la respuesta predominantemente muy positiva que se ha registrado en las evaluaciones experimentales y pruebas de usuario realizadas.

- Uno de los elementos más limitantes a la hora de implementar experiencias musicales interactivas realistas es el aspecto del tiempo. Incluso en casos en lo que no existía una apreciación consciente de ello, la presencia de una ínfima latencia en el sistema es suficiente para que la percepción del usuario de su experiencia interactiva se degrade.

- La combinación de técnicas de procesado de seal conjuntamente con algoritmos de aprendizaje máquina permite conseguir altas tasas de reconocimiento gestual en aplicaciones interactivas en tiempo real.

- El uso de técnicas de procesado digital junto con interfaces cerebro-computadora de bajo coste permite desarrollar paradigmas de interacción para estimación emocional.

JOURNAL AND CONFERENCE PAPERS DERIVED FROM THESE WORKS

- Drum-hitting gesture recognition and prediction system using Kinect Rosa-Pujazón et al. [2013b].
- Conducting a virtual ensemble with a Kinect device Rosa-Pujazón et al. [2013a].
- Fast-gesture recognition and classification using Kinect: an application for virtual reality drumkits Rosa-Pujazón et al. [2014a].
- A virtual reality drumkit simulator system with a Kinect device Rosa-Pujazón et al. [2014b].
- Drumkit simulator from everyday desktop object Herrero et al. [2014].
- Low-cost step aerobics system with virtual aerobics trainer Rosa-Pujazón et al. [2014c].
- Human-Computer interaction and Music Barbancho et al. [2013b].
- Virtual conductor for string quartet practice Baez et al. [2013]
- Correction system for polyphonic piano recordings Martin-Erdozain et al. [2013].
- Solfabot – Low-cost support tool for solfeo training Barbancho et al. [2013a].

REFERENCES

G Agostini, M Longari, and E Pollastri. Musical instrument timbres classification with spectral features. In *Multimedia Signal Processing, 2001 IEEE Fourth Workshop on*, pages 97–102, 2001. doi: 10.1109/MMSP.2001.962718. 60

DW Aha, D Kibler, and MK Albert. Instance-based learning algorithms. *Machine learning*, 6(1):37–66, 1991. 42, 45

M Akin. Comparison of wavelet transform and fft methods in the analysis of eeg signals. *Journal of Medical Systems*, 26(3):241–247, 2002. 101

NS Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992. 42, 45

AN Antle, M Droumeva, and G Corness. Playing with the sound maker: do embodied metaphors help children learn? In *Proceedings of the 7th international conference on Interaction design and children*, pages 178–185. ACM, 2008. 6

E Ardizzone, A Chella, and R Pirrone. Pose classification using support vector machines. In *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, volume 6, pages 317–322. IEEE, 2000. 8

T Asfour, P Azad, F Gyarfas, and R Dillmann. Imitation learning of dual-arm manipu-

lation tasks in humanoid robots. *International Journal of Humanoid Robotics*, 5(02): 183–202, 2008. 8

H Aurlien, IO Gjerde, JH Aarseth, G Eldøen, B Karlsen, H Skeidsvoll, and NE Gilhus. Eeg background activity described by a large computerized database. *Clinical Neurophysiology*, 115(3):665–673, 2004. 96

T Baba, M Hashida, and H Katayose. "virtualphilharmony": A conducting system with heuristics of conducting an orchestra. In *Proceedings of the 2010 Conference on New Interfaces for Musical Expression (NIME 2010)*, pages 263–270, 2010. 7, 86

R Baez, AM Barbancho, A Rosa-Pujazón, I Barbancho, and LJ Tardón. Virtual conductor for string quartet practice. In *SMAC 2013 - Stockholm Music Acoustics Conference 2013*, pages 292–298, 2013. 121

P Bakanas, J Armitage, J Balmer, P Halpin, K Hudspeth, and K Ng. mconduct: Gesture transmission and reconstruction for distributed performance. In *ECLAP 2012 Conference on Information Technologies for Performing Arts, Media Access and Entertainment*, page 107. Firenze University Press, 2012. 7, 86

S Bakker, E van den Hoven, and AN Antle. Moso tangibles: evaluating embodied learning. In *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*, pages 85–92. ACM, 2011. 6

JP Bandera, Rebeca Marfil, Antonio Bandera, Juan Antonio Rodríguez, Luis Molina-Tanco, and F Sandoval. Fast gesture recognition based on a two-level representation. *Pattern Recognition Letters*, 30(13):1181–1189, 2009. 8, 51

JP Bandera, JA Rodríguez, L Molina-Tanco, and A Bandera. A survey of vision-based architectures for robot learning by imitation. *International Journal of Humanoid Robotics*, 9(01), 2012. 7, 8

AM Barbancho, I Barbancho, B Soto, and LJ Tardón. Sic receiver for polyphonic piano music. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 377–380. IEEE, 2011. 13

AM Barbancho, A Ortiz, I Barbancho, A Rosa-Pujazón, and LJ Tardón. Solfabot – low-cost support tool for solfeo training. In *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pages 400–404, 2013a. 121

I Barbancho, AM Barbancho, A Jurado, and LJ Tardón. Transcription of piano recordings. *Applied acoustics*, 65(12):1261–1287, 2004. 11

I Barbancho, A Rosa-Pujazón, LJ Tardón, and AM Barbancho. Human–computer interaction and music. In *Sound-Perception-Performance*, pages 367–389. Springer, 2013b. 121

R Basili, A Serafini, and A Stellato. Classification of musical genre: a machine learning approach. In *Proceedings of ISMIR*. Citeseer, 2004. 60

JP Bello, L Daudet, S Abdallah, C Duxbury, M Davies, and MB Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035–1047, 2005. 11, 58

E Benetos and S Dixon. Polyphonic music transcription using note onset and offset detection. In *Proceedings of 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 37–40. IEEE, 2011. 11

CG Boogaart and R Lienhart. Note onset detection for the transcription of polyphonic piano music. In *Proceedings of the 2009 IEEE international conference on Multimedia and Expo*, pages 446–449. IEEE Press, 2009. 11

J Borchers, E Lee, W Samminger, and M Mühlhäuser. Personal orchestra: A real-time audio/video system for interactive conducting. *Multimedia Systems*, 9(5):458–465, 2004. 7, 85, 86

DO Bos. Eeg-based emotion recognition. *The Influence of Visual and Auditory Stimuli*, pages 1–17, 2006. 100

MM Bradley and PJ Lang. The international affective digitized sounds: affective ratings of sounds and instruction manual. *University of Florida, Gainesville, FL, Technical Report B-3*, 2007. 98

D Bradshaw and K Ng. Analyzing a conductor's gestures with the Wiimote. In *Proceedings of EVA London 2008: the International Conference of Electronic Visualisation and the Arts*, 2008. 7, 86

J Breebaart and M Mckinney. Features for audio classification. In *Proceedings of the Philips Symposium of Intelligent Algorithms, Eindoven*, 2002. 60

BW Brown. *Beyond ANOVA: basics of applied statistics*, volume 40. CRC Press, 1997. 53, 83

JC Brown. Computer identification of musical instruments using pattern recognition with cepstral coefficients as features. *The Journal of the Acoustical Society of America*, 105: 1933, 1999. 60

G Buzsaki. *Rhythms of the Brain*. Oxford University Press, 2009. 96

S Calinon. Continuous extraction of task constraints in a robot programming by demonstration framework. *Unpublished doctoral dissertation, Ecole Polytechnique Fédérale de Lausanne (EPFL)*, 2007. 7

S Calinon and A Billard. Stochastic gesture production and recognition model for a humanoid robot. In *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004).*, volume 3, pages 2769–2774. IEEE, 2004. 8

D Cao, OT Masoud, D Boley, and N Papanikolopoulos. Human motion recognition using support vector machines. *Computer Vision and Image Understanding*, 113(10):1064–1075, 2009. 8, 51

G Castellano, R Bresin, A Camurri, and G Volpe. Expressive control of music and visual media by full-body movement. In *Proceedings of the 7th international conference on New interfaces for musical expression*, pages 390–391. ACM, 2007. 6

S Celebi, AS Aydin, TT Temiz, and T Arici. Gesture recognition using skeleton data with weighted dynamic time warping. *Computer Vision Theory and Applications. Visapp*, 2013. 8

G Chanel, J Kronegg, D Grandjean, and T Pun. Emotion assessment: Arousal evaluation using eegs and peripheral physiological signals. In *Multimedia content representation, classification and security*, pages 530–537. Springer, 2006. 103

G Chanel, JJM Kierkels, M Soleymani, and T Pun. Short-term emotion assessment in a recall paradigm. *International Journal of Human-Computer Studies*, 67(8):607–627, 2009. 103

C Chen, J Liang, H Zhao, H Hu, and J Tian. Factorial HMM and parallel HMM for gait recognition. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 39(1):114–123, 2009. 8

L Chen, MT Özsu, and V Oria. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 491–502. ACM, 2005. 8

MH Cheng, MF Ho, and CL Huang. Gait analysis for human identification through manifold learning and HMM. *Pattern Recognition*, 41(8):2541–2553, 2008. 8

C Cortes and V Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. 42

A Croitoru, P Agouris, and A Stefanidis. 3d trajectory matching by pose normalization. In *Proceedings of the 13th annual ACM international workshop on Geographic information systems*, pages 153–162. ACM, 2005. 8

JD Deng, C Simmermacher, and S Cranefield. A study on feature analysis for musical instrument classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 38(2):429–438, 2008. ISSN 1083-4419. doi: 10.1109/TSMCB.2007.913394. 60

S Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30(1):39–58, 2001. 89

S Dixon. Onset detection revisited. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx-06)*, pages 133–137, 2006. 88

AH El-Baz and AS Tolba. An efficient algorithm for 3d hand gesture recognition using combined neural classifiers. *Neural Computing and Applications*, pages 1–8, 2013. 8

A Eronen. Comparison of features for musical instrument recognition. In *In Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2001. 59

G Essl and M Rohs. Interactivity for mobile music-making. *Organised Sound*, 14(02):197–207, 2009. 6

O Gillet and G Richard. Automatic transcription of drum loops. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '04)*, volume 4, pages iv–269 – iv–272 vol.4, may 2004. doi: 10.1109/ICASSP.2004.1326815. 59

G Goertzel. An algorithm for the evaluation of finite trigonometric series. *American Mathematical Monthly*, 65(1):34–35, 1958. 19

I Gold. Does 40-hz oscillation play a role in visual consciousness? *Consciousness and Cognition*, 8(2):186 – 195, 1999. ISSN 1053-8100. doi: http://dx.doi.org/10.1006/ccog.1999.0399. URL

http://www.sciencedirect.com/science/article/pii/S1053810099903999. 100

G Gómez-Herrero, W De Clercq, H Anwar, O Kara, K Egiazarian, S Van Huffel, and W Van Paesschen. Automatic removal of ocular artifacts in the eeg without an eog reference channel. In *Proceedings of the 7th Nordic Signal Processing Symposium, NORSIG 2006*, pages 130–133. IEEE, 2006. 100

M Goto et al. Development of the rwc music database. In *Proceedings of the 18th International Congress on Acoustics (ICA 2004)*, volume 1, pages 553–556, 2004. 20

F Gouyon, F Pachet, and Olivier D. On the use of zero-crossing rate for an application of classification of percussive sounds. In *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00)*, 2000. 58

L Gower and J McDowall. Interactive music video games and children's musical development. *British Journal of Music Education*, 29(01):91–105, 2012. 6

S Grollmisch, C Dittmar, and G Gatzsche. Concept, implementation and evaluation of an improvisation based music video game. In *Proceedings of the International IEEE Consumer Electronics Society's Games Innovations Conference, ICE-GIC 2009*, pages 210–212. IEEE, 2009. 5

MK Halpern, J Tholander, M Evjen, S Davis, A Ehrlich, K Schustak, E.P.S. Baumer, and G. Gay. Moboogie: creative expression through whole body musical interaction. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 557–560. ACM, 2011. 6

JC Hansen. *LEGO Mindstorms NXT Power Programming: Robotics in C*. Variant Press, 2007. 17

S Harada, JO Wobbrock, and JA Landay. 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*, pages 27–34. ACM, 2007. 6

S Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994. 42, 48

SS Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall Englewood Cliffs, NJ, 2007. 42, 48

P Herrera, A Yeterian, and F Gouyon. Automatic classification of drum sounds: a comparison of feature selection methods and classification techniques. *Music and Artificial Intelligence*, pages 69–80, 2002. 59, 60

P Herrera-Boyer, G Peeters, and S Dubnov. Automatic classification of musical instrument sounds. *Journal of New Music Research*, 32(1):3–21, 2003. 59

G Herrero, I Barbancho, LJ Tardón, A Rosa-Pujazón, and AM Barbancho. Drumkit simulator from everyday desktop object. In *Journal on Multimedia Tools and Applications, under review process*, 2014. 121

S Holland, AJ Bouwer, M Dalgelish, and TM Hurtig. Feeling the beat where it counts: fostering multi-limb rhythm skills with the haptic drum kit. In *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction*, pages 21–28. ACM, 2010. 6

A Höofer, A Hadjakos, and M Mühlhäuser. Gyroscope-Based Conducting Gesture Recognition. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 175–176, 2009. URL http://www.nime.org/proceedings/2009/nime2009_175.pdf. 6

DW Hosmer Jr, S Lemeshow, and RX Sturdivant. *Applied logistic regression*. Wiley. com, 2013. 40, 42, 48

DC Howell. *Statistical methods for psychology*. Cengage Learning, 2011. 53, 82

JR Hughes. Gamma, fast, and ultrafast waves of the brain: their relationships with epilepsy and behavior. *Epilepsy & Behavior*, 13(1):25–31, 2008. 96

JY Hwang and G Yang. Mobile virtual guitar having an auto-fingerboard, and method therefor, August 31 2012. WO Patent 2,012,115,299. 6

M Ihara, S Maeda, and S Ishii. Instrument identification in monophonic music using spectral information. In *Proceedings of the 2007 IEEE International Symposium on Signal Processing and Information Technology*, pages 595–599, 2007. doi: 10.1109/ISSPIT.2007.4458100. 59

II Itauma, H Kivrak, and H Kose. Gesture imitation using machine learning techniques. In *Signal Processing and Communications Applications Conference (SIU), 2012 20th*, pages 1–4. IEEE, 2012. 8, 51

MG Jacob and JP Wachs. Context-based hand gesture recognition for the operating room. *Pattern Recognition Letters*, 2013. 7, 8, 51

A Jensen and A la Cour-Harbo. *Ripples in mathematics: the discrete wavelet transform*. springer, 2001. 101

K Jensen. Envelope model of isolated musical sounds. In *Proceedings of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx99)*, 1999. 12

GH John and P Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 338–345. Morgan Kaufmann Publishers Inc., 1995. 42

S Jordà. The reactable: tangible and tabletop music performance. In *Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems*, pages 2989–2994. ACM, 2010. 6

G. Junker. *Pro OGRE 3D programming*. Apress, 2006. 34, 74

ET Khoo, T Merritt, VL Fei, W Liu, H Rahaman, J Prasad, and T Marsh. Body music: physical exploration of music theory. In *Proceedings of the 2008 ACM SIGGRAPH symposium on Video games*, pages 35–42, 2008. 6

D Kim, J Song, and D Kim. Simultaneous gesture segmentation and recognition based on forward spotting accumulative HMMs. *Pattern Recognition*, 40(11):3012–3026, 2007. 7, 51

Elif Kirmizi-Alsan, Zubeyir Bayraktaroglu, Hakan Gurvit, Yasemin H Keskin, Murat Emre, and Tamer Demiralp. Comparative analysis of event-related potentials during go/nogo and cpt: decomposition of electrophysiological markers of response inhibition and sustained attention. *Brain research*, 1104(1):114–128, 2006. 96

NP Lago and F Kon. The quest for low latency. In *Proceedings of the International Computer Music Conference*, pages 33–36, 2004. 36

PJ Lang, MM Bradley, and BN Cuthbert. International affective picture system (iaps): Technical manual and affective ratings, 1999. 98

J Laroche and M Dolson. New phase-vocoder techniques for pitch-shifting, harmonizing and other exotic effects. In *Applications of Signal Processing to Audio and Acoustics, 1999 IEEE Workshop on*, pages 91–94. IEEE, 1999. 70

S Le Cessie and JC Van Houwelingen. Ridge estimators in logistic regression. *Applied statistics*, pages 191–201, 1992. 42, 48

E Lee, TM Nakra, and J Borchers. You're the conductor: a realistic interactive conducting system for children. In *Proceedings of the 2004 conference on New interfaces for musical expression*, pages 68–73. National University of Singapore, 2004. 7, 86

HK Lee and JH Kim. An HMM-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):961–973, 1999. 7, 51

G Levin and Z Lieberman. In-situ speech visualization in real-time interactive installation and performance. In *Non-Photorealistic Animation and Rendering: Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, volume 7, pages 7–14, 2004. 6

H Li and M Greenspan. Model-based segmentation and recognition of dynamic gestures in continuous video streams. *Pattern Recognition*, 44(8):1614–1628, 2011. 8, 51

M Li, Q Chai, T Kaixiang, A Wahab, and H Abut. Eeg emotion recognition system. In *In-vehicle corpus and signal processing for driver behavior*, pages 125–135. Springer, 2009. 103

MA Livingston, J Sebastian, Z Ai, and JW Decker. Performance measurements for the microsoft Kinect skeleton. In *Proceedings of the 2012 IEEE Virtual Reality Workshops*, pages 119–120. IEEE Computer Society, 2012. 36

AA Livshin and X Rodet. Musical instrument identification in continuous recordings. In *In Proc. of DAFX*, 2004. 60

B Logan et al. Mel frequency cepstral coefficients for music modeling. In *ISMIR*, 2000. 59

D Malah. Time-domain algorithms for harmonic bandwidth reduction and time scaling of speech signals. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 27 (2):121–133, 1979. 75, 76

M Mandanici and S Sapir. Disembodied voices: A kinect virtual choir conductor. *http://www.smcnetwork.org/system/files/smc2012-174.pdf, last retrieved 20/09/2013*, 2012. 6

A Mannini and AM Sabatini. Machine learning methods for classifying human physical activity from on-body accelerometers. *Sensors*, 10(2):1154–1175, 2010. 7, 51

M. Martin-Erdozain, I Barbancho, A Rosa-Pujazón, and AM Barbancho. Correction system for polyphonic piano recordings. 2013. 121

H Morita, S Hashimoto, and S Ohteru. A computer music system that follows a human conductor. *Computer*, 24(7):44–53, 1991. 7

M Muhlig, M Gienger, S Hellbach, J J Steil, and C Goerick. Task-level imitation learning using variance-based movement optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA'09.*, pages 1177–1184. IEEE, 2009. 8, 51

T Nakra, Y Ivanov, P Smaragdis, and C Ault. The UBS virtual maestro: An interactive conducting system. *Proceedings of the International Conference on New Interfaces for Musical Expression, NIME2009*, pages 250–255, 2009. 7, 86

J Neter, W Wasserman, and MH Kutner. *Applied linear statistical models: regression, analysis of variance, and experimental designs*. 1990. 53, 83

KC Ng. Music via motion: transdomain mapping of motion and sound for interactive performances. *Proceedings of the IEEE*, 92(4):645–655, 2004. 6

E Niedermeyer. Alpha rhythms as physiological and abnormal phenomena. *International Journal of Psychophysiology*, 26(1-3):31–49, 1997. 96

G Odowichuk, S Trail, P Driessen, W Nie, and W Page. Sensor fusion: Towards a fully expressive 3d music control interface. In *Proceedings of the 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PacRim)*, pages 836–841. IEEE, 2011. 6, 36

OpenNI User Guide. OpenNI organization, November 2010. URL http://www.openni.org/documentation. Last viewed 19-12-2013. 37

AV Oppenheim, RW Schafer, JR Buck, et al. *Discrete-time signal processing*, volume 5. Prentice hall Upper Saddle River, 1999. 19

D O'shaughnessy. *Speech communication: human and machine*. Universities press, 1987. 59

G Padmavathi, D Shanmugapriya, and M Kalaivani. Acoustic signal based feature extraction for vehicular classification. In *Proceedings of the 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, volume 2, pages V2–11 – V2–14, aug. 2010. doi: 10.1109/ICACTE.2010.5579804. 58

L Peng and D Gerhard. A Wii-based gestural interface for computer-based conducting systems. In *Proceedings of the 2009 Conference on New Interfaces For Musical Expression*, 2009. 7, 86

RT Pivik and K Harman. A reconceptualization of eeg alpha activity as an index of arousal during sleep: all alpha activity is not equal. *Journal of sleep research*, 4(3): 131–137, 1995. 96

R Plutchik. *Emotion: A psychoevolutionary synthesis*. Harper & Row New York, 1980. 99

Prime Sensor NITE 1.3 Algorithms notes. PrimeSense Inc., 2010. URL http://www.primesense.com. Last viewed 19-12-2013. 37

Y Qin. A study of Wii/Kinect controller as musical controllers. *http://www.music.mcgill.ca/ ying/McGill/MUMT620/Wii-Kinect.pdf, last retrieved 20/09/2012*. 6

JR Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993. 42, 46

M Rangaswamy, B Porjesz, DB Chorlian, K Wang, KA Jones, LO Bauer, J Rohrbaugh, SJ OConnor, S Kuperman, T Reich, et al. Beta power in the eeg of alcoholics. *Biological psychiatry*, 52(8):831–842, 2002. 96

I Rish. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001. 42

A Rosa-Pujazón, I Barbancho, LJ Tardón, and AM Barbancho. Conducting a virtual ensemble with a kinect device. In *SMAC 2013 - Stockholm Music Acoustics Conference 2013*, pages 284–291, 2013a. 121

A Rosa-Pujazón, I Barbancho, LJ Tardón, and AM Barbancho. Drum-hitting gesture recognition and prediction system using Kinect. In *I Simposio Espaol de Entrenimiento Digital SEED 2013*, pages 108–118, 2013b. 121

A Rosa-Pujazón, I Barbancho, LJ Tardón, and AM Barbancho. Fast-gesture recognition and classification using Kinect: an application for virtual reality drumkits. In *Acta Acustica, under review process*, 2014a. 121

A Rosa-Pujazón, I Barbancho, LJ Tardón, and AM Barbancho. A virtual reality drumkit simulator system with a Kinect device. In *International Journal of Creative Interfaces and Computer Graphics, accepted for publication*, 2014b. 121

A Rosa-Pujazón, I Barbancho, LJ Tardón, and AM Barbancho. Low-cost step aerobics system with virtual aerobics trainer. In *in preparation*, 2014c. 121

S Sanei and JA Chambers. *EEG signal processing*. Wiley. com, 2008. 97, 100

E Scheirer and M Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 2, pages 1331–1334. IEEE, 1997. 58

C Schuldt, I Laptev, and B Caputo. Recognizing human actions: a local SVM approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004. 8

D. Shreiner. *OpenGL reference manual: The official reference document to OpenGL, version 1.2*. Addison-Wesley Longman Publishing Co., Inc., 1999. 34, 74

C Stanton, A Bogdanovych, and E Ratanasena. Teleoperation of a humanoid robot using full-body motion capture, example movements, and machine learning. In *Proc. Australasian Conference on Robotics and Automation*, 2012. 8, 51

H Stark and JW Woods. *Probability, random processes, and estimation theory for engineers*. Prentice-Hall Englewood Cliffs (NJ), 1986. 59, 60

I Steinwart and A Christmann. *Support vector machines*. Springer, 2008. 42

MB Sterman. Physiological origins and functional correlates of eeg rhythmic activities: implications for self-regulation. *Biofeedback and self-regulation*, 21:3–33, 1996. 96

C Stierman. *KiNotes: Mapping musical scales to gestures in a Kinect-based interface for musican expression*. PhD thesis, MSc Thesis University of Amsterdam, 2012. 6

K Takahashi. Remarks on SVM-based emotion recognition from multi-modal bio-potential signals. In *Robot and Human Interactive Communication, 2004. ROMAN 2004. 13th IEEE International Workshop on*, pages 95–100. IEEE, 2004. 103

LJ Tardón, S Sammartino, and I Barbancho. Design of an efficient music-speech discriminator. *The Journal of the Acoustical Society of America*, 127:271, 2010. 57, 59

M Teplan. Fundamentals of eeg measurement. *Measurement science review*, 2(2):1–11, 2002. 96

S Theodoridis and K Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, 4th edition, 2008. ISBN 1597492728, 9781597492720. 58

LS Theremin. Theremin, February 14 1928. US Patent 1,658,953. 70

T Todoroff, J Leroy, and C Picard-Limpens. Orchestra: Wireless sensor system for augmented performances & fusion with kinect. *QPSR of the numediart research program*, 4(2), 2011. 6

S Trail, M Dean, TF Tavares, G Odowichuk, P Driessen, WA Schloss, and G Tzanetakis. Non-invasive sensing and gesture control for pitched percussion hyper-instruments using the Kinect. 2012. 6

CH Vanderwolf. Are neocortical gamma waves related to consciousness? *Brain research*, 855(2):217–224, 2000. 96

CN von Ende. Repeated-measures analysis. *Design and analysis of ecological experiments. Oxford University Press, New York*, pages 134–157, 2001. 53, 82

CY Wang and AF Lai. Development of a mobile rhythm learning system based on digital game-based learning companion. *Edutainment Technologies. Educational Games and Virtual Reality/Augmented Reality Applications*, pages 92–100, 2011. 6

N Wiener. Extrapolation, interpolation, and smoothing of stationary time series. 1964. 38

J Yamato, J Ohya, and K Ishii. Recognizing human action in time-sequential images using hidden markov model. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 379–385. IEEE, 1992. 8

MJ Yoo, JW Beak, and IK Lee. Creating musical expression using kinect. *Proceedings of the 2011 Conference on New Interfaces for Musical Expression, Oslo, Norway*, 2011. 6

HS Yoon, J Soh, YJ Bae, and H Seung Yang. Hand gesture recognition using combined features of location, angle and velocity. *Pattern Recognition*, 34(7):1491–1501, 2001. 7, 51

YX Zhao, CH Chou, MC Su, YZ Hsieh, et al. Portable virtual piano design. *World academy of science, engineering and technology*, 67:1024–1027, 2010. 6

U Zölzer, X Amatriain, and J Wiley. *DAFX: digital audio effects*, volume 1. Wiley Online Library, 2002. 70, 75, 76